

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD POLITÉCNICA DE CARTAGENA



## **Generación automática de código para la plataforma domótica KNX/EIB en un marco dirigido por modelos.**



*AUTOR:*

David Montoro Mouzo

*DIRECTORES:*

Dr. Manuel Jiménez Buendía

Dr. Pedro Sánchez Palma

Septiembre 2010





<b>Autor</b>	David Montoro Mouzo
<b>E-mail del Autor</b>	david@montoro.com
<b>Director(es)</b>	Manuel Jiménez Buendía, Pedro Sánchez Palma
<b>E-mail del Director</b>	manuel.jimenez@upct.es
<b>Título del PFC</b>	Generación automática de código para la plataforma domótica KNX/EIB en un marco dirigido por modelos
<b>Descriptor</b>	MDA, MDE, Eclipse, EMF, JET, domótica, KNX/EIB, ETS3, ITTools, generación automática código, plantilla, macro.
<b>Resumen:</b>	
<p>Actualmente el desarrollo de sistemas domóticos es un campo sometido a intensa investigación y el desarrollo de aplicaciones domóticas es un área en crecimiento. Para este desarrollo, se utilizan habitualmente tecnologías de bajo nivel de abstracción o frameworks de implementación específicos; haciendo del desarrollo una tarea difícil donde pueden surgir numerosos problemas. Así, hoy día cada fabricante o consorcio proporciona sus propias bibliotecas y programas de aplicación de las bibliotecas a los desarrolladores. Esta falta de un estándar común y este bajo nivel de abstracción se traducen en una metodología de desarrollo con grandes taras. Es inherente a la misma la falta de reutilización, la falta de flexibilidad y la falta de robustez (tres de los objetivos primordiales de la ingeniería del software). Además, todo el proceso de diseño y desarrollo debe realizarse por un experto en el dominio domótico. Indudable es que sería conveniente desarrollar una nueva metodología que permitiese en la medida de lo posible paliar y empezar a sentar las bases para en un futuro resolver definitivamente estos problemas.</p> <p>El Desarrollo Dirigido por Modelos (MDD) es un enfoque del desarrollo de software actual que propone la utilización de modelos como el artefacto principal para el desarrollo de software. Actualmente es una línea de investigación en alza debido a que permite una gran reusabilidad y robustez. La arquitectura para implementar el enfoque MDD propuesta por el Object Management Group (OMG) es conocida como Model Driven Architecture (MDA). Esta propuesta añade al enfoque dirigido por modelos la inclusión de varios niveles de abstracción (CIM -Computation Independent Model-, PIM -Platform Independent Model-, PSM -Platform Specific Model-) y varias transformaciones entre niveles; realizando de esta manera descripciones del sistema a varios niveles de complejidad y permitiendo aislar en ciertos dominios la descripción de los sistemas del conocimiento profundo de la tecnología.</p> <p>En este TFM se busca completar en lo máximo posible la metodología descrita por el investigador Manuel Jiménez en su Tesis Doctoral, de tal manera que se consiga trasladar la información del modelo EMF a la aplicación ETS3. Estos trabajos se basarán en primer lugar en programar Makros del plug-in ITT de ETS3 para poder realizar todas las acciones básicas en ETS3 necesarias para la realización de un proyecto domótico (véase añadir un dispositivo y modificar sus direcciones o parámetros, configurar los objetos de comunicación, etc.). Estas Makros permitirán interactuar con la plataforma ETS3 de KNX/EIB y serán el corazón de la transformación M2T que se debe implementar para lograr el objetivo. De manera paralela al desarrollo de las Makros, se depurará el DSL (Domain Specific Language) a nivel PSM para KNX propuesto por María Francisca Rosique Contreras de tal manera que de soporte a estas Makros (ya que al ser estas fruto de un trabajo de ingeniería inversa sobre las bases de datos de la plataforma ETS3 incluirán conceptos funcionales que sobrepasarán a los conceptos propios del dominio tecnológico KNX/EIB). El modelo final PSM se presentará y se explicará, y se analizará detalladamente su implementación en EMF del entorno de desarrollo Eclipse. Una vez realizado esto, se utilizará el plug-in JET de Eclipse para definir una plantilla que permita la transformación automática de descripciones realizadas usando el DSL a nivel PSM a una Makro que podrá ser ejecutada en ETS3 para crear el proyecto domótico de forma automática.</p>	
<b>Titulación</b>	Máster en Tecnologías de la Información y las Comunicaciones
<b>Departamento</b>	Electrónica, Tecnología de Computadoras y Proyectos
<b>Fecha de Presentación</b>	Septiembre - 2010



<Página intencionalmente en blanco>



# Agradecimientos

A Paqui, Manolo y Pedro.

*La mejor escuela de la vida es el poder trabajar codo con codo con gente experta y enamorada de lo que hace. En este sentido, este año yo he tenido la suerte de poder aprender de la mejor forma posible: junto a vosotros. Por ello, gracias.*

A Aurora.

*En época de cambios e incertidumbre como la que nos toca vivir, tener algo que te transmita seguridad es providencial para seguir avanzando. En este sentido, tú eres el ancla de mi vida y me permites tener la estabilidad y la ilusión necesaria para hacer cosas cada día. Gracias por estar a mi lado preciosa. Sin ti nada tendría sentido.*

A mi familia y amigos.

*Si un hombre pudiese medirse por la gente que lo rodea, no me cabe duda de que seríais gente cotizada. Gracias por seguir a mi lado, soportándome y apoyándome. Todo esto es también en parte gracias a vosotros.*





<Página intencionalmente en blanco>



# ÍNDICE DE CONTENIDOS

<b>1. Introducción</b>	<b>31</b>
1.1. Motivación y objetivos	33
1.2. Estructura del presente documento	34
<b>2. Estado del arte</b>	<b>37</b>
2.1. Sistemas domóticos	39
2.1.1. Misión de los sistemas domóticos	39
2.1.2. Dispositivos del sistema	40
2.1.3. Tipos de arquitectura	40
2.1.4. Medios de transmisión	42
2.1.5. Tecnologías existentes	44
2.1.6. Estándar EIB/KNX	45
2.2. Desarrollo dirigido por modelos (MDE)	56
2.2.1. Conceptos básicos sobre MDE	56
2.2.2. Model Driven Architecture (MDA)	61
2.2.3. Herramientas que dan soporte a MDE	65
2.3. Desarrollo de sistemas domóticos dirigido por modelos	66
2.3.1. DSL para domótica propuesto por Jiménez	67
2.3.2. DSL para KNX/EIB utilizado en este trabajo	73
2.4. Generación automatizada de código	79
2.4.1. Uso de herramientas CASE	79
2.4.2. Uso de herramientas MDA	80
2.4.3. Técnicas basadas en OCL	82
2.4.4. Técnicas basadas plantillas	82

## **3. Herramientas 85**

<b>3.1. Plataforma ETS</b>	<b>87</b>
3.1.1. <i>Características principales</i>	87
3.1.2. <i>Versiones</i>	87
3.1.3. <i>Estructura del ETS3</i>	88
3.1.4. <i>Áreas de aplicación del ETS 3</i>	93
3.1.5. <i>Requisitos mínimos del sistema</i>	93
3.1.6. <i>Otros Software complementarios</i>	93
<b>3.2. IT Tools</b>	<b>94</b>
3.2.1. <i>Características principales</i>	94
3.2.2. <i>Design</i>	95
3.2.3. <i>Reconstruction</i>	96
3.2.4. <i>Makros</i>	97
<b>3.3. Entorno de desarrollo Eclipse</b>	<b>100</b>
3.3.1. <i>Introducción</i>	100
3.3.2. <i>Arquitectura</i>	101
3.3.3. <i>Comenzando con Eclipse</i>	102
3.3.4. <i>Eclipse Modeling Framework (EMF)</i>	104
3.3.5. <i>Graphical Modeling Framework (GMF)</i>	105
3.3.6. <i>Java Emitter Template (JET)</i>	107

## **4. Metodología 109**

<b>4.1. Tutorial para uso conjunto de EMF y JET</b>	<b>111</b>
4.1.1. <i>Creación de meta-modelos con EMF</i>	111
4.1.2. <i>Modelos a partir de metamodelos usando EMF</i>	112
4.1.3. <i>Proyecto JET que use como fuente un modelo</i>	123
<b>4.2. Uso Makros y ETS3</b>	<b>133</b>
4.2.1. <i>Nociones de programación de Makros</i>	133

## **5. Trabajos realizados** **139**

<b>5.1. Macros programadas</b>	<b>141</b>
5.1.1. <i>Estructura general</i>	141
5.1.2. <i>Macros funcionales</i>	145
5.1.2.1. Macro añadir dispositivo	145
5.1.2.2. Macro modificar dirección física	145
5.1.2.3. Macro modificar parámetro	146
5.1.2.4. Macro añadir dirección de grupo	146
5.1.2.5. Macro enlazar dirección de grupo	147
5.1.3. <i>Macros auxiliares</i>	148
5.1.3.1. Macro obtener el nombre de todos los dispositivos en la base de datos	148
5.1.3.2. Macro obtener información de los dispositivos agregados al proyecto	148
5.1.3.3. Macro obtener información del dispositivo activo	148
5.1.4. <i>Implementación de un proyecto ETS3 utilizando macros</i>	149
<b>5.2. Plantilla JET</b>	<b>153</b>
5.2.1. <i>Introducción</i>	153
5.2.2. <i>Descripción de la plantilla programada.</i>	153
<b>5.3. Implementación práctica de la metodología</b>	<b>156</b>
5.3.1. <i>Maleta KNX/EIB</i>	157
5.3.2. <i>Proyecto a implementar</i>	158
5.3.3. <i>Análisis de los dispositivos implicados en el proyecto</i>	159
5.3.4. <i>Análisis de las direcciones de grupo presentes en el proyecto</i>	161
5.3.5. <i>Análisis de los parámetros asociados a los dispositivos</i>	163
5.3.5.1. <i>Análisis de los parámetros del dispositivo “Regulador de fluorescencia”</i>	163

5.3.5.2.	Análisis de los parámetros del dispositivo “Actuador de persianas”	169
5.3.5.3.	Análisis de los parámetros del dispositivo “Actuador de 6 salidas (1)”	174
5.3.5.4.	Análisis de los parámetros del dispositivo “Actuador de 6 salidas (2)”	178
5.3.5.5.	Análisis de los parámetros del dispositivo “Teclado una fase”	178
5.3.5.6.	Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (1)”	179
5.3.5.7.	Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (2)”	182
5.3.5.8.	Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (3)”	185
5.3.5.9.	Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (4)”	188
5.3.5.10.	Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (5)”	188
5.3.5.11.	Análisis de los parámetros del dispositivo “Pulsador 4 fases”	188
5.3.6.	<i>Análisis de los objetos de comunicación</i>	189
5.3.6.1.	Análisis de los objetos de comunicación del dispositivo “Regulador de fluorescencia”	189
5.3.6.2.	Análisis de los objetos de comunicación del dispositivo “Actuador de persianas”	190
5.3.6.3.	Análisis de los objetos de comunicación del dispositivo “Actuador de 6 salidas (1)”	191
5.3.6.4.	Análisis de los objetos de comunicación del dispositivo “Actuador de 6 salidas (2)”	192
5.3.6.5.	Análisis de los objetos de comunicación del dispositivo “Teclado una fase”	193
5.3.6.6.	Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (1)”	193
5.3.6.7.	Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (2)”	194
5.3.6.8.	Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (3)”	195
5.3.6.9.	Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (4)”	196

5.3.6.10. Análisis de los objetos de comunicación del dispositivo “Entrada binara 4 canales (5)”	197
5.3.6.11. Análisis de los objetos de comunicación del dispositivo “Pulsador 4 fases”	198
5.3.7. <i>Transformación JET</i>	198
5.3.8. <i>Proyecto ETS3 resultado</i>	198
<b>6. Conclusiones y trabajos futuros</b>	<b>201</b>
<b>7. Bibliografía</b>	<b>205</b>
<b>8. Abreviaturas</b>	<b>209</b>

## ÍNDICE DE ANEXOS

<b>ANEXO I. Macros programadas</b>	<b>212</b>
<b>ANEXO II. Plantilla JET</b>	<b>289</b>
<b>ANEXO III. Código generado</b>	<b>303</b>
<b>ANEXO IV. Planos de la maleta EIB/KNX</b>	<b>647</b>





<Página intencionalmente en blanco>



# ÍNDICE DE FIGURAS

## 1. Introducción

## 2. Estado del arte

<i>Figura 2.1. Diagrama representativo del tipo de arquitectura centralizada</i>	41
<i>Figura 2.2. Diagrama representativo del tipo de arquitectura descentralizada</i>	41
<i>Figura 2.3. Diagrama representativo del tipo de arquitectura distribuida</i>	42
<i>Figura 2.4. Diagrama representativo del tipo de arquitectura híbrida</i>	42
<i>Figura 2.5. Niveles de conexionado contemplados en KNX/EIB</i>	47
<i>Figura 2.6. Direcciones de grupo de dos (izquierda) o tres niveles (derecha)</i>	48
<i>Figura 2.7. Campos de la trama KNX/EIB</i>	50
<i>Figura 2.8. Catálogo de combinaciones de bits de control y su prioridad</i>	50
<i>Figura 2.9. Principales fabricantes del sistema EIN/KNX</i>	53
<i>Figura 2.10. Ejemplo ilustrativo de la relación entre modelo y metamodelo en MDE</i>	58
<i>Figura 2.11. Ejemplo ilustrativo de la relación entre modelo, metamodelo y transformación en MDE</i>	59
<i>Figura 2.12. Logotipo/resumen de MDA</i>	61
<i>Figura 2.13. Organización del desarrollo software en capas especificado por MDA</i>	62
<i>Figura 2.14. Niveles de modelado en las familias CIM, PIM y PSM</i>	64
<i>Figura 2.15. Etapas de diseño de MDA (CIM, PIM y PSM) aplicadas a un modelo de negocios cualquiera</i>	64
<i>Figura 2.16. Proyección del catálogo de Unidades Funcionales: elementos pasivos</i>	70
<i>Figura 2.17. Proyección del catálogo de Unidades Funcionales: controladores</i>	71
<i>Figura 2.18. Metamodelo para el DSL propuesto por Jiménez</i>	72
<i>Figura 2.19. Metamodelo utilizado para el DSL KNX/EIB</i>	74
<i>Figura 2.20. Sintaxis abstracta: KNXProject</i>	75
<i>Figura 2.21. Sintaxis abstracta: Device</i>	75
<i>Figura 2.22. Sintaxis abstracta: BusDevice</i>	76
<i>Figura 2.23. Sintaxis abstracta: Channel</i>	76
<i>Figura 2.24. Sintaxis abstracta: Connection</i>	76
<i>Figura 2.25. Sintaxis abstracta: CommObject</i>	77
<i>Figura 2.26. Sintaxis abstracta: Parameter</i>	77
<i>Figura 2.27. Sintaxis abstracta: Association</i>	78
<i>Figura 2.28. Sintaxis abstracta: ChannelType</i>	78
<i>Figura 2.29. Sintaxis abstracta: KNXDPT</i>	78

### 3. Herramientas

<i>Figura 3.1. Ventana principal del ETS</i>	89
<i>Figura 3.2. Barra de herramientas del ETS</i>	89
<i>Figura 3.3. Vista de edificios de ETS</i>	90
<i>Figura 3.4. Vista de Aparatos de ETS</i>	91
<i>Figura 3.5. Vista de Topología de ETS</i>	91
<i>Figura 3.6. Barra de estado de ETS</i>	92
<i>Figura 3.7. Interacción de IT Tools con ETS</i>	95
<i>Figura 3.8. Design para ETS3</i>	96
<i>Figura 3.9. Reconstruction para ETS3</i>	96
<i>Figura 3.10. Barra de herramientas de Makros</i>	98
<i>Figura 3.11. Macros preprogramadas</i>	98
<i>Figura 3.12. Editor de Makros</i>	100
<i>Figura 3.13. Arquitectura de Eclipse</i>	103
<i>Figura 3.14. Splash-screen de Eclipse Ganymede</i>	103
<i>Figura 3.15. Logotipo de GMF</i>	105
<i>Figura 3.16. Interdependencias entre GMF, GEF, EMF y la plataforma Eclipse</i>	106
<i>Figura 3.17. Flujo de diseño para el desarrollo de un editor gráfico usando GMF</i>	106

### 4. Metodología

<i>Figura 4.1. Comprobación de la correcta instalación de EMF runtime en el entorno Eclipse</i>	111
<i>Figura 4.2. Plug-ins de EMF runtime para el entorno Eclipse</i>	112
<i>Figura 4.3. Creación de un nuevo proyecto que importe el metamodelo (i)</i>	112
<i>Figura 4.4. Creación de un nuevo proyecto que importe el metamodelo (ii)</i>	113
<i>Figura 4.5. Creación de un nuevo proyecto que importe el metamodelo (iii)</i>	113
<i>Figura 4.6. Importar el metamodelo (i)</i>	114
<i>Figura 4.7. Importar el metamodelo (ii)</i>	114
<i>Figura 4.8. Importar el metamodelo (iii)</i>	115
<i>Figura 4.9. Archivos generados para el metamodelo</i>	115
<i>Figura 4.10. Modelo generador</i>	116
<i>Figura 4.11. Propiedades de un objeto</i>	116
<i>Figura 4.12. Generación del código del modelo</i>	117
<i>Figura 4.13. Generación del editor para el modelo</i>	117
<i>Figura 4.14. Proyectos generados con los sufijos .edit y .editor</i>	118
<i>Figura 4.15. Ejecutar el editor de modelos</i>	118

<i>Figura 4.16. Plug-ins generados presentes en la nueva instancia Eclipse</i>	119
<i>Figura 4.17. Crear un nuevo proyecto para generar el modelo</i>	119
<i>Figura 4.18. Añadir una instancia del meta-modelo (i)</i>	120
<i>Figura 4.19. Añadir una instancia del meta-modelo (ii)</i>	121
<i>Figura 4.20. Añadir una instancia del meta-modelo (iii)</i>	122
<i>Figura 4.21. Añadir una instancia del meta-modelo (iv)</i>	122
<i>Figura 4.22. Instancia (modelo) del meta-modelo</i>	123
<i>Figura 4.23. Crear un proyecto JET (i)</i>	124
<i>Figura 4.24. Crear un proyecto JET (ii)</i>	124
<i>Figura 4.25. Editar el archivo plugin.xml</i>	125
<i>Figura 4.26. Ejecutar la transformación (i)</i>	126
<i>Figura 4.27. Ejecutar la transformación (ii)</i>	127
<i>Figura 4.28. Ejemplo de plantilla main</i>	128
<i>Figura 4.29. Sentencia para controlar la generación de código</i>	128
<i>Figura 4.30. Sintaxis general de una directiva JET</i>	128
<i>Figura 4.31. Ejemplo de scriptlet Java en JET</i>	129
<i>Figura 4.32. Ejemplo de expresión Java en JET</i>	129
<i>Figura 4.33. Manufacturer Domain</i>	135
<i>Figura 4.34. Project Domain</i>	136
<i>Figura 4.35. EIBA Domain</i>	137

## 5. Trabajos realizados

<i>Figura 5.1. Comentario inicial de la macro Modificar dirección física</i>	142
<i>Figura 5.2. Módulo ‘inicialización de variables’ de la macro Añadir dispositivos</i>	143
<i>Figura 5.3. Módulo ‘Asignación de valores a variables entrada’ de la macro Añadir dispositivos</i>	143
<i>Figura 5.4. Módulo ‘Asignación de valores a variables auxiliares’ de la macro Añadir dispositivos</i>	144
<i>Figura 5.6. Módulo ‘Lógica de la macro’ de la macro Añadir dispositivos</i>	144
<i>Figura 5.7. Parámetros de entrada para la macro ‘Modificar dirección física’</i>	145
<i>Figura 5.8. Parámetros de entrada para la macro ‘Modificar parámetro’</i>	146
<i>Figura 5.9. Parámetros de entrada para la macro ‘Añadir dirección de grupo’</i>	147
<i>Figura 5.10. Parámetros de entrada para la macro ‘Enlazar dirección de grupo’</i>	147
<i>Figura 5.11. Proyecto a replicar</i>	149
<i>Figura 5.12. Objetos de comunicación asociados a la dirección de grupo 1/1/14</i>	150
<i>Figura 5.13. Objetos de comunicación asociados a la dirección de grupo 1/1/15</i>	150
<i>Figura 5.14. Iterador que recorre los proyectos</i>	153
<i>Figura 5.15. Inicialización de las variables</i>	153

<i>Figura 5.16. Añadir los dispositivos</i>	154
<i>Figura 5.17. Añadir las direcciones de grupo</i>	154
<i>Figura 5.18. Configurar parámetros</i>	155
<i>Figura 5.19. Enlazar direcciones de grupo</i>	156
<i>Figura 5.20. Panel superior (izqda.) y panel inferior (dcha.) de la maleta EIB/KNX</i>	157
<i>Figura 5.21. Dispositivos presentes en el proyecto</i>	161
<i>Figura 5.22. Direcciones de grupo del proyecto en EMF</i>	163
<i>Figura 5.23. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (I)</i>	164
<i>Figura 5.24. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (II)</i>	164
<i>Figura 5.25. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (III)</i>	165
<i>Figura 5.26. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (IV)</i>	165
<i>Figura 5.27. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (V)</i>	166
<i>Figura 5.28. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (VI)</i>	166
<i>Figura 5.29. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (VII)</i>	166
<i>Figura 5.30. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (VIII)</i>	167
<i>Figura 5.31. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (IX)</i>	167
<i>Figura 5.32. Descripción de los parámetros para el dispositivo “Regulador de fluorescencia” utilizando el DSL propuesto en EMF</i>	168
<i>Figura 5.32. Análisis de los parámetros para el dispositivo “Actuador de persianas” (I)</i>	169
<i>Figura 5.33. Análisis de los parámetros para el dispositivo “Actuador de persianas” (II)</i>	169
<i>Figura 5.34. Análisis de los parámetros para el dispositivo “Actuador de persianas” (III)</i>	170
<i>Figura 5.35. Análisis de los parámetros para el dispositivo “Actuador de persianas” (IV)</i>	170
<i>Figura 5.36. Análisis de los parámetros para el dispositivo “Actuador de persianas” (V)</i>	171
<i>Figura 5.37. Análisis de los parámetros para el dispositivo “Actuador de persianas” (VI)</i>	171
<i>Figura 5.38. Análisis de los parámetros para el dispositivo “Actuador de persianas” (VII)</i>	172
<i>Figura 5.39. Análisis de los parámetros para el dispositivo “Actuador de persianas” (VIII)</i>	172
<i>Figura 5.40. Análisis de los parámetros para el dispositivo “Actuador de persianas” (IX)</i>	173
<i>Figura 5.41. Análisis de los parámetros para el dispositivo “Actuador de persianas” (X)</i>	173
<i>Figura 5.42. Descripción de los parámetros para el dispositivo “Actuador de persianas” utilizando el DSL propuesto</i>	174
<i>Figura 5.43. Análisis de los parámetros para el dispositivo “Actuador de 6 salidas (1)” (I)</i>	175
<i>Figura 5.44. Análisis de los parámetros para el dispositivo “Actuador de 6 salidas (1)” (II)</i>	175
<i>Figura 5.45. Análisis de los parámetros para el dispositivo “Actuador de 6 salidas (1)” (III)</i>	176
<i>Figura 5.46. Análisis de los parámetros para el dispositivo “Actuador de 6 salidas (1)” (IV)</i>	176
<i>Figura 5.47. Análisis de los parámetros para el dispositivo “Actuador de 6 salidas (1)” (V)</i>	177
<i>Figura 5.48. Análisis de los parámetros para el dispositivo “Actuador de 6 salidas (1)” (VI)</i>	177
<i>Figura 5.49. Descripción de los parámetros para el dispositivo “Actuador de 6 salidas (1)” utilizando el DSL propuesto</i>	178
<i>Figura 5.50. Análisis de los parámetros para el dispositivo “Teclado una fase”</i>	179

<i>Figura 5.51. Descripción de los parámetros para el dispositivo “Teclado una fase” utilizando el DSL propuesto</i>	179
<i>Figura 5.52. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” (I)</i>	180
<i>Figura 5.53. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” (II)</i>	180
<i>Figura 5.54. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” (III)</i>	181
<i>Figura 5.55. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” (IV)</i>	181
<i>Figura 5.56. Descripción de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” utilizando el DSL propuesto</i>	182
<i>Figura 5.57. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” (I)</i>	183
<i>Figura 5.58. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” (II)</i>	183
<i>Figura 5.59. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” (III)</i>	184
<i>Figura 5.60. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” (IV)</i>	184
<i>Figura 5.61. Descripción de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” utilizando el DSL propuesto</i>	185
<i>Figura 5.62. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (I)</i>	186
<i>Figura 5.63. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (II)</i>	186
<i>Figura 5.64. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (III)</i>	186
<i>Figura 5.65. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (IV)</i>	187
<i>Figura 5.66. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (V)</i>	187
<i>Figura 5.67. Descripción de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” utilizando el DSL propuesto</i>	188
<i>Figura 5.68. Objetos de comunicación y asociaciones para el dispositivo “Regulador de fluorescencia”</i>	189
<i>Figura 5.69. Descripción en EMF de los objetos de comunicación para el dispositivo “Regulador de fluorescencia” usando el DSL propuesto</i>	190
<i>Figura 5.70. Objetos de comunicación y asociaciones para el dispositivo “Actuador de persianas”</i>	190
<i>Figura 5.71. Descripción en EMF de los objetos de comunicación para el dispositivo “Actuador de persianas” usando el DSL propuesto</i>	191
<i>Figura 5.72. Objetos de comunicación y asociaciones para el dispositivo “Actuador de 6 salidas (1)”</i>	191
<i>Figura 5.73. Descripción en EMF de los objetos de comunicación para el dispositivo “Actuador de 6 salidas (1)” usando el DSL propuesto</i>	191
<i>Figura 5.74. Objetos de comunicación y asociaciones para el dispositivo “Actuador de 6 salidas (2)”</i>	192
<i>Figura 5.75. Descripción en EMF de los objetos de comunicación para el dispositivo “Actuador de 6 salidas (2)” usando el DSL propuesto</i>	193
<i>Figura 5.76. Objetos de comunicación y asociaciones para el dispositivo “Teclado una fase”</i>	193
<i>Figura 5.77. Descripción en EMF de los objetos de comunicación para el dispositivo “Teclado una fase” usando el DSL propuesto</i>	193
<i>Figura 5.78. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (1)”</i>	194
<i>Figura 5.79. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (1)” usando el DSL propuesto</i>	194
<i>Figura 5.80. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (2)”</i>	194
<i>Figura 5.81. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (2)” usando el DSL propuesto</i>	195

<i>Figura 5.82. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (3)”</i>	195
<i>Figura 5.83. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (3)” usando el DSL propuesto</i>	195
<i>Figura 5.84. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (4)”</i>	196
<i>Figura 5.85. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (4)” usando el DSL propuesto</i>	196
<i>Figura 5.86. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (5)”</i>	197
<i>Figura 5.87. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (5)” usando el DSL propuesto</i>	197
<i>Figura 5.88. Objetos de comunicación y asociaciones para el dispositivo “Pulsador 4 fases”</i>	198
<i>Figura 5.89. Descripción en EMF de los objetos de comunicación para el dispositivo “Pulsador 4 fases” usando el DSL propuesto</i>	198
<i>Figura 5.90. Proyecto ETS3 resultante (I)</i>	199
<i>Figura 5.91. Proyecto ETS3 resultante (II)</i>	199

## **6. Conclusiones y trabajos futuros.**



<Página intencionalmente en blanco>



# ÍNDICE DE TABLAS

## 1. Introducción

## 2. Estado del arte

<i>Tabla 2.3. Definición de la sintaxis concreta para el DSL de Jiménez, ejemplo representativo</i>	50
<i>Tabla 2.2. Problemas relativos al desarrollo de sistemas reactivos. Aportaciones de MDA</i>	67
<i>Tabla 2.1. Tipos de datos normalizados junto con su tamaño y función</i>	73

## 3. Herramientas

## 4. Metodología

## 5. Trabajos realizados

<i>Tabla 5.1. Parámetros de los dispositivos</i>	150
<i>Tabla 5.2. Información de los dispositivos en el proyecto</i>	151
<i>Tabla 5.3. Análisis de los parámetros de los dispositivos</i>	151
<i>Tabla 5.4. Direcciones de grupo en el proyecto</i>	151
<i>Tabla 5.5. Análisis de los objetos de comunicación presentes en el proyecto</i>	152
<i>Tabla 5.6. Pulsadores analógicos, control de luces</i>	158
<i>Tabla 5.7. Pulsadores analógicos, control de persianas</i>	158
<i>Tabla 5.8. Teclados KNX/EIB, controles de luces y persianas centralizado</i>	159
<i>Tabla 5.9. Información de los dispositivos en el proyecto</i>	160
<i>Tabla 5.10. Direcciones de grupo en el proyecto</i>	162
<i>Tabla 5.11. Resumen de los parámetros para el dispositivo “Regulador de fluorescencia”</i>	168
<i>Tabla 5.12. Resumen de los parámetros para el dispositivo “Actuador de persianas”</i>	174
<i>Tabla 5.13. Resumen de los parámetros para el dispositivo “Actuador de 6 salidas (1)”</i>	178
<i>Tabla 5.14. Resumen de los parámetros para el dispositivo “Teclado una fase”</i>	179
<i>Tabla 5.15. Resumen de los parámetros para el dispositivo “Entrada binaria 4 canales (1)”</i>	182
<i>Tabla 5.16. Resumen de los parámetros para el dispositivo “Entrada binaria 4 canales (2)”</i>	185
<i>Tabla 5.17. Resumen de los parámetros para el dispositivo “Entrada binaria 4 canales (3)”</i>	188

## 6. Conclusiones y trabajos futuros



<Página intencionalmente en blanco>



## *Introducción*

---

Este capítulo expone en primer lugar la motivación que ha llevado al desarrollo de este Trabajo Fin de Máster y los objetivos perseguidos con su realización. Posteriormente, se describe la estructura del presente documento de una manera general.





## 1.1. Planteamiento inicial.

Actualmente el desarrollo de sistemas domóticos es un campo sometido a intensa investigación siendo el desarrollo de aplicaciones domóticas es un área en constante crecimiento. Para este desarrollo, se utilizan habitualmente tecnologías de bajo nivel de abstracción o *frameworks* de implementación específicos; haciendo del desarrollo una tarea difícil donde pueden surgir numerosos problemas. Así, hoy día cada fabricante o consorcio proporciona sus propias bibliotecas y programas de aplicación de las bibliotecas a los desarrolladores.

Esta falta de un estándar común y este bajo nivel de abstracción se traducen en una metodología de desarrollo con grandes taras. Es inherente a la misma la falta de reutilización, la falta de flexibilidad y la falta de robustez (tres de los objetivos primordiales de la ingeniería del software). Además, todo el proceso de diseño y desarrollo debe realizarse por un experto en el dominio domótico. Indudable es que sería conveniente desarrollar una nueva metodología que permitiese en la medida de lo posible paliar y empezar a sentar las bases para en un futuro resolver definitivamente estos problemas.

El *Desarrollo Dirigido por Modelos (MDD)* es un enfoque del desarrollo de software actual que propone la utilización de modelos como el artefacto principal para el desarrollo de software. Actualmente es una línea de investigación en alza debido a que dota al desarrollo de software de una gran reusabilidad y robustez. La arquitectura para implementar el enfoque *MDD* propuesta por el *Object Management Group (OMG)* es conocida como *Model Driven Architecture (MDA)* [OMG 10]. Esta propuesta añade al enfoque dirigido por modelos la inclusión de varios niveles de abstracción (*CIM -Computation Independent Model-*, *PIM -Platform Independent Model-*, *PSM -Platform Specific Model-*) y varias transformaciones entre niveles; realizando de esta manera descripciones del sistema a varios niveles de complejidad y permitiendo aislar en ciertos dominios la descripción de los sistemas del conocimiento profundo de la tecnología. Estos conceptos hacen que el enfoque *MDD* y la arquitectura *MDA* sean candidatos de primer orden para cimentar una nueva metodología de desarrollo de sistemas domóticos.

La *División de Sistemas e Ingeniería Electrónica (DSIE)* surge en 1999 en el seno de la *Universidad Politécnica de Cartagena*, como un Grupo de Investigación que agrupa a profesores e investigadores de los departamentos de Tecnología Electrónica, Tecnologías de la Información y las Comunicaciones (Área Lenguajes y Sistemas Informáticos) y de Ingeniería de Sistemas y Automática; y que se encuentra adscrito administrativamente al Departamento de Tecnología Electrónica. En la actualidad, las principales líneas de investigación de este grupo se centran en el desarrollo de nuevas metodologías de desarrollo y nuevas aplicaciones que apliquen conceptos de ingeniería dirigida por modelos (*MDE*) a estos campos de investigación.

En concreto, el Doctor Manuel Jiménez Buendía y la investigadora María Francisca Rosique Contreras se encuentran actualmente desarrollando una nueva metodología de diseño de aplicaciones domóticas basada en *MDE* [Jiménez 09]. Así, uno de los trabajos enmarcados en esta investigación es encontrar una transformación final modelo a texto (*M2T*) que permita trasladar los modelos a aplicaciones de desarrollo actuales como *ETS3* [ETS3 07] u

otras. La línea de investigación actual busca definir una descripción del modelo del proyecto a nivel *PSM* para la tecnología *KNX/EIB* [KNX 10a] y posteriormente otra descripción *PSM2* en texto que nos permita trasladar el modelo del proyecto a la plataforma comercial *ETS3* que nos permitirá programar los dispositivos.

## 1.2. Objetivos.

De esta forma, en este documento se describen los trabajos realizados en la línea de completar la metodología descrita por el Doctor Manuel Jiménez en su Tesis Doctoral [Jiménez 09]. Estos trabajos se basarán en primer lugar en programar *Macros IT* [Makros 08] para poder realizar todas las acciones básicas en *ETS3* necesarias para la realización de un proyecto domótico (véase añadir un dispositivo y modificar sus direcciones o parámetros, configurar los objetos de comunicación, etc.). Estas *Makros* permitirán interactuar con la plataforma *ETS3* de *KNX/EIB* y serán el corazón de la transformación *M2T* final.

De manera paralela al desarrollo de las *Makros*, se depurará el *DSL (Domain Specific Language)* a nivel *PSM* para *KNX* propuesto por María Francisca Rosique Contreras de tal manera que de soporte a estas *Makros* (ya que al ser estas fruto de un trabajo de ingeniería inversa sobre las bases de datos de la plataforma *ETS3* incluyen conceptos funcionales que se alejan de los conceptos propios del dominio tecnológico *KNX/EIB*). El modelo final *PSM* se presentará y explicará también en este documento, analizando detalladamente su implementación en *EMF* [EMF 10a] del entorno de desarrollo *Eclipse* [Eclipse 10a]. Una vez realizado esto, se utilizará el *plug-in JET* [M2T 10a] de *Eclipse* para definir una plantilla que permita la transformación automática de descripciones realizadas usando el *DSL* a nivel *PSM* a una *Macro IT* que podrá ser ejecutada en *ETS3* para crear el proyecto domótico de forma automatizada.

Tras la programación de estas herramientas, se describirá la nueva metodología de desarrollo de sistemas domóticos *EIB/KNX* y se ilustrará con un proyecto ejemplo.

Además, en un afán de complementar lo máximo posible a [Jiménez 09] y aprovechando el hecho de que este documento tiene una esencia menos seria que la de una tesis, se va a incluir toda la información posible respecto al desarrollo de *Makros* para *ETS3*, plantillas *JET* y desarrollo basado en un enfoque dirigido por modelos. En este sentido se incluye un amplio capítulo de estado del arte, un capítulo de metodología y otro de herramientas. Especial mención en este contexto los dos tutoriales incluidos en este documento: uno para uso de *JET* y *EMF* y otro para uso de *Makros* en *ETS3*. Con esto, este documento pretende trascender en cierta medida la realización de un desarrollo en concreto y servir como base para futuras investigaciones enmarcadas en la aplicación de los conceptos *MDE* al desarrollo de sistemas domóticos.

## **1.2. Estructura del presente documento.**

El presente documento se estructura como sigue: en el segundo capítulo se da una introducción y se analiza el estado actual de las investigaciones en lo relacionado con los trabajos que se pretenden realizar (en el campo de la domótica, de la ingeniería dirigida por modelos y de la generación automatizada de código), en el tercer capítulo se describen las herramientas que se pretende utilizar (ETS, ITTools y Eclipse), en el cuarto capítulo se explican las metodologías que se van a emplear (uso de EMF y JET y uso de Makros y ETS3), en el quinto capítulo se exponen los trabajos realizados (las Makros para modificar los proyectos de ETS3, la plantilla JET para realizar automáticamente la transformación y un proyecto ejemplo), y por último en el sexto capítulo se analizan las conclusiones derivadas de estos trabajos y las líneas futuras que se derivan de ellos.



## *Estado del arte*

---

Este capítulo pretende introducir al lector en el contexto en el que se desarrolla este Trabajo Fin de Máster. En primer lugar, se exponen las líneas generales de los sistemas domóticos haciendo especial hincapié en la tecnología *KNX/EIB*. En segundo lugar se explican los fundamentos de *MDE* y *MDA*. El capítulo finaliza explicando los trabajos más relevantes en el desarrollo de sistemas domóticos usando un enfoque dirigido por modelos.



## 2.1. Sistemas domóticos.

Desde el punto de vista etimológico, el término domótica fue acuñado en Francia (país pionero en Europa en temas domóticos) y está formada por la contracción de *domus* (vivienda) más automática. Se entiende por domótica al conjunto de sistemas capaces de automatizar una vivienda, aportando servicios de gestión energética, seguridad, bienestar y comunicación, y que pueden estar integrados por medio de redes de comunicación pudiendo ser controlados desde dentro y fuera del hogar. De forma breve, podemos decir que la domótica es el medio por el cual se automatiza y controla cualquier sistema del hogar. El uso de la domótica viene precedido de tres principales objetivos: la seguridad, el confort y el ahorro energético.

En el presente epígrafe se va a realizar una breve introducción teórica al campo de la domótica para permitir al lector afrontar con solvencia el resto del presente documento. Debido a que esta es la finalidad única de este epígrafe, la información en él contenida no es completa sino un resumen; de tal forma que si el lector busca ampliar sus conocimientos en estos temas se insta a leer el *Capítulo 2* de [Jiménez 09], fuente de la mayoría de la información a continuación detallada. Otras fuentes son [Meseguer 07], [Rojo 09] y [KNX 10a].

### 2.1.1. Misión de los sistemas domóticos.

La domótica tiene como misión encargarse de gestionar cuatro aspectos fundamentales:

**Confort:** la domótica aporta la automatización de servicios como la iluminación, regulación de temperatura (calor/frío), la subida y bajada de persianas, la apertura o cierre de accesos, el despliegue o el repliegue de un toldo... Todo ello de manera individual (apretando pulsadores o mediante sensores aislados) o bien mediante la creación de escenas (que permitirán mediante una sola acción, por ejemplo, todos los servicios que se necesitan al ir a dormir o al levantarse por la mañana).

**Energía:** la domótica se encarga de gestionar el consumo de energía aplicando el uso de temporizadores, relojes programadores, termostatos, entre otros, con el fin de eliminar los usos innecesarios de luz, calefacción, aire acondicionado, etc.

**Seguridad:** se puede clasificar en tres tipos de seguridades, detalladas a continuación.

- *Seguridad de bienes:* tiene como misión salvaguardar todo lo que incluye el interior de la vivienda.
- *Seguridad de las personas:* tiene como misión velar por la integridad de las personas, enlazándolas con urgencias, policía y bomberos.
- *Seguridad de alarmas:* tiene como misión detectar incendios, escapes de gas e inundaciones, a través de sensores conectados a alarmas que enlazan con la competencia correspondiente para paliar el problema.

**Comunicaciones:** comunicación interna del sistema, telecontrol y telemetría, SMS, señales acústicas, etc.

Todos estos objetivos se pueden alcanzar mediante tres tipos de formas de control remoto. Se puede aplicar el control remoto desde dentro de la vivienda (mando a distancia), desde fuera de la vivienda (telefónicamente), o bien, se pueden programar escenas (escena levantarse, escena dormir...) o programar funciones según se cumplan condiciones horarias o climatológicas (encender la refrigeración si hace más de 24º C o cuando sean las tres de la madrugada).

### 2.1.2. Dispositivos del sistema.

Para llevar a cabo su misión, la amplitud de una solución de domótica puede variar desde un único dispositivo hasta amplios sistemas que controlan prácticamente todas las instalaciones dentro de la vivienda. Los distintos dispositivos de los sistemas de domótica se pueden clasificar en los siguientes grupos:

- **Controladores:** son los dispositivos que gestionan el sistema según la programación y la información que recibe. Puede haber un controlador solo, o varios distribuidos por el sistema como se verá en el siguiente sub-apartado.
- **Actuadores:** son los dispositivos capaces de recibir y ejecutar una orden del controlador, realizando una acción sobre un aparato o sistema (encendido/apagado, subida/bajada, apertura/cierre, etc.).
- **Sensores:** son los dispositivos que monitorizan el entorno, captando información del entorno y transmitiéndola al sistema (sensores de agua, gas, humo, temperatura, viento, humedad, lluvia, iluminación, etc.).
- **Bus:** es el medio de transmisión que transporta la información entre los distintos dispositivos por un cableado propio, por la redes de otros sistemas (red eléctrica, red telefónica, red de datos) o de forma inalámbrica.

Esta clasificación no es excluyente. Dependiendo de la tecnología un dispositivo puede ser clasificado en más de una de las categorías aquí expuesto, pudiendo por ejemplo ser un actuador y un sensor a la vez.

### 2.1.3. Tipos de arquitectura.

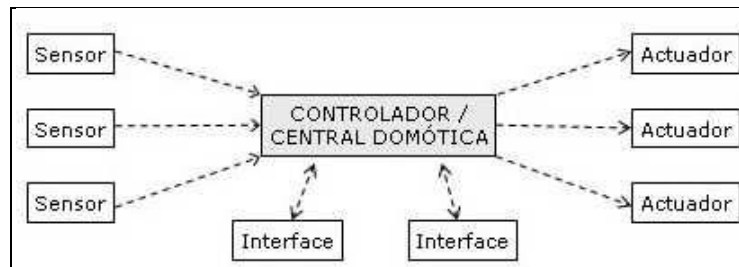
El tipo de arquitectura de un sistema domótico aporta información de cómo será la distribución y la ubicación de los elementos de control. Así, se pueden destacar cuatro tipos principales: arquitectura centralizada, arquitectura descentralizada, arquitectura distribuida y arquitectura híbrida o mixta.

#### 2.1.3.1. Arquitectura centralizada.

En un sistema domótico de arquitectura centralizada, un controlador centralizado,



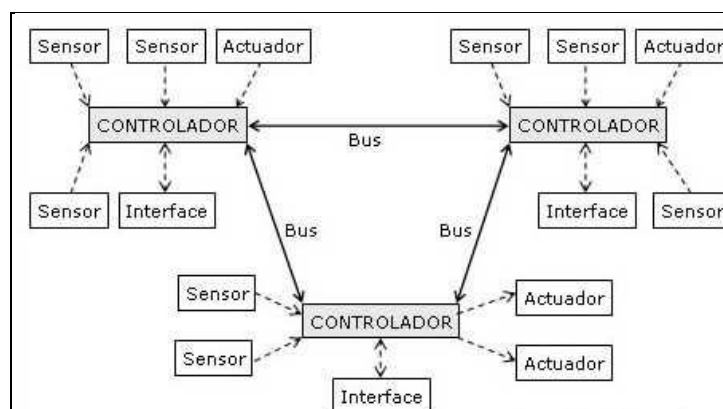
envía la información a los actuadores e interfaces según el programa, la configuración y la información que recibe de los múltiples sensores, sistemas interconectados y usuarios. Un diagrama representativo puede observarse en la *Figura 1.1*.



*Figura 2.1. Diagrama representativo del tipo de arquitectura centralizada*

### 2.1.3.2. Arquitectura descentralizada.

En un sistema domótico de arquitectura descentralizada, hay varios controladores interconectados por un bus. Estos controladores se envían información entre ellos y a los actuadores e interfaces conectados a los controladores, según el programa, la configuración y la información que reciben de los sensores, sistemas interconectados y usuarios. Un diagrama representativo puede verse en la *Figura 2.2*.



*Figura 2.2. Diagrama representativo del tipo de arquitectura descentralizada*

### 2.1.3.3. Arquitectura distribuida.

En un sistema domótico de arquitectura distribuida, cada sensor y actuador es también un controlador capaz de actuar y enviar información al sistema según el programa, la configuración, la información que capta por sí mismo y la que recibe de los otros dispositivos del sistema. Un diagrama representativo de este tipo de arquitectura puede observarse en la *Figura 2.3*.

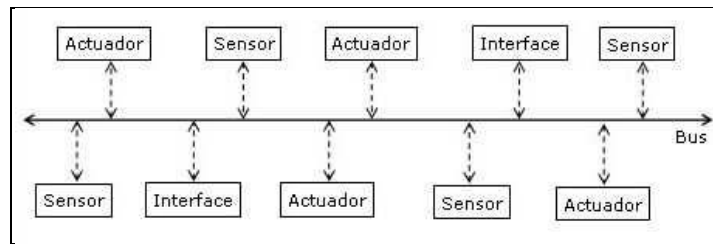


Figura 2.3. Diagrama representativo del tipo de arquitectura distribuida

2.1.3.4. Arquitectura Híbrida o Mixta.

En un sistema domótico de arquitectura híbrida (también denominado de arquitectura mixta), se combinan las arquitecturas de los sistemas centralizadas, descentralizadas y distribuidas. Así, se puede disponer de un controlador central o varios controladores descentralizados o puede darse el caso de que los dispositivos de interfaces, sensores y actuadores puedan también ser controladores (como en un sistema distribuido); pudiendo procesar la información según el programa, la configuración, la información que captan por sí mismos, y tanto actuar como enviarla a otros dispositivos de la red, sin que necesariamente pase por otro controlador. Un diagrama representativo de este tipo de arquitectura puede observarse en la Figura 2.4.

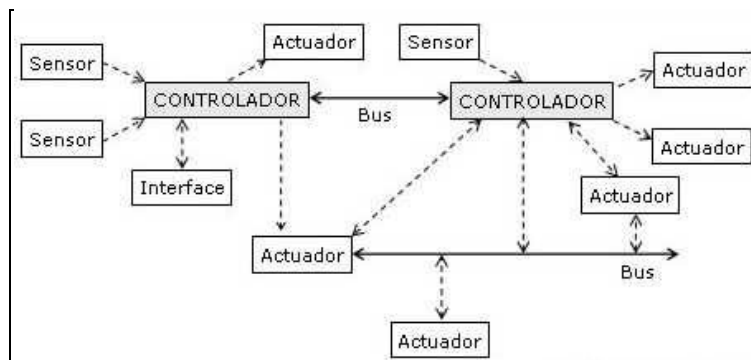


Figura 2.4. Diagrama representativo del tipo de arquitectura híbrida

2.1.4. Medios de transmisión.

En todo sistema domótico los diferentes dispositivos deben intercambiar información unos con otros a través de diferentes medios de transmisión. Los principales se detallan a continuación:

**Medios dedicados:** La transmisión a través de medios dedicados es la implementación más común que toman los sistemas de domótica. Algunos de los principales medios se detallan a continuación:

- *Fibra Óptica*: está constituida por un material dieléctrico transparente conductor de luz, compuesto por un núcleo con un índice de refracción menor que el del revestimiento, que envuelve el núcleo. Estos dos elementos forman una guía para que la luz se desplace por la fibra. Aporta fiabilidad a la transferencia de datos, es resistente a cualquier tipo de interferencia, puede transferir grandes cantidades de datos y es muy segura. Por el contrario, el coste de los cables y conexiones es elevado.
- *Cable Coaxial*: Es un cable utilizado para transportar señales eléctricas de alta frecuencia que posee dos conductores concéntricos, uno central, denominado positivo, encargado de llevar la información, y uno exterior, de aspecto tubular, denominado malla o blindaje, que sirve como referencia de tierra y retorno de las corrientes. Entre ambos se encuentra una capa aislante denominada dieléctrico, de cuyas características dependerá principalmente la calidad del cable. Todo el conjunto suele estar protegido por una cubierta aislante. En sistemas domóticos se usan fundamentalmente para la distribución de servicios de televisión y radio.
- *Par Trenzado*: Es una forma de conexión en la que dos conductores son entrelazados para cancelar las interferencias electromagnéticas (IEM) de fuentes externas y la diafonía de los cables adyacentes. El entrelazado de los cables disminuye la interferencia debido a la reducida área de bucle entre los cables (la cual determina el acoplamiento magnético en la señal). En la operación de balanceado de pares, los dos cables suelen llevar señales iguales y opuestas (modo diferencial), las cuales son combinadas mediante sustracción en el destino. El ruido de los dos cables se cancela mutuamente en esta sustracción debido a que ambos cables están expuestos a interferencias electromagnéticas similares. La tasa de trenzado, usualmente definida en vueltas por metro, forma parte de las especificaciones de un tipo concreto de cable. Cuanto mayor es el número de vueltas, mayor es la atenuación de la diafonía. El par trenzado es la solución más utilizada en sistemas domóticos.
- *Cables paralelos*: Son los utilizados tradicionalmente en telefonía. Son más sensibles a la diafonía y a las interferencias que los pares trenzados. En aplicaciones domóticas se utilizan para la conexión de sensores controladores o conexiones eléctricas auxiliares.

**Medios compartidos**: Varias soluciones utilizan cables compartidos y/o redes ya existentes para la transmisión de su información. Se pueden destacar las siguientes.

- *Red eléctrica*: Es la solución idónea para instalaciones existentes. El coste que acarrea es bajo y tiene un fácil conexionado. Por el contrario, la fiabilidad y la velocidad a la hora de transferir datos son bajas.
- *Inalámbrica*: Muchos sistemas de domótica utilizan soluciones de transmisión inalámbrica entre los distintos dispositivos, principalmente tecnologías de radiofrecuencia o infrarrojo.

- **Infrarrojo:** es el más extendido. Sus puntos fuertes son la comodidad y la flexibilidad, además de admitir un importante número de aplicaciones y ser resistente a cualquier tipo de interferencia electromagnética. Por el contrario, va a requerir visibilidad entre emisor y receptor.
- **Radiofrecuencia:** su aplicación aporta mucha flexibilidad al usuario pero tiene el gran inconveniente de la sensibilidad a las interferencias electromagnéticas y la complejidad en cuanto a temas de seguridad y fiabilidad.

### 2.1.5. Tecnologías existentes.

Actualmente existen numerosos sistemas domóticos comerciales, cada uno de ellos está orientado a un segmento concreto del mercado. Desde el punto de vista comercial, se puede decir que los tres sectores más importantes que precisan actualmente de estos sistemas son las casas ya construidas, las casas nuevas y los grandes edificios (hoteles, oficinas, residencias). Cada uno de estos sectores utiliza una tecnología específica, adaptada a las necesidades del usuario final.

En una casa construida, se suelen utilizar sistemas denominados de corrientes portadoras (traducción del francés "*courant porteur*"), que tienen como soporte de comunicación la propia red de alimentación de baja tensión (BT) de 220 V, presente en la vivienda. El motivo del empleo de este tipo de tecnología es el elevado coste, y en muchos casos la imposibilidad, de realizar un nuevo cableado para el sistema domótico. En este caso, los sistemas mayoritariamente adoptados por los instaladores son el sistema europeo CAD de Legrand y el americano X-10 de Home Systems, comercializado en Europa por Niessen.

Si se trata de una casa nueva, dependiendo de su tamaño y de los requisitos, los sistemas centralizados comerciales (SCC) suelen ser bastante apropiados. Las gamas bajas de SCC se suelen aplicar a nuevas viviendas de tamaño pequeño sin grandes requerimientos. Las gamas altas de SCC se emplean en viviendas nuevas de tamaño medio-grande con necesidades más avanzadas, aunque se está extendiendo cada vez más la implantación de sistemas distribuidos en este tipo de viviendas.

Existe un producto centralizado muy popular entre los instaladores europeos denominado IHC (*Innovation House Control*); que en España ha sido adoptado por la empresa Simon, comercializándose bajo el nombre de SimonVIS. Tiene la ventaja de tener un coste muy reducido y no requiere ningún tipo de especialización para su instalación. También existen otros sistemas menos populares como Amigo (*Merlín Gerin*), Microdelta (*Delta Dore*), Domoconcept, y otros muchos propietarios de diferentes fabricantes.

En el caso de un edificio, las necesidades son más complejas que las de una casa. En este caso, y teniendo en cuenta la cantidad de cableado necesario, son los sistemas en bus los que ganan terreno respecto a los demás, aunque en algunos casos las gamas altas de SCC también se pueden aplicar si la relación cableado/componentes lo permite. Los sistemas tipo

bus más instalados en Europa eran el *BatiBus* de *Merlin Gerin*, y el *EIB* desarrollado por un consorcio europeo que engloba empresas como *Siemens*, *Niessen*, *ABB*, *Legrand*, *Hager*, etc... Actualmente se encuentran englobados en el estándar *Konnex*. El sistema más popular en Estados Unidos es el *Lonworks* de *Echelon*. En Europa está poco introducido. Otros sistemas aplicables en este tipo de instalaciones son *CEBus* de la *EIA*, *EHS* de *EHSA*, *Smart House* de la *NAHB*, y en el caso de *SCC* de gama alta: *Sysmac* de *Omron*, *B3d* de *Performer 2000*, *D2B* de *Philips*, etc.

En resumen, se puede decir que los sistemas más instalados en Estados Unidos son *CEBus*, *X-10*, *Lonworks* y *Smart House*. A nivel europeo, los sistemas más importantes son *EIB*, *SimonVIS*, *Batibus* y *EHS*. Cabe destacar que los sistemas *Batibus*, *EIB* y *EHS* se han unido formando un consorcio para conseguir la compatibilidad de productos entre ellos. Este proceso ha sido denominado convergencia (*Konnex*, *KNX*), siendo el sistema *EIB* el que lidera la iniciativa y prevaleciendo sobre los otros dos.

Así, por ser el campo de directa aplicación de este estudio, se dedicará a continuación un apartado para tratar el sistema *EIB/KNX* de forma extensa. Para obtener una información más detallada del resto de tecnologías se recomienda la lectura del *Capítulo 2* de [Jiménez 09].

#### **2.1.6. Estándar EIB/KNX.**

El Bus de Instalación Europeo (*EIB*, *European Installation Bus*) surgió con la idea de introducir en el mercado un sistema unificado para la gestión de edificios, creado por el consorcio europeo *EIBA* (*European Installation Bus Association*) en 1990 por más de setenta compañías (*ABB*, *Siemens*, *Jung...*).

En la actualidad la asociación tiene más de cien miembros, existiendo unas veinte empresas que suministran productos, siendo las más importantes *Siemens*, *ABB*, *Temper*, *Grasslin* y *Niessen*. También existen miembros científicos que colaboran en el desarrollo de actividades de I+D, especialmente universidades y centros de investigación.

Las funciones de la asociación son básicamente el soporte para la preparación de normas unificadas y la definición de las pruebas y requisitos de homologación que garanticen la calidad y compatibilidad de los productos. En la actualidad, *EIBA* se ha integrado dentro de la asociación *Konnex* (*KNX*) para la convergencia de sistemas europeos.

Se trata, además, de un sistema abierto sobre las mismas premisas que otros sistemas de comunicación como los buses de campo abiertos: tanto las especificaciones del protocolo como los procedimientos de verificación y certificación están disponibles, así como los componentes críticos del sistema (microprocesadores específicos con la pila del protocolo y electrónica de acoplamiento al bus).

*KNX/EIB* está recogido en normas europeas e internacionales (*EN 50090* e *ISO/IEC 14543-3-X*), y es la tecnología domótica abierta más instalada en España y a nivel europeo. Por este motivo, se ha escogido trabajar con la tecnología *KNX/EIB* en el presente estudio.

### 2.1.6.1. Aspectos tecnológicos.

El *KNX/EIB* es un sistema distribuido (no requiere de un controlador central de la instalación), en el que todos los dispositivos que se conectan al bus de comunicación de datos tienen su propio microprocesador y electrónica de acceso al medio.

En una red *EIB* podemos encontrar cuatro tipos de componentes: módulos de alimentación de la red, acopladores de línea para interconectar diferentes segmentos de red, y elementos sensores y actuadores.

El estándar *KNX* permite el uso de cuatro medios de comunicación para la ya mencionada interconexión de dispositivos:

- *TP (Twisted Pair, TP1)*: par trenzado, de velocidad de transmisión *9600 bits/s*, con origen en la tecnología *EIB*. Todos los productos certificados *EIB TP1* y *KNX TP1* podrán operar en un mismo bus, así como intercambiarán información.
- *PL (Power Line, PL110)*: *power line*, de velocidad de transmisión *1200 bits/s*, también con origen en el estándar *EIB*. Los productos certificados *EIB* y *KNX PL110* operarán y se comunicarán los unos con los otros en una misma red de distribución eléctrica.
- *RF (Radio Frequency)*: Los dispositivos *KNX* admiten este medio de transmisión que emplea señales de radio para transmitir telegramas *KNX*. Dichos telegramas son transmitidos en la banda de frecuencia *868 MHz* (dispositivos de corto alcance), con una fuerza máxima irradiada de *25 mW* y velocidad de transmisión de *16.384 kBit/sec*. Permite implementaciones tanto unidireccionales como bidireccionales, se caracteriza por su bajo nivel de consumo energético y está destinado a pequeñas y medianas instalaciones que sólo necesitan transmisores en casos excepcionales.
- *IP (Ethernet)*: Los telegramas *KNX* pueden ser también encapsulados en telegramas *IP*. De esta forma, se podrán transportar a través de redes *LAN*, así como de Internet.

### 2.1.6.2. Topología

Para el conexionado de dispositivos del bus en cada línea se permite cualquier topología: árbol, estrella, bus o anillo, lo que facilita la instalación en viviendas y edificios. Existe una única restricción: no se permite cerrar anillos entre líneas situadas topológicamente en diferentes subredes.

La topología de conexión de dispositivos contempla tres niveles de conexionado:

- **Línea:** Es la unidad mínima de instalación. En ella se pueden conectar hasta 64 dispositivos (dependiendo de la capacidad de la fuente de alimentación y de la carga máxima producida por los dispositivos existentes). Si se desean conectar más componentes al bus, se habrá de instalar una nueva línea, que se acoplará, junto con la primera, a una línea principal mediante acopladores de línea (AL). Se pueden acoplar hasta 15 líneas en la línea principal, constituyendo un área. Cada línea, tanto la principal como las secundarias, deben tener su propia fuente de alimentación. Además, la línea principal puede tener conectados directamente hasta 64 dispositivos (incluyendo los acopladores de línea).
- **Área:** Como se acaba de comentar, se constituye acoplando hasta 15 líneas en la línea principal. De este modo, en un área se pueden conectar hasta 960 dispositivos.
- **Sistema completo:** Formado por la unión de hasta un total de 15 áreas distintas mediante los denominados Acopladores de Área (AA), que permitiría integrar hasta un máximo de 14.400 dispositivos.

Además, mediante pasarelas pueden interconectarse distintos sistemas completos, consiguiendo de este modo que el número de componentes y las posibilidades sean ilimitadas.

Un diagrama ilustrativo de todos estos niveles de conexión puede observarse en la *Figura 2.5.* [Jiménez 09].

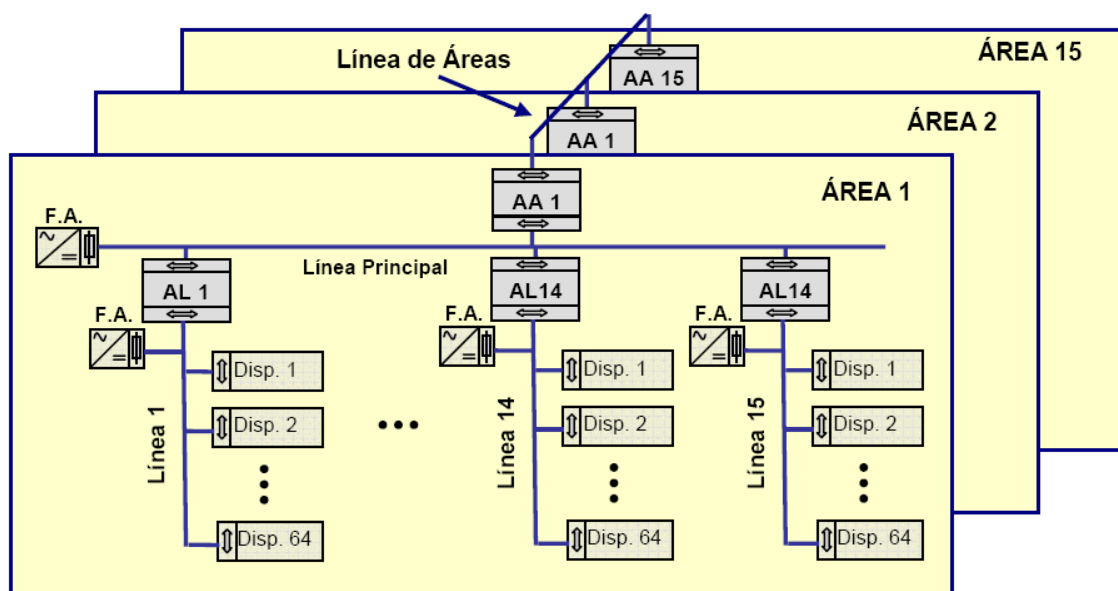


Figura 2.5. Niveles de conexión contemplados en KNX/EIB [Jiménez 09]

### 2.1.6.3. Direccionamiento

Los diferentes elementos existentes en una instalación *KNX/EIB* quedan perfectamente identificados gracias al sistema de direccionamiento. Existen dos tipos de direcciones: direcciones físicas y direcciones de grupo.

- **Direcciones físicas:** Las direcciones físicas identifican unívocamente cada dispositivo y corresponden con su localización en la topología global del sistema (área - línea secundaria - dispositivo). La dirección física consta de tres campos, que se representan separados por puntos:
  - *Área (4 bits)*. Identifica una de las 15 áreas.  $A=0$  corresponde a la dirección de la línea de áreas del sistema.
  - *Línea (4 bits)*. Identifica cada una de las 15 líneas en cada área.  $L=0$  se reserva para identificar a la línea principal dentro del área.
  - *Dispositivo (8 bits)*. Identifica cada uno de los posibles dispositivos dentro de una línea, hasta 255.  $D=0$  se reserva para el acoplador de línea.
- **Direcciones de grupo:** Las direcciones de grupo se emplean para definir funciones específicas del sistema, y son las que determinan las asociaciones de dispositivos en funcionamiento (y la comunicación entre sus objetos de aplicación). Las direcciones de grupo asignan la correspondencia entre elementos de entrada al sistema (sensores) y elementos de salida (actuadores).

Se pueden utilizar dos tipos de direccionamiento de grupo: de dos y tres niveles, dependiendo de las necesidades en la jerarquización de las funciones del sistema (Figura 2.6.) [Jiménez 09].

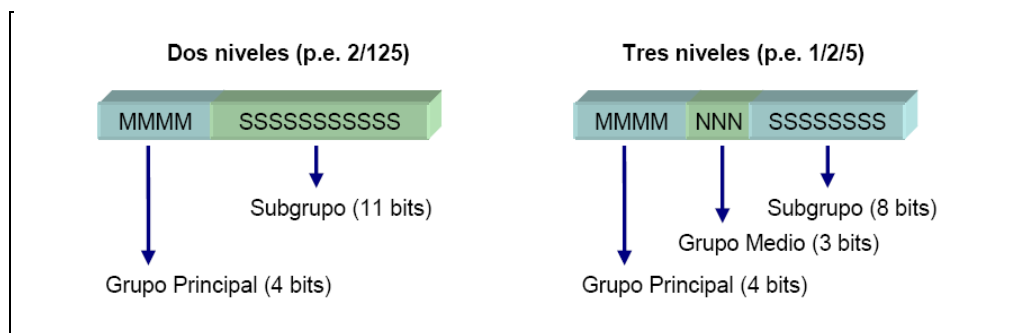


Figura 2.6. Direcciones de grupo de dos (izquierda) o tres niveles (derecha) [Jiménez 09]

Habitualmente el campo de grupo principal se utiliza para englobar grupos de funciones (alarmas, iluminación, control de persianas, etc.). Se pueden emplear valores de 1 a 13, los valores 14 y 15 no deben emplearse, ya que no son filtrados por los acopladores y podrían afectar a la dinámica de funcionamiento de todo el sistema. En todos los campos la dirección 0 está reservada para funciones del sistema.

En la configuración de una instalación *EIB*, la asignación de direcciones de grupo es básica para asegurar su correcto funcionamiento. Las direcciones de grupo, que



asocian sensores con actuadores, se pueden asignar a cualquier dispositivo en cualquier línea (son independientes de las direcciones físicas), con las siguientes condiciones:

- Los sensores sólo pueden enviar una dirección de grupo (sólo se les puede asociar una dirección de grupo).
- Varios actuadores pueden tener la misma dirección de grupo, es decir, responden a un mismo mensaje o telegrama.
- Los actuadores pueden responder a más de una dirección de grupo (pueden estar direccionados o asociados a varios sensores simultáneamente).

#### 2.1.6.4. Acceso al medio y telegramas

El **método de acceso al medio** empleado en EIB es de tipo *CSMA/CA* (*Carrier Sense Multiple Access/Collision Avoidance*): Acceso múltiple por detección de portadora, evitando colisiones.

La codificación se realiza de modo que el estado lógico *0* es dominante (impulso simétrico) sobre el *1*, que se denomina recesivo (no hay impulso). El mecanismo de resolución de colisiones es el siguiente:

- El dispositivo comprueba el bus, y si está libre comienza la transmisión.
- Durante el envío cada dispositivo escucha los datos presentes en el bus, comparándolos en todo momento con los que ha transmitido.
- Si no se producen colisiones, el envío se completa sin contratiempos.
- Si, por el contrario, se produce una colisión con los datos enviados por otro equipo, el arbitraje se resuelve por prioridad de los bits dominantes sobre los recesivos.

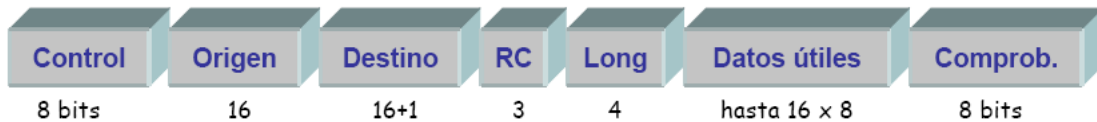
Por lo tanto, tendrán mayor prioridad aquellas tramas que presenten un mayor número de ceros en su inicio.

El envío de un mensaje o **telegrama** en un sistema EIB se realiza cuando se produce un evento, por ejemplo la activación de un pulsador o la detección de presencia. La transmisión se inicia después de que el bus haya permanecido desocupado por lo menos durante un periodo de *50 bits*, y después de la transmisión los componentes utilizan otro tipo de *13 bits* para comprobar si el telegrama se ha recibido correctamente. Todos los componentes direccionados envían un acuse de recibo.

Los telegramas se transmiten en modo asíncrono, a una velocidad de *9600 baudios*, donde cada carácter o byte consta de *1 bit* de inicio, *8 bits* de datos, *1 bit* de paridad par, *1 bit* de parada y una pausa de *2 bits* hasta la siguiente transmisión. De este modo la transmisión

de un *byte* supone un tiempo de *1,35 ms*, y la de un telegrama completo entre *20 y 40 ms* (la mayoría de las órdenes son de marcha-paro y suponen un tiempo de envío de *20 ms*).

El telegrama que se transmite por el bus, y que contiene la información específica sobre el evento que se ha producido, tiene siete campos: seis de control para conseguir una transmisión fiable y un campo de datos útiles con el comando a ejecutar. Estos campos pueden verse en la *Figura 2.7*. [Jiménez 09].



*Figura 2.7. Campos de la trama KNX/EIB [Jiménez 09]*

El campo de *control* (*8 bits*) incluye la prioridad que dicho telegrama tiene al ser enviado según el tipo de función (alarma, servicios del sistema o servicios habituales). El bit de repetición se pone a cero en caso de repetirse algún envío a causa del no reconocimiento de alguno de los destinatarios. De este modo se evita que los mecanismos que ya ha ejecutado la orden la vuelvan a repetir. El catálogo de combinaciones de bits de control y su prioridad puede apreciarse en la *Figura 2.8*.

1	0	W	1	P	P	0	0	Prioridad de transmisión
				0	0			Funciones del sistema (prioridad máx.)
				1	0			Funciones de alarma
				0	1			Prioridad de servicio elevada (manual)
				1	1			Prioridad de servicio baja (automático)
		0						Repetición

*Figura 2.8. Catálogo de combinaciones de bits de control y su prioridad*

El campo *origen* (*16 bits*) incluye la dirección física del dispositivo que transmite la trama: *4 bits* con el área, *4 bits* de identificador de línea y *8 bits* de identificador de dispositivo.

El campo *destino* (*17 bits*) incluye la dirección de destino, que será de dos tipos en función del valor que tome el *bit* más significativo (el *bit* número *17*). Si tiene valor *0*, se trata de una dirección física, y el telegrama se dirige únicamente a un dispositivo. Si tiene valor *1*, se trata de una dirección de grupo, y el telegrama se dirige a todos los mecanismos que deben escucharlo (los que tengan esa dirección de grupo). El resto de bits codifican la dirección de la misma manera que el campo anterior.

El campo *RC* es un contador de ruta y el campo *long* indica la longitud del campo de datos en *bytes*. El campo *datos útiles* contiene el tipo de comando (hay cuatro tipos) y los datos, de acuerdo con el *EIB Interworking Standard (EIS)*. El campo *comprob.* es un *byte* de comprobación.

El *EIB Interworking Standard (EIS)* recoge los tipos de datos normalizados junto con su tamaño y función, tal como se puede apreciar en la *Tabla 2.1*. [Jiménez 09].

Nº EIS	Función EIB	Nº bytes	Descripción
EIS 1	Conmutación (switching)	1 bit	Encendido/apagado, habilitar/deshabilitar, alarma/no alarma, verdadero/falso
EIS 2	Regulación (dimming)	4 bit	Se puede utilizar de 3 formas distintas: como interruptor, como valor relativo y como valor absoluto.
EIS 3	Hora (time)	3 bytes	Día de la semana, hora, minutos y segundos.
EIS 4	Fecha (date)	3 bytes	Día/mes/año (el margen es de 1990 a 2089).
EIS 5	Valor (value)	2 bytes	Para enviar valores físicos con representación S,EEEE,MMMMMMMMMM.
EIS 6	Escala (scaling)	8 bit	Se utiliza para transmitir valores relativos con una resolución de 8 bit. Por ejemplo, FF = 100 %.
EIS 7	Control motores (control drive)	1 bit	Tiene dos usos: Mover, arriba/abajo o extender/retraer y Paso a Paso.
EIS 8	Prioridad (priority)	1 bit	Se utiliza en conjunción con EIS 1 ó EIS 7.
EIS 9	Coma flotante (float value)	4 bytes	Codifica un número en coma flotante según el formato definido por el IEEE 754.
EIS 10	Contador 16 bit (16b-counter)	2 bytes	Representa los valores de un contador de 16 bit (tanto con signo como sin signo).
EIS 11	Contador 32 bit (32b-counter)	4 bytes	Representa los valores de un contador de 32 bit (tanto con signo como sin signo).
EIS 12	Acceso (access)	4 bytes	Se usa para conceder accesos a distintas funciones.
EIS 13	Caracter ASCII (Character)	8 bit	Codifica según el formato ASCII.
EIS 14	Contador 8 bit (8b-counter)	8 bit	Representa los valores de un contador de 8 bit (tanto con signo como sin signo).
EIS 15	Cadena (Character String)	14 bytes	Transmite un cadena de caracteres ASCII de hasta 14 bytes.

Tabla 2.1. Tipos de datos normalizados junto con su tamaño y función [Jiménez 09]

El *EIS* contiene los datos útiles para cada función asignada a los objetos de comunicación. Según este estándar existen siete tipos diferentes, cada uno asignado a un tipo de acción de control (conmutación, regulación de luz, envío de valor absoluto, envío de valor en punto flotante, etc.). De este modo se garantiza la compatibilidad entre dispositivos del mismo tipo de diferentes fabricantes.

Los objetos de comunicación son instancias de clases definidas en el estándar, y se incluyen en los programas almacenados en la memoria de los dispositivos para realizar una determinada acción. Normalmente, el programa de aplicación que se ejecuta en un dispositivo dispone de varios objetos de comunicación, que pueden ser de diferentes tipos *EIS*.

Por ejemplo, un pulsador de dos teclas con un programa de control de iluminación puede tener cuatro objetos: dos de conmutación (uno para cada tecla), tipo *EIS 1*, que envían

las órdenes de encendido-apagado, y otros dos de regulación (uno para cada tecla), tipo *EIS 2*, para el envío de órdenes de incremento-decremento de luminosidad.

Las asociaciones de direcciones de grupo, descritas con anterioridad, se realizan para cada uno de estos objetos de comunicación, de modo que un componente *EIB*, con una única dirección física, contiene varios sensores o varios actuadores, cuyo funcionamiento lógico es independiente.

#### 2.1.6.5. Programación de los dispositivos

La programación supone la etapa final de la realización de un proyecto de instalación *KNX/EIB*. El estándar *KNX* permite tres modos de programación:

- *Easy installation (E-mode)*: Este mecanismo de configuración está destinado a instaladores con una formación básica y provee una rápida evolución del aprendizaje pero con funciones limitadas. Los componentes *E-mode* están ya preprogramados y cargados con una serie de parámetros. Con un configurador simple, cada componente puede ser en parte reconfigurado (principalmente los parámetros de configuración y los enlaces de comunicación). La compatibilidad de los productos *E-mode* normalmente tiene limitada su funcionalidad y ha sido pensada para instalaciones de tamaño medio.
- *Automatic mode (A-mode)*: Este mecanismo de configuración está desarrollado especialmente para aplicaciones de usuario final, por ejemplo, dispositivos domésticos o pequeño material de instalación, vendido a través de los canales de venta al gran público. Los componentes *A-mode* disponen de mecanismos de configuración automática, que adaptan sus enlaces de comunicación al resto de componentes *A-mode* en la red. Cada componente contiene un número determinado de parámetros fijados y una librería con instrucciones de cómo comunicar con otros componentes *A-mode*.
- *System installation (S-Mode)*: El diseño de la instalación y la configuración se hace a través de un ordenador con el software *ETS (Engineering Tool Software*, descrita en el *Capítulo 3, Apartado 1* del presente documento) [ETS3 07], que sirve de herramienta para usar las bases de datos de producto de cada fabricante. *S-mode* está pensado para integradores de sistemas certificados y para grandes instalaciones. Los integradores podrán realizar instalaciones sofisticadas. Se realiza habitualmente conectando un ordenador personal al bus mediante una pasarela (*RS232, USB o IP*) y requiere realizar las siguientes tareas:
  - Programar las direcciones físicas de los dispositivos de acuerdo con la topología utilizada.

- Cargar los programas de aplicación que se van a utilizar para los dispositivos en el ya mencionado entorno *ETS*. Estos programas de aplicación son proporcionados por los fabricantes en forma de librerías.
- Configurar una serie de parámetros que definen el funcionamiento del programa de aplicación.
- Realizar los enlaces lógicos entre sensores, actuadores y controladores mediante la asignación de direcciones de grupo a los objetos de comunicación de los dispositivos.
- Descargar los programas de aplicación con los parámetros y asociaciones definidas en los dispositivos de la red.

#### 2.1.6.6. Principales fabricantes EIB/KNX

Los principales fabricantes del sistema EIB/KNX quedan reflejados en el cuadro de la Figura 2.9.

<i>ABB</i> 	<i>ARCUS</i> 	<i>B+B</i> 	<i>BERKER</i> 
<i>BUSH-JAEGER</i> 	<i>ELSNER</i> 	<i>GIRA</i> 	<i>HAGER</i> 
<i>IPAS</i> 	<i>JUNG</i> 	<i>LINGG &amp; JANKE</i> 	<i>MERTEN</i> 
<i>SIEMENS</i> 	<i>THEBEN</i> 	<i>WHD</i> 	<i>ZENNIO</i> 

Figura 2.9. Principales fabricantes del sistema EIB/KNX

A continuación se detallan las principales características de algunos de ellos:

- **ABB:** Multinacional con un gran abanico de productos, y destaca por los aparatos de carril *DIN*. Especial mención a la calidad y seguridad que ofrece.
- **ARCUS:** Destaca por su amplia gama de sensores *KNX*, y su producto estrella es la pantalla *MicroVis Logic+*. Se trata de una pequeña pantalla (tamaño de cajetín universal) que se maneja con una pequeña rueda-cursor y se adapta a la estética de muchas gamas de pulsadores *KNX*. Es muy completa, ofrece una gran funcionalidad y tiene una calidad de imagen excelente.
- **GIRA:** Fabricante de origen en Alemania/Austriaco, con un gran abanico de de productos en su catálogo.

- *IPAS*: Fabricante alemán dinámico, innovador, flexible y que cuenta con unos ingenieros de desarrollo sumamente especializados. Sus pulsadores de diseño decoran las instalaciones *KNX* más elegantes del mundo, sus pantallas táctiles con software de visualización son la más alta gama *KNX* y su control de accesos por tarjeteros es único. Sin embargo, lo que ha dado a *IPAS* fama mundial son sus pasarelas *TCP/IP ComBridge*, en sus distintas versiones.
- *JUNG*: Jung es uno de los fabricantes *KNX* que más destacan por sus diseños de componentes empotrables (elegantes y no demasiado caros). En general, posee un catálogo muy completo de productos *KNX*.
- *LINGG & JANKE*: Fabricante alemán especializado en componentes de carril *DIN* robustos, económicos, completos y fiables.
- *MERTEN*: Adquirida por el grupo *Schneider*, se ha convertido en uno de los gigantes *KNX*. Su catálogo es muy completo, tanto en lo que al diseño se refiere, como a los aparatos de carril *DIN*.
- *SIEMENS*: Sin duda, de uno de los grandes pilares de *KNX*. Se le puede conceder el mérito de haber sido el pionero del *EIB*. Actualmente, posee un catálogo muy variado y amplio de productos. Destacan, entre otras cosas, su *KNX-IP Router*, y sus aparatos de carril *DIN*.
- *ZENNIO*: Fabricante español que en sus inicios sorprendió al mercado del *KNX*, ha revolucionado la forma de concebir las instalaciones *KNX* y en la actualidad se ha consolidado gracias al buen resultado de sus primeros productos.

#### 2.1.6.7. Ventajas derivadas de la utilización de *EIB/KNX*

Se pueden destacar las siguientes ventajas:

- *Estándar internacional* que garantiza su continuidad en el futuro:
  - *ISO/IEC*: Aprobó la tecnología *KNX* como el Estándar Internacional *ISO/IEC14543-3* en 2006.
  - *CENELEC*: Aprobó la tecnología *KNX* como el Estándar Europeo *EN 50090* en 2003.
  - *CEN*: Aprobó la tecnología *KNX* como *EN 13321-1* (simple referencia a *EN50090*) y *EN13321-2 (KNXnet/IP)* en 2006.
  - *SAC*: Aprobó la tecnología *KNX* como Estándar *GB/Z20965* en China en 2007.

- *ANSI/ASHRAE*: Aprobó la tecnología *KNX* como el Estándar Estadounidense *ANSI/ASHRAE135* en 2005.
  
- *Interoperabilidad & Interworking* de productos: El proceso de certificación *KNX* asegura que funcionarán y se comunicarán diferentes productos de diferentes fabricantes usados en diferentes aplicaciones. Esto asegura un alto grado de flexibilidad en la extensión y modificaciones de las instalaciones. Laboratorios neutrales (terceras compañías) son quienes analizan la conformidad del producto. *KNX* es el único estándar para el control de casas y edificios que lleva a cabo un plan de certificación para productos, centros de formación (instituciones profesionales y privadas) e incluso personas (electricistas, proyectistas).
  
- *Calidad elevada* del producto: La *KNX Association* exige un alto nivel de producción y control de calidad durante todas las etapas de la vida del producto. Así, todos los miembros fabricantes tienen que mostrar conformidad a la norma *ISO 9001* incluso antes de que soliciten la certificación para productos *KNX*. Además de la conformidad del fabricante a la norma *ISO 9001*, los productos tienen que cumplir con los estándares tanto Europeos como Internacionales para el control de casas y edificios. En caso de duda, la *KNX Association* tiene el derecho de volver a analizar el producto o puede exigir al fabricante el informe de conformidad de dicho hardware.
  
- *Software unificado*: La herramienta software *ETS* (ya mencionada) permite proyectar, diseñar y configurar todos los productos certificados *KNX*. Dicha herramienta es además independiente del fabricante: el integrador de sistemas podrá combinar los productos de varios fabricantes en una instalación.
  
- *Sistema completo*: *KNX* puede ser usado para el control de todas las posibles funciones/aplicaciones en casas y edificios desde iluminación, contraventanas, control de seguridad y alarmas, calefacción, ventilación, aire acondicionado, control de agua y dirección de energía, medición, hasta aplicaciones para el hogar, audio, etc. *KNX* mejora el confort y la seguridad con sus instalaciones así como contribuye al ahorro energético y la protección del clima.
  
- *Adaptación a cualquier tipo de construcción*: *KNX* puede ser usado tanto en nuevas construcciones como en las ya existentes. Las instalaciones *KNX* pueden ser fácilmente extendidas y adaptadas a las nuevas necesidades, con una pequeña inversión de tiempo y dinero. Igualmente *KNX* puede ser instalado tanto en pequeñas casas como en grandes edificios.
  
- *Diferentes modos de configuración*: El estándar *KNX* permite a cada fabricante seleccionar su modo de configuración idóneo de los ya mencionados que le permita elegir su combinación ideal según el segmento del mercado que desee orientarse.

- *Diferentes medios de comunicación:* Cada medio de comunicación puede ser usado en combinación con uno o más modos de configuración, lo que permite a cada fabricante elegir la combinación perfecta para su segmento de mercado y aplicaciones.
- *Posibilidad de ser acoplados a otros sistemas:* Distintos fabricantes ofrecen pasarelas a otros sistemas, es decir ofrecen enlaces a otros sistemas de automatización de edificios, redes de telefonía, redes multimedia, redes IP, etc.

## 2.2. Desarrollo dirigido por modelos (MDE).

El Desarrollo de Software Dirigido por Modelos (o *MDE* del inglés *Model Driven Engineering*) y más concretamente la propuesta *MDA (Model Driven Architecture)* de *OMG (Object Management Group)* constituyen una aproximación para el desarrollo de sistemas software, basada en la separación entre la especificación de la funcionalidad esencial del sistema y la implementación de dicha funcionalidad usando plataformas de implementación específicas.

La iniciativa *MDA* cubre un amplio espectro de áreas de investigación (metamodelos basados en *MOF –Meta Object Facility-*, perfiles *UML –Unified Modeling Language-*, transformaciones de modelos, definición de lenguajes de transformación –*QVT, Query View and Transformation-*, construcción de modelos *PIM –Platform Independent Model-* y *PSM –Platform Specific Model-* y transformaciones entre ellos, construcción de herramientas de soporte, aplicación en métodos de desarrollo y en dominios específicos, etc.). Algunos de estos aspectos están bien fundamentados y se están empezando a aplicar con éxito, otros sin embargo están todavía en proceso de definición. En este contexto son necesarios esfuerzos que conviertan *MDA* y sus conceptos y técnicas relacionados en una aproximación coherente, basada en estándares abiertos, y soportada por técnicas y herramientas maduras.

A lo largo de este apartado se aclararán algunos conceptos sobre el desarrollo dirigido por modelos, aparte de ahondar sobre la propuesta *MDA* de *OMG* que sirve de referencia para la realización de la investigación que engloba este *TFM*. Se puede ampliar información en el *Capítulo 3* de [Jiménez 09], o en [Schmidt 06], [OMG 10] y [Poole 01].

### 2.2.1. Conceptos básicos sobre MDE.

Como ya se ha comentado, el *Desarrollo de Software Dirigido por Modelos* o *MDE* es una metodología de desarrollo software que se centra en crear modelos, o abstracciones, más cercanas a un concepto particular del dominio que a conceptos computacionales o algorítmicos. Esto se traducirá en un incremento de productividad derivado de maximizar la compatibilidad entre sistemas y de la simplificación del proceso de diseño.

Una de las premisas básicas de *MDE* es la obtención (semi)automática de código a través de las transformaciones de modelos. Así, la cantidad de código que debe escribirse manualmente para una aplicación se reduce drásticamente, y el desarrollador debe centrarse únicamente en el código personalizado, que no suele superar el 10% del total (en las



aplicaciones, en torno al 50% del código es genérico y el 40% semi-genérico). Esto repercute directamente en un aumento de la productividad.

La fiabilidad del código es otro de los atributos que mejora con el empleo de *MDE*. El código generado automáticamente requiere menos depuración y verificación, y además, los modelos no se utilizan sólo para la generación de código, sino que pueden producir como salida herramientas de comprobación utilizando restricciones escritas en lenguajes de definición de restricciones (*OCL*, *Object Constraint Language*). También se produce una mejora en la consistencia y la calidad, ya que las transformaciones de modelos a código se basan en patrones de diseño que permiten generar un código estructurado conforme a una arquitectura.

Por otra parte, la reusabilidad es una de las promesas de esta metodología. Los patrones de diseño pueden evolucionar en los niveles del modelo para definir cómo estructurar las clases del dominio y los componentes para las tareas estándar. Los modelos del dominio se pueden almacenar en librerías para ser reutilizados en nuevas aplicaciones. Además, el modelo de negocio se mantiene separado en todo momento de la arquitectura de la tecnología utilizada en la implementación, favoreciendo la reutilización.

#### *2.2.1.1. Conceptos de modelo y metamodelo.*

Modelar, en el sentido más amplio de la palabra, es el uso (con el fin de economizar), de algo en lugar del original para algún propósito cognitivo. Permite utilizar algo más simple, seguro y barato que la realidad para un propósito dado. Un modelo representa la realidad para dicho propósito; el modelo es una abstracción de la realidad, ya que no representa todos sus aspectos. Esto permite tratar con el mundo de una forma simplificada, evitando la complejidad, peligro e irreversibilidad de la realidad.

Así, un **modelo** se refiere siempre a un sistema (su referente). El referente debe estar bien definido y ser verificable de manera objetiva (aunque no tiene por qué existir, se puede modelar realidad virtual).

En el caso del software, éste es un modelo complejo y compuesto que puede representar diferentes aspectos de un sistema con distintos niveles de abstracción. En función del nivel de abstracción, el software puede ser una representación de los requisitos de la aplicación, la plataforma de ejecución o la lógica de negocio. Además, estará escrito en un lenguaje de programación (conforme a un lenguaje).

El concepto de **metamodelo** es uno de los pilares sobre los que descansa la metodología dirigida por modelos, y por tanto, uno de los fundamentos básicos para la actividad del modelado. Un metamodelo recoge todos los conceptos de un dominio de sistemas y las relaciones entre ellos que son importantes para el usuario. De este modo se garantiza la conformidad entre un modelo y su metamodelo.

A su vez, el metamodelo no tiene por qué ser único. Para un dominio pueden existir varios metamodelos que permitan describirlo desde varios puntos de vista y con diferentes niveles de detalle. Conviene puntualizar, que la creación de modelos a partir de modelos es muy habitual y no debe confundirse con la relación de conformidad (modelo conforme a metamodelo).

Centrándonos en esta línea de investigación, en **MDE** los modelos se pueden definir de forma estructural como multigrafos dirigidos. Un multigrafo dirigido  $G = (N_G, E_G, \Gamma_G)$  consiste en un conjunto finito de nodos  $N_G$ , un conjunto finito de aristas  $E_G$  y una función de asignación de aristas a pares de nodos  $\Gamma_G: E_G \times N_G \rightarrow N_G$ . Así, un modelo  $M = (G, \omega, \mu)$

- $G = (N_G, E_G, \Gamma_G)$  es un multigrafo dirigido.
- $\omega$  es un modelo, llamado modelo de referencia (metamodelo) de  $M$ , asociado al grafo  $G_\omega = (N_\omega, E_\omega, \Gamma_\omega)$ .
- $\mu: N_G \cup E_G \rightarrow N_\omega$  es una función que asocia elementos (nodos y aristas) de  $G$  a nodos de  $G_\omega$  (metaelementos).

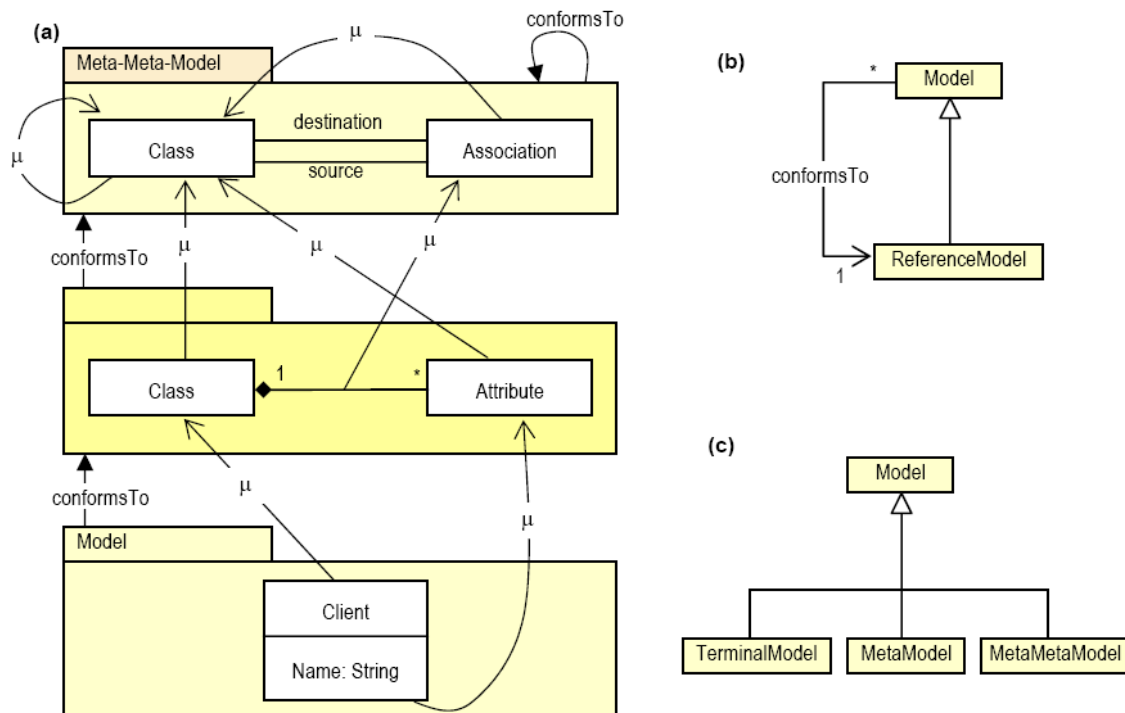


Figura 2.10. Ejemplo ilustrativo de la relación entre modelo y metamodelo en MDE [Jiménez 09]

La Figura 2.10. [Jimenez 09] ilustra a la perfección las relaciones entre modelos y metamodelos en MDE. Así, en (a) se puede ver que *Client* es un modelo conforme al metamodelo donde se define el concepto de clase y atributo (*Client* tiene el atributo nombre de tipo *String*, aunque según el metamodelo podría tener muchos más). A su vez, el metamodelo es un multigrafo dirigido conforme a su meta-metamodelo, que especifica un lenguaje gráfico para la descripción de metamodelos, según el cual, los metamodelos se componen de clases y asociaciones.

Según la relación de conformidad, todo modelo tiene un modelo de referencia (metamodelo) conforme al cual se ha especificado, pero el metamodelo también debe tener su propio meta-meta-modelo (que contiene los elementos básicos para describir metamodelos) y al que tiene que ser conforme. Además, el concepto de metamodelo no es absoluto, sino que depende del nivel de abstracción utilizado. Así, lo que en un determinado nivel es un metamodelo puede convertirse en un modelo para el nivel superior.

En la *Figura 2.10.* (b) y (c) se muestran estas relaciones: un modelo es siempre conforme a su modelo de referencia, pero además, ese modelo de referencia es a su vez un tipo de modelo. Según esto, se pueden distinguir tres tipos de modelos: terminales (su referencia es un metamodelo), metamodelos (su referencia es un metametamodelo) y metametamodelos (se suelen referenciar a sí mismos, es decir, son conforme a sí mismos).

### 2.2.1.2. Transformaciones de modelos.

En *MDE* las transformaciones de modelos son esenciales para dar soporte a todo el proceso de desarrollo. Las transformaciones permiten realizar conversiones de un modelo origen a otro destino entre diferentes metamodelos, manteniéndolos sincronizados. En el ámbito del desarrollo software esto permite, en una última instancia, convertir los modelos que utilizan los desarrolladores en modelos de la plataforma de implementación (código ejecutable).

Una transformación es la generación automática de un modelo a partir de otro modelo fuente, de acuerdo con una serie de reglas. Una definición de transformación es un conjunto de reglas de transformación que juntas describen cómo se transforma un modelo descrito en el lenguaje origen a un modelo descrito en el lenguaje destino. Una regla de transformación es una descripción de cómo se transforman una o más construcciones del lenguaje origen en una o más construcciones en el lenguaje destino.

Siguiendo la búsqueda de unificación en *MDE* (todo es un modelo), las transformaciones también se consideran modelos. Por lo tanto, el modelo de la transformación deberá también ser conforme a un metamodelo. En la *Figura 2.11* [Jiménez 09] se esquematiza la transformación de modelos y las relaciones a diferentes niveles de abstracción.

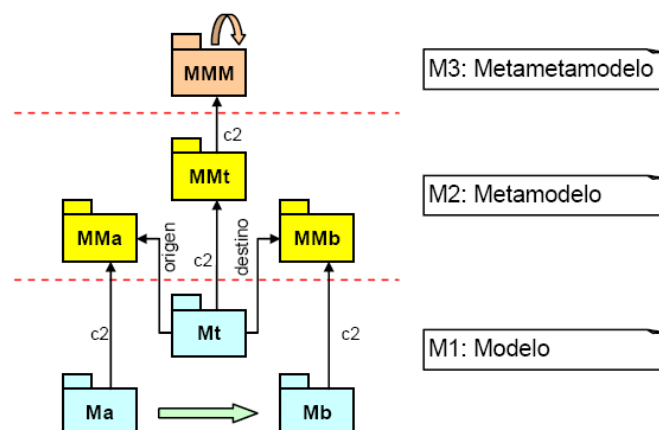


Figura 2.11. Ejemplo ilustrativo de la relación entre modelo, metamodelo y transformación en MDE [Jiménez 09]

Entonces, se puede concluir que:

- Una transformación genera un modelo destino  $M_b$  a partir de un modelo origen  $M_a$ . La transformación es en sí un modelo  $M_t$ .
- Puesto que los modelos pueden tener una representación visual, la consideración de las transformaciones como modelos proporciona la posibilidad de expresarlas de forma gráfica.
- Como todo modelo es conforme a un metamodelo, los modelos fuente y destino serán conformes a los metamodelos  $MM_a$  y  $MM_b$ . De forma análoga, la transformación  $M_t: M_a \rightarrow M_b$  es también conforme a un metamodelo  $MM_t$  que define el lenguaje común de transformación de modelos.
- Esto abre la posibilidad de realizar transformaciones de más alto nivel, es decir, transformaciones que tomen otras transformaciones como entrada y generen transformaciones como salidas.
- Las transformaciones no tienen por qué ser uno a uno, ya que es posible generar varios modelos a partir de un modelo único, o bien unificar modelos, es decir, generar un único modelo a partir de varios modelos de entrada (por ejemplo fusión de varias vistas de un mapa en uno sólo).

La clasificación de transformaciones de modelos más interesante que se puede realizar en el proceso de desarrollo MDE distingue dos tipos en base a la fase en que se realiza y al tipo de artefacto generado:

- **Modelo a modelo (*Model-To-Model (M2M)*)**. Son las que tienen como salida otro modelo. Este tipo de transformaciones es el empleado para hacer evolucionar los modelos en el proceso de desarrollo. Se diferencian cuatro formas de realizar la transformación: manipulación directa de la representación de bajo nivel del modelo (posible con cualquier lenguaje), mediante la teoría de transformaciones de grafos, utilizando lenguajes declarativos, relaciones matemáticas o reglas de conversión y aproximaciones híbridas que combinan cualquiera de las anteriores.
- **Modelo a texto (*Model-To-Text (M2T)*)**. Es un caso particular de las anteriores para generar una representación textual del modelo origen sin metamodelo de destino. Se suelen utilizar para generar código en las etapas finales de desarrollo, aunque también se utilizan para otros fines, como generar documentación. Se distingue dos métodos para realizar estas transformaciones: utilizando el patrón *Visitor* para recorrer el modelo y generar texto, o utilizando soluciones basadas en plantillas, que mezclan el texto de la salida con comandos para recorrer el modelo. Este *TFM* busca describir la metodología e implementar las soluciones necesarias para realizar una transformación *M2T* basada en plantillas.

### 2.2.2. Model Driven Architecture (MDA).

La arquitectura dirigida por modelos (*Model-Driven Architecture* o *MDA*) es un acercamiento al diseño de software, propuesto y patrocinado por el *Object Management Group* (*OMG*, un consorcio de empresas como *IBM*, *Borland*, *Hewlett-Packard* o *Boeing* entre otras). *MDA* se ha concebido para dar soporte a la ingeniería dirigida a modelos de los sistemas de software. *MDA* es una arquitectura que proporciona un conjunto de guías para estructurar especificaciones expresadas como modelos. De esta manera, *MDA* no pretende reemplazar paradigmas previos, lenguajes o herramientas, sino que pretende armonizarlos e integrarlos en base al enfoque dirigido por modelos. El logo de *MDA* puede verse en la *Figura 2.12*. [OMG 10].

En una sesión plenaria celebrada en Montreal entre los días 23 y 26 de Agosto de 2004, se acordó la siguiente definición de *MDA* (documento de *OMG ormsc/04-08-02*):

"*MDA is an OMG initiative that proposes to define a set of non-proprietary standards that will specify interoperable technologies with which to realize model-driven development with automated transformations. Not all of these technologies will directly concern the transformations involved in MDA. MDA does not necessarily rely on the UML, but, as a specialized kind of MDD (Model Driven Development), MDA necessarily involves the use of model(s) in development, which entails that at least one modeling language must be used. Any modeling language used in MDA must be described in terms of the MOF language, to enable the metadata to be understood in a standard manner, which is a precondition for any ability to perform automated transformations.*"

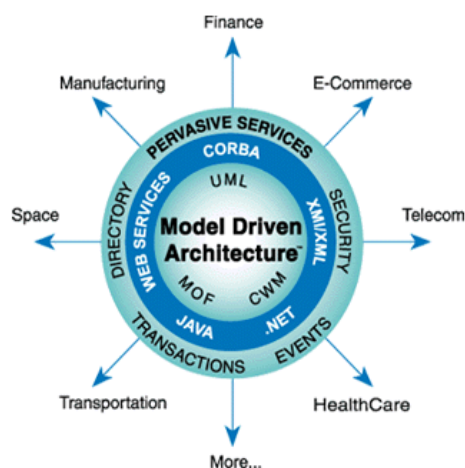


Figura 2.12. Logotipo/resumen de *MDA* [OMG 10]

Así, la organización del desarrollo software en capas que especifica *MDA* es consecuencia (casi) directa de esta definición (*Figura 2.13*. [Jiménez 09]). De esta manera, se puede puntualizar que:

- La capa superior *M3* contiene el metametamodelo reflexivo denominado *Meta-Object Facility* (*MOF*). *MOF* es común a todos los estándares *OMG*.

- En la capa *M2* se encuentran los metamodelos (como *UML* o *CWM*) que son utilizados por el usuario para crear los modelos que describen sistemas reales.
- La capa *M1* alberga los modelos que representan sistemas reales.

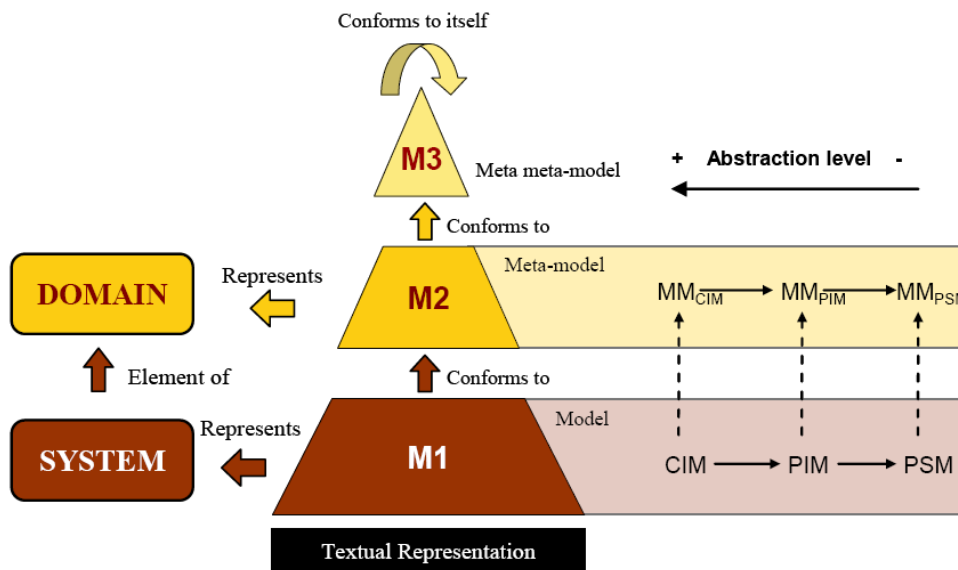


Figura 2.13. Organización del desarrollo software en capas especificado por MDA [Jiménez 09]

Otro aspecto muy importante que se aprecia en la *Figura 2.13*, es la organización del desarrollo software de manera horizontal en tres familias de modelos en función del nivel de abstracción: *CIM*, *PIM* y *PSM*. Este es un hecho diferenciador de *MDA* con respecto a *MDE*.

El **Computer Independent Model** o *Modelo Independiente de la Computación (CIM)* se centra en el dominio del sistema así como en sus requisitos, detalles de la estructura y procesamiento. El *CIM* se centra en los requerimientos y representa el nivel más alto del modelo de negocios. Además, el *CIM* trasciende los sistemas. Un objetivo fundamental del *CIM* es que cualquiera que pueda comprender el negocio y sus procesos pueda comprender un *CIM*, ya que éste evita todo tipo de conocimiento especializado de la tecnología *hardware* o *software* en la que se vaya a implantar. Los *CIMs* se asocian a los modelos de dominio y es recomendable que sean especificados a partir de un vocabulario común para todos los elementos implicados en el desarrollo del sistema.

El **Platform-Independent Model** o *PIM*, que se traduce al castellano como *Modelo Independiente de la Plataforma*, representa el modelo de procesos de negocio que va a ser implementado. Muestra la especificación del sistema teniendo en cuenta no sólo las especificaciones de funcionamiento propias – descritas en el *CIM* – sino también las especificaciones para la implementación en un medio informático. Comúnmente se usa *UML* o un derivado de él para describir el *PIM*.

El *PIM* modela los procesos y la estructuras del sistema, sin hacer ninguna referencia a la plataforma en la (o las) que será desplegada la aplicación. Ignora los sistemas operativos, los lenguajes de programación, el hardware y/o la topología de red. El *PIM* representa los aspectos que no cambiarán de una plataforma a otra, de acuerdo a una tecnología o método

de implantación escogido, pero no muestra aspectos de implementación. Suele ser el punto de entrada de todas las herramientas para *MDA* e, incluso, de muchos artículos que hablan de *MDA* y dejan de lado *CIM*.

Los ***Platform Specific Model*** o *Modelo Dependiente de la Plataforma (PSM)* tienen que lidiar explícitamente con los sistemas operativos, los lenguajes de programación, las plataformas (*CORBA, .NET, Java EE*), etc.

La transformación *PIM* a *PSM* no tiene por qué ser uno a uno, de hecho, una de las principales aportaciones de este enfoque es la posibilidad de realizar la implementación en distintas plataformas partiendo del mismo nivel *PIM* intermedio. Estas transformaciones se realizan mediante modelos de marcas para incluir la información adicional necesaria para poder realizar la transformación. Cada nivel del enfoque *MDA* (*CIM, PIM, PSM*) involucra la definición de modelos conformes a sus respectivos metamodelos. A su vez, cada metamodelo estará definido conforme a un metametamodelo. Dado un metamodelo podremos definir un conjunto de modelos posibles a partir de las facilidades que nos ofrezcan los lenguajes definidos a tal efecto. Este esquema se repite en todos los niveles de *MDA* tal y como se detalla en la *Figura 2.14*. [Jiménez 09]. En cada nivel los modelos son conformes a sus respectivos metamodelos, y a su vez, dichos modelos pertenecen al conjunto de todos los posibles modelos conformes a dicho metamodelo. Los metamodelos representan el conjunto de todos los metamodelos que se pueden definir conforme a ellos. Las transformaciones necesarias entre los modelos definidos en el nivel *CIM* hacia modelos destino en los niveles inferiores (*PIM, PSM*) vendrán dadas a partir de las respectivas correspondencias entre los conceptos que se manejan a nivel de metamodelo.

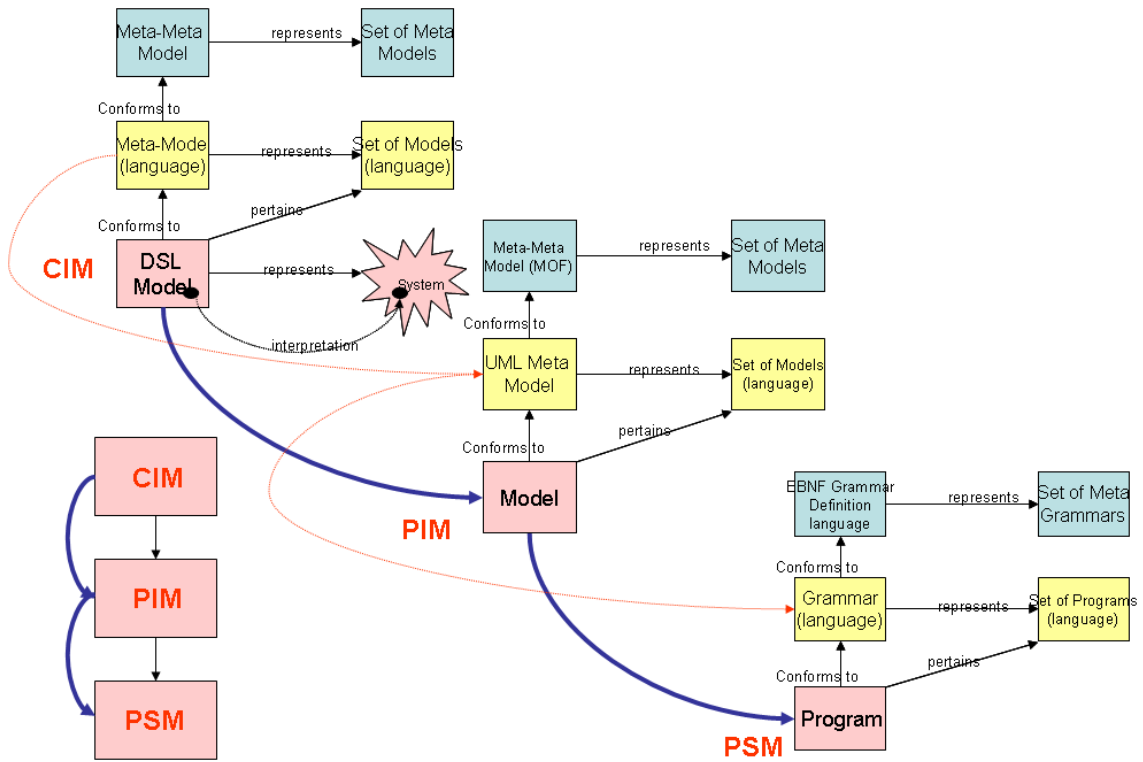


Figura 2.14. Niveles de modelado en las familias CIM, PIM y PSM. [Jiménez 09]

Un ejemplo ilustrativo de esta organización horizontal del desarrollo software aplicada a un modelo de negocios cualquiera puede verse en la Figura 2.15. En ella, se aprecian los conceptos básicos de cada etapa y la interrelación entre ellas.

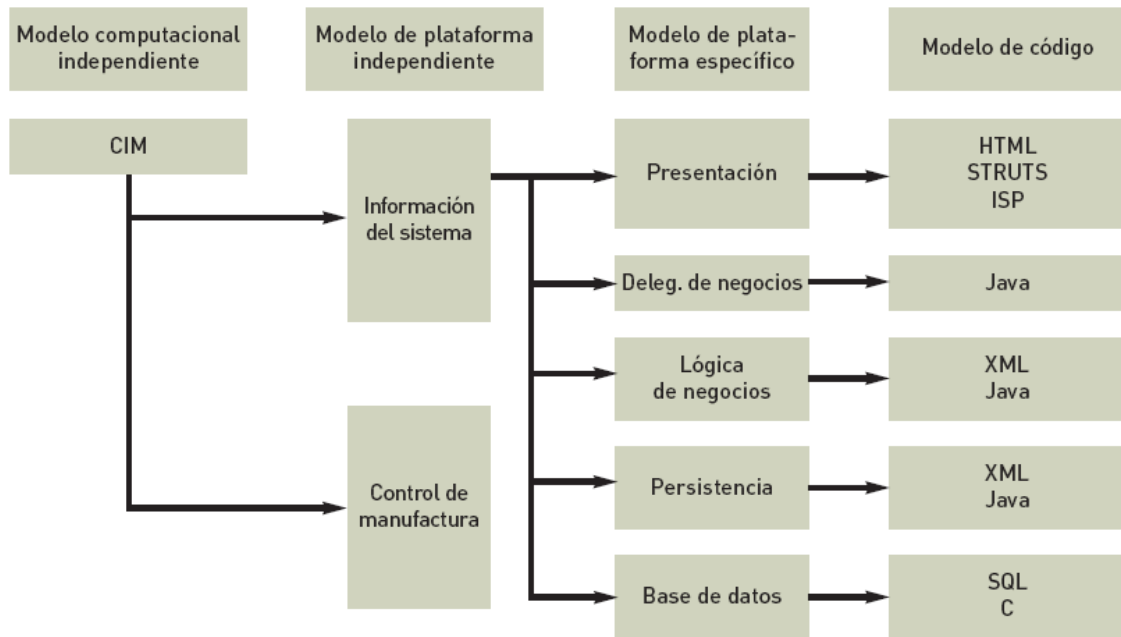


Figura 2.15. Etapas de diseño de MDA (CIM, PIM y PSM) aplicadas a un modelo de negocios cualquiera



La ventaja principal de *MDA* radica en una clara y estricta separación de responsabilidades. Por un lado, modelaremos los *PIMs*, que representan los modelos de nuestro negocio, y por el otro, los *PSMs* con las preocupaciones tecnológicas. Esto permitirá que ambos modelos evolucionen por separado. De esta manera, si quisiéramos, por ejemplo, modificar un aspecto técnico, bastaría con modificar el *PSM* sin que éstos tuvieran impacto en la lógica de negocios.

*MDA* también permite lidiar con la complejidad del negocio, modelando a éste por separado, y permitiendo su análisis y mejora; disminuir costos, si se cuenta con una herramienta *MDA* adecuada a nuestras necesidades; y mejorar la calidad de nuestros modelos y procesos, mediante su análisis y la separación de responsabilidades.

Las tecnologías más importantes involucradas para poder llevar a la práctica los conceptos subyacentes en *MDA* son:

- *Meta Object Facility (MOF)*
- *Unified Modeling Language (UML)*
- *XML Model Interchange (XMI)*
- *Common Warehouse Meta-model (CWM)*
- *Software Process Engineering Meta-model (SPEM)*
- *Action Semantics Language (ASL)*
- *Query-View-Transformation (QVT)*
- *UML profiles.*

Para obtener información sobre estas tecnologías se recomienda la lectura de [Jiménez 09, páginas de 78 a 80].

### 2.2.3. Herramientas que dan soporte a *MDE*.

Ninguna de las promesas y realidades de *MDE* (y en concreto *MDA*) sería hoy posible sin herramientas que den soporte a este nuevo enfoque. La *OMG* proporciona severas especificaciones sobre la implementación de estas herramientas, a menudo como respuestas a las *RFPs (Request for Proposals)*. Estas herramientas de desarrollo de software que permiten realizar tareas de modelado, transformaciones de modelos, verificación de modelos, *DSLs* (Lenguajes Específicos de Dominio), generación de código, etc., por lo que el auge de *MDA* ha impulsado en los últimos años el desarrollo de tecnología (herramientas, lenguajes, etc.) tanto de carácter comercial como de propósito general que implementan total o parcialmente los conceptos de *MDA* y *MDE*.

Las implementaciones de las especificaciones *OMG* son realizadas tanto por compañías privadas como por grupos *open source*. Una implementación importante es la de la *Fundación Eclipse* [Eclipse 10a]. Muchas de las implementaciones de los estándares de modelado de la *OMG* pueden encontrarse en el *Eclipse Modeling Framework (EMF)* [EMF 10a] o en el *Graphical Modeling Framework (GMF)* [GMF 10a] de esta fundación. Dado que el entorno

*Eclipse* es la principal herramienta utilizada para el desarrollo de este trabajo, en el siguiente capítulo se realizará un análisis más detenido de la misma.

Además de *EMF* y *GMF*, otras herramientas a considerar son *NetBeans MetaData Repository*, *OptimalJ*, *ArcStyler* y *AndroMDA*. *NetBeans MetaData Repository* da soporte tecnológico al estándar *MOF* constituyendo un repositorio de metainformación que puede ser modelada en *UML*. *OptimalJ* y *ArcStyler* son comerciales y *AndroMDA*, al igual que *ArcStyler* está basada en el concepto de cartucho como artefacto que engloba el lenguaje de modelado y las transformaciones para una determinada plataforma, existiendo en la actualidad cartuchos para la plataforma Java y para servicios web. En cambio, *Optimal-J* está dirigida a una arquitectura de tres capas con *Struts* (herramientas que dan soporte al desarrollo de aplicaciones web) y *EJB*.

### 2.3. Desarrollo de sistemas domóticos dirigido por modelos.

La aplicación de *MDE* al diseño de sistemas reactivos tiene una joven historia [Alonso 08] [Vicente 07] [Losilla 07a] [Losilla 07b] [Ortiz 07]. En concreto, su aplicación a sistemas domóticos ha sido estudiada en la Tesis que sirve de base para el desarrollo de este *TFM* [Jiménez 09]. Otra propuesta, aunque infinitamente más limitada puede ser la de Markus Voelter [Voelter 07]. Mejor propuesta que esta última es la de Muñoz [Muñoz 07], aunque también muy limitada comparada con [Jiménez 09].

Un sistema reactivo es por definición un sistema heterogéneo cuyo rol fundamental es mantener una interacción continua con su entorno a través de dispositivos sensores y actuadores. Como se puede ver, los sistemas domóticos encajan perfectamente en esta definición. Para el desarrollo de estos sistemas se sigue actualmente una metodología de diseño tradicional, basada en construir aplicaciones a medida para solucionar cada problema en concreto; de manera que rara vez es posible la reutilización de software.

Mediante la aplicación del enfoque dirigido por modelos se busca cambiar este hecho. En la *Tabla 2.2.* [Jiménez 09] se pueden observar las ventajas que esta aplicación traería consigo.

Problema Sistemas Reactivos	Implicaciones en el desarrollo	Aportaciones de MDA
Soluciones a medida de la plataforma utilizada en cada momento.	Aplicaciones poco flexibles y reutilizables.	Incremento de la flexibilidad y reutilización. Soluciones genéricas e independientes de la plataforma (niveles CIM y PIM).
Se requiere un alto grado de experiencia en la infraestructura desde las fases iniciales del diseño.	El proceso de diseño es lento y poco eficiente.	Aumento de la eficiencia en el proceso de diseño. En las fases iniciales del diseño se utilizan conceptos del dominio (lenguajes específicos de dominio), que son independientes de la plataforma.
Algunas de las herramientas utilizadas son muy dependientes de la infraestructura y su uso requiere amplios conocimientos de programación muy próximos a la plataforma de implementación.	El conocimiento en profundidad de una infraestructura es poco reutilizable en otras soluciones.	Herramientas genéricas. Soluciones reutilizables en distintas plataformas. La generación de código se realiza de forma (semi) automática en las fases finales del diseño (PSM).

*Tabla 2.2. Problemas relativos al desarrollo de sistemas reactivos. Aportaciones de MDA.* [Jiménez 09]

En los siguientes sub-apartados se resumirá brevemente la propuesta de metodología de diseño realizada por Jiménez, y se expondrán los cambios de enfoque realizados para adaptar su propuesta al caso particular de los sistemas *KNX/EIB*. Se hará especial hincapié en el *DSL* y el metamodelo propuestos por Jiménez, para posteriormente explicar el *DSL* utilizado

para el caso de sistemas *KNX/EIB* así como el metamodelo nivel *PIM* que le proporciona soporte.

### 2.3.1. DSL para domótica propuesto por Jiménez [Jiménez 09].

De manera previa a la propuesta realizada por Jiménez [Jiménez 09], era un hecho la práctica inexistencia de lenguajes específicos en el campo de la domótica que permitieran una captura de requisitos con cierto nivel de abstracción e independencia de la plataforma.

Efectivamente, existen varias herramientas comerciales que permiten describir sistemas domóticas (como por ejemplo *ETS* o *LonMaker*). Sin embargo, estas herramientas son fuertemente dependientes de la plataforma y por lo general requieren una formación muy especializada y una cierta experiencia tal y como ya se ha mencionado.

Antes de [Jiménez 09], no existía un *DSL* para domótica basado en *MDA*; de manera que el poco nivel de abstracción proporcionado por las herramientas comerciales era un problema aún sin solución. Para llenar todo este vacío, Jiménez propuso su "*DSL para Domótica*".

#### 2.3.1.1. Conceptos del *DSL*.

Este *DSL* se fundamenta básicamente en la existencia de componentes domóticos reutilizables con la posibilidad de modelar nuevos componentes. Los conceptos fundamentales definidos en el *DSL* son:

- **Unidades funcionales:** Son el elemento básico que puede formar parte de un dispositivo. En general, un dispositivo de una tecnología va a integrar varias unidades funcionales. Existirá un catálogo de unidades funcionales reutilizables que dará soporte al *DSL*. Se subdividen a su vez en Unidades funcionales pasivas y Unidades funcionales activas.
  - **Unidades funcionales pasivas:** Su actividad se reduce a su presencia física. Ofrecen servicios pero no requieren implementación software. Pueden ser de *entrada* (sensores, pulsadores, interruptores, etc.) o de *salida* (bombillas, motores, electroválvulas, etc.).
  - **Controladores:** Requieren programación para implementar su funcionamiento. Podrán ser de *entrada* (permiten integrar dispositivos pasivos de entrada en el dominio domótico; ofrecen servicios a dispositivos pasivos de entrada), de *salida* (permiten integrar dispositivos pasivos de salida en el dominio domótico; ofrecen servicios a dispositivos pasivos de salida), de *entrada/salida* (puertas lógicas, temporizadores y filtros) o *pasarelas* (permiten la comunicación con redes externas a la red

domótica; por ejemplo con redes de infrarrojos, redes RS232 o redes de telefonía).

- **Servicios:** Cada unidad funcional dispondrá de uno o varios servicios que serán utilizados para interactuar con otras. Un servicio podrá ser requerido o implementado. Existirá un catálogo de servicios reutilizables que dará soporte al DSL.
- **Parámetros:** Las unidades funcionales disponen de parámetros para su configuración.
- **Enlaces o asociaciones:** Mediante estas asociaciones se puede describir cualquier acción-reacción de la instalación domótica basada en la generación de eventos en el entorno. Podrá haber asociaciones *hardware* (cableadas) o asociaciones *software* (de servicios).
- **Escenas:** Las escenas son un conjunto de eventos encadenados que se generan como consecuencia de una única orden.

A modo ilustrativo, se muestran tanto el catálogo de unidades funcionales como el sub-catálogo de controladores en la *Figura 2.16.* y la *Figura 2.17* [Jiménez 09].

### 2.3.1.2. Sintaxis abstracta: Metamodelo de soporte al DSL.

Todo lenguaje específico de dominio debe tener una sintaxis abstracta que defina de manera formal los conceptos del lenguaje y las relaciones entre ellos. Esta sintaxis abstracta respalda a los conceptos definidos en el epígrafe anterior: establece una definición para estos conceptos y marca las relaciones entre ellos, incluyendo además reglas para determinar cuando un modelo está bien formado.

El metamodelo definido por Jiménez para domótica se puede apreciar en la *Figura 2.18.* [Jiménez 09]. Así, se divide en dos partes claramente diferenciadas: una parte para el catálogo de unidades funcionales y otra parte para la aplicación domótica. Esto dota al metamodelo de gran modularidad, de tal manera que se permite su utilización por parte de diseñadores de dispositivos así como de diseñadores de aplicaciones.

Este metamodelo se basa en cinco tipos de elementos principales: unidades funcionales (que disponen de dos elementos para ser modeladas: *FUnitDefinition* y *FUnitInstance*), catálogo de unidades funcionales (*FUnitCatalog*), catálogo de servicios (*ServiceCatalog*), enlaces (*FULink*) y las escenas (*Scene*). No es cometido de este documento explicar detenidamente estos elementos, por lo que para una mejor comprensión de los mismos por parte del lector se recomienda la lectura del *Capítulo 4* de [Jiménez 09].

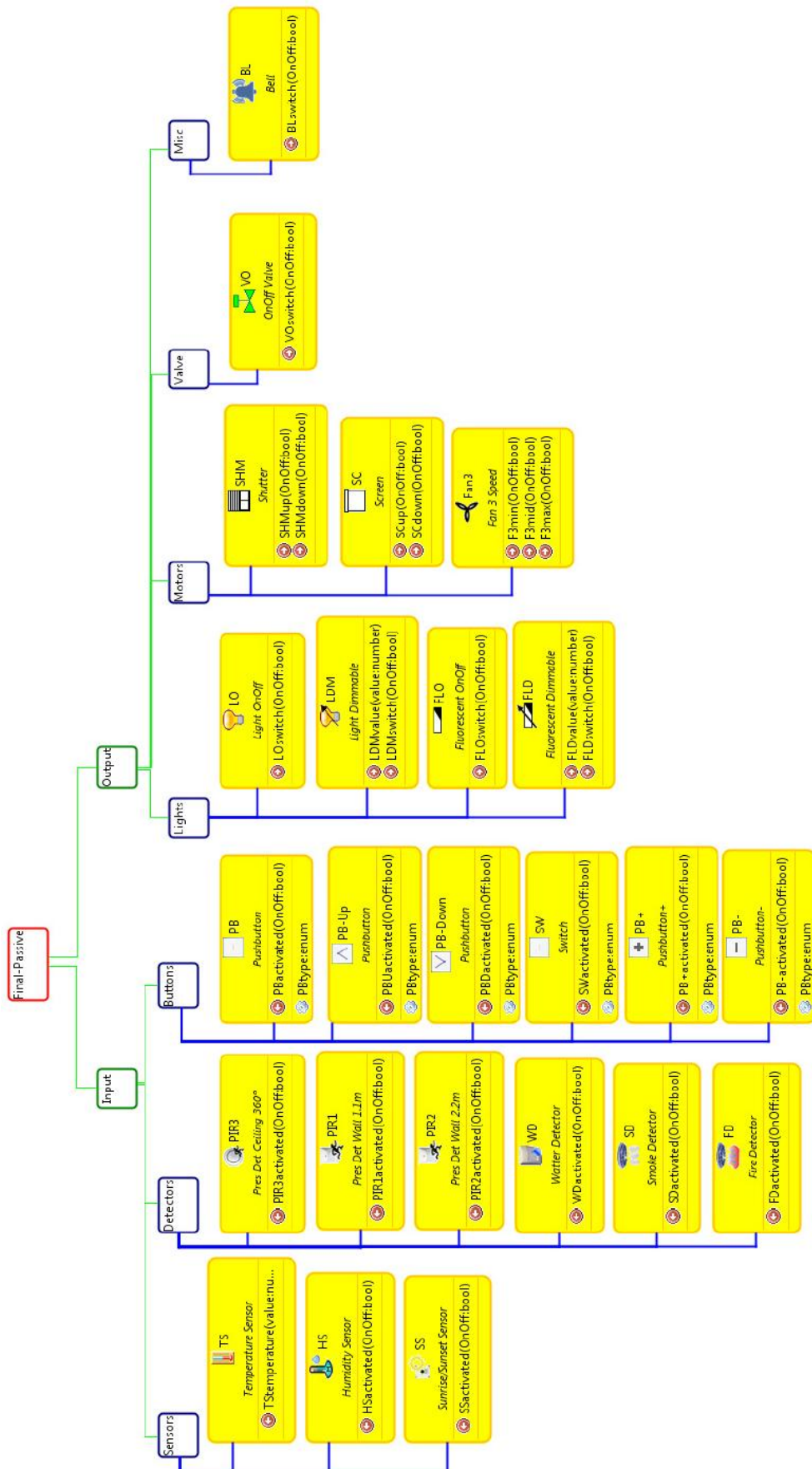


Figura 2.16. Proyección del catálogo de Unidades Funcionales: elementos pasivos. [Jiménez 09]

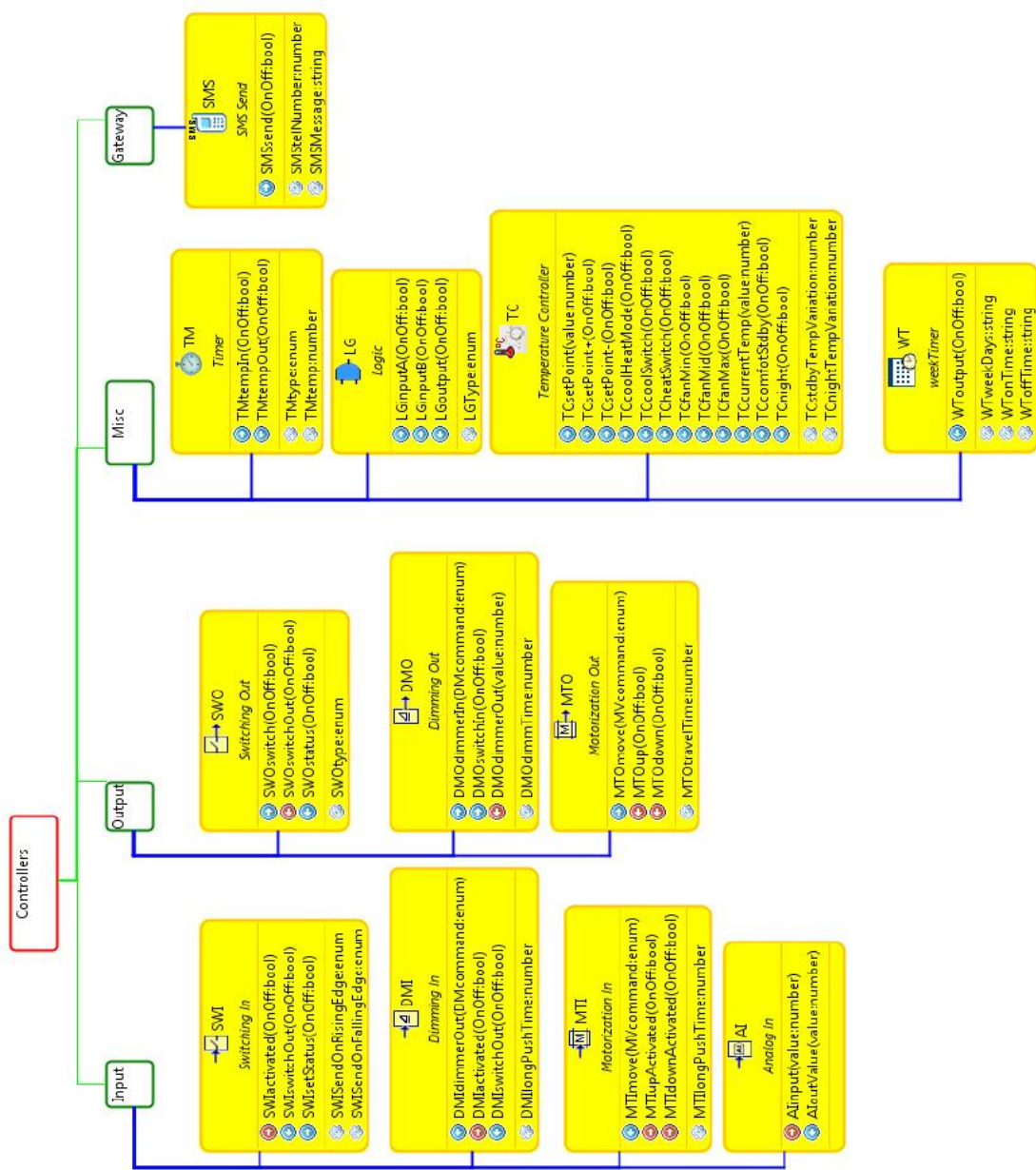


Figura 2.17. Proyección del catálogo de Unidades Funcionales: controladores. [Jiménez 09]

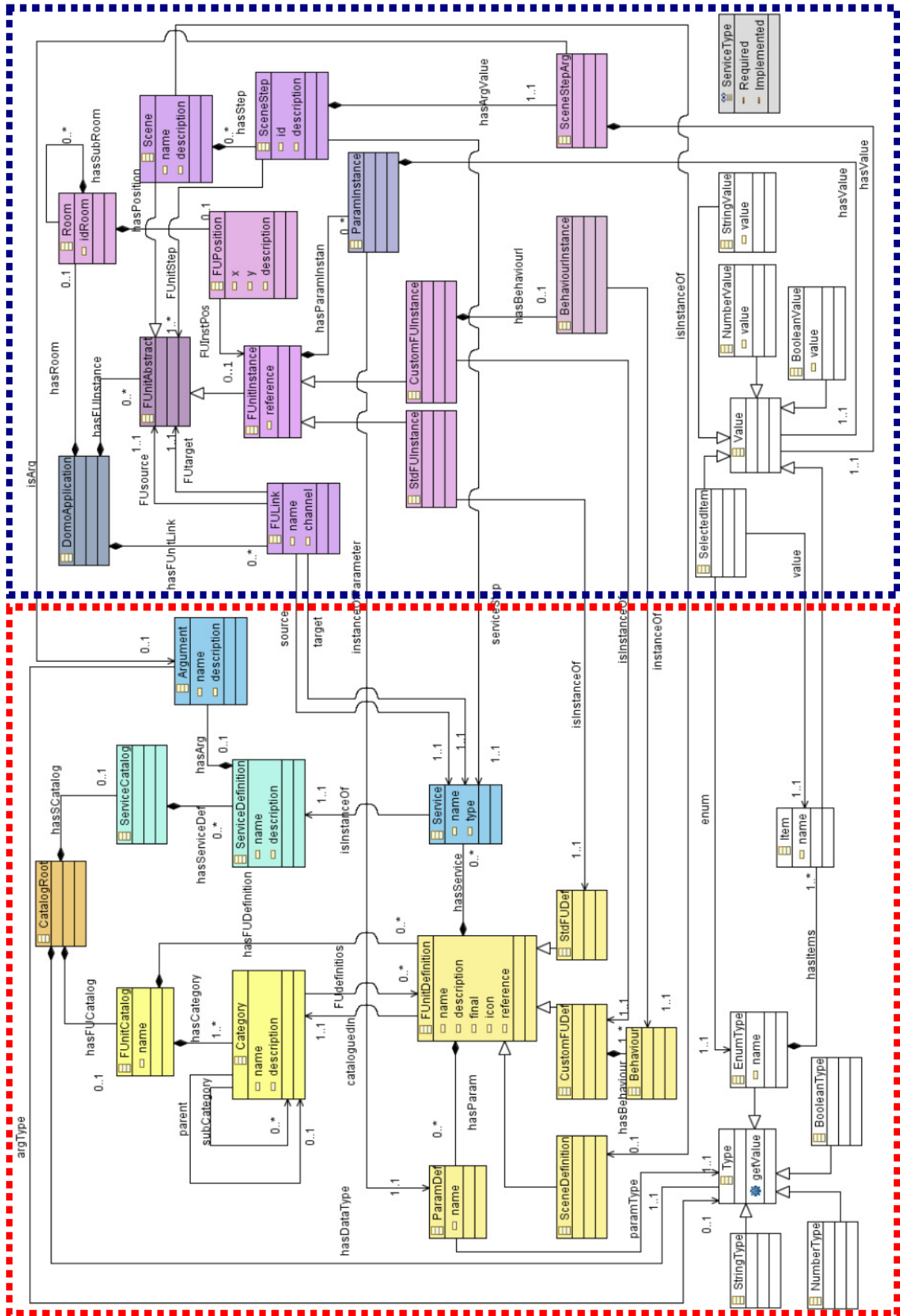


Figura 2.18. Metamodelo para el DSL propuesto por Jiménez. [Jiménez 09]



### 2.3.1.3. Sintaxis concreta: Lenguaje gráfico.

Jiménez también define en un lenguaje gráfico para capturar la descripción del sistema por parte del usuario, a fin de mantener dicha captura lo más intuitiva posible. Este lenguaje gráfico así como el metamodelo fueron implementados por Jiménez haciendo uso de los *plugins* de Eclipse GEF y EMF (respectivamente).

De igual forma que en el epígrafe anterior, tampoco es cometido de este análisis explicar los pormenores de este lenguaje gráfico. A modo ilustrativo se incluye una tabla (Tabla 2.3. [Jiménez 09]) donde se puede ver la sintaxis concreta para determinados conceptos del DSL. Para profundizar en esta sintaxis concreta, se remite al lector al mismo capítulo que en el epígrafe anterior.


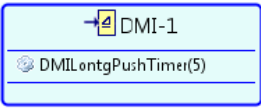

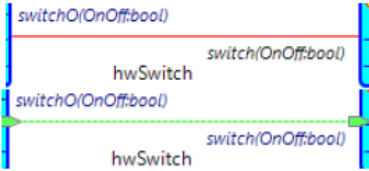



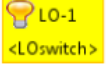

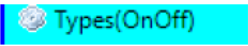
Icono	Representación Gráfica	Descripción
		<b>Instancia de unidad funcional:</b> muestra el icono de la unidad funcional que se instancia junto con los parámetros y sus valores.
		<b>Enlace entre unidades funcionales:</b> dependiendo de si se trata de un canal o un enlace, se representa como una línea roja que conecta dos unidades funcionales (una de ellas pasiva-final) o con una línea discontinua verde con terminaciones. En dicho enlace se muestran tres etiquetas, una para cada servicio que participa en el enlace y una tercera etiqueta central que muestra la definición del servicio que actúa en el enlace.
		<b>Escena:</b> contiene los pasos de escenas que se ejecutarán cuando arranque dicha escena.
		<b>Paso de escena:</b> muestra el servicio que se llamará junto con el icono y referencia de la unidad funcional a la que pertenece.
		<b>Parámetro:</b> se representan dentro de las unidades funcionales y a su lado el valor que se le ha asignado.

Tabla 2.3. Definición de la sintaxis concreta para el DSL de Jiménez, ejemplo representativo. [Jiménez 09]

### 2.3.2. DSL para KNX/EIB utilizado en este trabajo.

La investigadora María Francisca Rosique Contreras se encuentra actualmente desarrollando un *DSL* para describir sistemas *KNX/EIB*. Este *DSL* da soporte a todos los conceptos de los sistemas domóticos *KNX/EIB* descritos en el apartado de este mismo capítulo 2.1.6. *Estándar EIB/KNX* y es el utilizado para el desarrollo descrito en este *TFM*.

Este *DSL* sirve para completar la metodología propuesta por Jiménez, sirviendo como metamodelo *PSM* para posteriormente ser la base para finalizar el desarrollo con la generación automática de código definiendo una transformación *M2T* desde un modelo *PSM* conforme a ese metamodelo.

Para documentar este *DSL* seguiremos una estructura similar a la descripción del *DSL* propuesto por Jiménez, aunque para evitar redundancia con los conceptos explicados en el apartado 2.1.6. *Estándar EIB/KNX* se pasará directamente a describir la sintaxis abstracta.

**2.3.2.1. Sintaxis abstracta: Metamodelo de soporte al *DSL*.**

Una imagen del metamodelo que da soporte a este *DSL* puede observarse en la *Figura 2.19*. Este metamodelo da respaldo a los conceptos tecnológicos específicos de *EIB/KNX* describiendo todos los elementos presentes en dicha tecnología: proyectos, dispositivos, objetos de comunicación, parámetros, áreas, líneas, direcciones lógicas, direcciones físicas, etc. Este hecho hace que sea posible describir cualquier instalación domótica de tecnología *KNX/EIB* utilizando este *DSL* con total precisión.

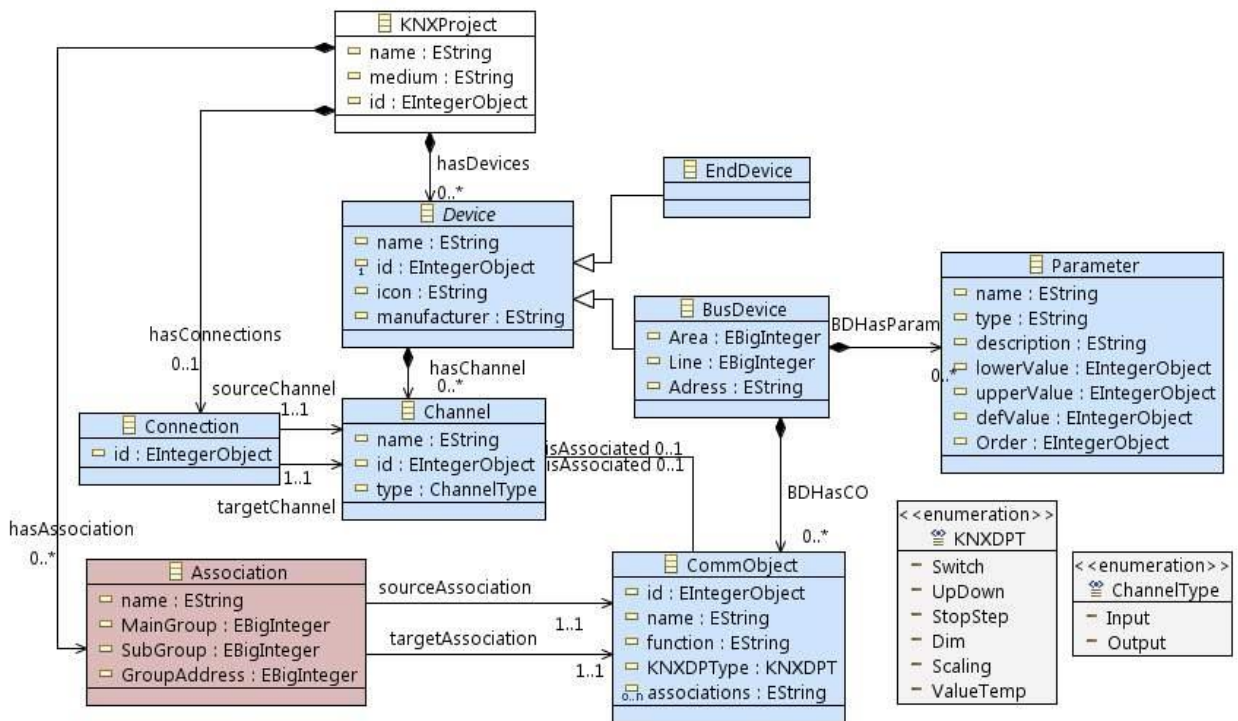
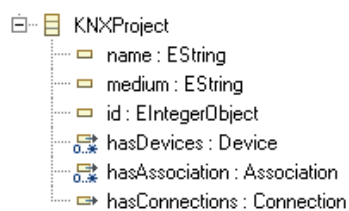


Figura 2.19. Metamodelo KNX/EIB.

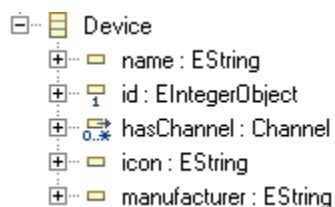
A continuación se pasará a describir cada uno de los conceptos presentes en el metamodelo, sus propiedades y asociaciones:

- **KNXProject**: Es el nodo padre de un modelo descrito conforme al *DSL* en cuestión. Representa el proyecto de instalación domótica que se quiere describir. Posee los siguientes atributos (ver también *Figura 2.20.*):
  - **name (EString)**: El nombre del proyecto.
  - **medium (EString)**: Hace referencia al medio de transmisión del proyecto (par trenzado, línea eléctrica, etc.).
  - **id (EIntegerObject)**: Identificador numérico único del proyecto.
  - **hasDevices (Device)**: Lista de referencias a los dispositivos del proyecto
  - **hasAssociation (Association)**: Lista de referencias a las asociaciones del proyecto.
  - **hasConnections (Connection)**: Lista de referencias a las conexiones del proyecto.



*Figura 2.20. Sintaxis abstracta: KNXProject.*

- **Device**: Nodo padre del que heredan dos tipos de dispositivos: *EndDevice* y *BusDevice*. Posee las características comunes a ambos (ver también *Figura 2.21.*):
  - **name (EString)**: El nombre del dispositivo.
  - **id (EIntegerObject)**: Identificador numérico único del dispositivo.
  - **hasChannel (Channel)**: Lista de referencias a los canales asociados al dispositivo.
  - **manufacturer (EString)**: El fabricante del dispositivo
  - **icon (EString)**: *Path* al icono de este dispositivo.

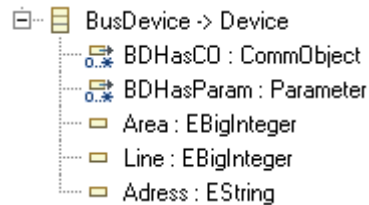


*Figura 2.21. Sintaxis abstracta: Device.*

- **EndDevice**: Representa a un dispositivo final pasivo (bombilla, interruptor, etc.). No posee atributos adicionales a los heredados de *Device*.
- **BusDevice**: Dispositivo *KNX/EIB* al uso. Además de los parámetros heredados de *Device* posee una dirección física (descrita en términos de *Área*, *Línea* y *Dirección*),

objetos de comunicación asociados y parámetros adicionales (ver también *Figura 2.22.*):

- **BDHasCO (*CommObject*):** Asociaciones a los objetos de comunicación el objeto.
- **BDHasParam (*Parameter*):** Asociaciones a los parámetros del objeto.
- **Area (*EBigInteger*):** Área del dispositivo.
- **Line (*EBigInteger*):** Línea del dispositivo.
- **Adress (*EString*):** Último octeto de la dirección física del dispositivo.



*Figura 2.22. Sintaxis abstracta: BusDevice.*

- **Channel:** Da soporte al concepto de canal en *KNX/EIB*. Entiéndase canal como conexión lógica entre dispositivos. Posee los siguientes atributos (ver también *Figura 2.23.*):

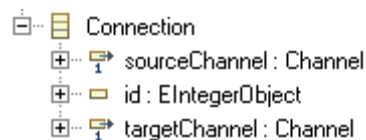
- **name (*EString*):** El nombre del canal.
- **id (*IntegerObject*):** Identificador numérico único del canal.
- **type (*ChannelType*):** Tipo del canal. Es del tipo enumerado *ChannelType* que se explica un poco más adelante.
- **isAssociated (*CommObject*):** Objeto de comunicación asociado al canal.



*Figura 2.23. Sintaxis abstracta: Channel.*

- **Connection:** Viene a representar las conexiones físicas entre los dispositivos (cableado “real”). Su estructura se aprecia en la *Figura 2.24.*

- **id (*IntegerObject*):** Identificador numérico único de la conexión.
- **sourceChannel (*Channel*):** Canal origen de la conexión.
- **targetChannel (*Channel*):** Canal destino de la conexión.



*Figura 2.24. Sintaxis abstracta: Connection.*

- **CommObject:** Representa los objetos de comunicación asociados a cada dispositivo. Posee los siguientes atributos (ver también *Figura 2.25.*):
  - **name (EString):** El nombre del objeto de comunicación.
  - **id (EIntegerObject):** Identificador numérico único del objeto de comunicación.
  - **function (EString):** Descripción de la función del objeto de comunicación.
  - **KNXDPTtype (KNXDPT):** Tipo del objeto de comunicación. Será del tipo de datos enumerado KNXDPT (que se explica más adelante).
  - **isAssociated (Channel):** Referencia al canal de comunicación al que está asociado el objeto de comunicación.
  - **associations (EString):** Lista de asociaciones del objeto de comunicación.

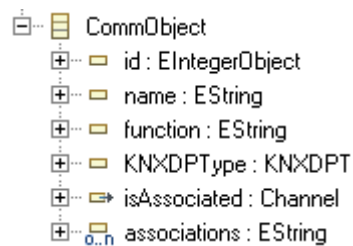


Figura 2.25. Sintaxis abstracta: CommObject.

- **Parameter:** Representa los parámetros que servirán para configurar los dispositivos. Posee los siguientes atributos (ver también *Figura 2.26.*):
  - **name (EString):** El nombre del parámetro.
  - **type (EString):** Tipo del parámetro.
  - **description (EString):** Descripción de la funcionalidad del parámetro.
  - **lowerValue (EIntegerObject):** Mínimo valor del parámetro.
  - **upperValue (EIntegerObject):** Máximo valor para el parámetro.
  - **defValue (EIntegerObject):** Valor por defecto para el parámetro.
  - **Order (EIntegerObject):** Orden de aparición del parámetro en la base de datos en ETS. Este parámetro es fruto de la ingeniería inversa realizada para trasladar los modelos al programa ETS3; para más explicaciones sobre él consultar el *Capítulo 5, Apartado 5.1.2.* de este documento.

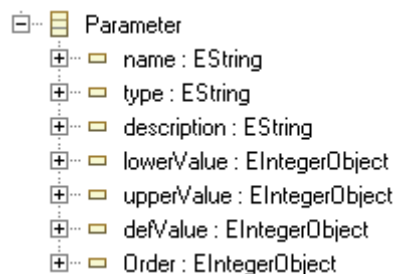


Figura 2.26. Sintaxis abstracta: Parameter.

- **Association:** Da soporte a las asociaciones entre objetos de comunicación. Este es un concepto del metamodelo que está estrechamente ligado al concepto de direcciones de grupo de *KNX/EIB*. Posee los siguientes atributos (ver también *Figura 2.27.*):

- **name (EString):** El nombre de la Association.
- **sourceAssociation (CommObject):** Referencia al objeto de comunicación que es el origen de la comunicación.
- **targetAssociation (CommObject):** Referencia al objeto de comunicación que es el destino de la comunicación.
- **MainGroup (EBigInteger):** Primer octeto de la dirección de grupo.
- **SubGroup (EBigInteger):** Segundo octeto de la dirección de grupo.
- **GroupAddress (EBigInteger):** Tercer octeto de la dirección de grupo.

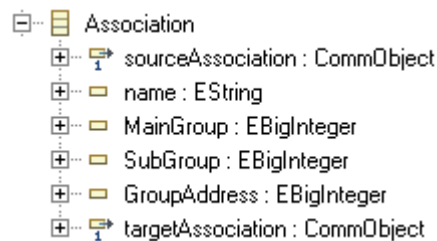


Figura 2.27. Sintaxis abstracta: Association.

Además de estos conceptos, se definen dos tipos de datos enumerados como ya se ha visto:

- **ChannelType:** Tipo del canal de comunicación. Podrá ser de entrada (*Input*) o de salida (*Output*). Ver Figura 2.28..

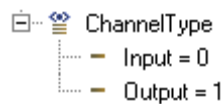


Figura 2.28. Sintaxis abstracta: ChannelType.

- **KNXDPT:** Tipo objeto de comunicación. Podrá ser *Switch*, *UpDown*, *StopStep*, *Dim*, *Scaling* o *ValueTemp*. Ver Figura 2.29.

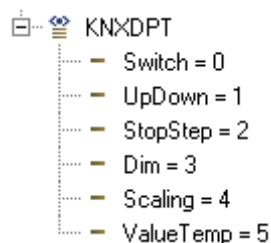


Figura 2.29. Sintaxis abstracta: KNXDPT.

## 2.4. Herramientas y técnicas para la generación automatizada de código.

La generación automática de código es ya un viejo sueño, al que la *MDA* ha dado un nuevo impulso. Últimamente han aparecido muchos métodos y herramientas que prometen la generación automática y completa del código de una aplicación a partir de su especificación en *UML*. Como ejemplo, es difícil encontrar una herramienta *CASE* que no se anuncie a sus posibles usuarios/compradores destacando sus capacidades de generación de código o su adhesión a la filosofía *MDA*.

No obstante aún queda mucho por hacer. Todos estos métodos y herramientas son capaces de generar clases en Java o tablas en una base de datos relacional (*BDR*) a partir de un esquema conceptual (*EC*) definido usando un diagrama de clases en *UML* y (algunos menos) de generar también la parte dinámica a partir de diagramas de estados o lenguajes de acciones. El problema es que la mayoría tienen limitaciones que no los harían aplicables a los trabajos a que se pretende realizar.

Es fácil intuir que a causa de esto es totalmente imposible evaluar todas las herramientas y métodos disponibles actualmente para generar código de manera automatizada. Así, a continuación se analizarán de manera resumida las herramientas y métodos más representativos de cada grupo (herramientas *CASE*, herramientas *MDA*, métodos de generación automática, herramientas basadas en *OCL*, etc.).

### 2.4.1. Uso de herramientas *CASE*.

Las herramientas *CASE* (*Computer Aided Software Engineering*, Ingeniería de Software Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

Aunque al principio su principal función era facilitar la modelización de los sistemas software, hoy en día todas las herramientas *CASE* incluyen capacidades más o menos potentes de generación de código a partir de los modelos especificados por el diseñador. Evidentemente es imposible evaluar todas las herramientas existentes (ver [UML 05] para una lista exhaustiva) por lo que se han seleccionado las siguientes:

- ***Poseidon for UML*** [Poseidon 09] es un software de modelización *UML*. Su origen está en el proyecto *ArgoUML*. En su edición profesional facilita generación de código *Java*, *C* y *C++* a partir de diagramas *UML*.
- ***Rational Rose*** [Rational 09] de *IBM* es una herramienta *CASE* comercial concebida para el despliegue, diseño, construcción, pruebas y administración de proyectos en el proceso desarrollo de software. Tiene un gran peso en el mercado siendo usada

por multitud de profesionales y de grandes empresas. Incorpora funciones de generación automática de código y documentación a partir de descripciones *UML* de todos los tipos.

- **MagicDraw UML** [MagicDraw 09] es una herramienta *CASE* desarrollada por *No Magic*. La herramienta es compatible con el estándar *UML 2.3*, e incluye generación de código para diversos lenguajes de programación (*Java*, *C++* y *C#*, entre otros). La herramienta es compatible con la mayoría de IDEs, tanto comerciales como no.
- **Objectteering/UML** [Objectteering 09], es otra herramienta *CASE* que en su edición comercial que también posibilita la generación automática de código *Java*, *C#*, *C++*, *SQL*, *CORBA*, etc. Existe un plugin para utilizarlo en eclipse a partir de descripciones *EMF*.

#### 2.4.2. Uso de herramientas MDA.

Consideramos como herramientas *MDA* a herramientas cuyo énfasis principal es dar soporte a la transformación de los modelos definidos por el diseñador de forma independiente de la tecnología (*PIMs*) a modelos dependientes de tecnología (*PSMs*) para pasar de éstos al código final. Han aparecido conjuntamente con la popularización de la propuesta *MDA*. En esta sección se evalúan tres de las herramientas *MDA* más conocidas: *MOFScript*, *OptimalJ* y *ArcStyler*.

**MOFScript** [MOFScript 10] es una herramienta para transformaciones *M2T*, como por ejemplo generación de código o generación de documentación. Proporciona un lenguaje independiente del metamodelo, de tal manera que permite utilizar cualquier tipo de metamodelos e instancias para generar el código. Se trata de un lenguaje cuyo uso está muy extendido, basado en reglas y con una sintaxis muy sencilla. Puede aprenderse más en [MOFScript 10].

**OptimalJ** [Optimal 01] de *Compuware* es una herramienta *MDA* que utiliza *MOF* para soportar estándares como *UML* y *XMI*. Se trata de un entorno de desarrollo que permite generar aplicaciones *J2EE* completas a partir de un *PIM*. En *OptimalJ* existen tres tipos de modelos:

- **Modelo del Dominio:** modelo que describe el sistema a un alto nivel de abstracción, sin detalles de implementación. Corresponde al *PIM* de la aplicación y su elemento principal es un modelo de clases del negocio.
- **Modelo de la Aplicación:** modelo del sistema desde el punto de vista de una tecnología determinada (*J2EE*). Contiene los *PSM* de la aplicación. Se genera automáticamente a partir de un modelo del dominio y está formado por tres modelos: modelo de base de datos, modelo de interfaz web y modelo *EJB*.
- **Modelo de Código:** código de la aplicación, generado a partir de un modelo de la aplicación.



*OptimalJ* distingue varios tipos de patrones: “patrones de transformación” entre modelos (para transformar un *PIM* en *PSM* y un *PSM* en código) y “patrones funcionales” que aplican transformaciones dentro de un modelo. Del proceso de desarrollo con *OptimalJ* se puede destacar:

- Generación automática a partir del *PIM* de los modelos *PSM* de la capa de presentación, capa de negocio *EJB* y base de datos, estableciendo la conexión entre las tres capas.
- Distinción entre bloques libres y protegidos en el código para impedir la modificación del código generado. Una nueva generación de la aplicación mantiene el código añadido en los bloques libres.
- La interfaz web generada proporciona una navegación por defecto para cada objeto de negocio, que permite el mantenimiento de los datos asociados a las clases del Modelo del Dominio. Esa interfaz es muy pobre pero existe la posibilidad de crear un patrón de presentación que se ajuste a unas necesidades concretas, o bien manualmente modificar el código de la aplicación.

**ArcStyler** [ArcStyler 03] de *iO-Software* es una herramienta *MDA* que también utiliza *MOF* para soportar estándares como *UML* y *XMI*, y además *JMI* para el acceso al repositorio de modelos. Integra herramientas de modelado (*UML*) y desarrollo (ingeniería inversa, explorador de modelos basado en *MOF*, construcción y despliegue) con la arquitectura *CARAT* que permite la creación, edición y mantenimiento de cartuchos *MDA* (*MDA-Cartridge*) que definen transformaciones. También incluye herramientas relacionadas con el modelado del negocio y el modelado de requisitos por lo que cubre todo el ciclo de vida. Vamos a comentar brevemente los aspectos más relevantes de *ArcStyler* e indicaremos cómo construir una aplicación de tres capas como la que soporta *OptimalJ*.

Un cartucho contiene un conjunto de reglas de transformación y se instala como un plugin. El lenguaje de script *Jpython* es el utilizado para la creación de cartuchos. Actualmente existen numerosos cartuchos para diferentes plataformas, como *J2EE*, *.NET*, servicios web, etc. Los cartuchos utilizan *Perfiles UML* para incluir en los modelos aspectos específicos de una plataforma, por ejemplo un perfil para *EJB* o *Java 2*. Una funcionalidad muy potente relacionada con los cartuchos es la herencia de cartuchos, siendo posible definir un cartucho a partir de otro existente. A través de cartuchos se pueden definir todos los modelos de *OptimalJ*.

Además, se establece un proceso bien definido para elaborar la capa de presentación de las aplicaciones, mediante la definición de un modelo *accesor* se define el flujo de control y el flujo de datos de la capa de presentación de un modo abstracto. En el proceso de generación de código, el modelo *Accessor* se implementa para una tecnología determinada.

### 2.4.3. Técnicas basadas en OCL.

*OCL (Object Constraint Language)* es un lenguaje notacional (subconjunto de UML estándar), que permite escribir restricciones sobre modelos de objetos. Estas restricciones rigen el aspecto de un objeto individual. Hoy día OCL es una técnica muy extendida para enriquecer los modelos UML con restricciones y extensiones. Las técnicas OCL se utilizan a la hora de generar código para conseguir un producto final libre de errores y más consistente.

A continuación evaluamos herramientas dedicadas específicamente a la definición y generación de restricciones *OCL*.

- **Dresden OCL** [Dresden 10] es sin lugar a dudas la más conocida. Este set de herramientas proporciona a los desarrolladores un soporte completo para el lenguaje *OCL*, y además incluye una herramienta de generación de código *Java* que, gracias al uso de diferentes módulos, un resultado funcional, coherente e interpretable por el programador.
- **OCLtoSQL** [Dresden 10] es una herramienta incluida en el anterior *kit* que permite la generación de código *SQL* a partir de un diagrama de clases con restricciones *OCL*.
- **BoldSoft ModelRun** [Bold 10] es otra herramienta importante, aunque más pensada para la definición de elementos derivados. Permite ejecutar una expresión *OCL* tanto sobre los objetos en memoria principal como sobre los objetos guardados de forma persistente en una base de datos. Incluye soporte para la generación automática de código *C++* compatible con la también plataforma de *Bold Delphi*.

### 2.4.4. Técnicas basadas plantillas.

Otra alternativa es la de generar código en base a plantillas. En este documento se va a entender por plantilla a una combinación de bloques de texto y lógica de control que puede generar un archivo de texto. Esta aproximación es la más válida para los trabajos que se pretenden realizar por las peculiaridades de los mismos. A continuación se analizarán dos posibilidades.

- **Visual Studio** [Visual 10a] de *Microsoft* proporciona la posibilidad de escribir generadores de código en base a la creación de plantillas. La lógica de control se escribe en forma de fragmentos de código de programa en *Visual C#* o *Visual Basic*. El archivo generado puede ser texto de cualquier tipo, como una página web o un archivo de recursos, o código fuente del programa escrito en cualquier lenguaje. Las plantillas de texto se pueden usar en tiempo de ejecución para generar parte del resultado de una aplicación. También se pueden usar para la generación de código, donde las plantillas ayudan a compilar parte del código fuente de una aplicación. En [Visual 10b] se puede encontrar un análisis detallado de estas posibilidades.

- **JET (Java Emitter Template)** [JET 04a] es una herramienta perteneciente al *Eclipse Modelling Framework* que posibilita la generación automatizada de código en base a plantillas. La lógica se describe utilizando una sintaxis *JSP*, y de manera sencilla se puede implementar plantillas que tomen como fuente de los datos archivos *XML* o modelos descritos usando *EMF*. Por esta integración completa con *EMF* se eligió *JET* como la herramienta que implementara la transformación *M2T* que posibilitara la generación automatizada de código que se estaba buscando.



## *Herramientas*

---

En este capítulo se describen las herramientas utilizadas en el desarrollo de la nueva metodología de desarrollo de sistemas domóticos. En primer lugar se va a describir la plataforma *ETS3* para realizar proyectos domóticos *KNX/EIB*, para en segundo lugar describir las herramientas *IT Tools* que complementan su funcionamiento. Se finaliza el capítulo analizando el entorno de desarrollo Eclipse y analizando los tres plugins utilizados en este TFM: *EMF*, *GMF* y *JET*.



### 3.1. Plataforma *ETS*.

Tal y como se ha explicado en el capítulo anterior, el *Bus Europeo de Instalación EIB/KNX* es un sistema integrado y comprensible para la automatización integral de edificios y viviendas, que proporciona soluciones flexibles, compatibles en el futuro y económicamente rentables. Su diversidad de funciones no sólo le permite ser usado en instalaciones sencillas y limitadas, sino que también permite soluciones para todo tipo de edificios. El sistema bus de instalación *EIB/KNX* está enfocado, por tanto, a satisfacer las necesidades de las instalaciones eléctricas y de edificios en general, desde la instalación, configuración y mantenimiento del sistema bus, a su puesta en servicio.

La planificación, diseño y puesta en marcha de una instalación *EIB/KNX* requiere de una herramienta software para el proyectista y el contratista eléctricos que esté bien estructurada y sea fácil de usar. Una herramienta que debe proporcionar las posibilidades de planificación, puesta en marcha y diagnóstico de las instalaciones *EIB/KNX*.

El programa *ETS* (acrónimo del inglés *Engineering Tool Software*) [KNX 10b] es la única herramienta software independiente del fabricante para diseñar y configurar instalaciones inteligentes para el control de casas y edificios hechas con el sistema *KNX*. La Herramienta *Software* para Ingeniería *ETS3* se estructura por tanto de manera flexible, extensible y modular para poder facilitar futuras ampliaciones de la tecnología *EIB/KNX*. Asimismo, se ofrece al usuario una amplia ayuda en línea que facilita toda la información necesaria en este contexto.

En los sucesivos sub-apartados se detallarán los aspectos más reseñables de esta herramienta.

#### 3.1.1. Características principales.

Las características principales del software *ETS* son:

- *ETS* es una herramienta unificada, esto es: en cualquier lugar del mundo se puede utilizar el mismo software.
- *ETS* es una herramienta del estándar *KNX*.
- Todas las bases de datos de los productos certificados por parte de los fabricantes de *KNX* pueden ser importadas al *ETS*.

#### 3.1.2. Versiones.

El software más actualizado del programa *ETS* es el *ETS 3*, la nueva generación de la familia de software *ETS*. Existe una solución para todos los niveles:

- *ETS3 Tester*: Para aprender.
- *ETS3 Starter*: Para pequeños proyectos en viviendas.

- *ETS3 Professional*: Para todas aquellas construcciones sin limitaciones en tamaño o complejidad.

### 3.1.3. Estructura del *ETS3*.

La ventana principal del *ETS* es como se muestra en la *Figura 3.1*. La interfaz de usuario del *ETS* ha sido diseñada deliberadamente teniendo en cuenta las pautas de diseño de *Microsoft*, para asegurar que el manejo general de la herramienta resulte similar al de otras herramientas de software. Como usuario es posible seleccionar qué ventana desea abrir en la ventana principal. Incluso es posible elegir abrir un tipo de ventana de entre varios tipos. La selección de la ventana es posible a través de la Barra de Herramientas o del menú de selección '*Ver/Vistas del Proyecto*'. La ventana principal consiste en los elementos siguientes:

- **Título:** El Título de una ventana contiene el nombre de la herramienta y, cuando esté disponible, el nombre de la vista actual y del proyecto.
- **Barra de Menús:** Contiene el nombre de los distintos menús.
- **Menú:** Aparece cuando se selecciona una entrada en la barra de menús. Un menú contiene funciones, tanto básicas como específicas, de la herramienta. Cada menú individual de la barra de menús del *ETS 3* profesional contiene varios elementos. Cuando un elemento no está visible (en gris-sombreado), éste no puede ser seleccionado. Sólo será posible seleccionar este elemento cuando se haya ejecutado previamente una acción que requiera de esta función. Téngase en cuenta que algunos menús dependen, de forma activa, del estado de la herramienta en cada momento.
- **Barras de herramientas:** Los botones de las barras de herramientas permiten la ejecución directa de diversas funciones del programa.
- **Iconos:** Un icono es un pequeño símbolo gráfico situado en las barras de herramientas. A través de los iconos, las órdenes y funciones más importantes del *ETS 3* pueden ser ejecutadas con un simple clic del ratón. Cuando los iconos estén sombreados en gris, no podrán ser usados. La posibilidad de utilizar las funciones que se esconden tras cada icono, depende de las funciones activas del programa en cada momento. Antes de que los iconos puedan ser de nuevo activados, puede ser necesario seleccionar otra ventana u otro elemento o función dentro de una ventana.
- **Barra de Estado:** Muestra la información del estado de las selecciones, órdenes y operaciones de la aplicación en cada momento.



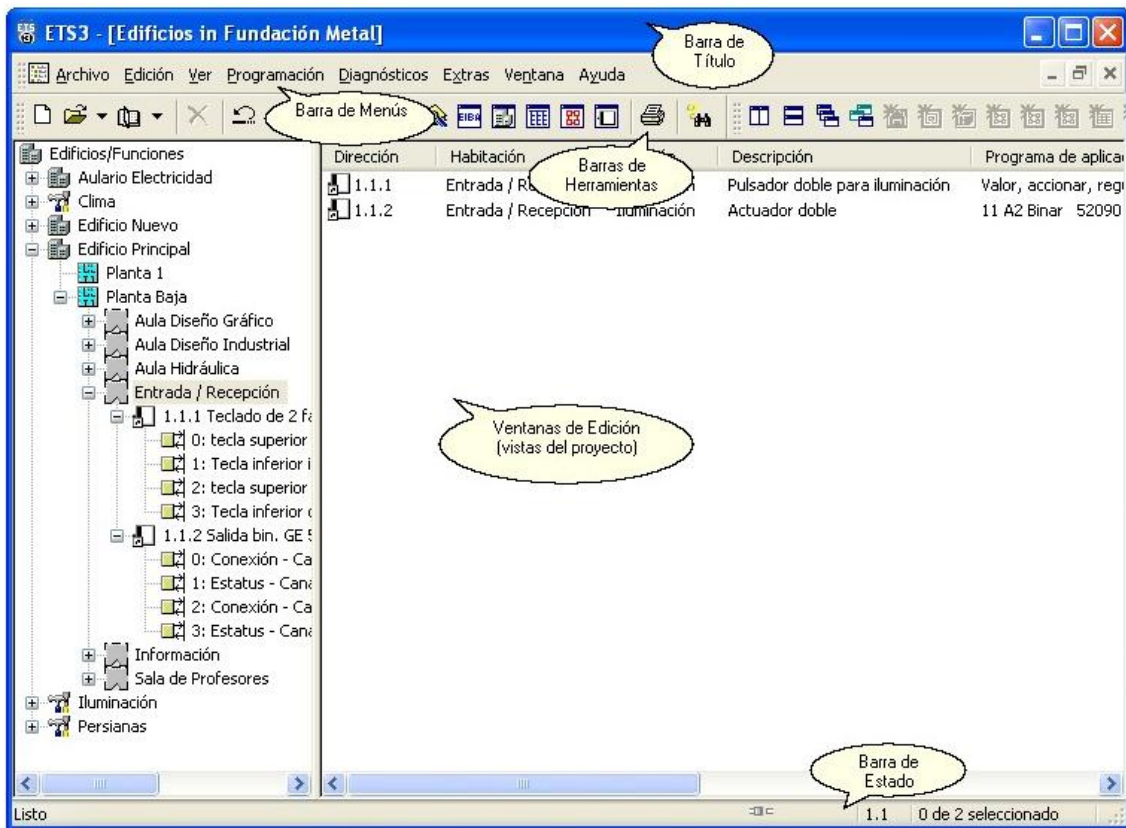


Figura 3.1. Ventana principal del ETS

Ahondando un poco más en las **barras de herramientas**, se ha de mencionar que estas contienen botones con símbolos que nos permiten un acceso rápido a las funciones más importantes del programa ETS3. Todas las funciones de las barras de herramientas son, asimismo, accesibles desde los correspondientes menús. Un ejemplo típico de una barra de herramientas puede verse en la Figura 3.2.



Figura 3.2. Barra de herramientas del ETS

Existen varias barras de herramientas, aunque sólo se muestran tres en la configuración por defecto del ETS3. Se pueden visualizar las barras de herramientas o incluso eliminarlas a través del menú Ver/Barras de Herramientas. Además, también es posible crear barras de herramientas propias y activarlas (o desactivarlas) de acuerdo con las necesidades puntuales del desarrollador. Esto puede hacerse en el elemento de menú *Personalizar*.

También puede suceder que algunas funciones contenidas en una barra de herramientas estén desactivadas (cuando esto ocurre se ven sombreadas en color gris), debido a que la ventana o función correspondiente a esa función no se encuentra seleccionada. Por ejemplo, las funciones "insertar grupo principal" o "insertar dirección de grupo" sólo estarán

disponibles cuando la ventana activa sea la vista de direcciones de grupo.

Otro punto importante son las denominadas **Vistas del Proyecto**. Las diferentes ventanas de edición del *ETS3* son llamadas también *Vistas del Proyecto*. Las Ventanas de Edición se abren desde el menú *Ver/Vistas del Proyecto* o a través de los correspondientes iconos de la barra de herramientas. Todas las ventanas de edición tienen en principio una estructura similar. En la parte izquierda de la ventana contienen un navegador con la estructura de la información en forma de árbol y en la parte derecha contienen información en forma de listados. Existen las siguientes ventanas de edición (vistas del proyecto):

- **Vista de Edificios:** En la parte izquierda de la *Vista de Edificios*, los datos se despliegan según se haya establecido la estructura del edificio. Asimismo, en esta ventana también es posible ordenar o mostrar el proyecto desde el punto de vista de las funciones que realizan los aparatos. Aquí se pueden insertar en la estructura del proyecto elementos como edificios, partes de edificio, habitaciones y armarios. En la esquina superior de la parte izquierda de la ventana se observa el símbolo a partir del cual se despliega la estructura de edificios y funciones. Si se hace *click* con el botón derecho aparece la posibilidad de insertar edificios o funciones. La apariencia clásica de este tipo de vista se ve en la *Figura 3.3*.



Figura 3.3. Vista de edificios de ETS

- **Vista de Aparatos:** La ventana *Todos los Aparatos* puede abrirse tanto desde el icono de la barra de herramientas como a través del elemento de menú *Ver/Vistas del Proyecto/Todos los Aparatos*. En la ventana *Todos los Aparatos* se muestran todos los aparatos contenidos en el proyecto activo, incluidos aquellos que no hayan sido aún asignados a una habitación, línea o función. Por medio de esta ventana se obtiene una buena visión general de nuestro proyecto, por ejemplo al apreciar fácilmente si existen aparatos sin dirección física asignada. Resulta evidente que esta ventana puede contener excesiva información, por lo que aquí es muy recomendable el uso de filtros. Es posible hacer las mismas operaciones que a través de las *Vistas de Edificio* o de *Topología* en los aparatos mostrados en esta ventana: editar aparatos, objetos de comunicación, etc. Si se desea asignar los aparatos a una habitación o armario o queremos asignar una dirección física, la manera más fácil es arrastrar con el ratón (*Arrastrar & Soltar - Drag & Drop*) el(los) aparato(s) a la ventana correspondiente. La apariencia clásica de este tipo de vista se ve en la *Figura 3.4*.



Figura 3.4. Vista de Aparatos de ETS

- **Topología:** En la *Vista de Topología*, los datos del proyecto son representados según la topología del bus (áreas y líneas). La estructura de la topología normalmente se genera de forma automática cuando se asignan a los aparatos sus direcciones físicas en otras partes del ETS 3. En función de las direcciones asignadas, van apareciendo en la vista de topología las áreas y las líneas principales. Cuando no se ha asignado la dirección física a un aparato, podemos construir explícitamente la estructura de áreas y líneas deseada y usar la vista de topología como medio de asignación de direcciones físicas fácil y rápido (utilizando *Drag & Drop*). Asimismo, resulta posible construir desde el principio un proyecto en esta ventana en lugar de en la vista de edificios. Esto se hace utilizando la *inserción de áreas*, la *inserción de líneas* y la *inserción de aparatos en la vista de topología*. La apariencia clásica de este tipo de vista se ve en la Figura 3.5.

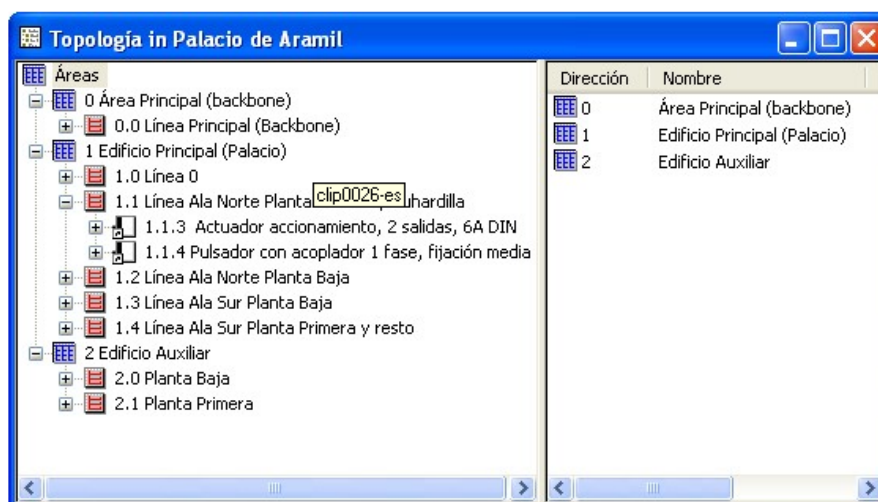


Figura 3.5. Vista de Topología de ETS

- **Direcciones de Grupo:** Las direcciones de grupo aparecen en la *Ventana de Direcciones de Grupo* y se representan - dependiendo de la configuración elegida - en 2 ó 3 niveles. La representación de las direcciones de grupo en diferente número de niveles no tiene




ninguna repercusión funcional. Sirve para mejorar la apreciación global del proyecto. Una dirección de grupo, como ya se vio en el epígrafe anterior, se representa mediante un valor de 15 bits. A modo de recordatorio, decir que en una representación en dos niveles, 4 bits constituyen el grupo principal (por consiguiente pueden representarse 16 grupos principales) y 11 bits constituyen los subgrupos o grupos secundarios (hasta 2048 subgrupos). También es posible representar las direcciones de grupo en 3 niveles. De esta forma, 4 bits constituyen de nuevo el grupo principal, 3 bits el grupo intermedio (hasta 8 grupos intermedios) y 8 bits constituyen los subgrupos o grupos secundarios (hasta 256 subgrupos). Esta flexibilidad permite al usuario una gestión más simple y eficiente de las direcciones de grupo. Además, se puede cambiar entre las dos formas de presentación sin pérdida de datos. La representación en 2 ó 3 niveles se establece en el menú *Extras/Opciones*, en la sección Presentación. Las direcciones de Grupo se crean normalmente en esta *Vista de Direcciones de Grupo*. Sin embargo, las direcciones de grupo pueden crearse de forma indirecta al seleccionar el comando 'Enlazar con...' asignando direcciones de grupo a los objetos de comunicación.

Otro aspecto muy útil de la interfaz de *ETS3* es la **Barra de Estado**, que muestra la información sobre el estado del programa y que puede apreciarse en la *Figura 3.6*. En el extremo izquierdo de esta barra, se muestran comentarios y explicaciones sobre la función actualmente seleccionada y las operaciones en marcha. Por ejemplo, se muestra información sobre cada elemento del menú, al moverse con el ratón sobre un menú abierto.



Figura 3.6. Barra de estado de ETS

La parte derecha de la barra de estado se divide por defecto en tres partes:

- *Estado de la conexión del PC con la instalación EIB/KNX* (  Online (en línea),  Offline (no en línea) y  Desconectado). Detrás del icono 'en línea', se muestra la dirección de la BCU local, si está disponible.
- *'Línea Actual'*. Este campo muestra cuál es la línea actual seleccionada. Esto es importante, ya que cuando un aparato es insertado en la ventana de componentes, la dirección física asignada automáticamente pertenecerá a esta línea. Cuando se selecciona una línea en la vista de *Topología*, puede determinarse que esta sea la *'Línea Actual'* a través del elemento de menú Edición *'Establecer como Línea Actual'* o por medio del Menú sensible al contexto (botón derecho del ratón).
- En la tercera parte, a la derecha de la barra de estado, se muestran *cuántos elementos contiene la lista de la ventana de edición y cuántos de estos se han seleccionado*.

### 3.1.4. Áreas de aplicación de ETS3.

Para los profesionales que trabajan con *ETS Professional*, es importante tener en cuenta la amplia elección de soluciones y productos que se puede encontrar en el mercado. A través del uso de este programa, se puede llevar a cabo soluciones de todas las áreas de aplicación para las cuales *ETS* tiene productos disponibles. Las áreas de aplicación del software *ETS3* son las siguientes:

- Control de iluminación (encendido/apagado, regulación, escenas).
- Control de persianas, contraventanas y toldos.
- Calefacción, ventilación y aire acondicionado (control individual de temperatura, control de radiadores, ventiladores, calderas...).
- Control de accesos y seguridad (detección de presencia, detección de fuego y robo, alarma, simulación de presencia, iluminación anti-pánico).
- Funciones de confort y control inteligente en todas las aplicaciones (control de usuario, escenarios para distintas situaciones, procesos de control inteligentes...).

### 3.1.5. Requisitos mínimos del sistema.

Los requisitos mínimos necesarios para esta herramienta son:

- *Sistema operativo: Windows 2000 Professional, Windows XP, Windows Vista.*
- *Hardware: Procesador de 1 GHz, 512 MB de memoria RAM, 3 GB de disco duro y una resolución de 1024x768.*
- *Interfaces de red KNX: RS232, USB, IP. No se debe usar un adaptador USB a puerto serie, ni puerto serie, ni LAN (Ethernet).*

### 3.1.6. Otros Software complementarios.

Además del software *ETS* también existen otros tipos de software creados para dar soporte a toda la tecnología que engloba este gran sistema. Los demás software que podemos encontrar son los siguientes:

- **iETS Server:** es un interfaz *KNX* que es instalado como parte de la instalación y permite al usuario conectar su red *KNX* a través de una *LAN* o Internet. *iETS Server* permite la programación y/o diagnóstico de su instalación vía el protocolo de internet, *IP (Internet Protocol)*. La *KNX Association* ofrece *iETS Server* como un programa *Windows*. *iETS* es una poderosa herramienta que aporta asistencia para los instaladores, proveedores de servicios, gestores de instalaciones y dueños o usuarios para la monitorización. Además aporta una resolución de problemas y actualizaciones del

software *KNX*. De esta forma, se evita perder el tiempo en desplazamientos para la reparación de cualquier mal funcionamiento de la instalación.

- **Falcon:** *Falcon Driver Library* es la potente biblioteca de *KNX* basada en *DCOM* (*Microsofts Distributed Component Object Model*) que posibilita el uso de *Falcon* mediante *LAN*. *Falcon* ofrece un API (*Application Program Interface*) o Interfaz de Programa de Aplicación para el envío y recepción de telegramas mediante la red *KNX*. Soporta el acceso *RS232*, *USB* e Internet. Las librerías *Falcon* son una herramienta extraordinaria, sobre todo para los programadores. Sienta las mejores bases para el acceso a *KNX*. Además de un sencillo API, ofrece una gestión del acceso para componentes Bus, para telegramas (direcciones de grupo), direcciones físicas, estados de los componentes y mucho más. El acceso al Bus posibilita la programación de lenguajes como *Visual C++*, *Delphi*, *Visual Basic* y otros. *Falcon* se utiliza como librería de acceso de red para *ETS3* y *EITT*. A partir de la versión 1.23, *Falcon* ofrece junto al *IConnection Manager* un interfaz de usuario fácil de usar por el integrador para crear y configurar la conexión entre *KNX* y otros medios.
- **IT Tools:** *ITT* es un conjunto de *add-in's* que aumentan la funcionalidad de *ETS3*. Nuestro grupo de investigación ha licenciado esta herramienta, para su uso en la investigación en la que este Trabajo Fin de Máster. Se ahondará más en este software complementario en el apartado siguiente.

### 3.2. IT Tools.

*IT Tools* es un conjunto de herramientas que extienden la funcionalidad de la herramienta *ETS* y que lo convierten en una herramienta flexible y a medida.

Nuestro grupo de investigación adquirió *Makros* de *IT Tools* para utilizarlo en la investigación en la que se enmarca el presente *TFM*; por lo que a continuación se procederá a describir las características más importantes del conjunto de herramientas, centrándonos posteriormente en la programación de macros en *Makros*.

#### 3.2.1. Características principales.

La funcionalidad de *ETS3 Professional* puede, como ya se ha comentado, ser extendida con software adicional. Estos programas software que extienden la funcionalidad son provistos específicamente para funcionar como extensiones. Se distinguen dos tipos de extensiones:

- **Plug-In:** Un *plug-in* es una extensión de producto provista específicamente por el desarrollador del producto para permitir el diseño y/o el trabajo con dispositivos especiales.

- **Add-In:** Un *add-in* es en general una extensión de la funcionalidad de *ETS3*, pero que no son incluidas en el estándar *ETS* porque su funcionalidad es especial o no es esencial.

Los componentes de *IT Tools* son *add-in's* provistos por *IT Gesellschaft für Informations technik mbH* (empresa alemana independiente de los desarrolladores de *ETS*).

Los *add-in's* de *IT Tools* para *ETS* son *Design*, *Reconstruction* y *Makro*. En la *Figura 3.7.* puede verse como estos *add-in's* interactúan con el entorno *ETS3*. A lo largo de los siguientes sub-apartados se expondrán las características principales de *Design* y *Reconstruction*, y se hará un análisis en profundidad de la programación en *Makro* (necesario para la comprensión del presente *TFM*).

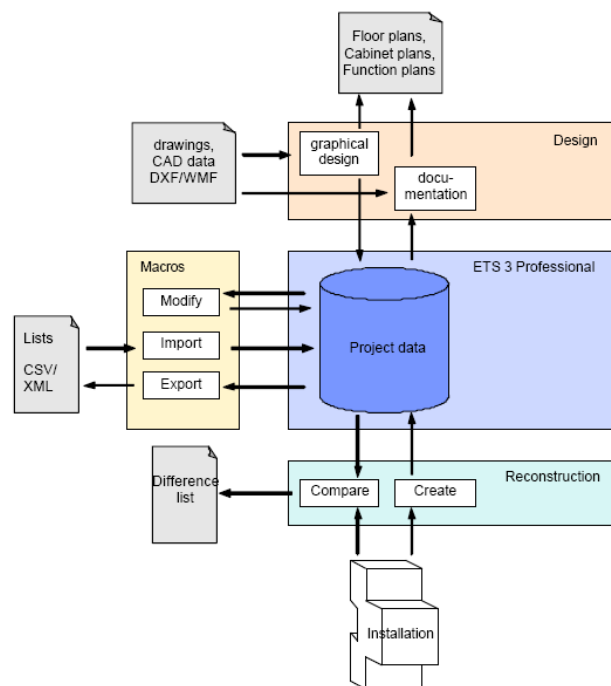


Figura 3.7. Interacción de IT Tools con ETS.

### 3.2.2. Design.

La herramienta de *IT* para el *ETS3 Design* es un módulo polifacético para la documentación y diseño gráfico de instalaciones *KNX*. El diseño se integra completamente en el *ETS3 Professional*. Las ventanas de *Design* muestran los mismos datos de proyecto que las ventanas del *ETS3*, siendo la representación, sin embargo, diferente. Con esta herramienta, los elementos del proyecto pueden representarse mediante símbolos gráficos, los cuales se complementan con otros elementos gráficos (planos de planta, armarios de distribución), dándoles así un orden lógico o apariencia de realidad. Esto puede apreciarse en la *Figura 3.8.*

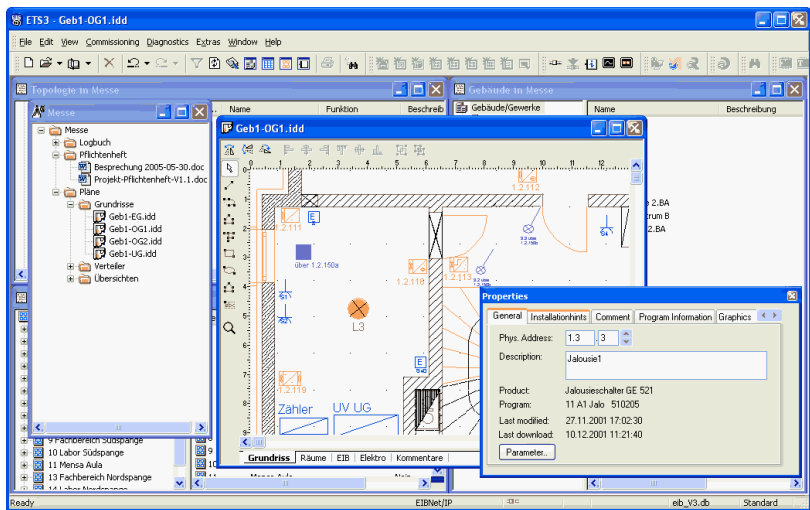


Figura 3.8. Design para ETS3.

*IT Design* expande el *ETS3* al rango funcional del sistema *CAD* para poder diseñar los proyectos *EIB* o documentarlos. Los datos del proyecto de la base de datos *ETS3* se asociarán con los datos gráficos directamente. No se necesita exportar ni importar datos. De este modo la documentación puede ser actualizada rápida y fácilmente. Los cambios en los datos del proyecto *ETS3* son tomados automáticamente por la documentación gráfica.

### 3.2.3. Reconstruction.

El módulo *IT* de reconstrucción para el *ETS3* se utiliza para recuperar la información de proyectos *ETS* perdidos o desactualizados obteniendo los datos de los componentes. *Reconstrucción* reconoce todos los componentes presentes en el Bus y recupera los datos relevantes que tienen en memoria. Este *add-in* puede apreciarse en la *Figura 3.9*.

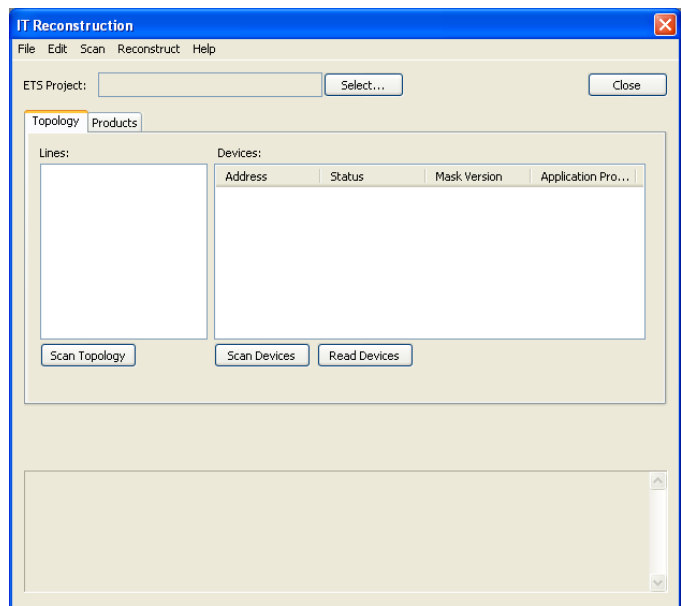


Figura 3.9. Reconstruction para ETS3.



Los datos obtenidos pueden ser utilizados para:

- Crear un nuevo proyecto ETS
- Comparar con un proyecto ETS existente
- Actualizar un proyecto ETS existente

Pueden reconstruirse los siguientes datos:

- *Topología* (áreas y líneas)
- *Componentes* (dirección física, programa de aplicaciones utilizado, ajustes de configuración, banderas y prioridad del objeto de comunicación, direcciones de grupo asignadas)
- *Direcciones de grupo*

En algunos casos es posible que la Reconstrucción sea incompleta o incorrecta:

- Los componentes con una identificación de *BCU* desconocida no aportan ninguna información.
- Los componentes programados con módulos complementarios sólo pueden aportar información y ser reconstruidos cuando esta característica es soportada por el software en cuestión, lo cual depende del fabricante (mediante el plug-in de reconstrucción correspondiente).

### 3.2.4. Makros.

El módulo *Makros* de *IT* para el *ETS3* convierte al *ETS3 Professional* en una herramienta muy flexible capaz de adaptarse a las necesidades propias de cada desarrollador de sistemas domóticos. Para satisfacer los requerimientos de un solo cliente se puede completar el proyecto con funciones útiles desarrolladas para tener una funcionalidad concreta.

#### 3.1.4.1. Características.

Las *macros* son pequeños programas fáciles de añadir al *ETS3 Professional* y que están pensados para facilitar el trabajo a los desarrolladores. Así, las *macros* facilitan la realización de tareas específicas de diseño del proyecto, documentación u obtención de datos estadísticos.

Las características de las *macros* se detallan a continuación:

- Dan solución a tareas específicas.
- Están concebidas y optimizadas para dar solución en muchos casos a problemas individuales.
- Son de rápida disponibilidad.
- Son una solución económica.
- Son fáciles de modificar.

Todas las operaciones realizadas por una *macro* pueden hacerse también “a mano”. Sin embargo, los *macros* son más rápidos y cometen menos errores. De esta manera, la principal ventaja del uso de *macros* es que ahorran tiempo.

Una *macro* está contenida en un archivo que se guarda en la carpeta *Makro* del *IT Tools for ETS3*. Para poder ejecutarlas hay que utilizar un *add-in* especial para el *ETS3* llamado *Makro Runtime*. El *Makro Runtime* se encarga de que aparezca el *macro* en el menú, además de crear la conexión de los *macros* con el modelo de objetos de la base de datos de *ETS3*. La barra de herramientas de *Makros* que aparece en *ETS3* se puede observar en la *Figura 3.10*.



Figura 3.10. Barra de herramientas de Makros.

#### 3.2.4.2. Macros preprogramadas.

Esta extensión de *ETS3* ofrece la posibilidad de adquirir hasta once macros preprogramadas, cuyo código no es visible pero que sí aumentan la funcionalidad de la herramienta. Se incluye un menú que permite el acceso a la ejecución de las mismas (*Figura 3.11*).



Figura 3.11. Macros preprogramadas.

A continuación se da una información básica sobre cada una de las macros preprogramadas:

- *Assign group globally*: Esta macro se usa para asignar direcciones de grupo globalmente a varios objetos de comunicación.

- *Export group list*: Esta macro se usa para exportar una lista de las direcciones de grupo presentes en el proyecto. El formato de salida es una hoja *Excel* (formato *.csv*).
- *Import group list*: Esta macro se usa para importar una lista de direcciones de grupo desde un archivo al entorno *ETS3*.
- *Change object flags*: Esta macro se usa para cambiar los *flags* (por ejemplo lectura o transmisión) del objeto u objetos seleccionados.
- *Change parameters*: Esta macro se usa para realizar cambios de parámetros en varios dispositivos del mismo tipo.
- *Change product*: Esta macro se usa para reemplazar el producto y/o el programa de aplicación de varios dispositivos al mismo tiempo.
- *Delete unused group address*: Esta macro elimina todas las direcciones de grupo del proyecto que no se encuentran asignadas a ningún objeto de comunicación.
- *Import device list*: Esta macro es útil cuando se está creando un sistema grande o cuando se busca automatizar el proceso de diseño *EIB*. Así, la macro crea dispositivos automáticamente en el proyecto *ETS3* siguiendo una serie de requisitos.
- *Export device list*: Esta macro genera el archivo necesario para utilizar la macro anterior, en el cual está presente la información sobre los dispositivos del proyecto.
- *Shift Group addresses*: Esta macro se usa para intercambiar dos o más direcciones de grupo, que deben de estar previamente seleccionadas en la ventana de proyecto.
- *Merge Projects*: Esta macro se usa para combinar dos proyectos *ETS3*, o para comparar dos proyectos *ETS3*.

Una información más detallada sobre las macros preprogramadas puede obtenerse en [Makros 08].

#### 3.2.4.3. Editor de Makros.

Es posible que para determinados desarrollos de proyectos, componentes específicos o requerimientos inusuales del proyecto sea necesario utilizar macros especiales. Todas las operaciones de *ETS3* que pueden ejecutarse a través de la superficie de trabajo de usuario (y algunas más) se pueden implementar también mediante una interfaz de programación. Las macros utilizan esta interfaz y o bien ejecutan funciones implementadas de forma fija, o bien utilizan datos y/o informaciones de control obtenidas de archivos.

El *Makro Editor* permite la programación de macros específicas utilizando lenguaje *VBScript*, lo cual dota a esta herramienta de una potencia mucho mayor. Así, se utilizará este editor para crear macros propias que después sirvan para implementar la traducción *M2T* pretendida. Estas macros se verán con más detalle en el *Capítulo 5* de este documento. Una imagen ilustrativa del editor de macros puede verse en la *Figura 3.12*.

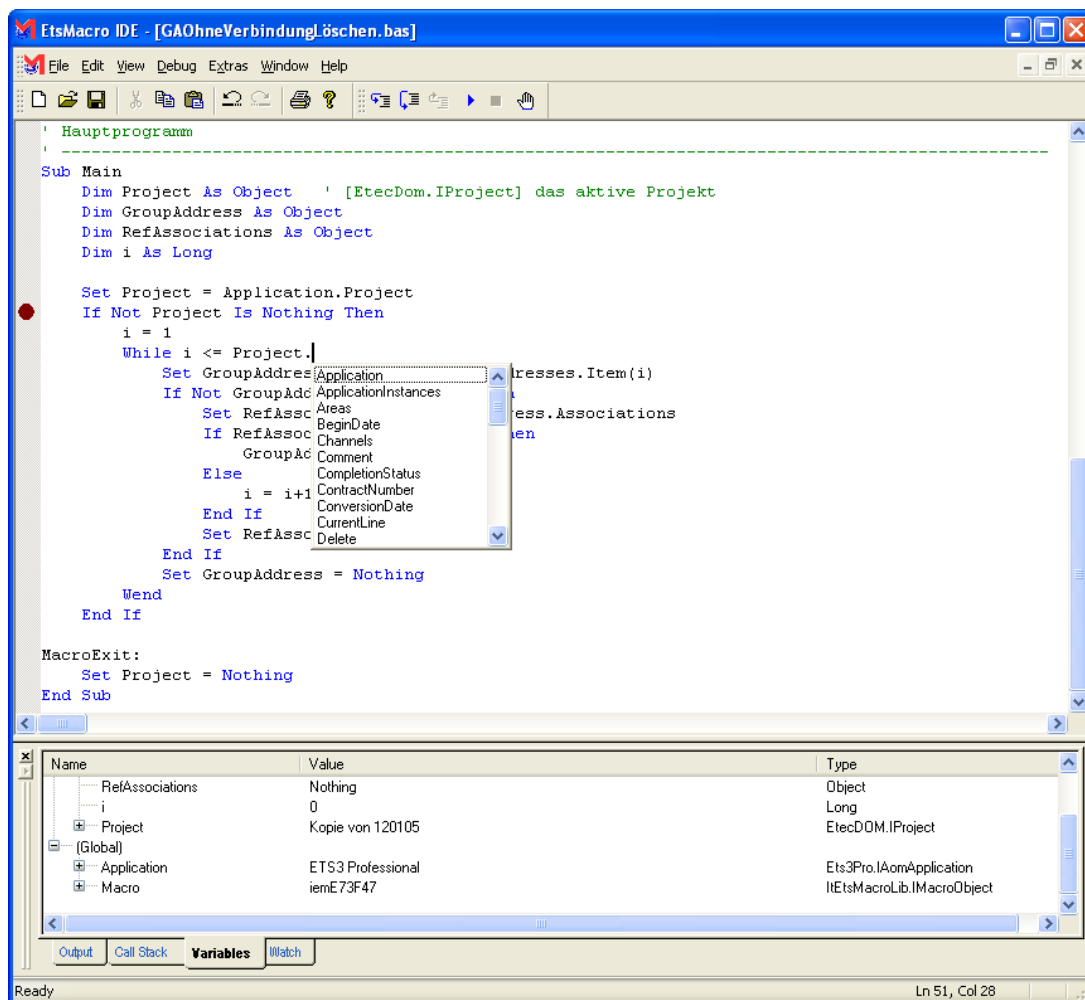


Figura 3.12. Editor de Makros.

### 3.3. Entorno de desarrollo *Eclipse*.

#### 3.3.1. Introducción.

*Eclipse* es una plataforma de desarrollo abierta y de libre distribución fruto del trabajo y la investigación de la *Eclipse Foundation*. Está compuesta de *frameworks* entendibles, herramientas y *runtimes* para desarrollar y gestionar toda la vida de un proyecto de desarrollo software.

La *Eclipse Foundation* es una organización sin ánimo de lucro cuyos miembros (como por ejemplo *Borland*, *IBM*, *Intel* o *Motorola*) financian la realización de diversos proyectos (más de 60) que evolucionan en paralelo de manera independiente y cooperativa. En un principio, *Eclipse* surgió como un *Java IDE*, pero posteriormente creció hasta convertirse en mucho, mucho más gracias a estos proyectos que implementan herramientas para programar casi cualquier cosa imaginable.

El *Eclipse Project* fue creado originalmente por *IBM* en el año 2001. Estaba financiado por un consorcio de vendedores de *software*. La *Eclipse Foundation* se creó en 2004 como una organización independiente sin ánimo de lucro para actuar como administrador de la comunidad *Eclipse*. En un principio, permitía a los vendedores el acceso a una comunidad en torno a *Eclipse* neutral, transparente y abierta para establecerse. Hoy día, la comunidad *Eclipse* está formada por organizaciones individuales y organizaciones compuestas por agrupaciones de vendedores/fabricantes de *software*.

La fundación *Eclipse* se financia con las aportaciones anuales de sus miembros y se encuentra gobernada por un conjunto de directivos representantes de las *fuerzas vivas* actuales del *software* y que conforman la denominada *Board of Direction*. Desarrolladores y consumidores importantes tienen cabida en ella, así como representantes de los comités *Open Source*.

*Eclipse* utiliza la licencia *EPL* (*Eclipse Public License*). La licencia *EPL* permite a las empresas incluir *Eclipse* en sus productos comerciales, aunque también exige contribuir con la comunidad a aquellos que creen trabajos derivados del código licenciado bajo licencia *EPL*. La naturaleza *commercial-friendly* de la licencia *EPL* puede verse en multitud de casos en que diversas compañías han desarrollado y comercializado productos basados en *Eclipse*.

### 3.3.2. Arquitectura.

La base para *Eclipse* es la *Plataforma de cliente enriquecido* (del inglés *Rich Client Platform* o *RCP*). Los siguientes componentes constituyen la plataforma de cliente enriquecido:

- **Plataforma principal.** Inicio de *Eclipse*, ejecución de *plugins*.
- **OSGi.** Una plataforma para *bundling* estándar.
- El **Standard Widget Toolkit** (*SWT*). Un *widget toolkit* portable.
- **JFace.** Manejo de archivos, manejo de texto, editores de texto.
- El **Workbench** de *Eclipse*. Vistas, editores, perspectivas, asistentes.

Los *widgets* de *Eclipse* están implementados como se acaba de mencionar por una herramienta de *widget* para *Java* llamada *SWT*; a diferencia de la mayoría de las aplicaciones *Java*, que usan las opciones estándar *Abstract Window Toolkit* (*AWT*) o *Swing*. La interfaz de usuario de *Eclipse* también tiene una capa *GUI* intermedia llamada *JFace*, la cual simplifica la construcción de aplicaciones basada en *SWT*.

Eclipse utiliza *plug-ins* para proporcionar toda la funcionalidad, a diferencia de otras aplicaciones donde la funcionalidad suele estar *hard coded*. El sistema *runtime* de eclipse está basado en *Equinox*, un estándar *OSGi*.

El mecanismo para implementar esta funcionalidad basada en *plug-ins* es un *framework* ligero de composición de software (basado en *Component-based software engineering* o *CBSE*, también conocido como *component-based development* o *CBD*). Gracias a este mecanismo, *Eclipse* ha sido extendido para usar otros lenguajes de programación como *C* y *Python*. El *plug-in framework* permite a *Eclipse* funcionar con lenguajes como *LaTeX*, aplicaciones de red como *Telnet* o sistemas de bases de datos. Por tanto, la arquitectura de *plug-in* posibilita escribir cualquier extensión deseada para el entorno.

Con la excepción de un pequeño *kernel run-time*, todo en *Eclipse* es un *plug-in*. Esto significa que cada *plug-in* desarrollado se integra con *Eclipse* y con los demás *plug-ins* de manera similar. *Eclipse* proporciona *plug-ins* para una amplia variedad de aplicaciones, como por ejemplo un *plug-in* para soportar el secuenciamiento de diagramas *UML*, un *plug-in* para explorar la base de datos, y otros muchos. Estos *plug-ins* dan soporte a los *plug-in* de usuario.

El *SDK* de *Eclipse* contiene también las *Herramientas de Desarrollo Java de Eclipse* (*Eclipse Java Development Tools* o *JDT*), ofreciendo un *IDE* con un compilador de *Java* y un modelo completo de los archivos fuente de *Java*. Esto permite técnicas avanzadas de refactorización y análisis de código. El *IDE* también hace uso de un espacio de trabajo, en este caso un grupo de *metadata* en un espacio para archivos plano, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente.

En cuanto a las aplicaciones clientes, eclipse provee al programador con *frameworks* muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc. Por ejemplo, *GEF* (*Graphic Editing Framework* - Framework para la edición gráfica) es un *plug-in* de Eclipse para el desarrollo de editores visuales que pueden ir desde procesadores de texto *wysiwyg* hasta editores de diagramas *UML*, interfaces gráficas para el usuario (*GUI*), etc. Dado que los editores realizados con *GEF* se encuentran “dentro” de Eclipse, además de poder ser usados conjuntamente con otros *plugins*, hacen uso de su interfaz gráfica personalizable y profesional. Todo lo explicado en este epígrafe puede observarse en la *Figura 3.13*.

### **3.3.3. Comenzando con Eclipse.**

En la web de *Eclipse* [Eclipse 10a] existen diversos paquetes de descargas, por lo que en primer lugar hay que descargar uno de ellos y descomprimirlo. A partir de ese momento *Eclipse* ya es plenamente funcional, ya que no precisa instalación.

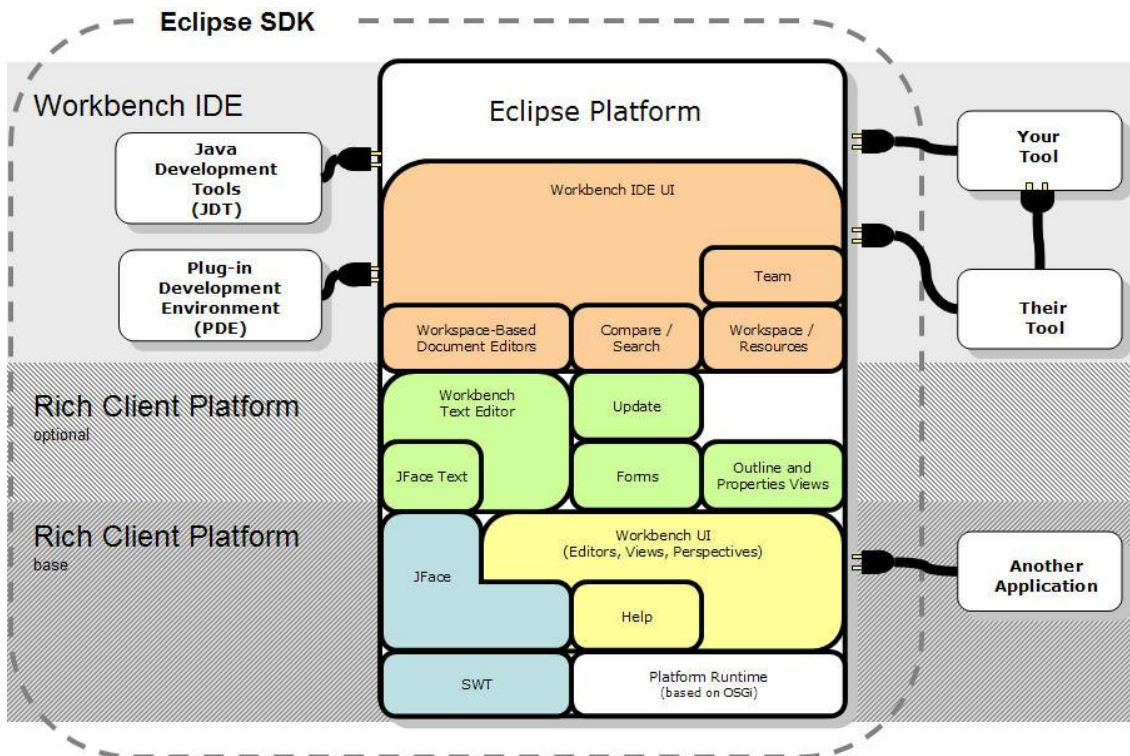


Figura 3.13. Arquitectura de Eclipse.

El paquete recomendado a fecha de realización de este TFM es el paquete *Eclipse Ganymede* [Ganymede 10] (Figura 3.14.). *Eclipse Ganymede* es la versión anual de *Eclipse* del año 2008 e incluye 23 de los proyectos mencionados en la introducción; entre ellos *Eclipse Modelling Framework* (del que se hablará a continuación).

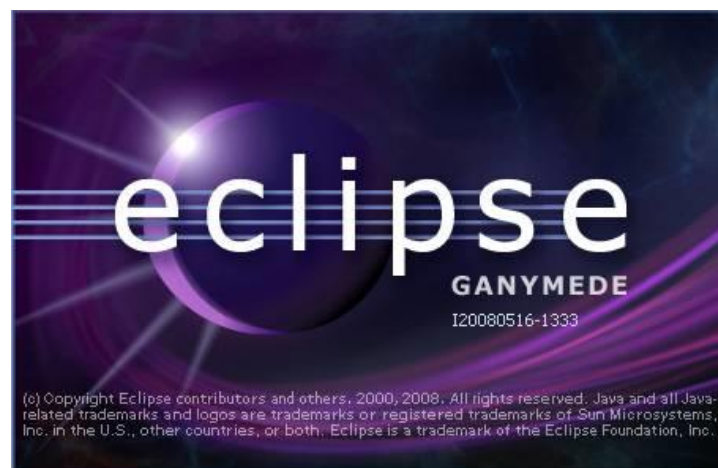


Figura 3.14. Splash-screen de Eclipse Ganymede

Los *plugins* pueden añadirse a *Eclipse* desde la misma plataforma, aunque los apartados de la web oficial *Eclipse* denominados *Eclipse Marketplace* y *Eclipse Projects* son también una forma de instalarlos.

Existe además la posibilidad de consultar una amplia oferta de información sobre el uso de la plataforma en la propia web oficial, destacando los apartados *The oficial Eclipse FAQ* y *Eclipse Wiki*. Existen también foros abiertos a las consultas.

### **3.3.4. Eclipse Modeling Framework (EMF).**

*Eclipse Modeling Framework* (Framework de modelado Eclipse, *EMF*) [EMF 10a] es un *framework* de modelado y generación de código para construir herramientas y otras aplicaciones basadas en un modelado de datos estructurado. Desde una especificación del modelo descrita en *XMI*, *EMF* suministra herramientas y soporte *runtime* para producir un conjunto de clases Java para el modelo, un conjunto de clases *Adapter* que permiten visualización y edición basándose en comandos del modelo, y un editor básico. Los Modelos pueden ser especificados usando *Java*, documentos *XML*, o herramientas de modelado como *Rational Rose*, y después ser importados a *EMF*. Lo más importante de todo es que *EMF* suministra las bases para la interoperabilidad con otras herramientas y aplicaciones basadas en *EMF*.

*EMF* está compuesto por tres piezas fundamentales:

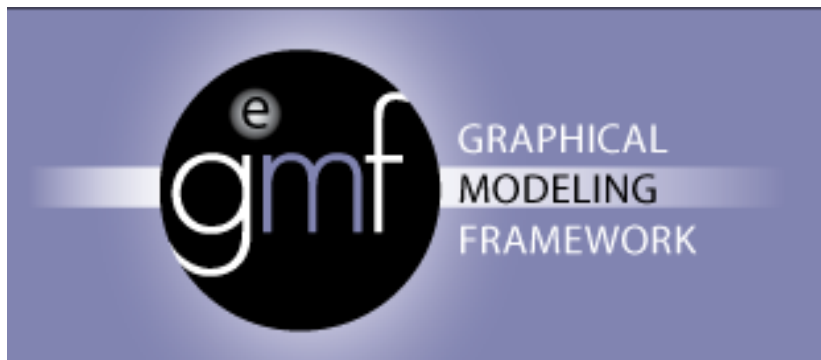
- ***EMF***. El núcleo del marco *EMF* incluye un meta modelo (*Ecore*) para describir modelos y soporte en tiempo de ejecución para dichos modelos; incluyendo notificación de cambios persistencia y consistencia entre las diversas vistas y una *API* muy eficiente para manipular objetos *EMF*.
- ***EMF.Edit***. El marco *EMF.Edit* incluye clases genéricas reutilizables para construir editores para los modelos *EMF*.
- ***EMF.Codegen***. Esta unidad de generación de código es capaz de generar todo lo necesario para obtener un editor completo para el modelo *EMF*. Incluye una *GUI* que permite especificar las opciones de generación y generadores que pueden ser invocados. La unidad de generación de código se sustenta en el componente *JDT* de *Eclipse* (*Java Development Tooling*). Los tres niveles de generación de código soportados son: *modelo* (proporciona interfaces *Java* y clases que implementan los modelos, además de un paquete factoría –meta data-), *adaptadores* (generan las clases de implementación –denominadas *ItemProviders*- que adaptan las clases de los modelos para ser editadas y mostradas) y *editor* (produce un editor de estructura propia de acuerdo a los editores generales *Eclipse EMF* y sirve de punto de partida para empezar a personalizar el editor). Todos los generadores permiten regeneración de código y conservación de cambios.



Para saber más sobre *EMF* y como modelar sistemas se recomienda la lectura de [EMF 10b].

### 3.3.5. Graphical Modeling Framework (GMF)

El *Eclipse Graphical Modeling Framework (GMF, Figura 3.15.)* [GMF 10a] proporciona soporte a la generación e infraestructura en tiempo de ejecución para el desarrollo de editores gráficos basados en *EMF* y *GEF (Graphical Editing Framework [GEF 10])*. *GMF* está dividido en dos componentes principales: la infraestructura que da soporte en tiempo de ejecución (el *runtime* [GMF 10b]) y las herramientas para generar editores capaces de utilizar ese *runtime* (el *tooling* [GMF 10c]). En la *Figura 3.16.* puede verse las dependencias entre estos *frameworks*: el editor gráfico generado mediante *GMF* depende del *runtime GMF*, pero también hace uso directo de *EMF*, *GEF* y de la plataforma eclipse.



*Figura 3.15. Logotipo de GMF.*

*GMF* permite un desarrollo rápido de editores gráficos para modelado en Eclipse, lo cual es de mucha importancia para el tema tratado en este *TFM*. Ejemplos de estos editores pueden ser editores *UML*, editores *ECore*, editores de procesos de negocios, editores de flujo, etc.

El flujo de trabajo usando *GMF* puede verse en la *Figura 3.17.* A grandes rasgos, en primer lugar habría que crear un modelo de dominio para definir la información no gráfica que manejará el editor. Después habría que definir un modelo de definición de diagrama que contendrá los elementos gráficos que se van a mostrar en el editor. Entonces hay que crear un modelo de mapeo de diagrama, que va a definir qué elementos del dominio se traducen en qué elementos gráficos. Con estos tres modelos se puede generar el editor gráfico básico, que puede ser mejorado y personalizado hasta obtener el editor gráfico definitivo.

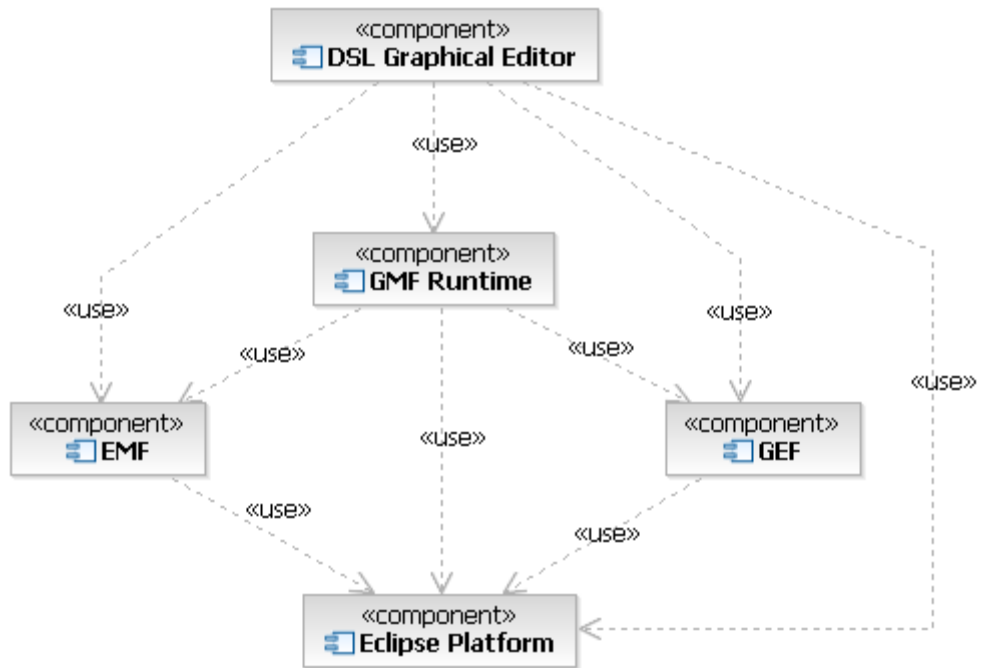


Figura 3.16. Interdependencias entre GMF, GEF, EMF y la plataforma Eclipse. [GMF 10c]

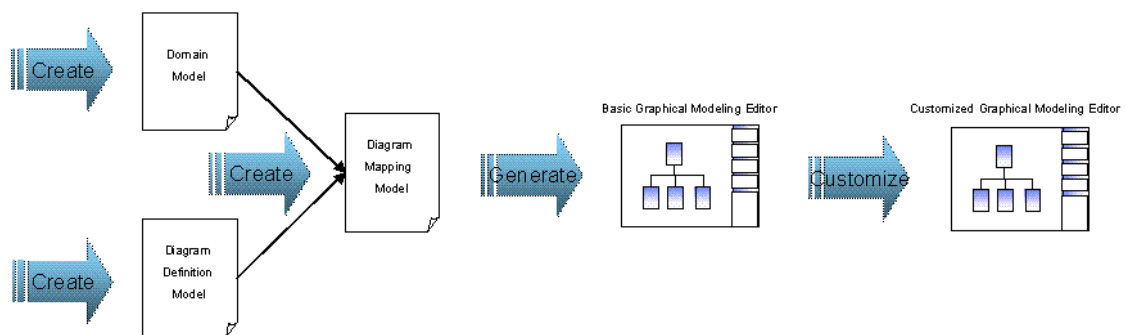


Figura 3.17. Flujo de diseño para el desarrollo de un editor gráfico usando GMF. [GMF 10c]

### **3.3.6. Java Emitter Template (JET)**

*Java Emitter Template (JET)* es una herramienta de *EMF* que se utiliza típicamente para la generación de código. La generación de código es una parte importante de *MDD*, ya que la meta de *MDD* es describir los sistemas software como modelos abstractos (como modelos *EMF/ECore* o modelos *UML*) y después definir una transformación de estos modelos a código. Sería deseable que este proceso se realizara dentro de lo posible de manera automática, en pos de un proceso más robusto y una mejor calidad del software. Las transformaciones automáticas como las realizadas mediante las plantillas de *JET* pueden lograr esto. En el *Epígrafe 4.1.3.* del presente documento se trata ampliamente esta herramienta y se dan nociones para su utilización.



## ***Metodologías de desarrollo***

---

Este capítulo se va a describir todos los pormenores tanto para desarrollar proyectos usando *JET* y *EMF* como para utilizar la plataforma *ETS3* y el *plug-in Makros*.



## 4.1. Uso conjunto de EMF y JET.

En este apartado se va a explicar la forma correcta de utilizar *EMF* y *JET*, partiendo desde cero. El lector experto en *EMF* puede directamente pasar al *Epígrafe 4.1.3*. El lector con experiencia en *JET* puede obviar el *Epígrafe 4.1.4*.

De esta manera, en este apartado se describirá en primer lugar como crear metamodelos usando *EMF* para luego explicar el proceso para crear modelos con ellos. Una vez hecho esto, se detallará como crear un proyecto *JET* que tome como fuente modelos *EMF*. Finalmente, se darán unos apuntes básicos sobre la programación de plantillas en *JET*.

Se va a partir de la información recogida en [JET 04a] [JET 04b] [M2T 10a] [M2T 10b] [Vogel 09] [JET 04c].

### 4.1.1. Creación de meta-modelos con EMF.

#### 4.1.1.1. Prerrequisitos.

Para crear meta-modelos utilizando EMF (*Eclipse Modelling Framework*) será necesario contar con el paquete *EMF runtime* que contiene el generador y varios *plug-ins* relacionados. Una vez se ha instalado este paquete, se debe verificar su disponibilidad en el entorno Eclipse (*Figura 4.1*, primero izquierda y luego derecha). Si *EMF* está correctamente instalado deben aparecer un número de *plug-ins* en torno a la centena con *plug-in ID* similar a *org.eclipse.emf.\** (*Figura 4.2*)

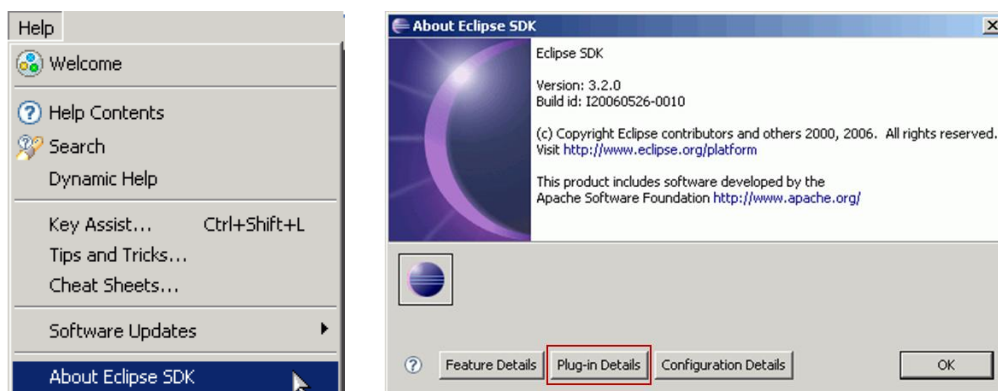


Figura 4.1. Comprobación de la correcta instalación de EMF runtime en el entorno Eclipse

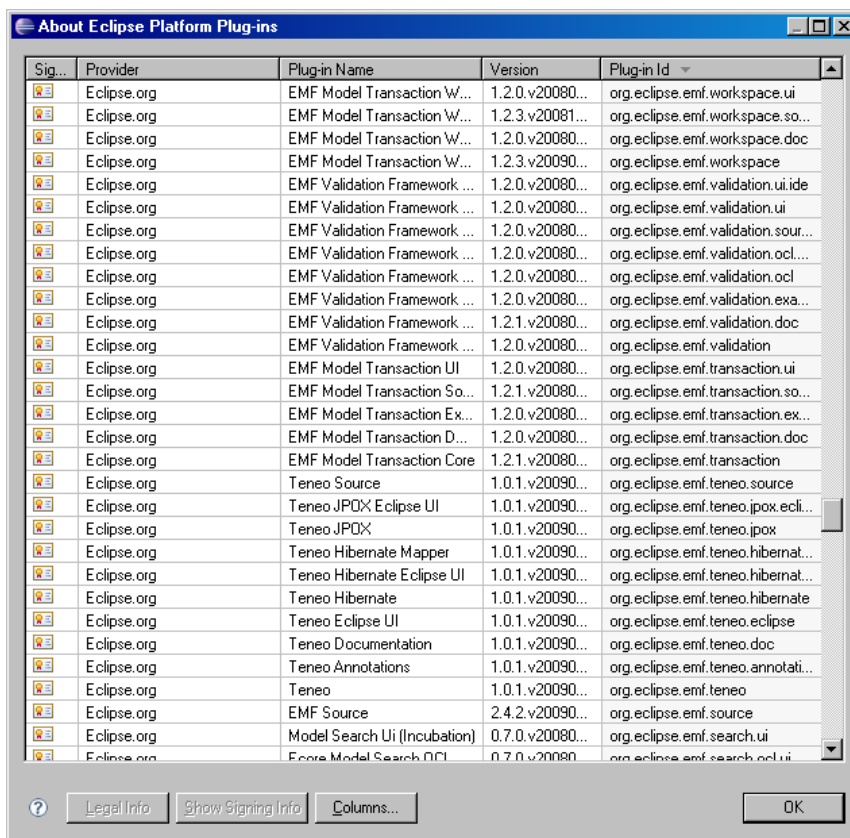


Figura 4.2. Plug-ins de EMF runtime para el entorno Eclipse.

#### 4.1.2. Modelos a partir de metamodelos usando EMF.

Supondremos en este epígrafe que el meta-modelo está descrito en otro proyecto EMF en pos de una mayor generalidad. Se creará un nuevo proyecto EMF que importará el meta-modelo de nuestra aplicación. Entonces, se generará el editor del modelo y se ejecutará en otra instancia de Eclipse donde se podrá describir el modelo.

##### 4.1.2.1. Creación de un nuevo proyecto EMF que importe el Metamodelo.

En primer lugar se debe crear el nuevo proyecto EMF. Para ello, hacer click en la barra de tareas del entorno Eclipse en File > New > Project... (Figura 4.3). Entonces, en la ventana que surge elegir EMF Project y hacer click en Next (Figura 4.4). En la ventana siguiente introducir el nombre del proyecto, la localización y hacer click en Next (Figura 4.5).



Figura 4.3. Creación de un nuevo proyecto que importe el metamodelo (i)



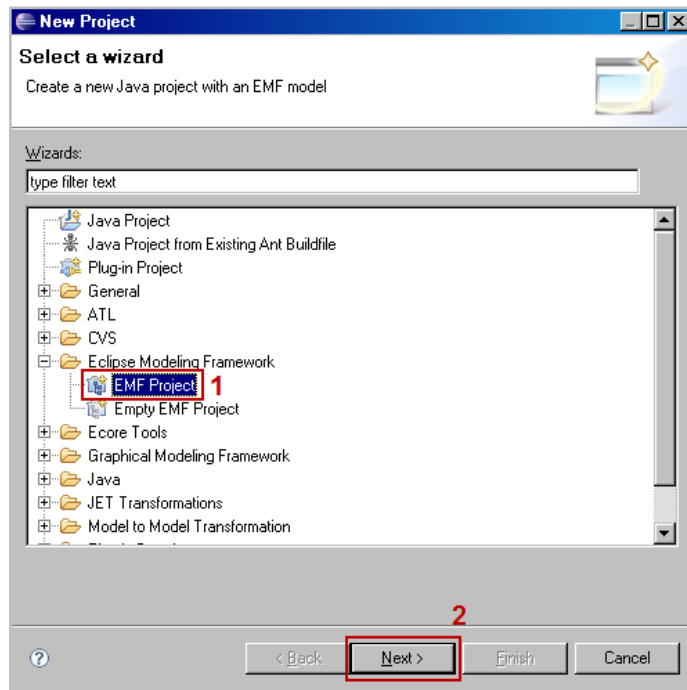


Figura 4.4. Creación de un nuevo proyecto que importe el metamodelo (ii)

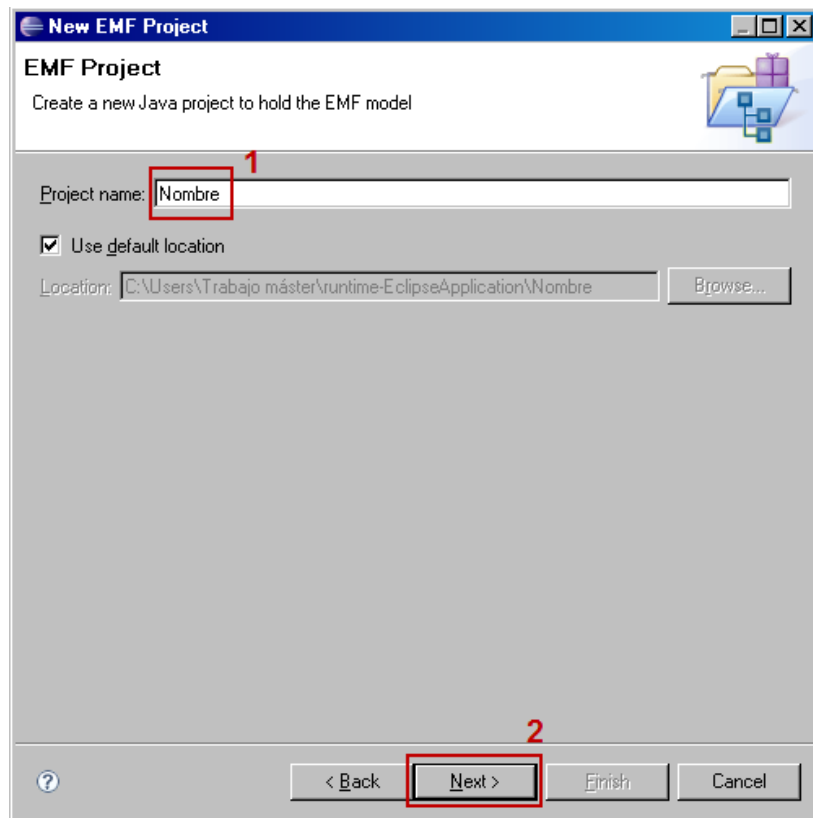


Figura 4.5. Creación de un nuevo proyecto que importe el metamodelo (iii)

En la ventana que aparece se debe elegir la forma en que se importará el meta-modelo que vamos a utilizar para crear nuestros modelos. En este caso, vamos a elegir un *Ecore model* (proveniente del apartado anterior). De cualquier manera, elegir el tipo de archivo que describe el meta-modelo y hacer click en *Next* (Figura 4.6).

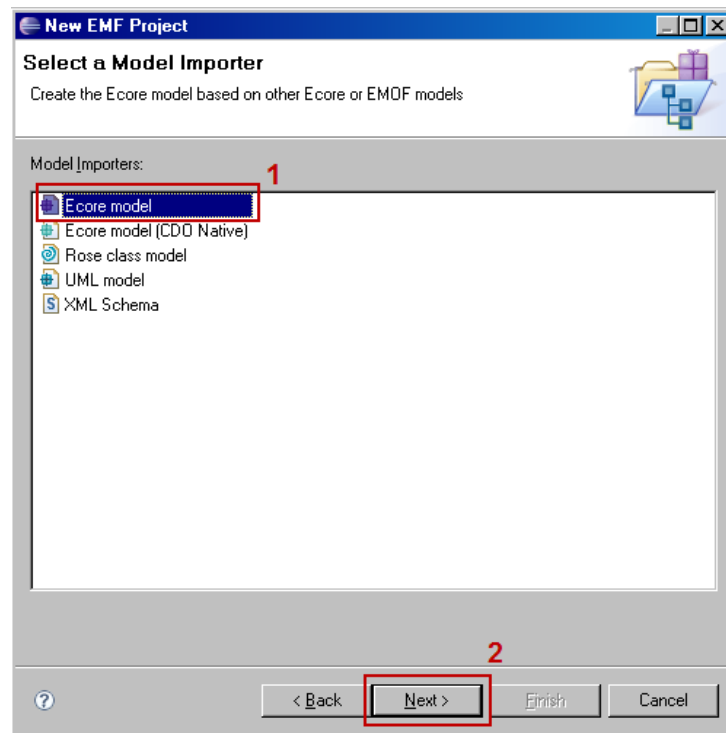


Figura 4.6. Importar el metamodelo (i)

Entonces, en la ventana siguiente hacer click en *Browse File System* y localizar y seleccionar el archivo que contiene al metamodelo. Hacer *click* en *Next* (Figura 4.7).

A continuación, seleccionar el paquete para el que se quiere generar el modelo y hacer click en *Next* (Figura 4.8). En este caso solo hay un paquete, pero pudiera ser que el meta-modelo contuviera más de un paquete.

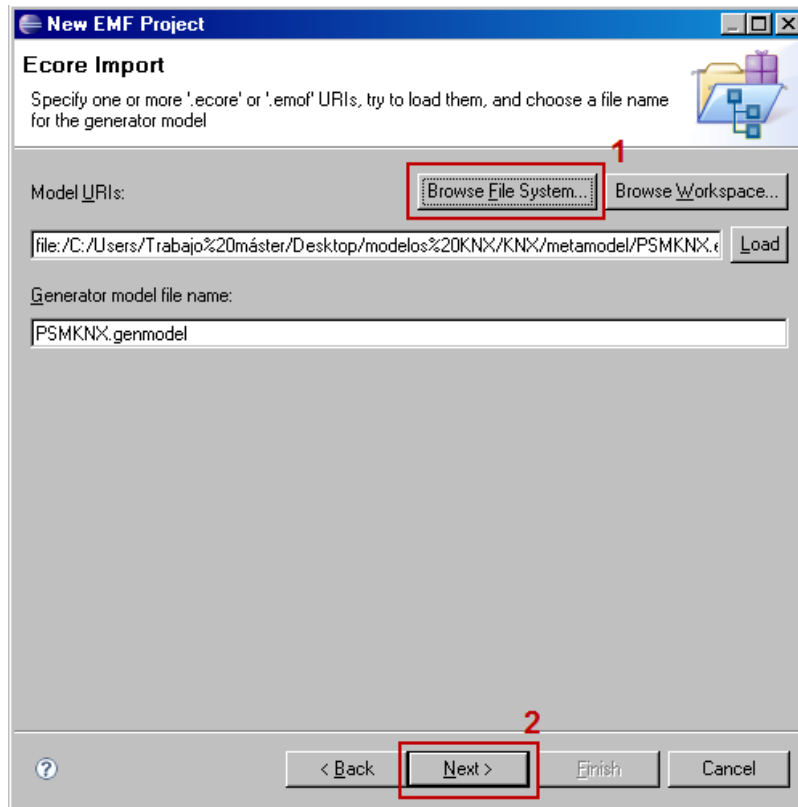


Figura 4.7. Importar el metamodelo (ii)

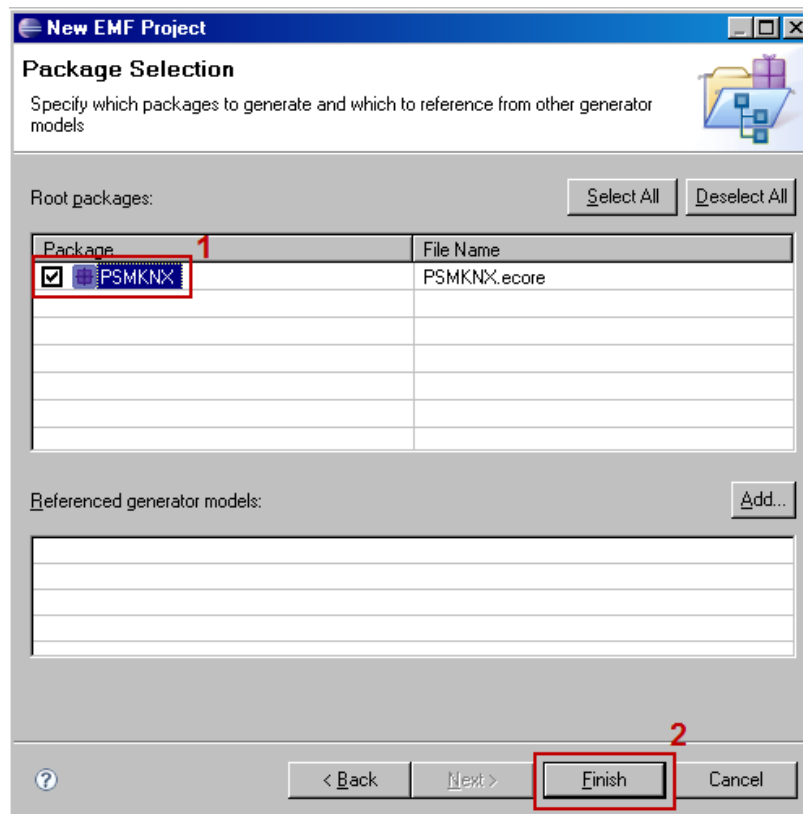


Figura 4.8. Importar el metamodelo (iii)

Si todo se ha realizado correctamente, se habrán creado un modelo *Ecore* (PSMKNX.ecore en el ejemplo, *Figura 4.9*) y un modelo generador (PSMKNX.genmodel en el ejemplo, *Figura 4.9*).

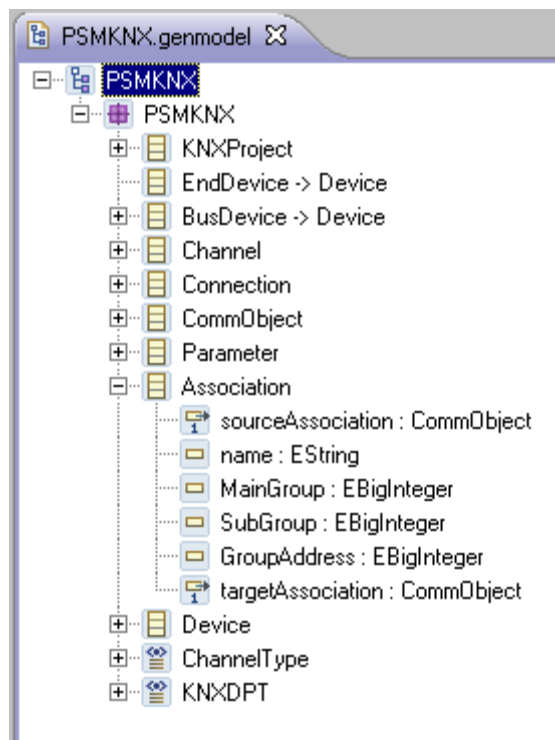


*Figura 4.9. Archivos generados para el metamodelo*

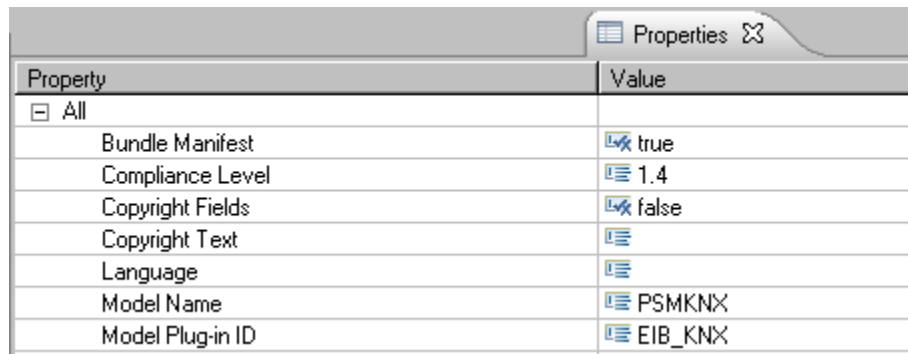
**4.1.2.2. Generación del código del meta-modelo EMF.**

El modelo generador muestra un objeto *root* que es el modelo completo (*Figura 4.10*). Este objeto modelo tiene hijos que representan sus paquetes, cuyos hijos a su vez representan clases y tipos de datos (incluyendo tipos enumerados). Los hijos de las clases son atributos de clase, referencias y operaciones; los hijos de los tipos enumerados son los literales de la enumeración.

Hay propiedades asociadas a cada objeto. Para verlas hay que activar la *properties View* haciendo click derecho sobre el objeto de modelo y seleccionando “*Show properties view*”. Se verá algo similar a la *Figura 4.11*.



*Figura 4.10. Modelo generador.*



Property	Value
[-] All	
Bundle Manifest	true
Compliance Level	1.4
Copyright Fields	false
Copyright Text	
Language	
Model Name	PSMKNX
Model Plug-in ID	EIB_KNX

Figura 4.11. Propiedades de un objeto.

Utilizando esta vista de propiedades podemos cambiar lo que consideremos conveniente del meta-modelo original antes de los siguientes pasos.

Hecho este inciso, se continuará con el procedimiento para la creación de modelos. Se debe ahora a partir del *generator model* generar el código. Para ello, hacer *click* derecho en el modelo generador y seleccionar *Generate Model Code* (Figura 4.12). Tras esto se crearán varias clases e interfaces de *java* así como los archivos *MANIFEST.MF* y *plugin.xml*.

Si se cambia el modelo hay que volver a generar el código siguiendo los mismos pasos para que los cambios se trasladen a este.

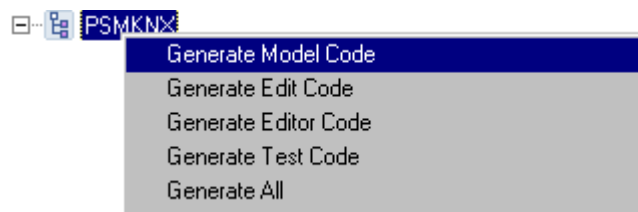


Figura 4.12. Generación del código del modelo.

#### 4.1.2.3. Generación de un editor para el modelo.

Se puede generar de manera sencilla un editor *Eclipse* completamente funcional para cualquier modelo. Por defecto, este editor se divide en dos *plug-ins*: un *plug-in* “edición” que provee una vista estructurada y permite una edición del modelo basada en comandos; y un *plug-in* “editor” que proporciona la interfaz de usuario para el editor y el *wizard*.

Para generar este editor, hacer *click* derecho en el paquete y seleccionar “*Generate Edit Code*” (Figura 4.13, izquierda). Posteriormente, volver hacer *click* derecho y seleccionar “*Generate Editor Code*” (Figura 4.14, derecha).

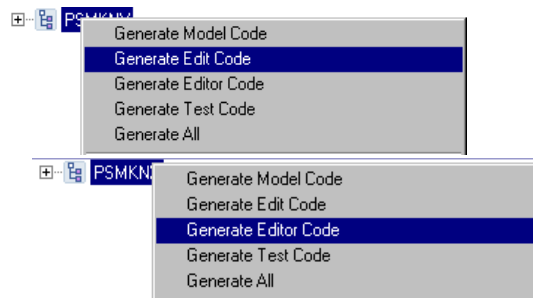


Figura 4.13. Generación del editor para el modelo.

Tras esto, se generaran dos proyectos más visibles en el *Package Explorer view*, con los sufijos *edit* y *editor* (Figura 4.14.).

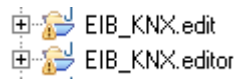


Figura 4.14. Proyectos generados con los sufijos *.edit* y *.editor*.

NOTA: Es posible resumir los pasos para generar los códigos para el modelo y el editor utilizando la opción *Generate All* presente en el diálogo emergente de las figuras de este apartado. En este caso, se generará además un código de test que en principio puede ser ignorado.

El código debe compilarse llegado este punto, y recompilarse para cada cambio que se produzca en él. Por defecto, *Eclipse* compila estos códigos según se generan. Si se diese el caso de que la opción de compilación automática se encontrase deshabilitada, se debería hacer una compilación manual haciendo *click* en la barra de tareas en *Project > Build All*.

#### 4.1.2.4. Ejecutar el editor generado.

En este punto se debe ejecutar el editor de modelos para nuestro meta-modelo generado en el paso anterior. Para ello, se debe hacer *click* en la barra de tareas en *Run* y después seleccionar *Run As > Eclipse Application* (Figura 4.15.).

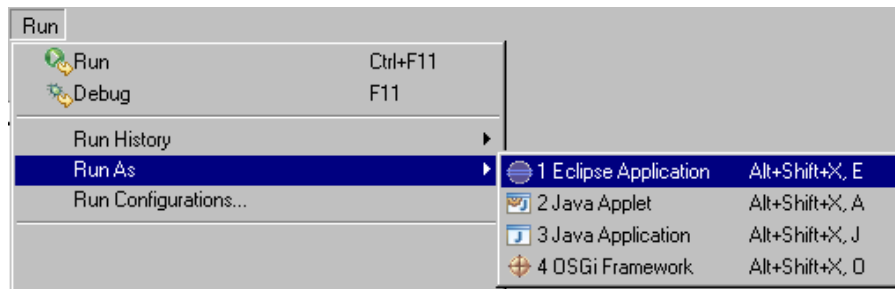


Figura 4.15. Ejecutar el editor de modelos.

Una segunda instancia de la plataforma *Eclipse* se ejecutará. Si examinamos los *plug-ins* presentes en esta instancia *Eclipse* de la misma forma que en el apartado 4.1.1.1. se puede observar como entre ellos están presentes los *plug-ins* generados en el apartado anterior (en la Figura 4.16., sombreados en tono magenta).

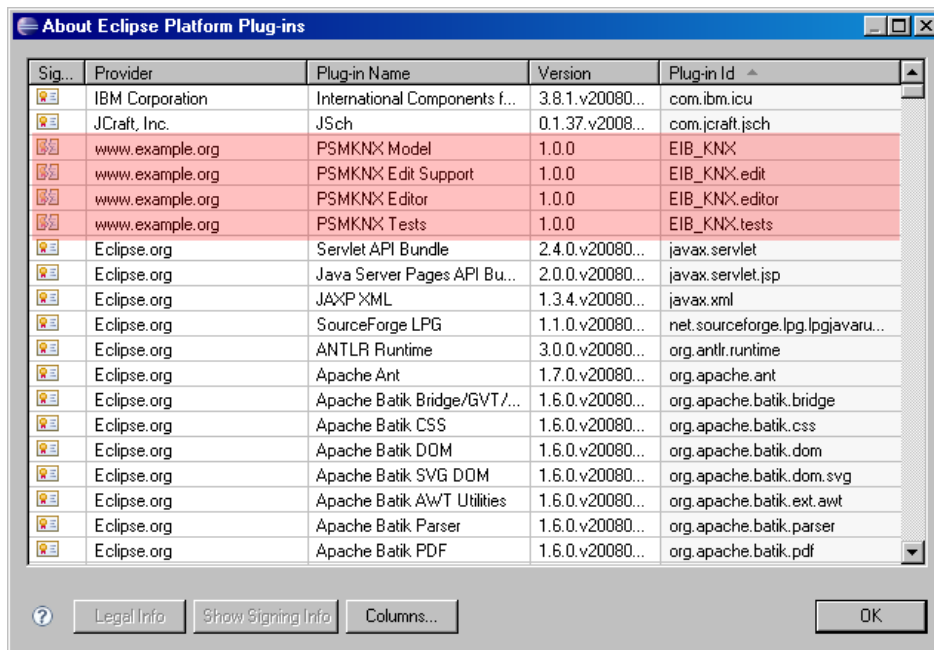


Figura 4.16. Plug-ins generados presentes en la nueva instancia Eclipse.

#### 4.1.2.5. Generar el modelo.

Llegado este punto, se puede generar el modelo a partir del meta-modelo original en la nueva plataforma *Eclipse*. Para ello, se usará el *wizard* del *plug-in* editor generado en el apartado 4.1.2.4.

Hacer *click* en *File > New > Project* y en la ventana emergente seleccionar *General > Project*. Hacer *click* en *Next* (Figura 4.17.).

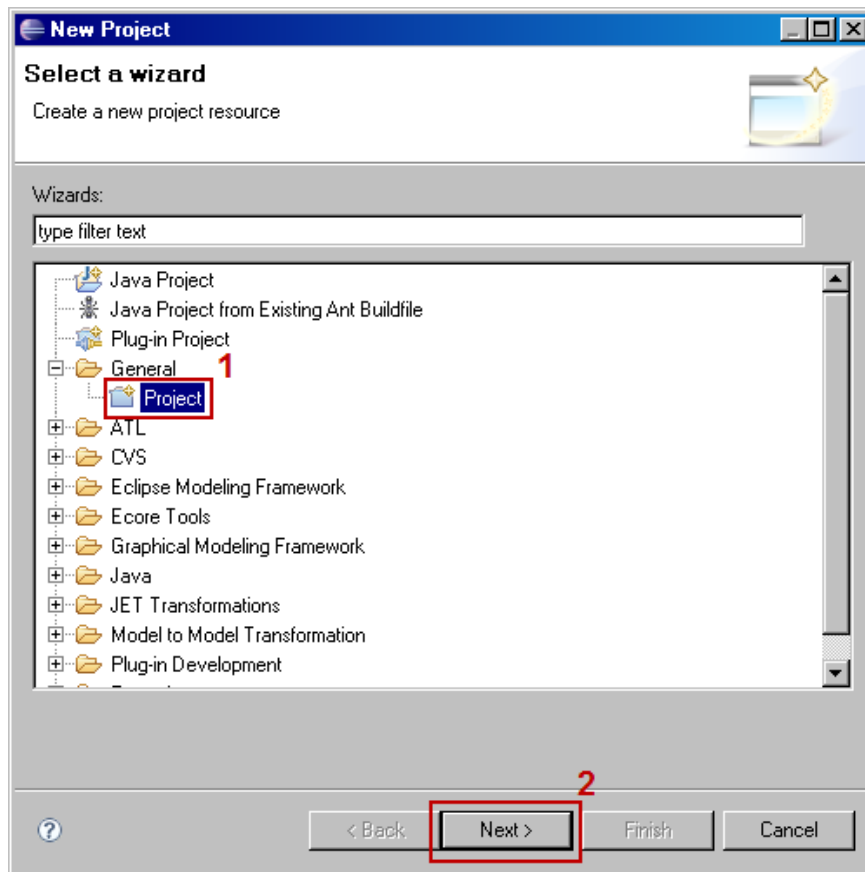


Figura 4.17. Crear un nuevo proyecto para generar el modelo.

Hacer *click* derecho en el nuevo proyecto y seleccionar *New > Other* del menú contextual desplegado (Figura 4.18.).

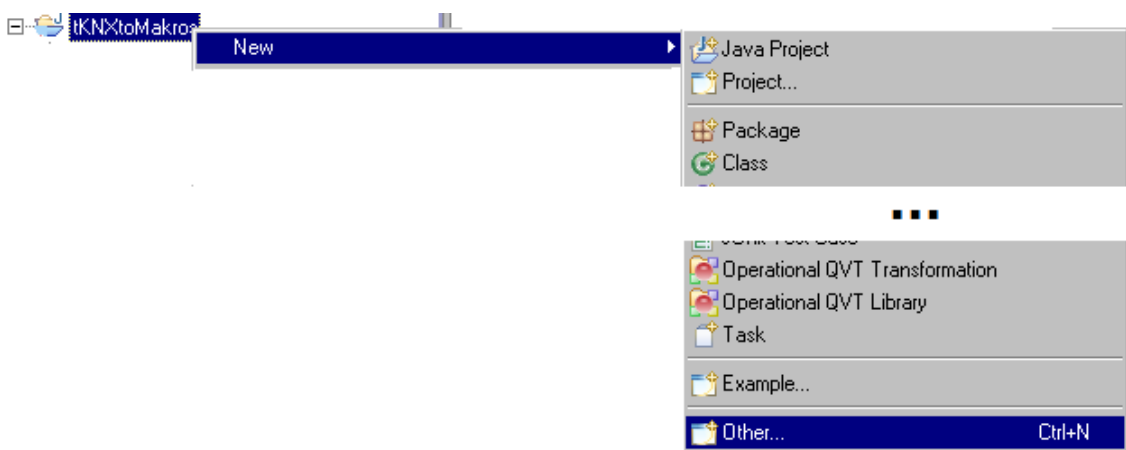


Figura 4.18. Añadir una instancia del meta-modelo (i)



En la ventana emergente, el *wizard* generado anteriormente se encuentra dentro de *Example EMF Model Creation Wizards*. Seleccionarlo y hacer *click* en *Next* (Figura 4.19.).

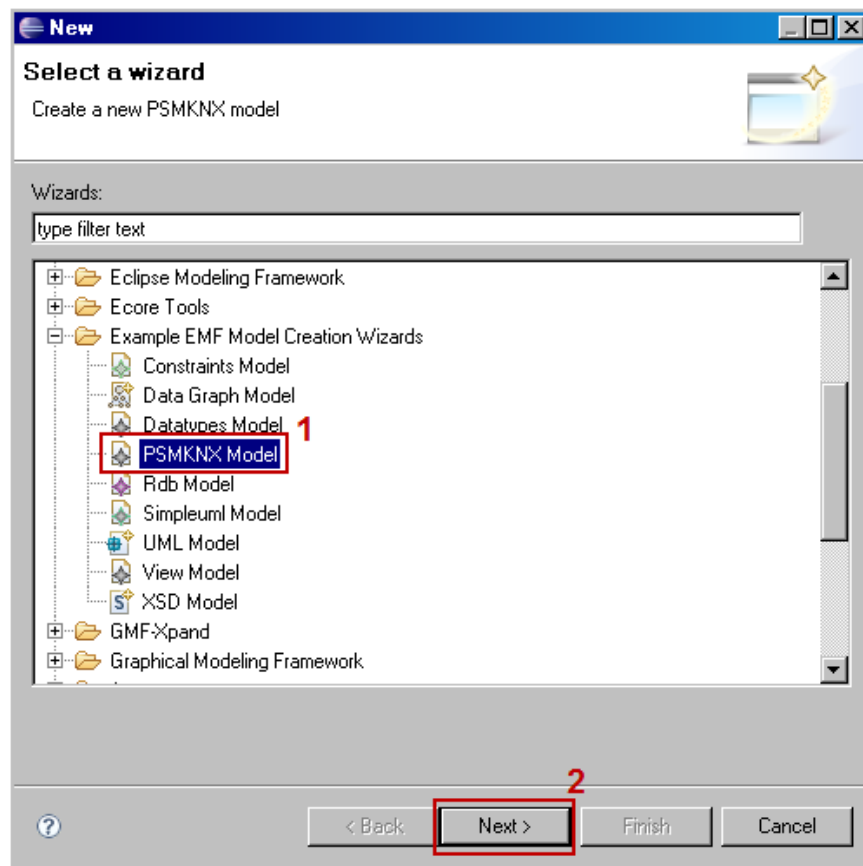


Figura 4.19. Añadir una instancia del meta-modelo (ii)

Deberemos introducir la siguiente ventana el nombre para el modelo y su ubicación dentro del proyecto, y hacer *click* en *Next* (Figura 4.20.). En la siguiente ventana, elegir el elemento *Root* que se creará y hacer *click* en *Finish* (Figura 4.21.). Tras esto, se habrá añadido un nuevo archivo a nuestro proyecto con la extensión igual al proyecto que originó los *plug-ins* en los apartados 4.1.2.2. y 4.1.2.3 (Figura 4.22. izquierda). Este archivo se podrá editar mediante un editor gráfico que permitirá ir agregando elementos de acuerdo al meta-modelo original (Figura 4.22. derecha).

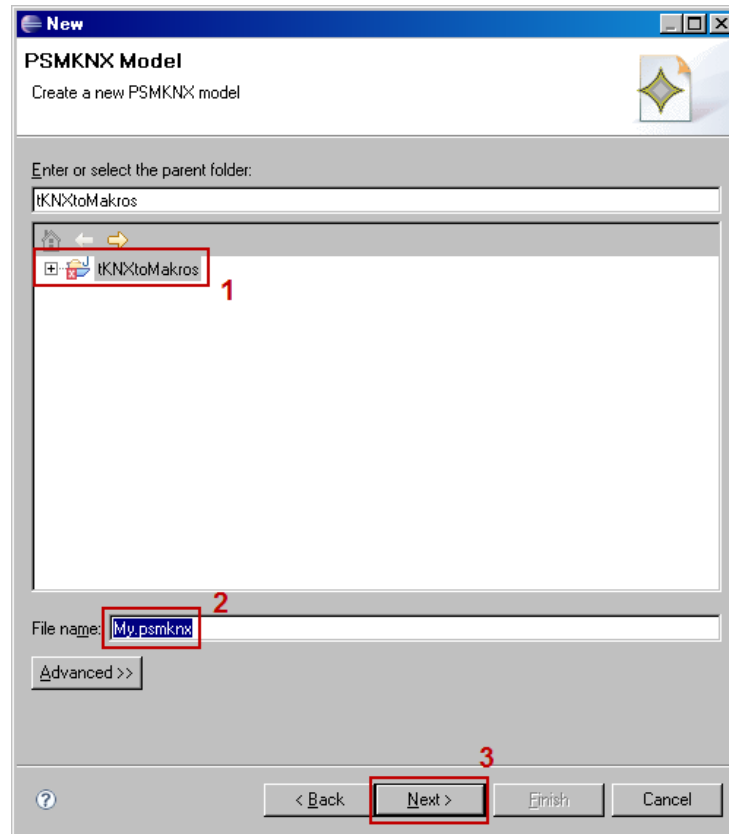


Figura 4.20. Añadir una instancia del meta-modelo (iii)

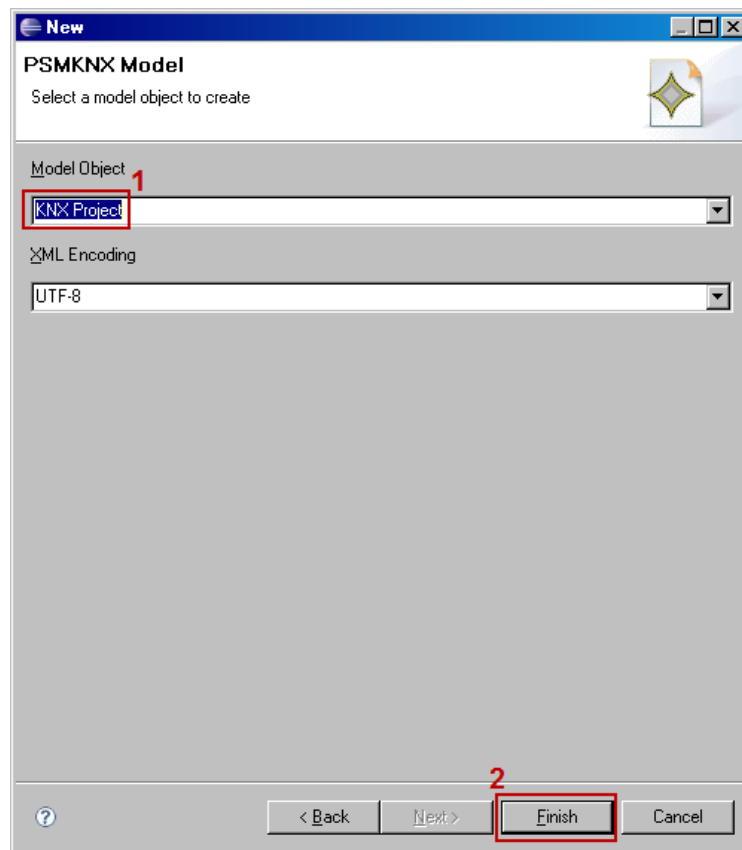


Figura 4.21. Añadir una instancia del meta-modelo (iv)

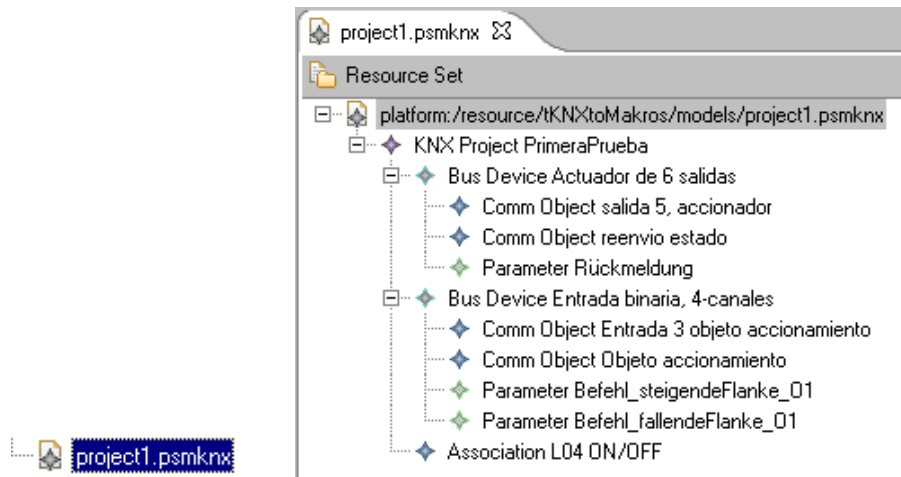


Figura 4.22. Instancia (modelo) del meta-modelo.

### 4.1.3. Proyecto JET que use como fuente un modelo.

#### 4.1.3.1. Introducción.

*Java Emitter Template (JET)* es un subproyecto de *Eclipse Modelling Framework (EMF)* centrado en simplificar el proceso de generación automática de código (*Java*, *XML*, *JSP*, etc.) a partir de plantillas. *JET* funciona a partir de plantillas muy similares a los *JSP* que, al igual que en esta tecnología, son traducidas a una clase *Java* para luego ser ejecutadas por medio de una clase generadora creada por el usuario.

En este *TFM* se pretende utilizar *JET* para traducir automáticamente los modelos *EMF* de sistemas *KNX/EIB* a una *Makro* ejecutable en *ETS3* que genere dicho proyecto. Para ello, en primer lugar, se ha de explicar cómo crear un proyecto *JET* y como configurarlo para poder traducir correctamente modelos a texto. Después, se introducirá al lector en la programación básica con *JET* y se darán apuntes prácticos para la programación de plantillas que tomen como fuente modelos. El primer punto se explica a continuación, el segundo en el apartado siguiente (*apartado 4.1.4*).

#### 4.1.3.2. Proyecto en JET compatible con EMF.

##### 4.1.3.2.1. Crear un proyecto JET.

El primer paso es crear un proyecto *JET* usando el *Wizard* existente para ello. De la barra de tareas elegir: *File > New > Project* y elegir el *Wizard* denominado *JET Transformation Project* bajo la carpeta *JET transformations* (Figura 4.23.). Hacer *click* en *Next*.

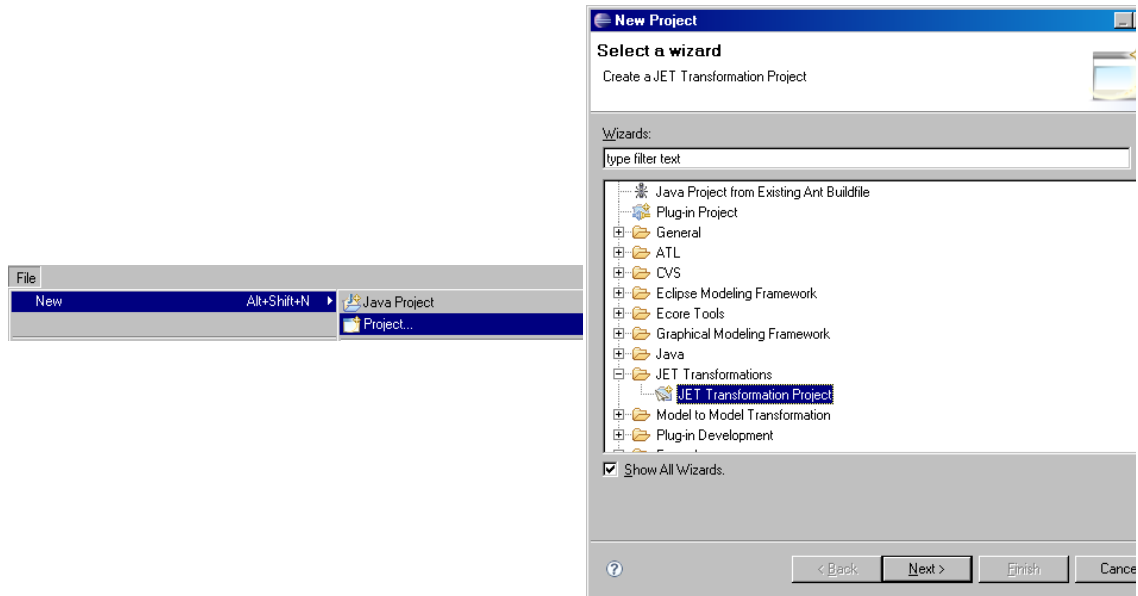


Figura 4.23. Crear un proyecto JET (i).

En la ventana siguiente introducir el nombre del proyecto y hacer *click* en *Next* (Figura 4.24., izquierda). En la siguiente ventana rellenar los campos relativos al nombre de la transformación, al ID de la transformación y a la descripción y hacer *click* en *Finish* (Figura 4.24., derecha).

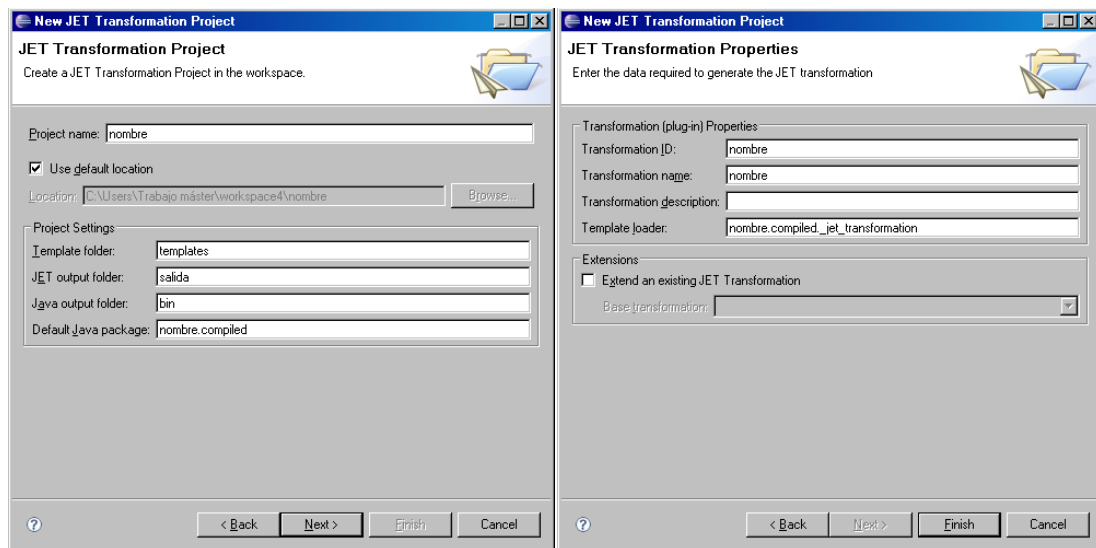


Figura 4.24. Crear un proyecto JET (ii).

La nueva transformación incluirá dos plantillas en un directorio denominado *templates*: la plantilla *main.jet* (que es el punto de entrada de la transformación y que generalmente llama a una o más plantillas que escriben el texto de salida) y la plantilla *dump.jet* (que se crea por defecto y es llamada por *main.jet*; en ella se supone que se va a describir la transformación utilizando *JET*).

En el proyecto de *JET* también se pueden encontrar otros archivos como el archivo *META-INF/MANIFEST.MF* (que es el manifiesto del programa y declara un identificador único para el proyecto y la transformación), el archivo *build.properties* (que es un archivo de texto que define las propiedades del proyecto), el archivo *plugin.xml* (que es un añadido al manifiesto y da información adicional sobre la transformación) o el archivo *sample.xml* (que es un documento que puede usarse como fuente para un *test* simple de la transformación).

#### 4.1.3.2.2. Editar el archivo *plugin.xml*.

Para posibilitar que una transformación *JET* utilice un documento *EMF*, se debe modificar el archivo *plugin.xml* para cambiar el *loader* de los archivos. Para esto, abrir el archivo *plugin.xml* (Figura 4.25., izquierda) y hacer *click* en la pestaña *Extensions*. Expandir el elemento *org.eclipse.jet.transform* en el árbol *All Extensions*. Hacer *click* en el elemento *Transform*. En el campo *modelLoader* introducir *org.eclipse.jet.emf* (Figura 4.25., derecha).

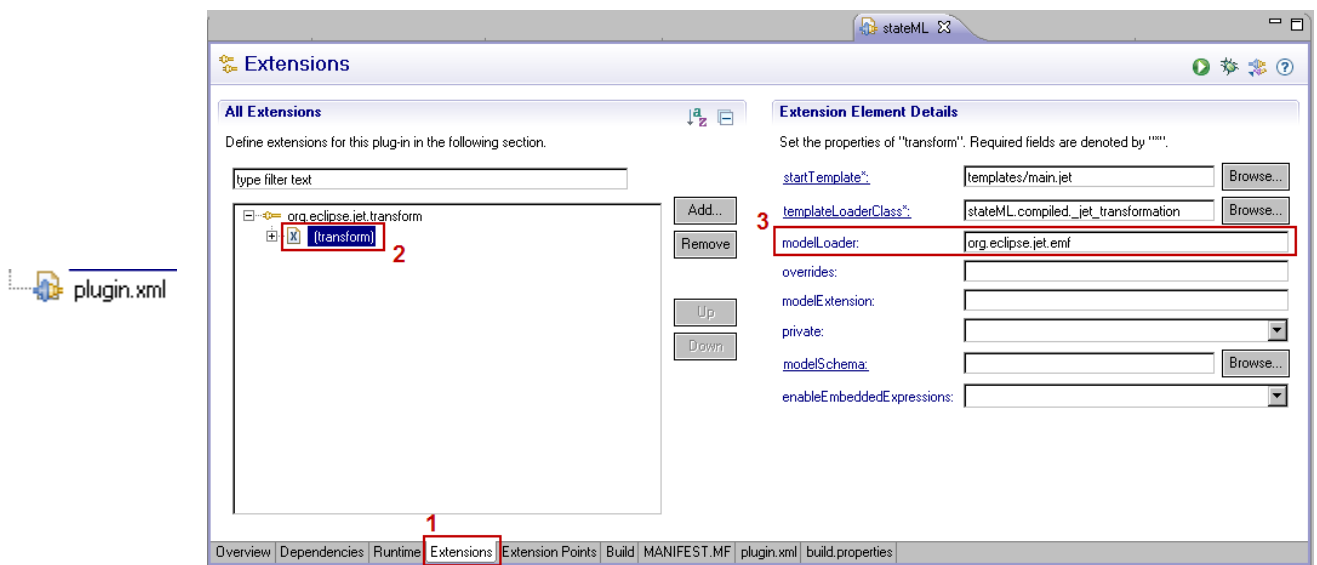


Figura 4.25. Editar el archivo *plugin.xml*

#### 4.1.3.2.3. Ejecutar la transformación.

Una vez descritas en las plantilla, las transformaciones *JET* se deben ejecutar mediante el menú *Run* de *Eclipse*. Para ello, seleccionar *Run > Run Configurations* del menú desplegable que surge al hacer *click* derecho en el objeto *EMF* que queremos que sea la fuente de la transformación (Figura 4.26.).

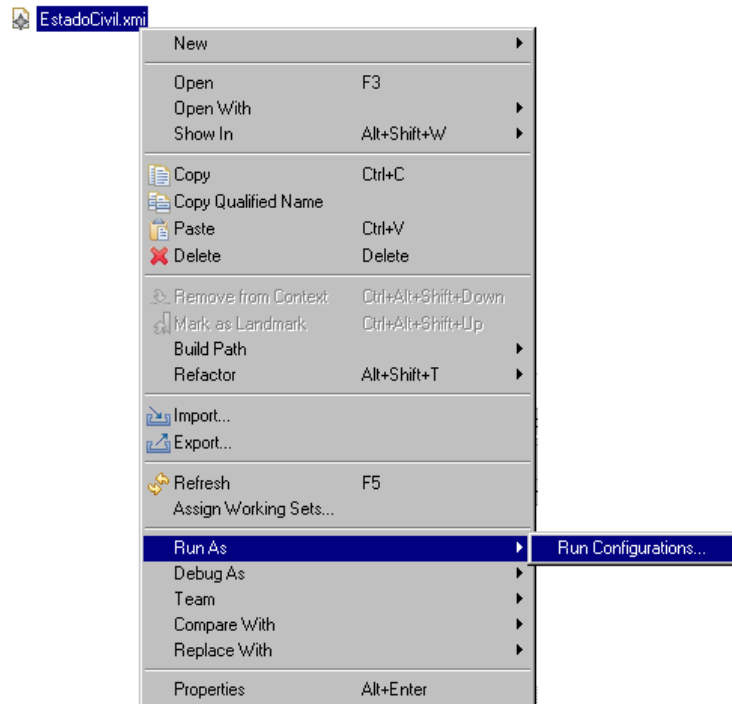


Figura 4.26. Ejecutar la transformación (i).

Entonces, en la ventana que surge se debe dar un nombre para la configuración (con el propósito de que se pueda a volver a ejecutar la transformación de una manera rápida) y seleccionar la categoría *JET Transformation category*. Proporcionar un nombre para la transformación, comprobar que el archivo de entrada es el correcto y hacer *click* en *Run* (Figura 4.27.). Si todo está correcto se ejecutará la transformación y se generará el archivo de salida descrito en el *main.jet* de la forma que nuestra plantilla indique y en base al archivo de entrada seleccionado.

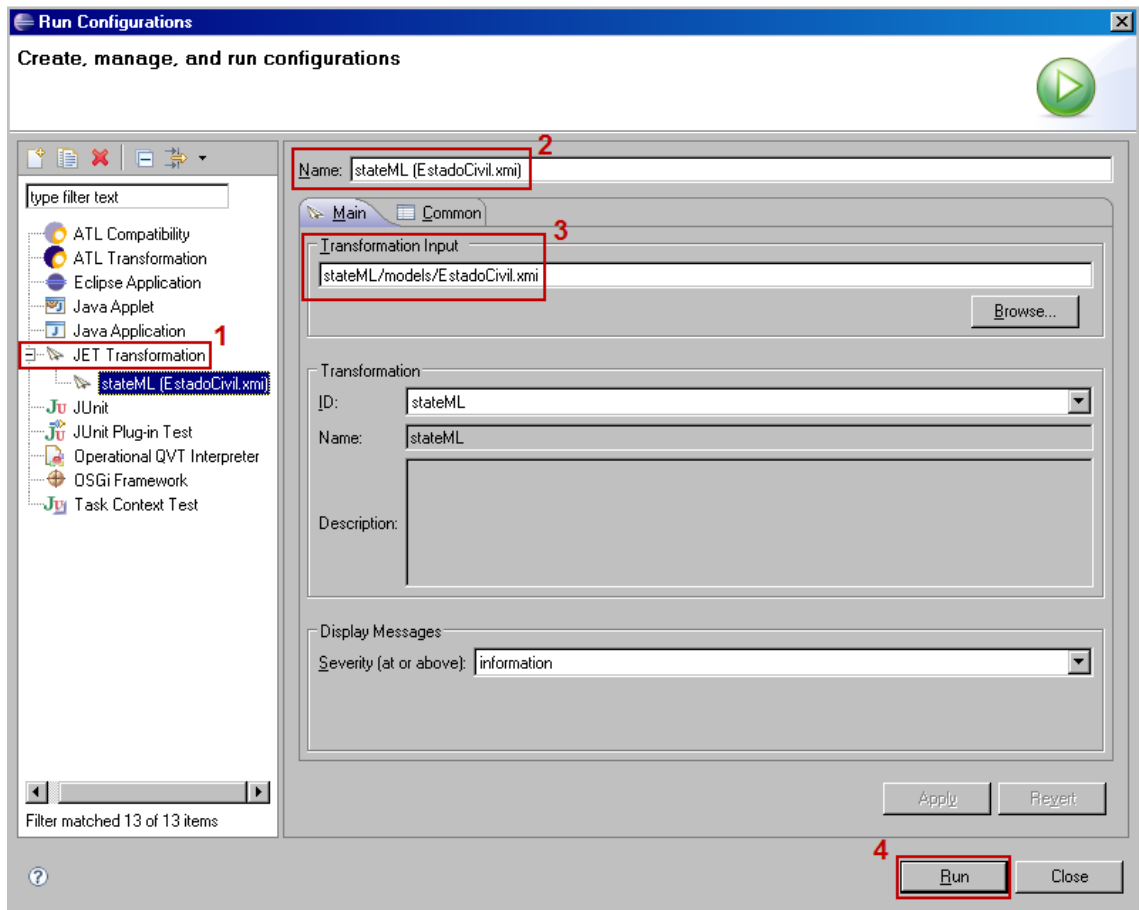


Figura 4.27. Ejecutar la transformación (ii).

#### 4.1.3.4. Nociones de programación en JET.

En *JET* se definen plantillas. Estas plantillas se usan para crear clases de implementación Java (que por defecto no se muestran en el *Package Explorer*) mediante un proceso denominado *translation*. Las clases *Java* se pueden utilizar para generar el *string* de salida con mediante el método *generate()* en el proceso denominado *generation*.

##### 4.1.3.4.1. Plantillas.

Las transformaciones en *JET* se describen mediante plantillas presentes por defecto en el directorio *templates* dentro del directorio de nuestro proyecto.

Lo habitual es utilizar para la transformación una plantilla *main* que llamará a su vez a otras plantillas *secundarias*. En esta plantilla se especificarán las plantillas secundarias que se utilizarán y como se llamará y donde se ubicará el archivo de salida de la transformación (el texto generado). En la *Figura 4.28*. se puede ver una plantilla *main* ejemplo.

```

<%@taglib prefix="ws" id="org.eclipse.jet.workspaceTags" %>
<c:setVariable var="org.eclipse.jet.taglib.control.iterateSetsContext"
select="true()"/>

<c:if test="isVariableDefined('org.eclipse.jet.resource.project.name')">
  <ws:file template="templates/dump.jet"
    path="{org.eclipse.jet.resource.project.name}/replicaProyecto3.vbs"/>
</c:if>

```

Figura 4.28. Ejemplo de plantilla main.

El esqueleto de esta plantilla se genera automáticamente al crear el proyecto *JET*. Dentro los *tags* `<c:if>` y `</c:if>` se deben especificar las plantillas de la transformación (mediante su *path* y nombre) y el archivo de salida que va a originar cada plantilla (también mediante su *path* y nombre) mediante el uso de la sentencia de la Figura 4.29.

```

<ws:file template="path/nombre_plantilla" path="path/nombre_archivo de salida"/>

```

Figura 4.29. Sentencia para controlar la generación de código.

La plantilla secundaria que define la transformación es la que se debe modificar para crear el código de salida deseado siguiendo las directrices expuestas en el siguiente epígrafe.

#### 4.1.3.4.2. Sintaxis *JET*.

El texto del archivo de salida se obtiene mediante la descripción presente en la plantilla que da origen a la transformación. Todo el texto “plano” que se incluya en dicha plantilla se incluirá en el archivo generado, de tal forma que se puede utilizar la sintaxis de *JET* para rellenar los huecos, iterar, tomar decisiones o hacer cálculo; logrando con ello una generación de código automatizada e inteligente.

Así, *JET* utiliza cuatro tipos de expresiones: directivas, declaraciones *java*, expresiones *java* y *scriptlets java*. Además se puede incluir comentarios.

Las **directivas** utilizan la sintaxis mostrada en la Figura 4.30. Así, son mensajes para el *engine* de *JET* y no se traducen en ningún tipo de código en el archivo de salida; únicamente afectan a como se traduce la plantilla.

```

<%@Directiva atributo1="valor1" atributo2="valor" ... %>

```

Figura 4.30. Sintaxis general de una directiva *JET*.



Las **declaraciones *java*** pueden contener cualquier sentencia *Java* entre los caracteres `<%!` y `%>`. Cada declaración debe contener una o más sentencias *Java* sintácticamente correctas. Se suelen utilizar para definir métodos a utilizar después mediante *scriptlets*.

Los ***scriptlets java*** pueden contener cualquier fragmento de código *Java*. Comienzan con `<%` y termina con `%>`. Cualquier salida que genere el código *Java* se plasmará en el archivo de salida, los *scriptlets* están concebidos para implementar lógica. Los *scriptlets* se ejecutan en la plantilla en tiempo de invocación. Que se produzca o no un resultado en el archivo de salida dependerá de dicha ejecución en dicho momento. Para usar *scriptlets* debe cumplirse que cuando todos los *scriptlets* presentes en la plantilla se pongan uno detrás de otro deberán formar un código *Java* con sentido como puede verse en la *Figura 5.2*. (de se generará un error en tiempo de ejecución). En ese código de ejemplo puede verse como utilizar sentencias *Java* y *scriptlets* para seleccionar entre dos posibles salidas en el archivo generado. Más adelante se verá cómo hacer esto usando los *tags* de *JET*.

```
<% if (Calendar.getInstance().get(Calendar.AM_PM) == Calendar.AM) {%>
Good Morning
<% } else { %>
Good Afternoon
<% } %>
```

*Figura 4.31. Ejemplo de scriptlet Java en JET.*

Las **expresiones *Java*** contienen expresiones *Java*. Las expresiones *Java* no son sentencias completas, en concreto no incluyen el punto y coma ( ; ) final. Las expresiones pueden acceder a cualquier elemento *Java* incluyendo campos y métodos declarados en declaraciones *java* o en *scriptlets java*. Comienzan con `<%=` y terminan con `%>`. El código *Java* resultante de la transformación (que posteriormente generará el código del archivo de salida) convertirá el resultado de la expresión a un *string* en caso de ser necesario. En la *Figura 4.32*. se muestra como obtener el nombre de la clase que se está ejecutando mediante el uso de expresiones.

```
<%= this.getClass().getName() %>
```

*Figura 4.32. Ejemplo de expresión Java en JET.*

Los **comentarios** en *JET* comienzan con `<!--` y finalizan con `-->`.

#### 4.1.3.4.3. Tags de JET.

JET incluye cuatro librerías de *tags*. Estas son:

- **Tags de control:** usadas para controlar la ejecución de la plantilla (el flujo de traducción) y para acceder al modelo fuente de la transformación.
- **Tags de formato:** usadas para alterar el texto en determinadas plantillas.
- **Tags Java:** útiles para generar código *Java*.
- **Tags de Workspace:** que se usan para acceder recursos del *workspace* como archivos, ficheros y proyectos.

Los **tags de control** más comunes son:

- **c:get**            `<c:get select="XPath_expression"/>`

Escribe el valor del modelo fuente indicado por *select* en el archivo de salida. Este valor deberá estar expresado en términos del estándar *XPath*.

- **c:set**            `<c:set select="XPath_expression" name="name">`  
                      valor del atributo  
                      `</c:set>`

Establece el atributo identificado por *name* del objeto seleccionado por la expresión *XPath*. El valor se determina por las expresiones contenidas entre los *tags*.

- **c:iterate**        `<c:iterate select="XPath_expression" var="name">`  
                      código iterado  
                      `</c:iterate>`

Recorre los elementos *select* (identificados por una expresión *XPath*) y ejecuta el código encerrado dentro de los *tags* por cada elemento. En cada iteración, se podrá acceder al elemento iterado con el identificador indicado en *var*.

- **c:if**             `<c:if test="value">`  
                      expresiones que se ejecutan si test==true  
                      `</c:if>`

Ejecuta el texto contenido entre los *tags* si el valor de la expresión *test* es igual a *true*. Si *test* es una expresión *XPath* se convierte según el estándar a *boolean*.

Hay muchos otros *tags*, pero para escribir plantillas sencillas estos son los más relevantes.

#### 4.1.3.4.4. Estándar *XPath*.

*JET* implementa el estándar *XPath 1.0* con pequeñas excepciones. Aunque el lenguaje *XPath* es rico, solo se requiere una pequeña parte para utilizar la mayoría de los *tags* de *JET*.

*XPath* (y por ende *JET*) asume que se está trabajando en un modelo que tiene estructura de árbol. Esto querrá decir que mediante *XPath* es posible ir moviéndose a lo largo del modelo fuente de la transformación para llegar a todos los nodos y propiedades. *XPath* además soporta expresiones matemáticas, booleanas y comparativas; así como llamadas a funciones.

La idea básica de *XPath* es el ***path expression***, que son similares los *paths* de cualquier S.O.; A destacar:

- Un *path* son una serie de pasos separados por barras (/).
- Los pasos se evalúan de izquierda a derecha.
- Cada paso identifica un nodo del árbol o una opción del lenguaje (como dos puntos para subir en el árbol, por ejemplo).
- Los pasos pueden tener una condición al final se escribe entre corchetes ([ y ]).
- Una barra inicial (/) indica que se parte de la raíz del modelo.
- Los *path* pueden empezar con una variable, cuyo nombre debe de ir precedido del símbolo *dollar* (\$).

Otro concepto importante es el concepto de **nodo**. Hay dos tipos básicos de nodos en el estándar *XPath*:

- **Nodo Raíz:** El nodo raíz es el nodo a partir del cual aparecen todos los demás. Se identifica con /. El nodo raíz del árbol contiene el elemento raíz del árbol.
- **Nodo elemento:** Cualquier elemento del modelo se convierte en un nodo elemento del árbol. Cada nodo elemento tiene su nodo padre. El nodo padre de cualquier elemento es, a su vez, un elemento; a excepción del elemento raíz cuyo padre es el nodo raíz. Los nodos elemento tienen a su vez hijos, que son: nodos elemento, nodos texto, nodos comentario y nodos de instrucciones de proceso. Los nodos elemento también tienen propiedades tales como su nombre, sus atributos e información sobre los “espacios de nombre” que tienen activos.

Las expresiones que utiliza *XPath* para navegar por el árbol se denominan **axes**. Los más relevantes son:

- **Child**                    /nodoPadre/nodoHijo

Se corresponde con la barra (/), aunque su sintaxis larga es /child:: . Se utiliza para seleccionar todos los nodos del tipo *nodoHijo* del *nodoPadre*.

- **Attribute**     / nodoPadre/nodoHijo/@Atributo  
                  / nodoPadre/nodoHijo[@Atributo]/\*  
                  / nodoPadre/nodoHijo[@Atributo=valor]

Se corresponde con la arroba ( @ ), aunque su sintaxis larga es *attribute::* . Sirve para seleccionar los nodos atributos utilizando su nombre. La arroba se puede utilizar para añadir un predicado a un paso del *path* de tal forma que se elija solamente los nodos hijo que tengan un determinado atributo o cuyo atributo tenga un determinado valor.

- **Descendant**   / nodoPadre//nodoHijo  
                  / nodoPadre//\*

Se corresponde con la doble barra ( // ), aunque su sintaxis larga es *descendant::* . Sirve para seleccionar a todos los hijos del nodo textual (los hijos de tipo *nodoHijo*, los hijos de los hijos de tipo *nodoHijo*, los hijos de los hijos de los hijos de tipo *nodoHijo*, etc.). Se pueden seleccionar todos los elementos que cuelgan de un nodo utilizando la doble barra y el asterisco.

- **Self**           ./

Se corresponde con el punto ( . ), y sirve para seleccionar el nodo contexto.

- **Parent**        / nodoPadre/..

Se corresponde con los dos puntos ( .. ), y realiza un paso hacia arriba en el árbol. Puede utilizarse sin problema como paso dentro de un *path* complejo para seleccionar hermanos del nodo contextual o descendientes de hermanos.

## 4.2. Uso de Makros y ETS3.

### 4.2.1. Nociones sobre la programación de macros.

La primera reseña que se ha de hacer sobre la programación de macros se deriva de manera directa del lenguaje de programación utilizado para ello: *Visual Basic Script (VBS)*. Se han de tener claras las nociones y conceptos de programación en este lenguaje, haciendo especial hincapié en el manejo de colecciones de objetos y el acceso a propiedades de objetos.

Esto es debido al *Domain Object Model (EtecDom)* que utiliza el *Macro Runtime Environment* para representar a través de objetos *entities* en el dominio *EIB*. Esto se realizará a través de objetos (con propiedades y métodos) y colecciones de objetos. Es providencial tener claro el funcionamiento de este *DOM*, ya que se deberá recorrer en las macros en busca de los objetos correctos a modificar para efectuar los cambios necesarios en el proyecto.

El *EtecDom* se descompone en tres dominios que describen tanto los dispositivos en la base de datos de *ETS (Manufacturer Domain, Figura 4.33.)*, las propiedades y los objetos de los proyectos (*Project Domain, Figura 4.34.*) y el dominio de *EIB* que da soporte a los dos dominios anteriores (*EIBA Domain, Figura 4.35.*).

En los diagramas de la *Figura 4.33.* a la *Figura 4.35.* en azul están los objetos del dominio y en amarillo las colecciones. Así, en estos diagramas de árbol se ven las relaciones entre los objetos que describen la base de datos de *ETS*. Hay colecciones de objetos, que tienen propiedades y que referencian otras colecciones y/o objetos (que a su vez pueden referenciar a otras colecciones y otros objetos, y que a su vez tienen también sus propiedades). Estas referencias se pueden apreciar muy bien en estos diagramas. Así, se ha de tener claro la manera en que se describe la funcionalidad de *ETS* en este *DOM*, para lo cual se han de tener más que asumidos los conceptos del punto uno y el punto dos de este capítulo; además de utilizar la ayuda del programa para ver las propiedades y objetos/colecciones asociadas a cada objeto del dominio.

El principal problema de la base de datos es el no poder visualizar de manera directa que hay exactamente en ella, ya que no se dispone de ninguna herramienta para ello. Este problema deberá de ser solucionado mediante la programación de macros para obtener las características de los dispositivos o para obtener lo que contienen las colecciones. Estas macros serán denominadas *macros auxiliares*.

Además, como la finalidad última de este *TFM* es desarrollar un sistema de diseño basado en plantillas, las macros que se programen deberán de cumplir una serie de requisitos que lo permitan; a saber:

- Separar las inicializaciones de variables del resto del código.
- Usar nombres únicos para todas variables.
- Diferenciar claramente el segmento de código que una sub-capa superior de la aplicación debería ir copiando

- Definir claramente las variables que dicha subcapa debe modificar

En el *Capítulo 5 Apartado Primero* se explica mediante ejemplos más en profundidad tanto las Makros programadas para la aplicación como los conceptos metodológicos para programarlas.

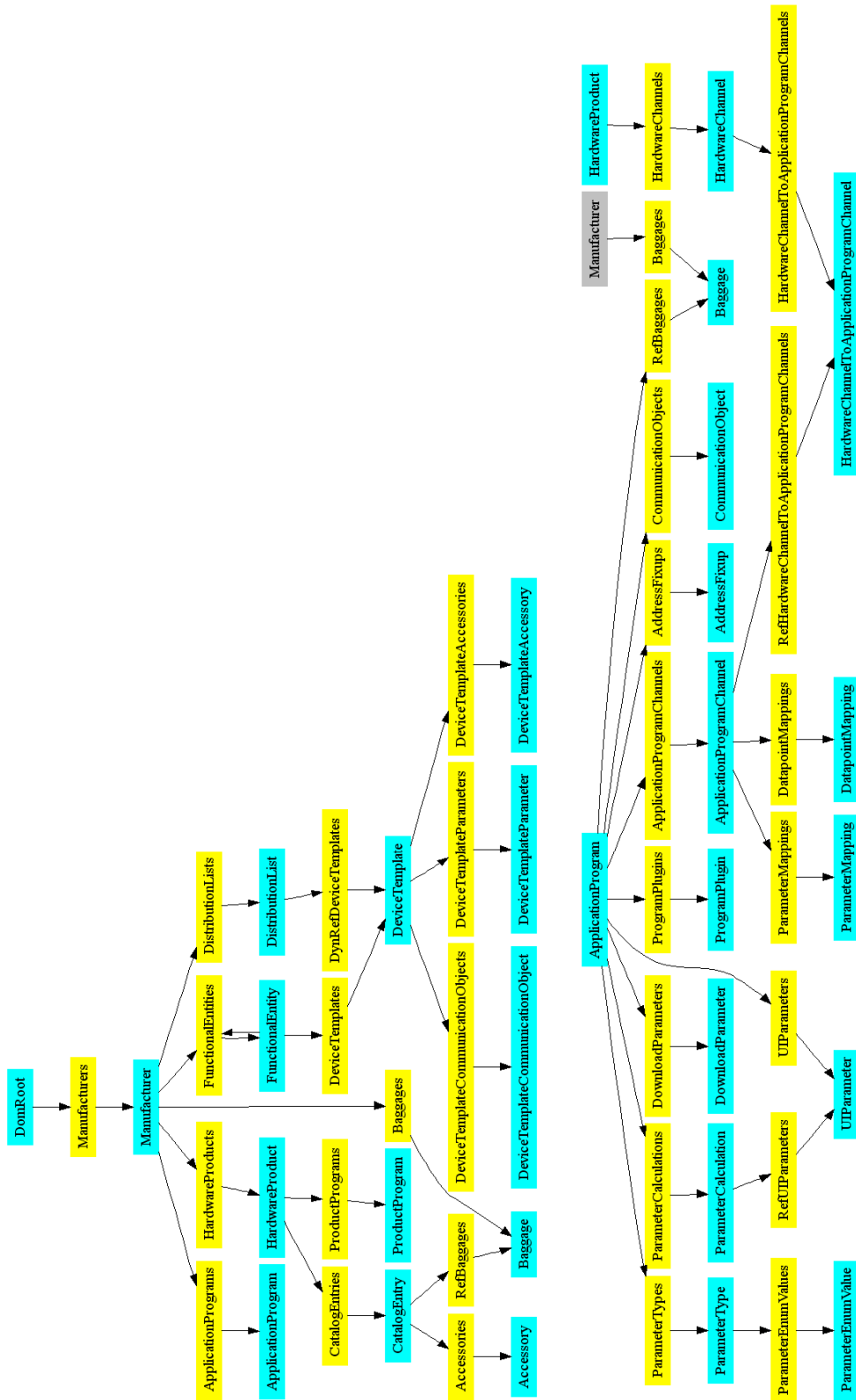


Figura 4.33. Manufacturer Domain

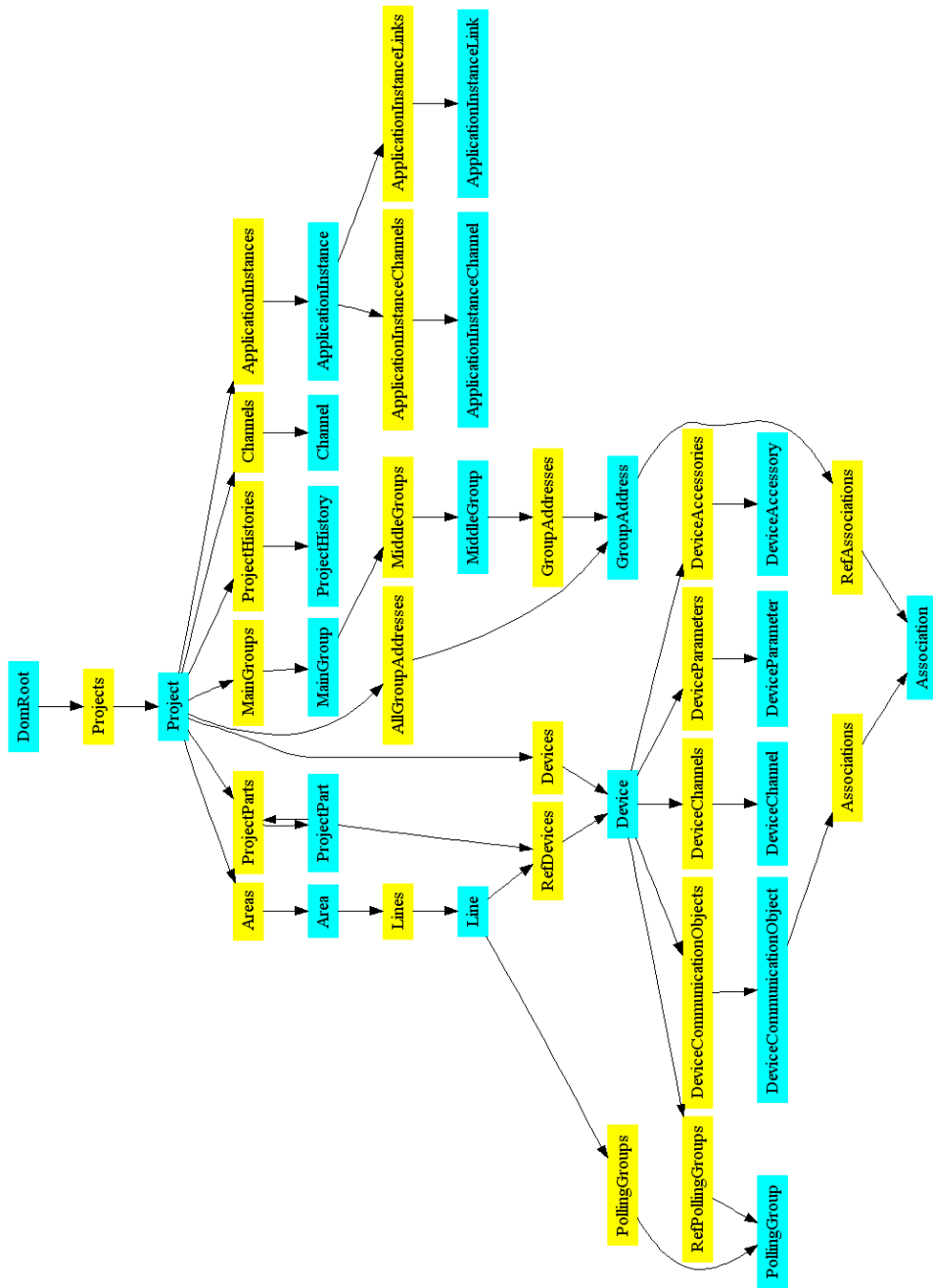


Figura 4.34. Project Domain



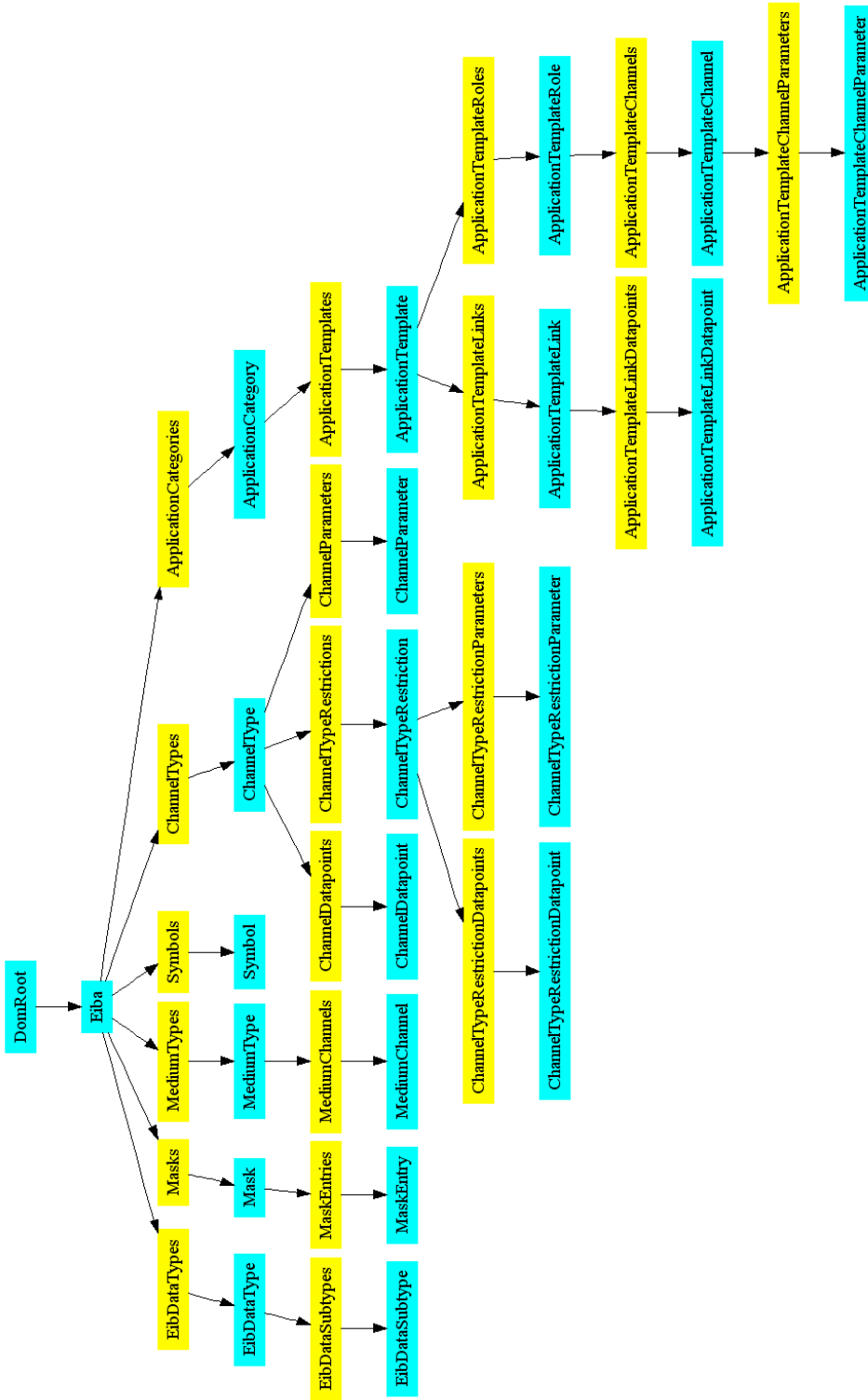


Figura 4.35. EIBA Domain



## ***Trabajos realizados***

---

En este capítulo se van a detallar en primer lugar los desarrollos acometidos para conseguir automatizar el desarrollo de proyectos domóticos. En primer lugar se van a analizar las macros programadas para *ETS3* así como en la plantilla *JET* programada. Después se describirá la metodología de desarrollo propuesta, que será ilustrada finalmente con un proyecto ejemplo.



## 5.1. Macros programadas.

En este apartado vamos a describir las Macros de *ETS3* programadas para poder luego ser utilizadas en la plantilla *JET* con la finalidad de poder trasladar de manera automática los proyectos descritos usando el *DSL* del *Apartado 2.3.2* a dicho entorno *ETS3*. Los fundamentos de la programación de la misma ya han sido relatados en este documento (*Capítulo 4, Apartado 4.2*), a pesar de lo cual no está de más hacer hincapié en algunos aspectos fundamentales. En primer lugar, estas Macros son fruto de un trabajo de ingeniería inversa sobre la base de datos del entorno *ETS3*, en la cual se almacenan todas las características tanto del proyecto bajo desarrollo como de los dispositivos disponibles para realizarlo. En segundo lugar, y como ya se ha comentado en capítulos anteriores, están programadas en *Visual Basic Script* y contienen tanto conceptos propios del dominio domótico como otros surgidos *ad hoc* fruto de la inferencia de los contenidos de la base de datos y los mecanismos diseñados para trabajar con ella. Por último, decir que las figuras que describen como se almacena la información en las bases de datos de *ETS3* (*Figura 4.33, Figura 4.34 y Figura 4.35*) van a ser claves a la hora de entender las macros mostradas a continuación.

Estas macros se pueden clasificar de manera general en dos categorías: las Macros utilizadas para modificar el proyecto de *ETS3* y las Macros auxiliares programadas para obtener la información necesaria para realizar la labor ingeniería inversa sobre la base de datos de *ETS3*. Las macros más relevantes de ambos conjuntos se encuentran recopiladas en el *Anexo I*.

### 5.1.1. Estructura general.

Las macros programadas de más valor son las macros para modificar proyecto. Todas estas macros siguen una estructura particular, fruto del orden lógico de acciones a seguir para implementarlas y fruto también de la necesidad de que sean validas para su uso en plantillas.

En primer lugar, hay que decir que al principio de cada macro hay un comentario que da todos los datos relativos a la misma en cuanto a su funcionamiento, la versión, requerimientos para su utilización, etc. Un ejemplo de estos comentarios extraídos de la macro *Modificar dirección física* (la cual nos va a servir para ilustrar la estructura general de las macros) puede verse en la *Figura 5.1*.

```
'David Montoro Mouzo
'2009
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
'
'Macro para modificar la dirección física de un dispositivo
'
'OJO: Para que "se nos permita" por parte del entorno modificar la dirección
'física el dispositivo ha de estar añadido a una línea
'
'Funcionando OK, pero hay que modificar el macro AnadirDispositivo para que
'los dispositivos se añadan a una línea --> Usar v3 de esta macro!!
'
'Versión para plantilla
'-----
'-----
'-----
```

Figura 5.1. Comentario inicial de la macro Modificar dirección física.

A partir de aquí, como ya se ha comentado, todas las macros siguen un patrón estructural similar definido para hacer posible su utilización en la plantilla de *JET* que implementa la transformación *M2T*. Esta estructura se compone de cuatro módulos principales: *inicialización de las variables*, *asignación de valores a las variables entrada*, *asignación de valores a las variables auxiliares* y *lógica funcional de la macro*. En este epígrafe se explicará cada una de estos módulos ilustrándolas con la Macro *Añadir dispositivo* presente en el Anexo I.

En el módulo '*inicialización de variables*' se inicializan las variables que se van a utilizar en la macro. Se separa la inicialización en dos: variables a modificar (que serán las variables que describirán los parámetros de entrada para la macro) y variables auxiliares (que son variables auxiliares necesarias para la lógica). Se inicializan todas con un nombre único que da idea de su función en la macro, ya que estará compuesto por el nombre de la macro al que se le concatena un nombre descriptivo de la función de la variable en el código. Es importante que los nombres sean únicos no solo dentro de una macro, sino dentro del conjunto de las macros; ya que de lo contrario no sería posible utilizar estas macros para generar una macro que genere un proyecto de manera automatizada utilizando la plantilla *JET*. La parte '*inicialización de variables*' de la macro *Añadir dispositivos* puede verse en la *Figura 5.2*.

```

'-----
'Iniciación de variables del macro AnadirDispositivo
'-----

'variables a modificar
Dim AnadirDispositivo_NombreReferencia
Dim AnadirDispositivo_DireccionFisica
Dim AnadirDispositivo_Fabricante

'variables auxiliares
Dim AnadirDispositivo_AuxiliarYaAnadido
Dim AnadirDispositivo_AuxiliarEncontrado
Dim AnadirDispositivo_AuxiliarNuevoDispositivo
Dim AnadirDispositivo_AuxiliarNombre
Dim AnadirDispositivo_AuxiliarFabricante
Dim AnadirDispositivo_AuxiliarReferencia
Dim AnadirDispositivo_auxDR
Dim inc
Dim numeroEntitys

```

Figura 5.2. Módulo 'inicialización de variables' de la macro Añadir dispositivos.

En el módulo 'asignación de valores a variables' entrada se introducen en la macro los datos necesarios para obtener de manera exacta y precisa la modificación que buscamos en el proyecto de *ETS3* (ya sea añadir un dispositivo en concreto, modificar un parámetro específico de un dispositivo, etc.). En todas las macros se indica en un comentario situado en la línea inmediatamente superior a cada asignación cuál es el contenido y formato de la especificación requerida. Es importante comprender bien estas asignaciones ya que serán las que a posteriori se realizarán de manera automática mediante la transformación *M2T* definida en *JET* a partir de la descripción del proyecto domótico realizada utilizando el *DSL*. La parte 'asignación de valores de entrada' de la macro *Añadir dispositivos* puede verse en la *Figura 5.3*.

```

'Valores a las variables del macro AnadirDispositivo
'-----

'Poner aquí el nombre (nombre del catalogo) de el dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Actuador de 6 salidas"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 33

```

Figura 5.3. Módulo 'Asignación de valores a variables entrada' de la macro Añadir dispositivos.

El módulo ‘*asignación de valores a las variables auxiliares*’ contiene asignaciones a variables auxiliares necesarias para la lógica funcional de la macro. Estas asignaciones se encuentran ya especificadas en las macros programadas, y no se deben modificar. Así, tampoco se modificarán con la transformación definida en *JET*. A pesar de todo, no se deben eliminar de la macro final porque inicializan las variables necesarias para lógica de la macro de tal manera que el funcionamiento y el resultado obtenido sobre el proyecto *ETS3* sean los correctos. La parte ‘*asignación de variables auxiliares*’ de la macro *Añadir dispositivos* puede verse en la *Figura 5.4*.

```
'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
```

Figura 5.4. Módulo ‘*Asignación de valores a variables auxiliares*’ de la macro *Añadir dispositivos*.

El último módulo presente en todas las macros es el módulo *lógica de la macro*, que es el que implementa las modificaciones en la base de datos de *ETS3* siguiendo los procedimientos descritos en el *Capítulo 4, Apartado 4.2*. Cada macro implementa una funcionalidad determinada, descrita al principio de la misma. Esta parte de la macro *Añadir dispositivos* puede verse en la *Figura 5.6*.

```
for each manufacturer in AnadirDispositivo_auxDR.manufacturers
for each functionalEntity in manufacturer.FunctionalEntities
if AnadirDispositivo_AuxiliarYaAnadido = false then
AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then
numeroEntities = functionalEntity.FunctionalEntities.count
for inc=1 to numeroEntities step 1
AnadirDispositivo_AuxiliarReferencia = LCase(CStr(FunctionalEntity.FunctionalEntities.Item(Cint(inc)).name))
AnadirDispositivo_AuxiliarEncontrado = InStrRev(AnadirDispositivo_AuxiliarReferencia,AnadirDispositivo_NombreReferencia)
if (AnadirDispositivo_AuxiliarEncontrado > 0) then
Set AnadirDispositivo_AuxiliarNuevoDispositivo = Project.Devices.Add(FunctionalEntity.FunctionalEntities.Item(Cint(inc)).DeviceTemplates.Item(1))
AnadirDispositivo_AuxiliarNuevoDispositivo.Name = AnadirDispositivo_AuxiliarNombre
Set AnadirDispositivo_AuxiliarNuevoDispositivo.line = Project.Areas.Item(2).Lines.Item(2)
AnadirDispositivo_AuxiliarNuevoDispositivo.Address = AnadirDispositivo_DireccionFisica
AnadirDispositivo_AuxiliarYaAnadido = true
end if
Next
end if
end if
Next
Next
```

Figura 5.6. Módulo ‘*Lógica de la macro*’ de la macro *Añadir dispositivos*.

Además, cabe reseñar que las inicializaciones de parámetros se separan del resto de la macro utilizando un comentario como el que se muestra en la *Figura 5.6., izquierda*; y que la macro se finaliza con el comentario *Figura 5.6., derecha*. Esto es para facilitar la identificación de la parte de la macro que va al principio de la macro global que generará transformación definida en *JET* (el módulo de inicializaciones), y para identificar rápidamente a que funcionalidad responde cada módulo de código tanto en la plantilla *JET* como en la macro global generada.



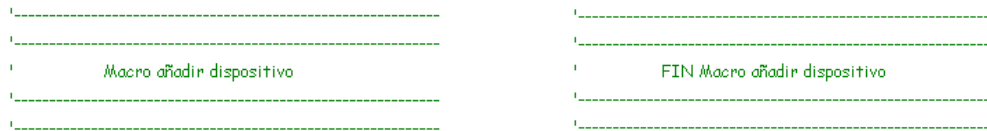


Figura 5.6. Delimitadores en las macros. Izquierda: delimitador de inicio; derecha: delimitador de fin.

### 5.1.2. Macros funcionales.

En este epígrafe se va a analizar las macros programadas para modificar el proyecto. Se detallarán su funcionamiento y los argumentos de entrada que utilizan.

#### 5.1.2.1. Macro añadir dispositivo.

Esta macro permite añadir un dispositivo presente en la base de datos *ETS3* a nuestro proyecto. El dispositivo se busca en la base de datos en función al fabricante y a su nombre y luego se añade. Los parámetros que toma como entrada pueden verse en la *Figura 5.3*. En *AnadirDispositivo\_NombreReferencia* se debe incluir el nombre del dispositivo total o parcialmente. En *AnadirDispositivo\_Fabricante* se debe especificar el nombre del fabricante de manera exacta. En *AnadirDispositivo\_DireccionFisica* se debe especificar la dirección física que va a tener el nuevo dispositivo añadido al proyecto. Todos los dispositivos se añaden por defecto a la primera área, primera línea del proyecto; ya que el *DSL* no contempla áreas y líneas en esta primera versión.

#### 5.1.2.2. Macro modificar dirección física.

Esta macro sirve para modificar la dirección física de un dispositivo añadido al proyecto. No se utiliza en la plantilla de *JET* pero se ha programado para posibles usos futuros. Es importante reseñar que para que esta macro arroje un resultado correcto el dispositivo debe estar añadido a una línea del proyecto (para poder ‘buscarlo’ en la base de datos). Los parámetros de entrada de esta macro pueden verse en la *Figura 5.7*. Es necesario especificar el nombre del dispositivo del que queremos modificar la dirección (*ModificarDireccionFisica\_nombreDispositivo*) y el valor de la nueva dirección física (*ModificarDireccionFisica\_valorDireccion*).

```
'Valores a las variables de el macro ModificarDireccionFisica
'-----

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarDireccionFisica_nombreDispositivo = "nuevoDispositivo entrada binaria"
ModificarDireccionFisica_nombreDispositivo = lcase (ModificarDireccionFisica_nombreDispositivo)

'Poner aquí el valor de la nueva dirección
ModificarDireccionFisica_valorDireccion = 21
```

Figura 5.7. Parámetros de entrada para la macro ‘Modificar dirección física’.

### 5.1.2.3. Macro modificar parámetro.

Un punto clave de un proyecto en *ETS3* es variar el comportamiento de los dispositivos configurando sus parámetros. Para poder hacer esto mediante una macro, se programó la macro modificar parámetro. Esta macro modifica los parámetros de un dispositivo añadido al proyecto y toma como entrada las variables mostradas en la *Figura 5.8*. Se debe especificar el nombre del dispositivo al que pertenece el parámetro (*ModificarParametro\_nombreDispositivo*), el nombre del parámetro de manera exacta (*ModificarParametro\_nombreParametro*), el valor que se quiera dar al parámetro (*ModificarParametro\_valorParametro*) y el orden de aparición del parámetro dentro de la base de datos de *ETS3* (*ModificarParametro\_ordenAparicion*). El orden de aparición se incluye debido a que dentro de los parámetros de un dispositivo hay varios parámetros con el mismo nombre y características, pero solo hay uno que cambia las características comportamentales para cada configuración de parámetros. Este tema se analizará más en detenimiento más adelante en este capítulo cuando se explique cómo implementar un proyecto utilizando estas macros (*Apartado 5.4* del presente capítulo), pero en general hay que decir que es necesario hacer un estudio previo de los parámetros de un dispositivo para utilizar esta macro.

```
'Valores a las variables de el macro #ModificarParametro
'-----

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = "entrada binaria"
ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-.
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 40
```

*Figura 5.8. Parámetros de entrada para la macro 'Modificar parámetro'.*

### 5.1.2.4. Macro añadir dirección de grupo.

Esta macro añade una dirección de grupo al proyecto. Esta macro es importante, ya que es necesario tener añadidas las direcciones de grupo al proyecto *ETS3* para poder enlazar los objetos de comunicación, los cuales son los que implementan realmente las comunicaciones entre dispositivos. Las variables de entrada que toma esta macro se pueden ver en la *Figura 5.9*. Hay que especificar valores para los tres campos de la dirección (se suponen direcciones de tres niveles) y el nombre de la dirección de grupo que se quiere añadir.

```
'Valores a las variables de el macro AnadirDireccionGrupo
'-----

AnadirDireccionGrupo_1erDigito = 2
AnadirDireccionGrupo_2oDigito = 4
AnadirDireccionGrupo_3erDigito = 7
AnadirDireccionGrupo_NombreDireccion = "nombre de la dirección"
```

Figura 5.9. Parámetros de entrada para la macro 'Añadir dirección de grupo'.

#### 5.1.2.5. Macro enlazar dirección de grupo.

Para implementar la comunicación entre dispositivos se debe asociar a los objetos de comunicación precisos a las direcciones de grupo adecuadas. Además, se debe indicar si se va a enviar información o si solo se va escuchar. Para esto se programó la macro enlazar dirección de grupo que toma como entrada los parámetros de la *Figura 5.10*. Se deben especificar el nombre del dispositivo que contenga el objeto de comunicación que queremos enlazar (*ModificarDireccionGrupo\_nombreDispositivo*), el número de objeto de comunicación que queremos enlazar (*ModificarDireccionGrupo\_numeroObjetoComunicacion*) y la dirección de grupo a la que queremos enlazarlo (mediante sus tres componentes). Además, se especifica si solo se escucha o si también se envía en la variable *ModificarDireccionGrupo\_SendDireccionGrupo*. Importante resaltar que será necesario un estudio de los objetos de comunicación previo a la utilización de esta macro, de manera similar a el caso de los parámetros.

```
'Valores a las variables de el macro ModificarDireccionGrupo
'-----

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarDireccionGrupo_nombreDispositivo = "actuador"
ModificarDireccionGrupo_nombreDispositivo = lcase (ModificarDireccionGrupo_nombreDispositivo)

'Poner aquí el número de objeto de comunicación a modificar
ModificarDireccionGrupo_numeroObjetoComunicacion = 4

'Poner aquí la dirección de grupo
ModificarDireccionGrupo_1 = 1
ModificarDireccionGrupo_2 = 1
ModificarDireccionGrupo_3 = 1

'Si queremos que sea send o no (booleano = [true, false])
ModificarDireccionGrupo_SendDireccionGrupo = true
```

Figura 5.10. Parámetros de entrada para la macro 'Enlazar dirección de grupo'.

### 5.1.3. Macros auxiliares.

Además de las macros con funcionalidad explícita y capacidad de modificar el proyecto, se programaron una serie de macros auxiliares con la finalidad principal de obtener información de la base de datos de *ETS3*. Esta información fue la que a la postre se utilizó para poder implementar macros que permitiesen modificar el proyecto de una manera adecuada. Además, se implementaron macros para obtener toda la información del proyecto y de los dispositivos y poder utilizarla para el análisis paramétrico o de los objetos de comunicación. Aunque se programaron muchas, se analizan las tres más importantes.

#### 5.1.3.1. Macro obtener el nombre de todos los dispositivos en la base de datos.

Esta macro crea un fichero de texto por cada fabricante en la base de datos en el cuál se almacenan los nombres de todos los dispositivos asociados a dicho fabricante presentes dicha base de datos. Estos nombres de dispositivos y fabricantes sirvieron para poder implementar en las macros que modifican el proyecto las búsquedas. Además, esta macro se utilizó con mínimas modificaciones para obtener información adicional sobre los dispositivos presentes en la base de datos de *ETS3*. Es importante reseñar que los ficheros se crean en la ruta "*C:\Info dispositivos\*", por lo que para utilizar esta macro se debe o bien crear una carpeta con el nombre '*Info dispositivos*' en la unidad *C:* o bien cambiar la ruta en la macro. Esta macro no requiere configuración adicional a parte de esta para poder ejecutarla.

#### 5.1.3.2. Macro obtener información de los dispositivos agregados al proyecto.

Esta macro se programó para poder realizar los análisis paramétricos y de objetos de comunicación previos a la implementación de una macro que implementara un proyecto. Esta macro escribe en un fichero ubicado en "*C:\Info dispositivos\*" toda la información de cada uno de los dispositivos agregados al proyecto en lo referente a nombre, fabricante, entrada de catálogo, dirección física, objetos de comunicación, direcciones de grupo, áreas, líneas, parámetros, etc. Mismas reseñas que para la macro anterior en términos de la ruta donde se crea el archivo y en cuanto a la no necesidad de una configuración previa.

#### 5.1.3.3. Macro obtener información del dispositivo activo.

A la hora de hacer un análisis paramétrico o de los objetos de comunicación de un dispositivo, es mejor obtener la información solo de ese dispositivo que la de todos los presentes en el proyecto, a fin de que el análisis sea más sencillo y se puedan obtener conclusiones de una manera más eficiente. Con esta finalidad se programó una macro que obtenía la misma información que la macro del apartado anterior, pero solamente para el dispositivo seleccionado en ese momento. Por tanto, para obtener la información de un dispositivo en concreto se deberá seleccionar haciendo *click* con el ratón sobre él y después se deberá ejecutar esta macro. Entonces se creará un archivo en la ruta "*C:\Info dispositivos\*"

con la información relativa a dicho dispositivo. Se deberán tener en cuenta las mismas consideraciones respecto a la no configuración y la ruta que para las dos macros anteriores.

#### 5.1.4. Implementación de un proyecto *ETS3* utilizando macros.

Con la finalidad de clarificar en lo máximo posible el uso de las macros que se acaban de describir, se va a realizar un pequeño ejemplo en el que se va a crear un proyecto de *ETS3* desde cero utilizando las macros aquí expuestas. Así, las usaremos como base para crear una macro global que tras su ejecución modificará un proyecto en blanco de *ETS3* de tal manera que el resultado final será el proyecto buscado. El proyecto se va a generar no tiene una funcionalidad concreta, por lo que todo lo que a continuación se expone debe ser visto solamente desde el plano instructivo: es un ejemplo sencillo pero que requiere realizar todas las acciones que serían necesarias para implementar un proyecto más grande y con una funcionalidad definida. La macro resultante se incluye en el Anexo I a modo ilustrativo.

El proyecto se va a realizar puede verse en la *Figura 5.11*. Vemos como se compone de dos dispositivos (un actuador de accionamiento con dirección física 1.1.4 y una entrada binaria con dirección física 1.1.11) y de dos direcciones de grupo (1/1/14 - L04 ON/OFF y 1/1/15 - L04 ESTADO). En la *Figura 5.12*. y *Figura 5.13*. se aprecian en más detalle los objetos de comunicación de los dispositivos asociadas a cada una de estas direcciones. Además, una descripción de los parámetros que se desean configurar para cada uno de los dispositivos puede verse en la *Tabla 5.1*.

The screenshot shows a software interface for project replication. The main window is titled 'Topología en Proyecto a replicar'. It features a tree view on the left and a table on the right. The tree view shows a hierarchy: 'Proyecto a replicar' -> '1 Áreal' -> '1.1 Lineal' -> '1.1.4 ACT2 Actuador accionamiento, 6 salidas, 6A DIN' and '1.1.11 IP3 P10-P12-P11-P09 Entrada binaria, 4 canales'. The table on the right has columns: 'Dirección', 'Habitación', 'Función', and 'Descripción'. It lists two entries: '1.1.4 Cuadro\_Inf... ACT2' and '1.1.11 Panel\_Supe... IP3 P10-P12-P...'. Below this, there is another section titled 'Direcciones de Grupo en Proyecto a replicar' with a tree view showing 'Grupos Principales' -> '1 Planta baja' -> '1 Iluminación' -> '14 L04 ON/OFF' and '15 L04 ESTADO'. To the right of this is another table with columns 'Objeto' and 'Aparato', listing '16: Salida 5 - Reenvío de estado' and '2: Entrada 3 - Objeto accionamiento 3.1'.

Dirección	Habitación	Función	Descripción
1.1.4	Cuadro_Inf...		ACT2
1.1.11	Panel_Supe...		IP3 P10-P12-P...

Objeto	Aparato
16: Salida 5 - Reenvío de estado	1.1.4 ACT2 Actuador accionamiento, 6 salidas, 6A DIN
2: Entrada 3 - Objeto accionamiento 3.1	1.1.11 IP3 P10-P12-P11-P09 Entrada binaria, 4 canales

*Figura 5.11. Proyecto a replicar.*

Objeto	Aparato	Envío	ACK (PL)	C	R
4: Salida 5 - Accionar	1.1.4 ACT2 Actuador accionamiento, 6 salidas, 6A DIN	S		C	-
2: Entrada 3 - Objeto accionamiento 3.1	1.1.11 IP3 P10-P12-P11-P09 Entrada binaria, 4 canales	S		C	-

Figura 5.12. Objetos de comunicación asociados a la dirección de grupo 1/1/14.

Objeto	Aparato	Envío	ACK (PL)	C	R
16: Salida 5 - Reenvío de estado	1.1.4 ACT2 Actuador accionamiento, 6 salidas, 6A DIN	S		C	-
2: Entrada 3 - Objeto accionamiento 3.1	1.1.11 IP3 P10-P12-P11-P09 Entrada binaria, 4 canales			C	-

Figura 5.13. Objetos de comunicación asociados a la dirección de grupo 1/1/15.

Dispositivo	Parámetro	Valor
<b>Actuador accionamiento 6 salidas</b>	Salida 5, "reacción después de fallo en la alimentación"	Ninguna
	Salida 5, "comportamiento al regreso de la tensión"	Abrir contacto
	Salida 5, "Temporización"	Ninguna
	No invertir	Salida 5, "Reenvío de estado"
	Salida 5, "Modo del relé"	Contacto normalmente abierto
<b>Entrada binaria 4 canales</b>	Entrada 3, función entrada 3	Accionar
	Entrada 3, comando al flanco ascendente.	Alternado
	Entrada 3, comando al flanco descendente	Sin reacción

Tabla 5.1. Parámetros de los dispositivos.

Antes de poder describir el proyecto utilizando las macros, hay que realizar un análisis para obtener todos los valores de las variables de entrada necesarios. Así, deberemos ver exactamente cuáles son los nombres de los dispositivos en la base de datos *EST3*, el fabricante de cada uno de ellos, su dirección física, los nombres de los parámetros que hay que modificar (también en la base de datos) y su orden de aparición, las direcciones de grupo que hay que añadir al proyecto, el número de los objetos de comunicación que hay que asociar a cada una de ellas y si se asocian como *sending* o no, etc. Para estos estudios se recomienda ser metódicos y precisos, ya que si no se recopilan bien estos datos el proyecto no se generará correctamente. Sirvan como ejemplo de cómo recopilar los datos de manera adecuada las tablas de *Tabla 5.2.* a *Tabla 5.5.*

Dispositivo	Nombre ETS3	Fabricante	Dirección física
<b>Actuador accionamiento</b>	Actuador de 6 salidas	Albrecht Jung	4
<b>Entrada binaria</b>	Entrada binaria, 4-canales	Albrecht Jung	6

Tabla 5.2. Información de los dispositivos en el proyecto.

Dispositivo	Parámetro	Valor	Por defecto	Nombre en ETS3	Valor ETS3	Orden aparición
<b>Actuador accionamiento</b>	Salida 5, "reacción después de fallo en la alimentación"	Ninguna	Sí	-	-	-
	Salida 5, "comportamiento al regreso de la tensión"	Abrir contacto	Sí	-	-	-
	Salida 5, "Temporización"	Ninguna	Sí	-	-	-
	Salida 5, "Reenvío de estado"	No invertir	No	Rückmeldung	0	2
	Salida 5, "Modo del relé"	Contacto normalmente abierto	Sí	-	-	-
<b>Entrada binaria</b>	Entrada 3, función entrada 3	Accionar	Sí	-	-	-
	Entrada 3, comando al flanco ascendente.	Alternado	No	Befehl_steigendeFlanke_O1	3	13
	Entrada 3, comando al flanco descendente	Sin reacción	No	Befehl_fallendeFlanke_O1	0	40

Tabla 5.3. Análisis de los parámetros de los dispositivos.

Nombre direcciones de grupo	Dirección
L04 ON/OFF	1/1/14
L04 ESTADO	1/1/15

Tabla 5.4. Direcciones de grupo en el proyecto.

Pertenece al dispositivo	Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
<b>Actuador accionamiento</b>	4	Salida 5, accionador	L04 ON/OFF	Sí
	16	Reenvío estado	L04 ESTADO	Sí
<b>Entrada binaria</b>	2	Entrada 3 objeto accionamiento	L04 ON/OFF	Sí
	2	objeto accionamiento	L04 ESTADO	No

Tabla 5.5. Análisis de los objetos de comunicación presentes en el proyecto.

Una vez hemos hecho un análisis de la información presente en el proyecto que queremos crear, deberemos utilizar las macros expuestas en el *Apartado 5.1.2* para crear una macro unificada que añada a un proyecto en blanco de *ETS3* todos los dispositivos y direcciones y configure todo lo necesario. Un ejemplo de este tipo de macro unificada se encuentra presente, como ya se ha comentado, en el *Anexo 1*. Para su creación se deben de seguir los siguientes pasos:

1. **Copiar las inicializaciones de las variables** de todas las macros al principio de la nueva macro.
2. Para **añadir los dispositivos** copiar la parte correspondiente de la macro Añadir dispositivos (*Apartado 5.1.2.1.*) tantas veces como dispositivos se vayan a añadir, utilizando los datos de entrada de la *Tabla 5.2.* para cada uno de los dispositivos.
3. Para **añadir las direcciones de grupo** copiar la parte correspondiente de la macro Añadir direcciones de grupo (*Apartado 5.1.2.4.*) tantas veces como direcciones se vaya a añadir, utilizando los datos de la *Tabla 5.4.* como datos de entrada.
4. Ahora se debe **configurar los dispositivos**, esto es fijar el valor de los parámetros. Para ello copiar la parte correspondiente de la macro Modificar parámetro (*Apartado 5.1.2.3.*) tantas veces como parámetros se vayan a modificar y utilizar los datos de la *Tabla 5.2.* Solo será necesario modificar aquellos parámetros que no vengan por defecto configurados al valor que se busca.
5. Por último, será necesario **enlazar las direcciones de grupo**. Es importante realizar este paso siempre después de la configuración de los dispositivos para que los objetos de comunicación en cuestión se encuentren visibles y manipulables. Para ello, copiar la parte correspondiente de la macro Enlazar direcciones de grupo (*Apartado 5.1.2.5.*) y utilizar la información contenida en la *Tabla 5.5.*

Una vez finalizado este proceso, se obtiene una macro que al ser ejecutada con un proyecto en blanco de *ETS3* abierto lo modifica hasta obtener el proyecto deseado. Es importante darse cuenta de que el procedimiento seguido para crear esta macro es un procedimiento basado en la replicación de fragmentos básicos de código reconfigurados de manera mínima. A este tipo de procedimiento se le podría denominar ‘procedimiento basado en plantillas’. En el siguiente apartado explicaremos como crear una plantilla de *JET* (cuya naturaleza y fundamentos ya han sido expuestos en el *Capítulo 3, Apartado 3.3.6* y en el *Capítulo 4, Apartado 4.1.3*) que realice todo este procedimiento de forma automática extrayendo los datos a partir de un sistema domótico descrito con el *DSL*. Es importante



entender todo el procedimiento de obtención de la macro descrita en este apartado para entender correctamente la plantilla que se va a exponer.

## 5.2. Plantilla JET

### 5.2.1. Introducción.

En el epígrafe anterior a este se expone como realizar una macro global que permita generar un proyecto *ETS3* a partir de un proyecto en blanco. En este epígrafe se expondrá como programar una plantilla *JET* que permita la implementación automática de estas macros globales y que tome los datos de una descripción de un proyecto realizada utilizando el *DSL* expuesto en el *Capítulo 3, Apartado 2.3.2*. Para entender la plantilla será imprescindible tener claros los conceptos inherentes a este *DSL*, así como las nociones de programación en *JET* y como implementar una macro que genere un proyecto *ETS3*.

### 5.2.2. Descripción de la plantilla programada.

En este apartado se va a describir el código de la plantilla programada para posibilitar su comprensión y posterior modificación. Se va a analizar el código de una manera secuencial haciendo hincapié en los conceptos del lenguaje de programación, del *DSL* y de las macros. La plantilla final se encuentra en el *Anexo II*. Se aconseja consultarla a la vez que se lee este apartado para una mejor comprensión.

En primer lugar hay un iterador que recorre los proyectos presentes en la descripción del *DSL* (*Figura 5.14*). Habitualmente solo habrá un proyecto descrito, pero se incluye para usos futuros.

```
<c:iterate select = "//KNXProject" var = "a"> <%-- iterador de los proyectos --%>
```

*Figura 5.14. Iterador que recorre los proyectos.*

A partir de aquí se van a seguir los pasos descritos en el Apartado 5.4. que describen como implementar la macro global para generar el proyecto. El primer paso es la inicialización de las variables que se van a necesitar para todas las macros (*Figura 5.15*).

```
<%--
PASO 0
Inicializaciones de la variables VBS para la Makro
--%>
...
```

*Figura 5.15. Inicialización de las variables.*



Tras esta parte encontramos la parte de la plantilla que modifica los parámetros del proyecto. Se recorren en primer lugar los dispositivos y para cada dispositivo encontrado se recorren sus parámetros. Para cada parámetro encontrado se introduce la parte necesaria de la macro modificar parámetro (*Apartado 5.1.2.3.*), obteniendo la información necesaria para las variables de entrada de la descripción realizada utilizando el DSL (*Figura 5.18.*).

```
<!--
PASO 3
Parte de la plantilla para modificar los parámetros
-->
.....
'
'                               Se configuran los dispositivos (parámetros)
'
'.....
<c:iterate select = "//BusDevice" var = "e"> <!-- Iterador 1 para modificar los parámetros (1º se recorren
los dispositivos) -->
<c:iterate select = "$e/Parameter" var = "f"> <!-- Iterador 2 para modificar los parámetros (Después, los
parámetros de cada dispositivo) -->
'
'Disp: <c:get select = "$e/@name"/>
'Descrip: <c:get select = "$f/@description"/>
'<c:get select = "$f/@name"/> [<c:get select = "$f/@Order"/>] = <c:get select = "$f/@defValue"/>
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = "<c:get select = "$e/@name"/>"
ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "<c:get select = "$f/@name"/>"
ModificarParametro_nombreParametro = lcase(ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner
el valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = <c:get select = "$f/@defValue"/>

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = <c:get select = "$f/@Order"/>
...

```

*Figura 5.18. Configurar parámetros.*

Por último, se implementan las asociaciones de los objetos de comunicación a las direcciones de grupo siguiendo un procedimiento similar (*Figura 5.19.*) y la macro *Enlazar Direcciones Grupo* (*Apartado 5.1.2.5.*).



### 5.3.1. Maleta KNX/EIB

Como ya se ha comentado en la introducción, para la finalidad de testear el correcto funcionamiento de la metodología propuesta se implementó un sistema domótico típico haciendo uso de la *Maleta KNX/EIB* a disposición del departamento. Esta maleta recoge una maqueta completa y funcional de una instalación domótica típica para una vivienda unifamiliar.

Esta maqueta se encuentra instalada en una maleta portátil, resultando de esta manera compuesta por dos paneles: el panel superior (*Figura 5.20., izquierda*) y el panel inferior (*Figura 5.20., derecha*).



*Figura 5.20. Panel superior (izqda.) y panel inferior (dcha.) de la maleta EIB/KNX.*

En el panel superior se puede apreciar un plano de la vivienda sobre el que se dispone de puntos de luz (regulados de tipo *on/off*), persianas (representadas mediante barras de leds) y pulsadores para su control. Asimismo se dispone de dos pulsadores y dos indicadores para generar y visualizar alarmas de fuego e inundación y dos indicadores para la activación de la máquina de frío y calor para la climatización. Una imagen detallada de este panel puede verse también en el ANEXO IV.

En el panel inferior se puede observar dos cuadros eléctricos con los dispositivos de carril *DIN*: protecciones, fuente de alimentación, actuador de regulación con tres salidas, actuador de persianas y actuadores de accionamiento. Además, en dicho panel se encuentra presente una bombilla fluorescente regulada (conectada al actuador de regulación) y un interfaz de programación *USB*. También se ha incluido en dicho panel tres teclados: uno con dos teclas con pulsación arriba y abajo, un teclado de una tecla con pulsación arriba/abajo y un teclado universal de 8 teclas (4 fases) con receptor de radiofrecuencia para el mando a distancia. Las teclas del mando a distancia realizan las mismas funciones que se hayan programado en este teclado.

Esta maleta es por tanto una completa instalación domótica completa y funcional compuesta por dispositivos de la tecnología *KNX/EIB*, que va a permitir la implementación de

cualquier instalación imaginable. Por último, reseñar que en el *ANEXO IV* se incluye también un mapa eléctrico de la maleta, en el cuál se pueden observar los aparatos arriba enumerados y ver qué dirección física tienen programada.

### 5.3.2. Proyecto a implementar.

Como ya se ha comentado, se pretende implementar un proyecto para controlar las luces y las persianas presentes en la maleta utilizando la metodología desarrollada en este trabajo fin de máster. De esta manera, el proyecto a implementar se describe en las siguientes tablas (*Tabla 5.6.* a *Tabla 5.8.*). En la *Tabla 5.6.* se describe que pulsadores distribuidos en el panel superior controlarán que puntos de luz y en la *Tabla 5.7.* se de forma análoga el control de las persianas mediante estos pulsadores; mientras que en la *Tabla 5.8.* se muestra que funciones de luz y persianas se centralizarán en los teclados *KNX/EIB*.

Punto de luz	Tipo	Ubicación	Controlada desde
LR01	Regulada	Dormitorio principal	P13
LR02	Regulada	Salón	P08
L04	On/Off	Apliques cocina	P11
L05	On/Off	Dormitorio	P14
L06	On/Off	Baño principal	P15
L07	On/Off	Dormitorio	P16
L08	On/Off	Hall	P09
L09	On/Off	Dormitorio	P10
L10	On/Off	Terraza	P07

*Tabla 5.6. Pulsadores analógicos, control de luces.*

Persiana	Ubicación	Controlada desde
Pers01	Dormitorio	P01: arriba P02: abajo
Pers02	Dormitorio principal	P03: arriba P04: abajo
Pers03	Dormitorio	P17: arriba P18: abajo
Pers04	Dormitorio	P19: arriba P20: abajo

*Tabla 5.7. Pulsadores analógicos, control de persianas.*

Teclado	Ubicación	Función
PUL-2F-1	Panel inferior de la maleta Se ubicará en Hall	Tecla izquierda arriba/abajo: general persianas Tecla derecha arriba/abajo: persianas Pers01 y Pers02
PUL-1F-1	Panel inferior de la maleta Se ubicará en Hall	General de iluminación para toda la vivienda: Tecla arriba: ON Tecla abajo: OFF
PUL-4F-1	Panel inferior de la maleta. Se ubicará en el dormitorio principal.	Tecla 1: control de LR01 Tecla 2: control de L06 Tecla 3: persiana Pers02 arriba Tecla 4: persiana Pers02 abajo Tecla 5: Todas las luces ON Tecla 6: Todas las luces OFF Tecla 7: Todas las persianas ARRIBA Tecla 8: Todas las persianas ABAJO
PUL-4F-2	Panel superior Se ubicará en Hall	Tecla 1: control de Pers01 Tecla 2: control de Pers02 Tecla 3: control de Pers03 Tecla 4: control de Pers04

Tabla 5.8. Teclados KNX/EIB, controles de luces y persianas centralizado.

### 5.3.3. Análisis de los dispositivos implicados en el proyecto.

Comenzando con la metodología descrita en el *Apartado 5.1.4.* para utilizar las herramientas diseñadas para implementar un proyecto domótico *EIB/KNX*, analizaremos los dispositivos presentes en el proyecto. Esta información se encuentra recogida en la *Tabla 5.9.*

Si se utiliza el *DSL* del *Apartado 2.3.2.* para describir estos dispositivos, se obtiene en *EMF* algo similar a la *Figura 5.21.* En cada dispositivo se introducirán los datos necesarios, explicados en el apartado anteriormente citado.

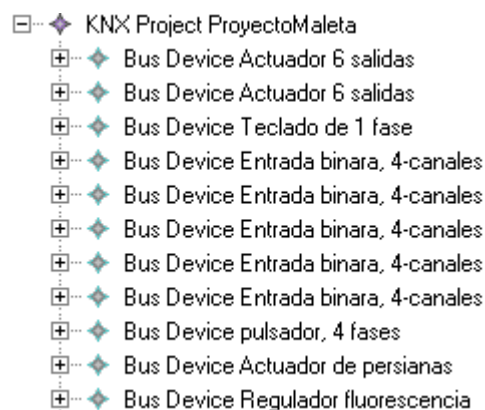


Figura 5.21. Dispositivos presentes en el proyecto.

Dispositivo	Nombre ETS3	Fabricante	Información de la entrada del catálogo	Dirección física
<b>Regulador fluorescencia</b>	Regulador fluorescencia 1-10V, 3 canales DIN	Albrecht Jung	Catalog Entry: Regulador fluorescencia 1-10V, 3 canales DIN Order Number: 2193REG Nombre programa: Regulador fluorescencia 301801 versión: 1	1
<b>Actuador persianas</b>	JA/S4.230.1M Shutter Actuador Man.,4f,230V AC,MDRC	ABB	Catalog Entry: JA/S4.230.1M Shutter Actuador Man.,4f,230VAC,MDRC Order Number: GH Q631 0064 R0111 Nombre programa: Shutter,4f M/2 versión: 32	2
<b>Actuador Accionamiento (1)</b>	Actuador accionamiento, 6 salidas, 6A DIN	Albrecht Jung	Catalog Entry: Actuador accionamiento, 6 salidas, 6A DIN Order Number: 2136.6REG Nombre programa: Accionamiento con reenvío, func. lógica, temporiza versión: 1	3
<b>Actuador Accionamiento (2)</b>	Actuador accionamiento, 6 salidas, 6A DIN	Albrecht Jung	Catalog Entry: Actuador accionamiento, 6 salidas, 6A DIN Order Number: 2136.6REG Nombre programa: Accionamiento con reenvío, func. lógica, temporiza versión: 1	4
<b>Pulsador 1 Fase</b>	Pulsador con acoplador 1 fase, fijación media	Albrecht Jung	Catalog Entry: Pulsador con acoplador 1 fase, fijación media Order Number: 2071.02LED Nombre programa: Accionar, regular, persiana 105601 versión: 1	6
<b>Entrada Binaria (1)</b>	Entrada binaria, 4 canales	Albrecht Jung	Catalog Entry: Entrada binaria, 4 canales Order Number: 2076-4T Nombre programa: 4 entradas, 2 salidas 705701 versión: 1	9
<b>Entrada Binaria (2)</b>	Entrada binaria, 4 canales	Albrecht Jung	Catalog Entry: Entrada binaria, 4 canales Order Number: 2076-4T Nombre programa: 4 entradas, 2 salidas 705701 versión: 1	10
<b>Entrada Binaria (3)</b>	Entrada binaria, 4 canales	Albrecht Jung	Catalog Entry: Entrada binaria, 4 canales Order Number: 2076-4T Nombre programa: 4 entradas, 2 salidas 705701 versión: 1	11
<b>Entrada Binaria (4)</b>	Entrada binaria, 4 canales	Albrecht Jung	Catalog Entry: Entrada binaria, 4 canales Order Number: 2076-4T Nombre programa: 4 entradas, 2 salidas 705701 versión: 1	12
<b>Entrada Binaria (5)</b>	Entrada binaria, 4 canales	Albrecht Jung	Catalog Entry: Entrada binaria, 4 canales Order Number: 2076-4T Nombre programa: 4 entradas, 2 salidas 705701 versión: 1	13
<b>Teclado 4 Fases (2)</b>	Tastensor 4fach Standard TSM	Albrecht Jung	Catalog Entry: Teclado estándar, 4 fases Order Number: 2074NABS Nombre programa: Persiana 102C01 versión: 1	16

Tabla 5.9. Información de los dispositivos en el proyecto.



### 5.3.4. Análisis de las direcciones de grupo presentes en el proyecto.

El siguiente punto será añadir las direcciones de grupo presentes en el proyecto y necesarias para lograr la funcionalidad deseada. Estas direcciones pueden apreciarse resumidas en la *Tabla 5.10*.

De forma análoga al apartado anterior, si se utiliza el *DSL* propuesto para incluir estas direcciones en el proyecto se obtendrá algo similar a la *Figura 5.22*.

- |  |   |
|--|---|
| ◆ Association Mover Todas Persianas            | ◆ Association L05 ESTADO                    |
| ◆ Association Stop Todas Persianas             | ◆ Association L06 ON/OFF                    |
| ◆ Association Iluminación todas                | ◆ Association L06 ESTADO                    |
| ◆ Association LR01 Dormitorio ON/OFF           | ◆ Association L07 ON/OFF                    |
| ◆ Association LR01 Dormitorio REGULACIÓN       | ◆ Association L07 ESTADO                    |
| ◆ Association LR01 Dormitorio ESTADO           | ◆ Association L08 ON/OFF                    |
| ◆ Association LR01 Dormitorio VALOR            | ◆ Association L08 ESTADO                    |
| ◆ Association LR02 Salon ON/OFF                | ◆ Association L09 ON/OFF                    |
| ◆ Association LR02 Salon REGULACIÓN            | ◆ Association L09 ESTADO                    |
| ◆ Association LR02 Salon ESTADO                | ◆ Association L10 ON/OFF                    |
| ◆ Association LR02 Salon VALOR                 | ◆ Association L10 ESTADO                    |
| ◆ Association Bombilla Fluorescente ON/OFF     | ◆ Association P01 Persianas1 Dorm MOV       |
| ◆ Association Bombilla Fluorescente REGULACIÓN | ◆ Association P01 Persianas1 Dorm STOP      |
| ◆ Association Bombilla Fluorescente ESTADO     | ◆ Association P02 Persianas2 Dorm MOV       |
| ◆ Association Bombilla Fluorescente VALOR      | ◆ Association P02 Persianas2 Dorm ppal STOP |
| ◆ Association L03 Aseo ON/OFF                  | ◆ Association P03 Persianas3 Dorm MOV       |
| ◆ Association L03 Aseo ESTADO                  | ◆ Association P03 Persianas3 Dorm STOP      |
| ◆ Association L04 ON/OFF                       | ◆ Association P04 Persianas4 Dorm MOV       |
| ◆ Association L04 ESTADO                       | ◆ Association P04 Persianas4 Dorm STOP      |
| ◆ Association L05 ON/OFF                       |   |

*Figura 5.22. Direcciones de grupo del proyecto en EMF.*

Nombre dirección de grupo	Dirección
Mover Todas Persianas	1/0/0
Stop Todas Persianas	1/0/1
Iluminación todas	1/0/2
LR01 Dormitorio ON/OFF	1/1/0
LR01 Dormitorio REGULACIÓN	1/1/1
LR01 Dormitorio ESTADO	1/1/2
LR01 Dormitorio VALOR	1/1/3
LR02 Salon ON/OFF	1/1/4
LR02 Salon REGULACIÓN	1/1/5
LR02 Salon ESTADO	1/1/6
LR02 Salon VALOR	1/1/7
Bombilla Fluorescente ON/OFF	1/1/8
Bombilla Fluorescente REGULACIÓN	1/1/9
Bombilla Fluorescente ESTADO	1/1/10
Bombilla Fluorescente VALOR	1/1/11
L03 Aseo ON/OFF	1/1/12
L03 Aseo ESTADO	1/1/13
L04 ON/OFF	1/1/14
L04 ESTADO	1/1/15
L05 ON/OFF	1/1/16
L05 ESTADO	1/1/17
L06 ON/OFF	1/1/18
L06 ESTADO	1/1/19
L07 ON/OFF	1/1/20
L07 ESTADO	1/1/21
L08 ON/OFF	1/1/22
L08 ESTADO	1/1/23
L09 ON/OFF	1/1/24
L09 ESTADO	1/1/25
L10 ON/OFF	1/1/26
L10 ESTADO	1/1/27
P01 Persiana1 Dorm MOV	1/2/0
P01 Persiana1 Dorm STOP	1/2/1
P02 Persiana2 Dorm MOV	1/2/2
P02 Persiana2 Dorm ppal STOP	1/2/3
P03 Persiana3 Dorm MOV	1/2/4
P03 Persiana3 Dorm STOP	1/2/5
P04 Persiana4 Dorm MOV	1/2/6
P04 Persiana4 Dorm STOP	1/2/7

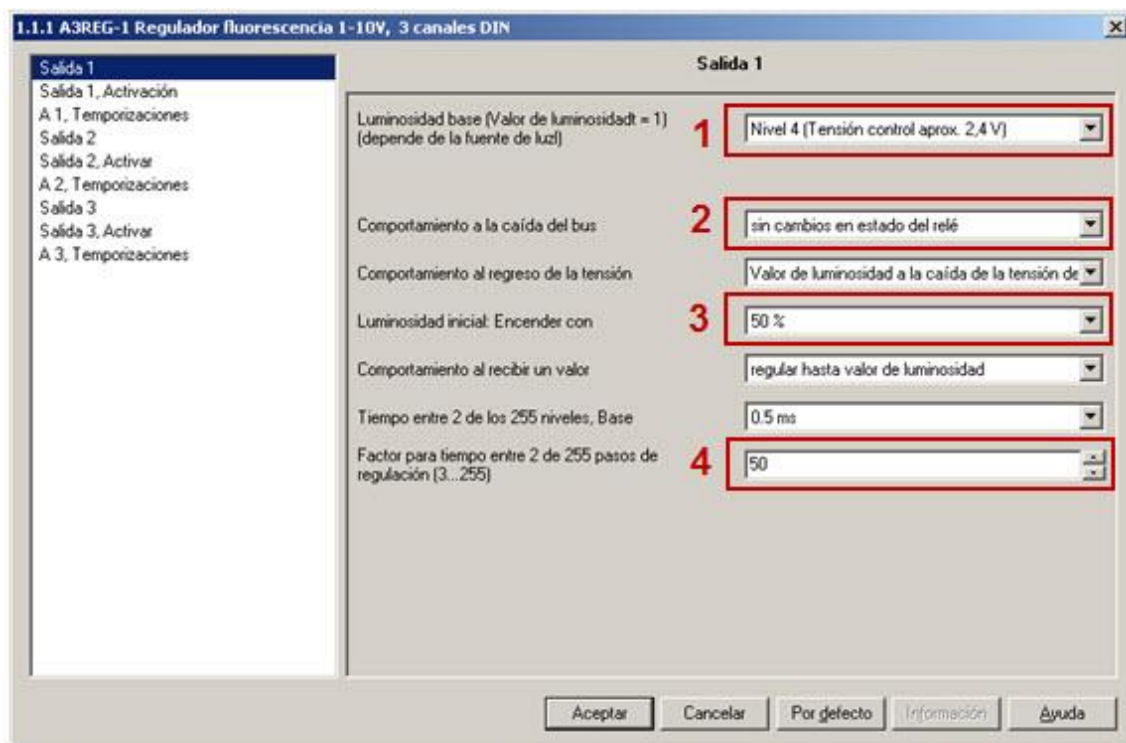
Tabla 5.10. Direcciones de grupo en el proyecto.

### 5.3.5. Análisis de los parámetros asociados a los dispositivos.

En este apartado se realizará un análisis de los parámetros que es necesario modificar para implementar el proyecto, evaluando los dispositivos uno a uno. En los sucesivos epígrafes se muestra de manera detallada estos análisis.

#### 5.3.5.1. Análisis de los parámetros del dispositivo “Regulador de fluorescencia”.

En las figuras de *Figura 5.23.* a *Figura 5.31.* se muestra de manera gráfica que parámetros es necesario cambiar en el dispositivo a partir de su configuración por defecto. En dichas figuras se numeran de forma gráfica dichos parámetros, de tal manera que en la *Tabla 5.11.* se recogen los datos a introducir en el *DSL* para programar este dispositivo. En la *Figura 5.32.* se muestra la descripción de estos parámetros utilizando el *DSL*.



*Figura 5.23. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (I).*

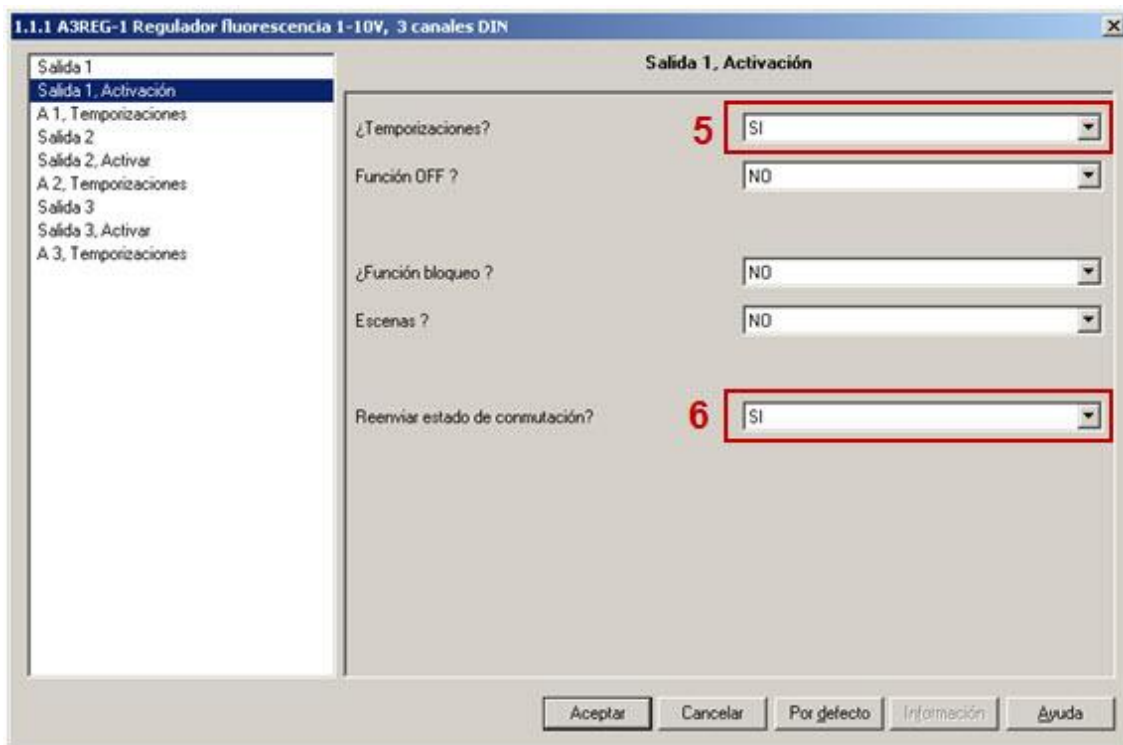


Figura 5.24. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (II).

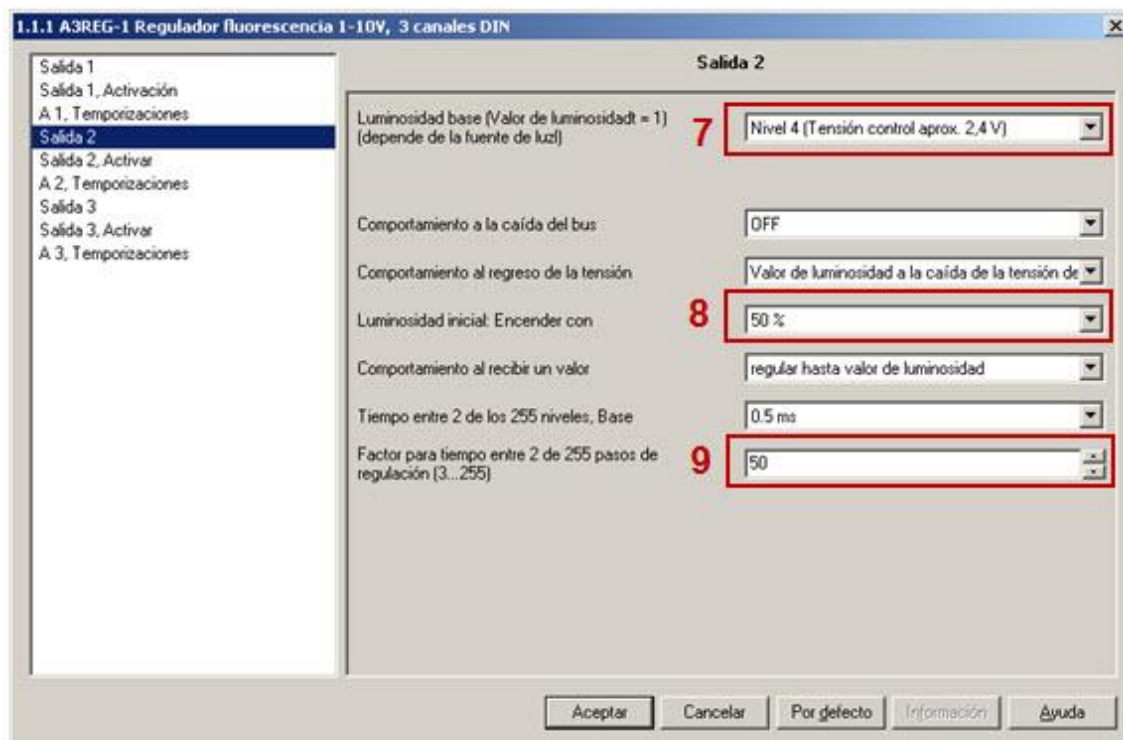


Figura 5.25. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (III).

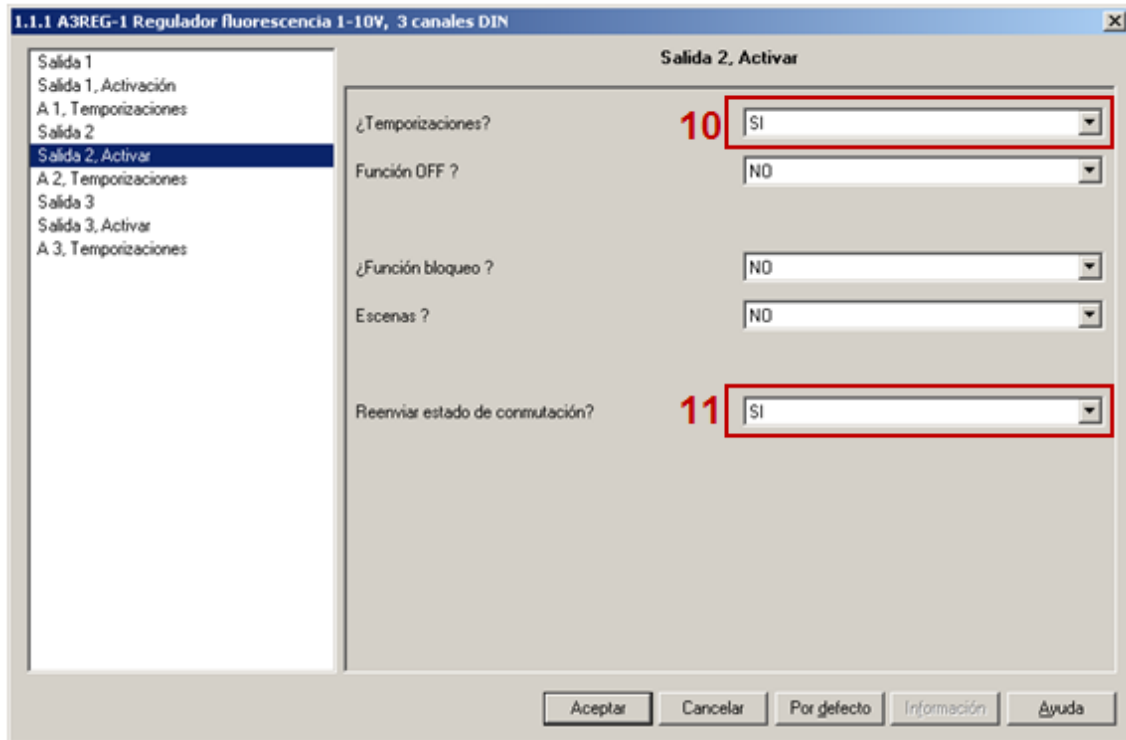


Figura 5.26. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (IV).

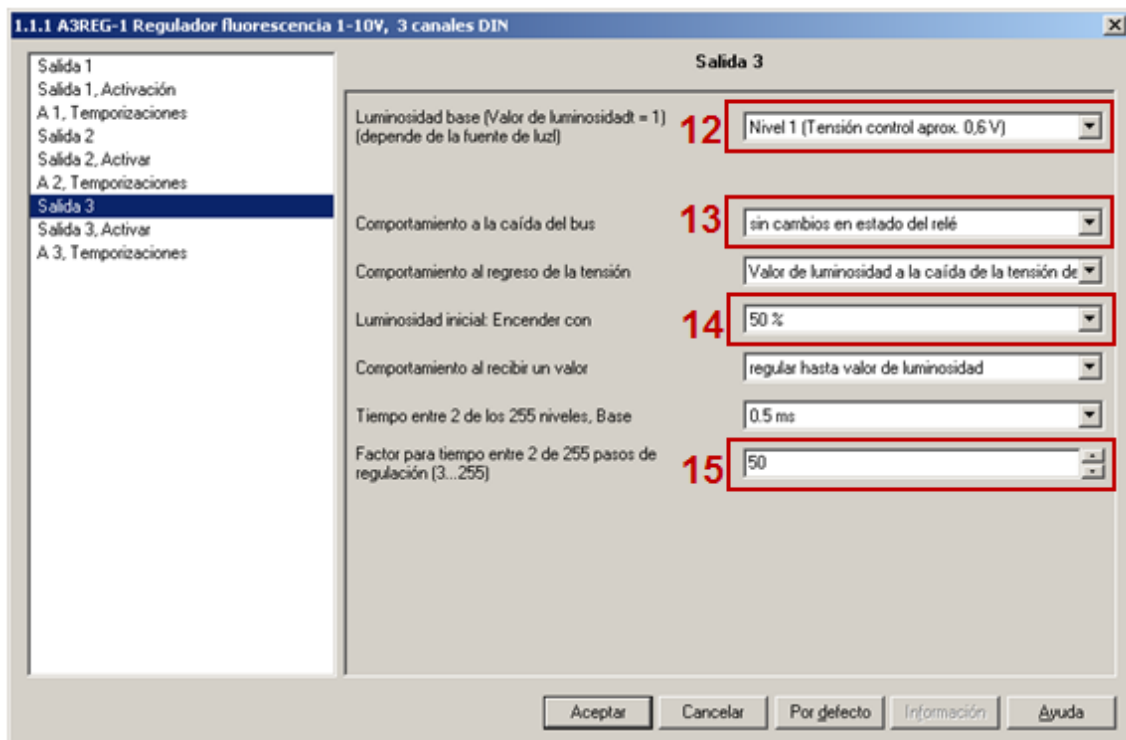


Figura 5.27. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (V).

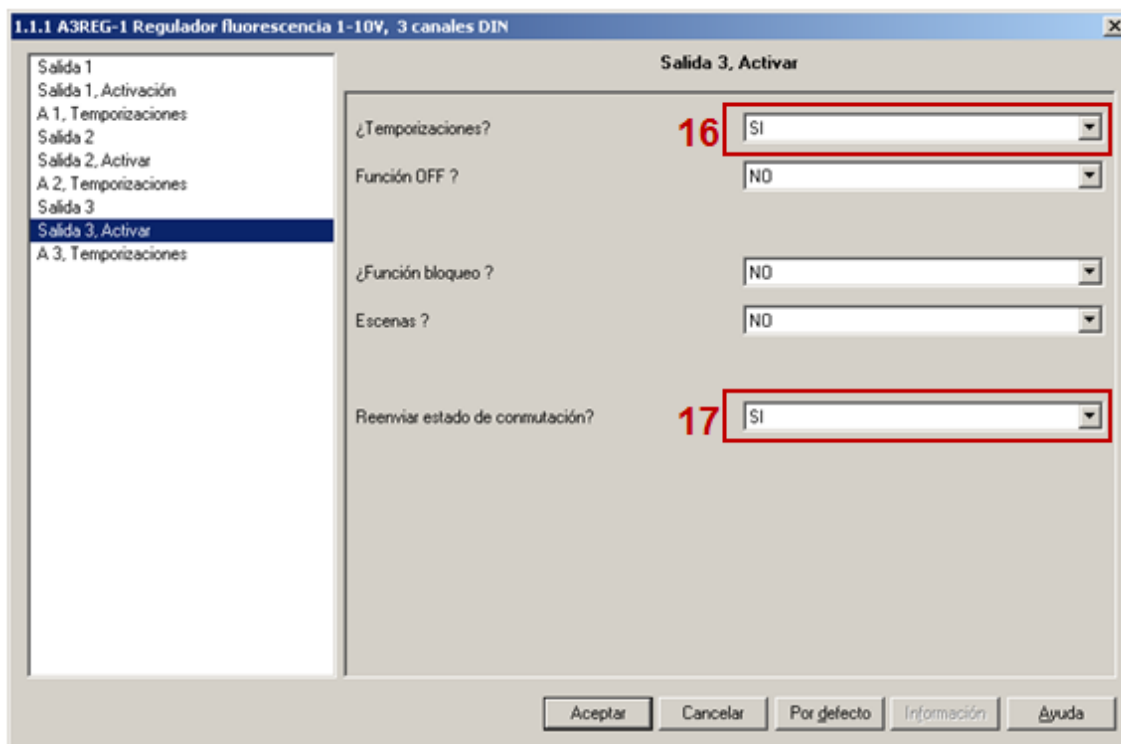


Figura 5.28. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (VI).

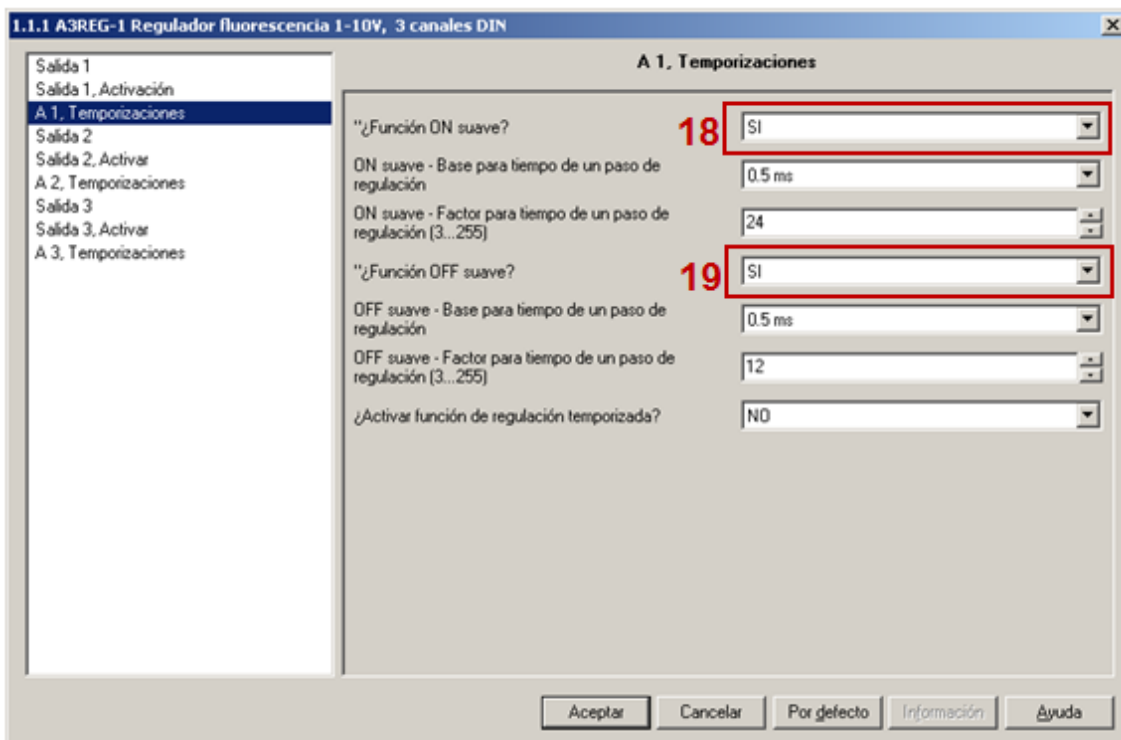


Figura 5.29. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (VII).

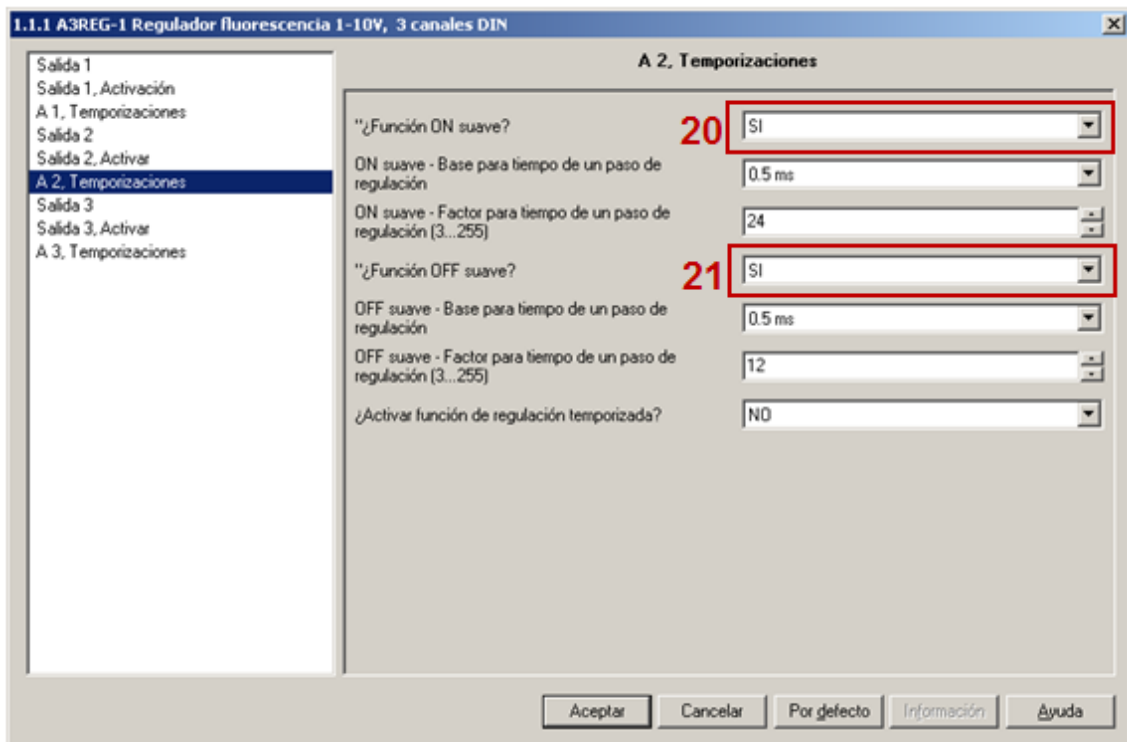


Figura 5.30. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (VIII).

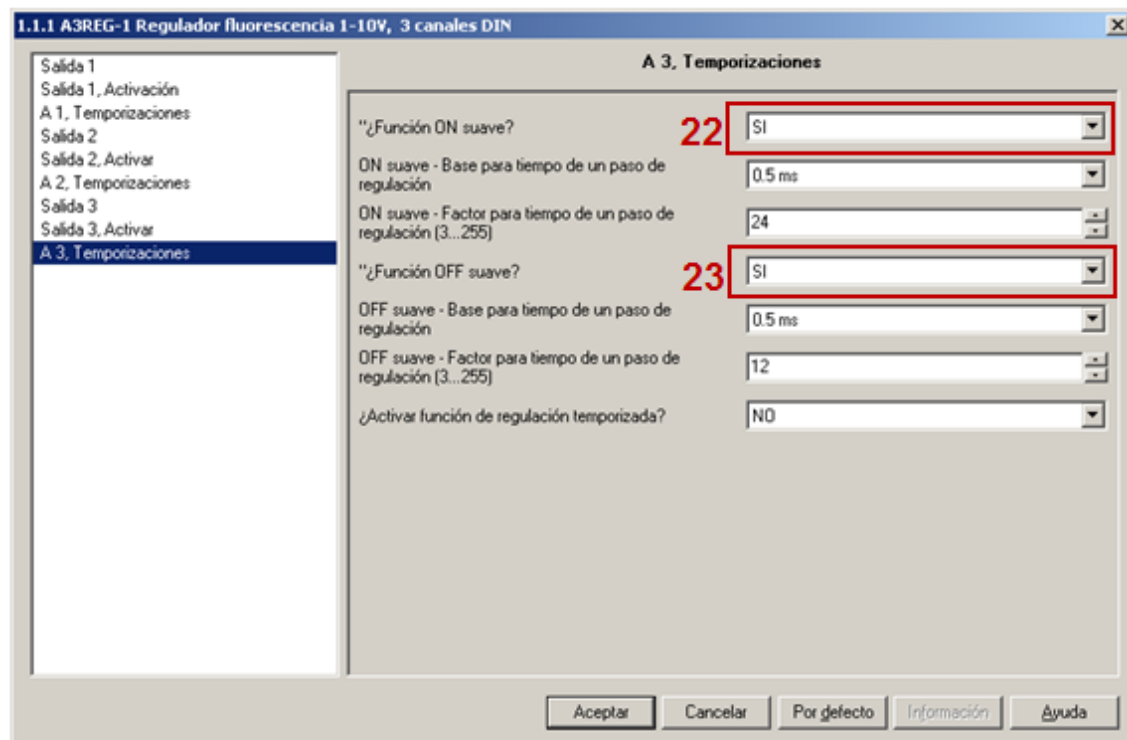


Figura 5.31. Análisis de los parámetros para el dispositivo “Regulador de fluorescencia” (IX).

Numero	Nombre	Valor	Orden aparición
1	Grundhelligkeit_Kanal_1	3	1
2	Wert_Busausfall	2	1
3	Einschalthelligkeit_Kanal_1	6	1
4	Faktor_Dimmschritt_Kanal_1	50	1
5	Ausg_1_Zeit	1	1
6	Ausg_1_Rück	1	1
7	Grundhelligkeit_Kanal_2	3	1
8	Einschalthelligkeit_Kanal_2	6	1
9	Faktor_Dimmschritt_Kanal_2	50	1
10	Ausg_2_Zeit	1	1
11	Ausg_2_Rück	1	1
12	Grundhelligkeit_Kanal_3	0	1
13	Wert_Busausfall	2	3
14	Einschalthelligkeit_Kanal_3	6	1
15	Faktor_Dimmschritt_Kanal_3	50	1
16	Ausg_3_Zeit	1	1
17	Ausg_3_Rück	1	1
18	SoftEIN_JA_NEIN	1	3
19	SoftAUS_JA_NEIN	1	3
20	SoftEIN_JA_NEIN	1	1
21	SoftAUS_JA_NEIN	1	1
22	SoftEIN_JA_NEIN	1	2
23	SoftAUS_JA_NEIN	1	2

Tabla 5.11. Resumen de los parámetros para el dispositivo “Regulador de fluorescencia”.

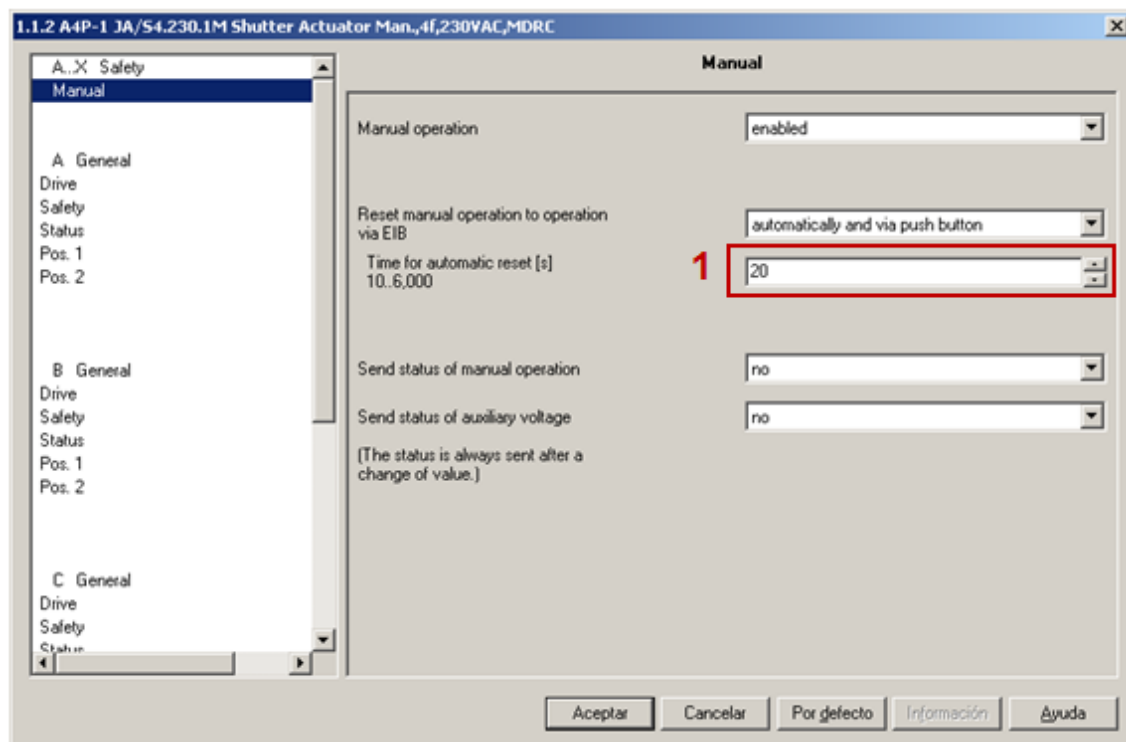
- ◆ Parameter Grundhelligkeit\_Kanal\_1
- ◆ Parameter Wert\_Busausfall
- ◆ Parameter Einschalthelligkeit\_Kanal\_1
- ◆ Parameter Faktor\_Dimmschritt\_Kanal\_1
- ◆ Parameter Ausg\_1\_Zeit
- ◆ Parameter Ausg\_1\_Rück
- ◆ Parameter Grundhelligkeit\_Kanal\_2
- ◆ Parameter Einschalthelligkeit\_Kanal\_2
- ◆ Parameter Faktor\_Dimmschritt\_Kanal\_2
- ◆ Parameter Ausg\_2\_Zeit
- ◆ Parameter Ausg\_2\_Rück
- ◆ Parameter Grundhelligkeit\_Kanal\_3
- ◆ Parameter Wert\_Busausfall
- ◆ Parameter Einschalthelligkeit\_Kanal\_3
- ◆ Parameter Faktor\_Dimmschritt\_Kanal\_3
- ◆ Parameter Ausg\_3\_Zeit
- ◆ Parameter Ausg\_3\_Rück
- ◆ Parameter SoftEIN\_JA\_NEIN
- ◆ Parameter SoftAUS\_JA\_NEIN
- ◆ Parameter SoftEIN\_JA\_NEIN
- ◆ Parameter SoftAUS\_JA\_NEIN
- ◆ Parameter SoftEIN\_JA\_NEIN
- ◆ Parameter SoftAUS\_JA\_NEIN

Figura 5.32. Descripción de los parámetros para el dispositivo “Regulador de fluorescencia” utilizando el DSL propuesto en EMF.



### 5.3.5.2. Análisis de los parámetros del dispositivo “Actuador de persianas”.

En las figuras de *Figura 5.33.* a *Figura 5.41.* se muestra de manera gráfica que parámetros es necesario cambiar en el dispositivo a partir de su configuración por defecto. En dichas figuras se numeran de forma gráfica dichos parámetros, de tal manera que en la *Tabla 5.12.* se recogen los datos a introducir en el *DSL* para programar este dispositivo. En la *Figura 5.42.* se muestra la descripción de estos parámetros utilizando el *DSL*.



*Figura 5.33. Análisis de los parámetros para el dispositivo “Actuador de persianas” (I).*

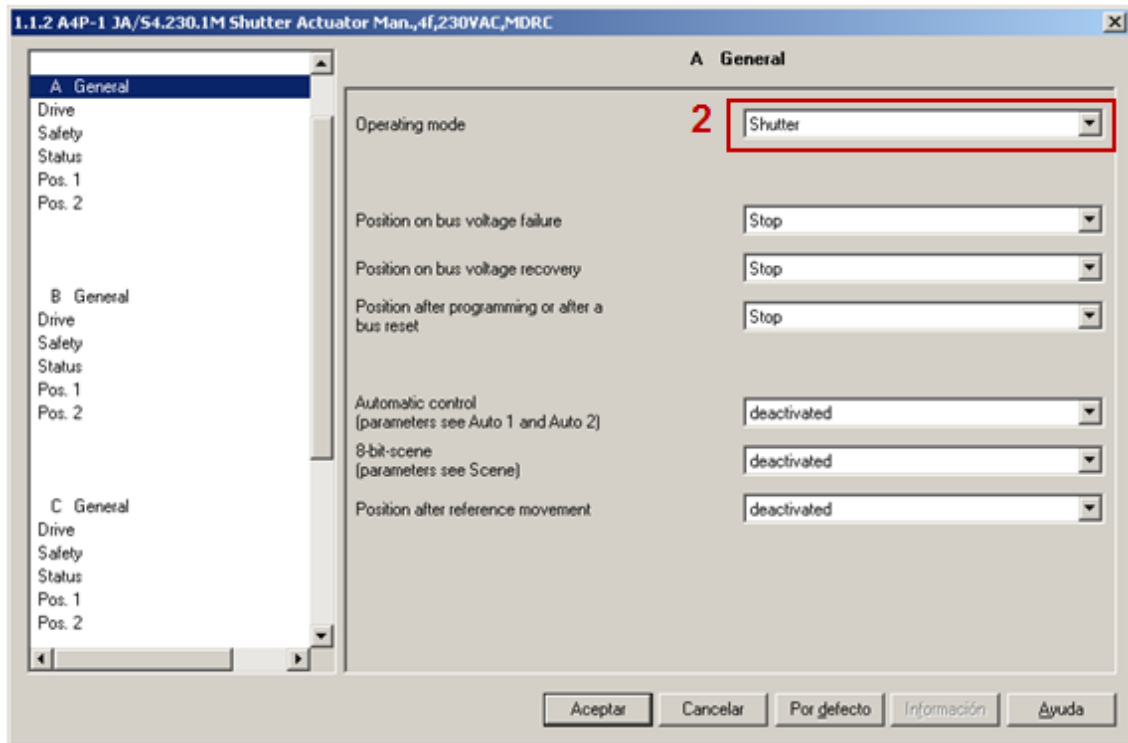


Figura 5.34. Análisis de los parámetros para el dispositivo “Actuador de persianas” (II).

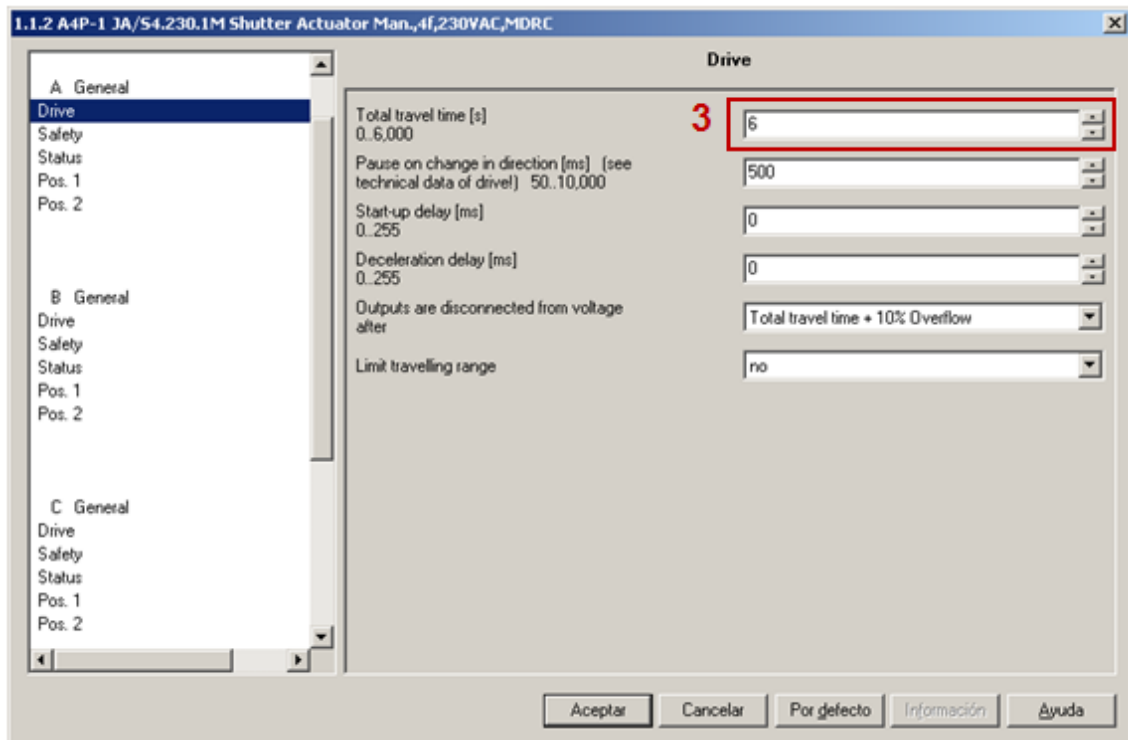


Figura 5.35. Análisis de los parámetros para el dispositivo “Actuador de persianas” (III).

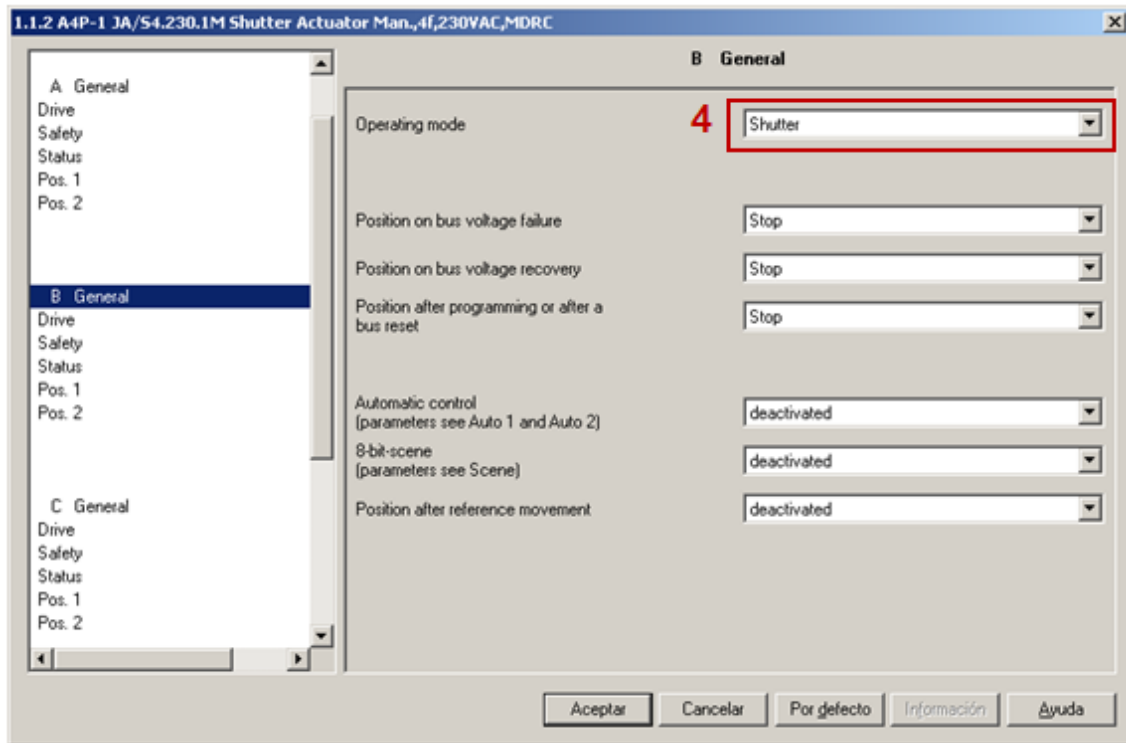


Figura 5.36. Análisis de los parámetros para el dispositivo “Actuador de persianas” (IV).

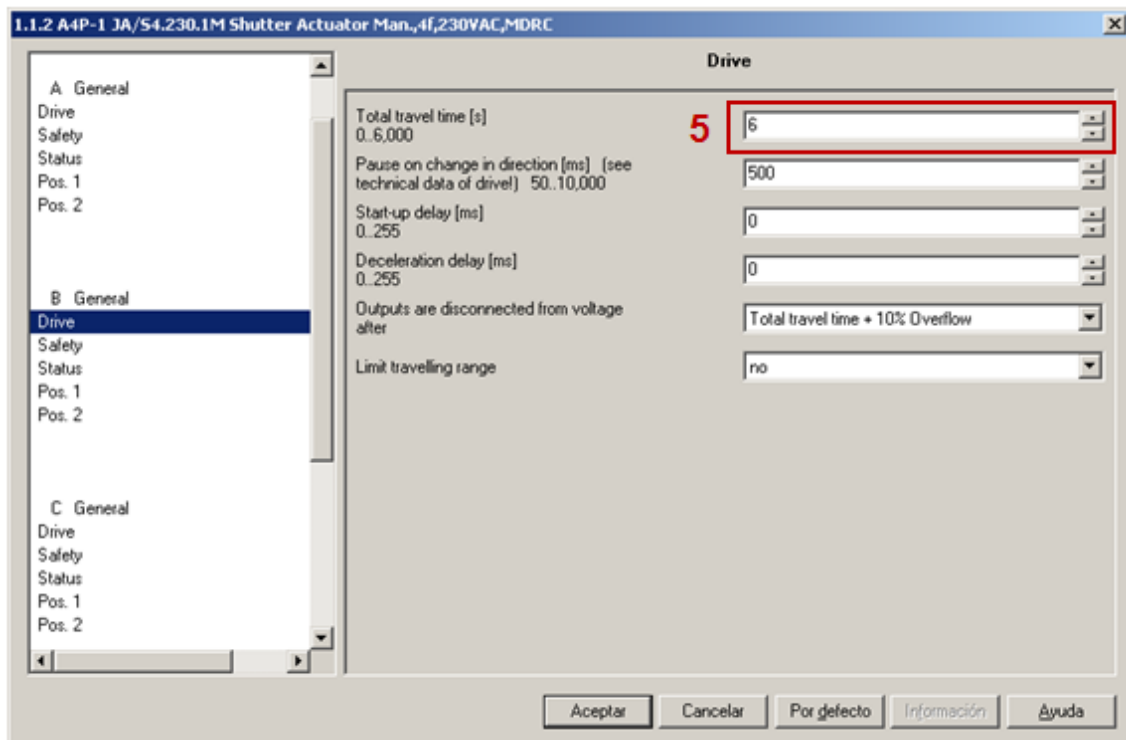


Figura 5.37. Análisis de los parámetros para el dispositivo “Actuador de persianas” (V).

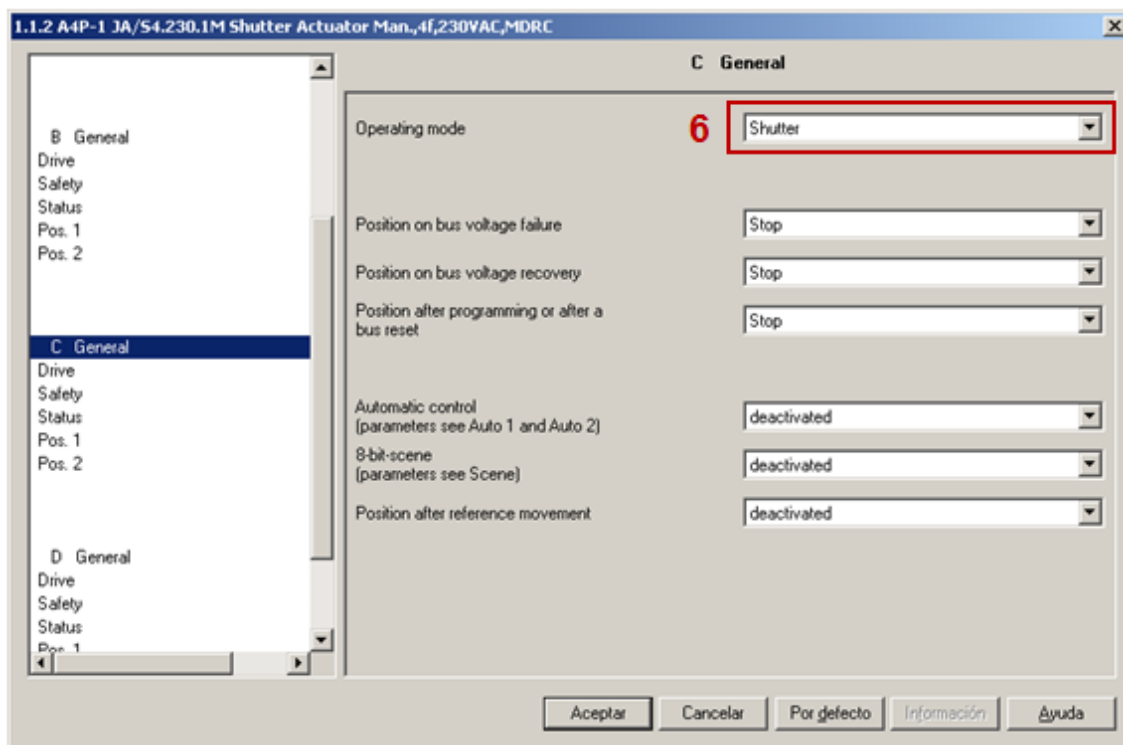


Figura 5.38. Análisis de los parámetros para el dispositivo “Actuador de persianas” (VI).

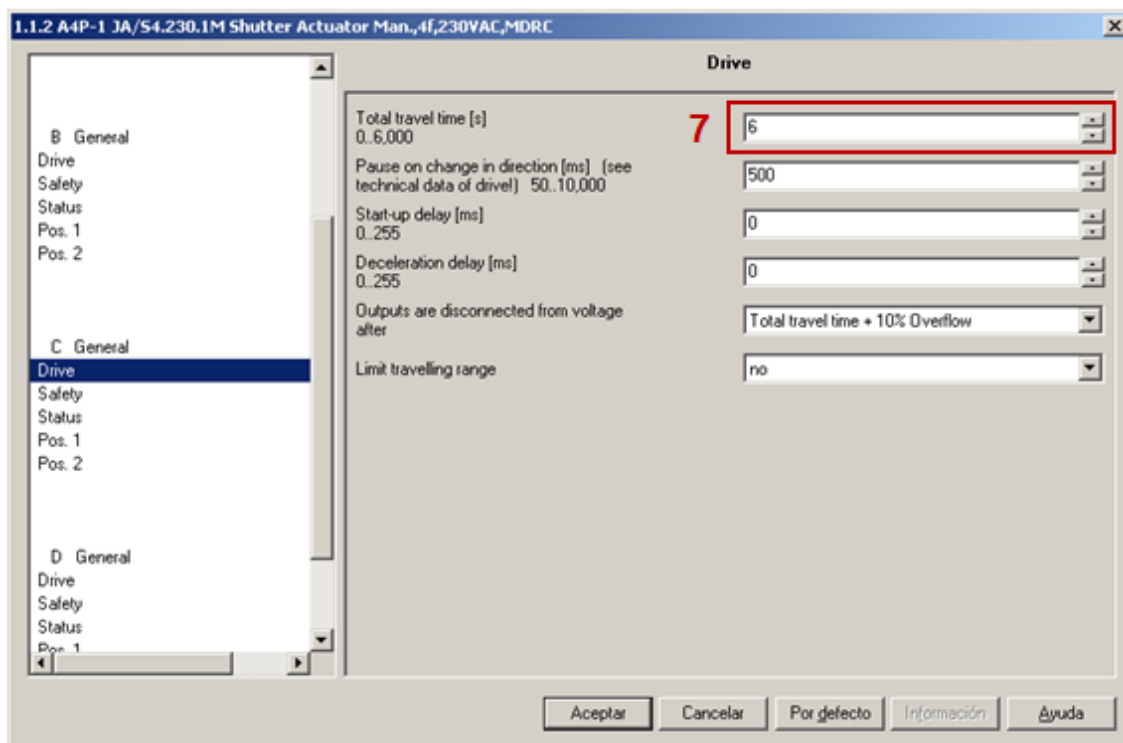


Figura 5.39. Análisis de los parámetros para el dispositivo “Actuador de persianas” (VII).

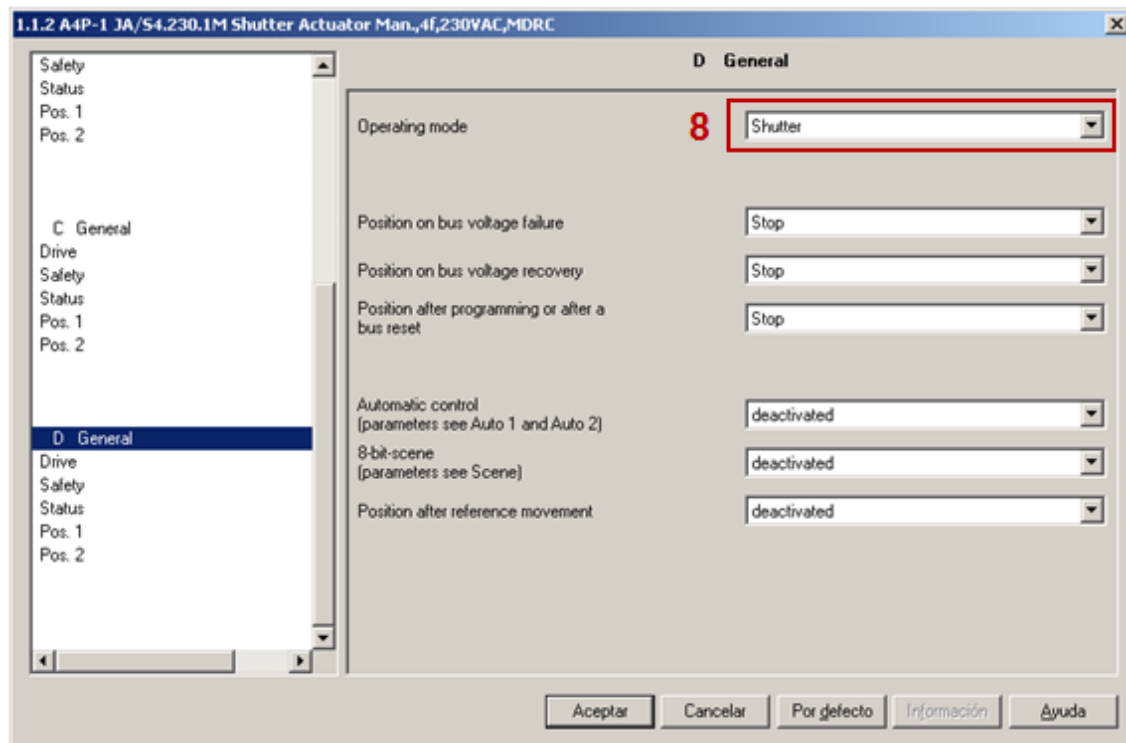


Figura 5.40. Análisis de los parámetros para el dispositivo "Actuador de persianas" (VIII).

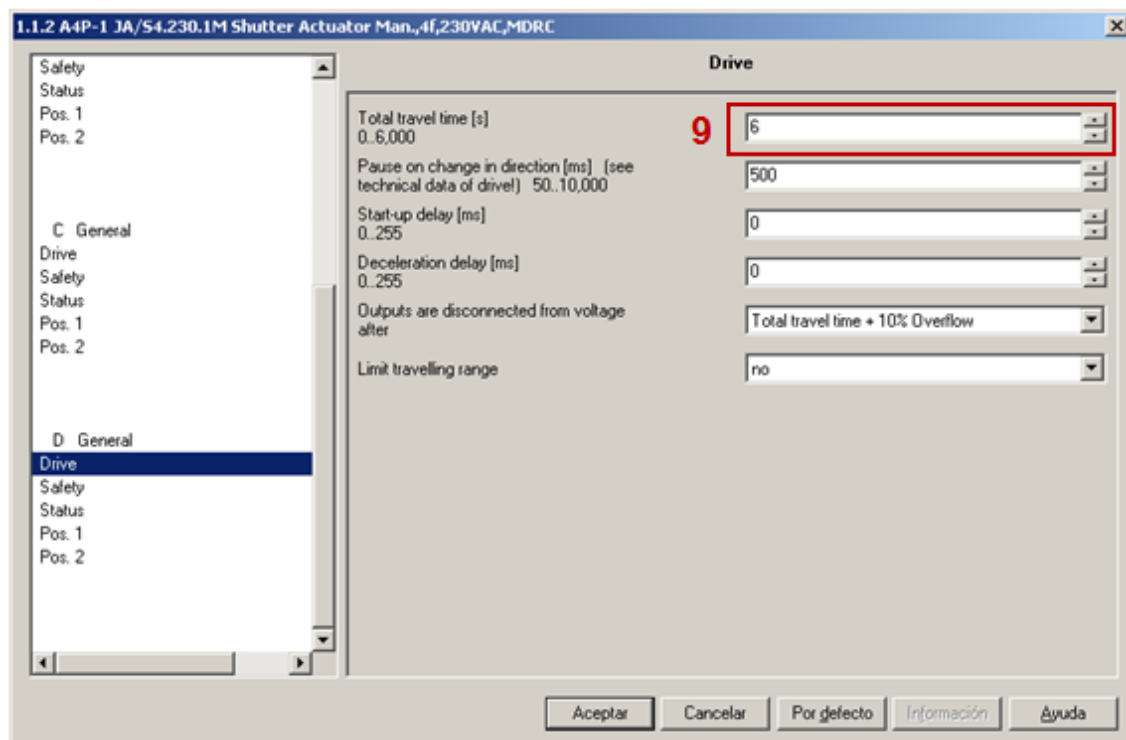


Figura 5.41. Análisis de los parámetros para el dispositivo "Actuador de persianas" (IX).

Numero	Nombre	Valor	Orden aparición
1	Zeit für aut. Zurücksetzen	20	1
2	Betriebsart	1	2
3	Gesamtverfahrzeit	6	3
4	Betriebsart	1	4
5	Gesamtverfahrzeit	6	2
6	Betriebsart	1	1
7	Gesamtverfahrzeit	6	5
8	Betriebsart	1	3
9	Gesamtverfahrzeit	6	7

Tabla 5.12. Resumen de los parámetros para el dispositivo “Actuador de persianas”.



Figura 5.42. Descripción de los parámetros para el dispositivo “Actuador de persianas” utilizando el DSL propuesto en EMF.

### 5.3.5.3. Análisis de los parámetros del dispositivo “Actuador de 6 salidas (1)”.

En las figuras de *Figura 5.43.* a *Figura 5.48.* se muestra de manera gráfica que parámetros es necesario cambiar en el dispositivo a partir de su configuración por defecto. En dichas figuras se numeran de forma gráfica dichos parámetros, de tal manera que en la *Tabla 5.13.* se recogen los datos a introducir en el *DSL* para programar este dispositivo. En la *Figura 5.49.* se muestra la descripción de estos parámetros utilizando el *DSL*.

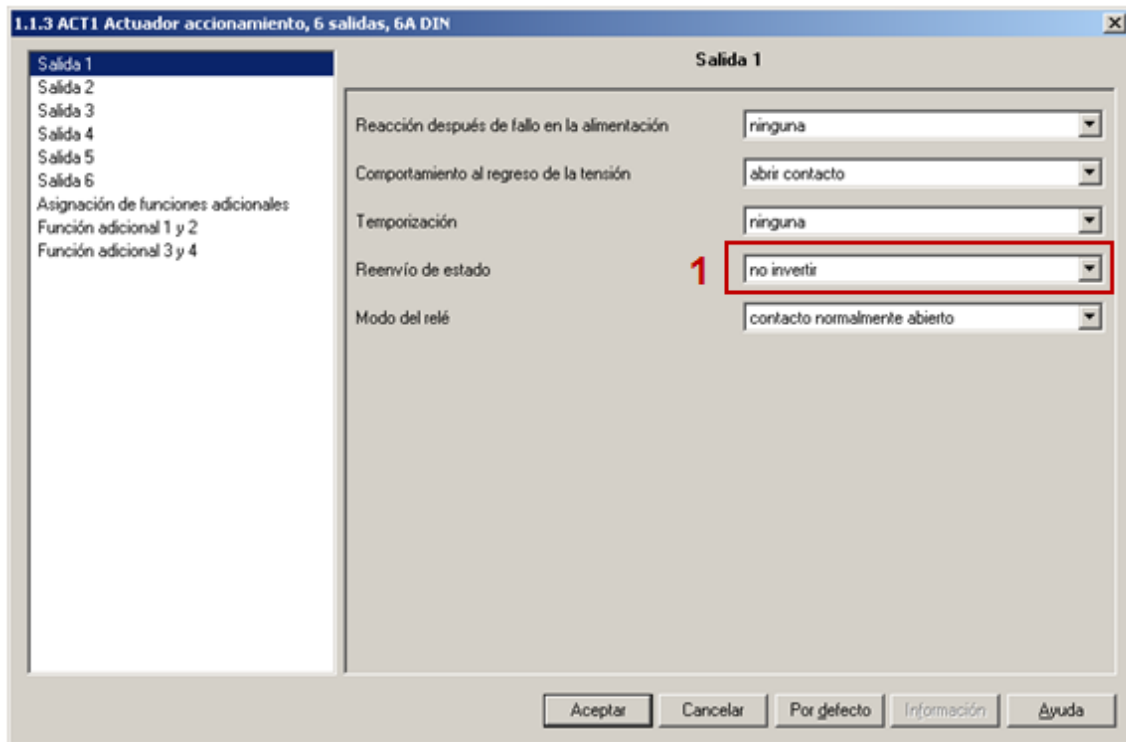


Figura 5.43. Análisis de los parámetros para el dispositivo "Actuador de 6 salidas (1)" (I).

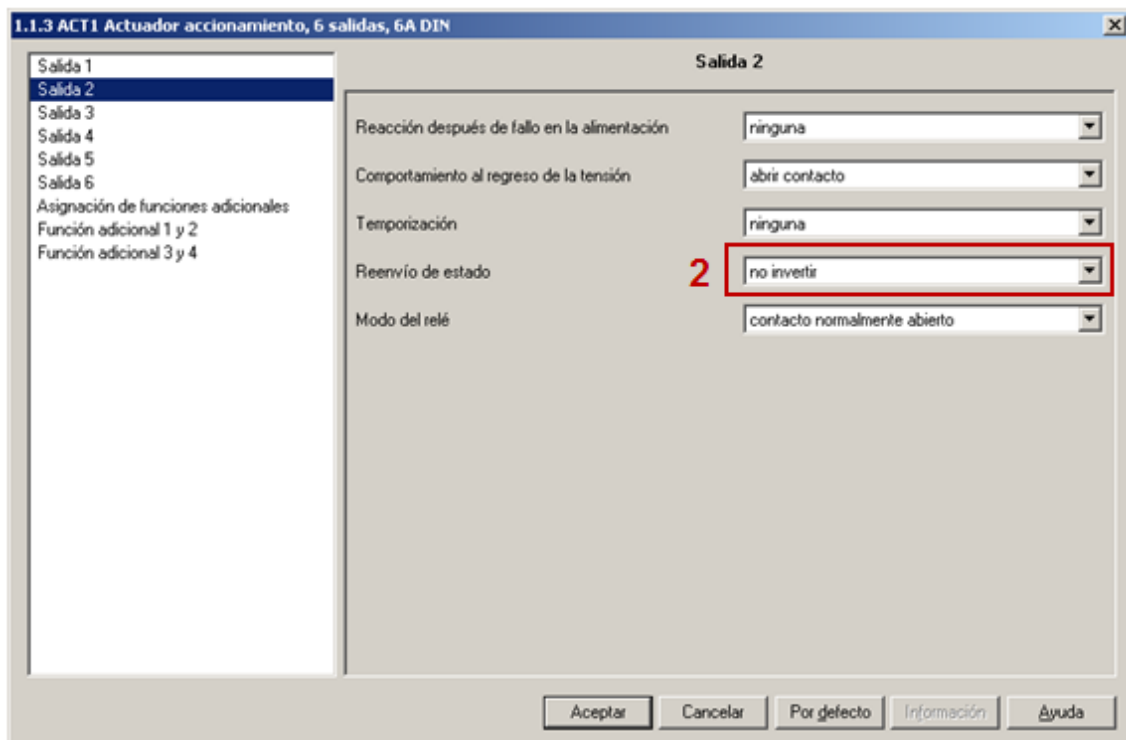


Figura 5.44. Análisis de los parámetros para el dispositivo "Actuador de 6 salidas (1)" (II).

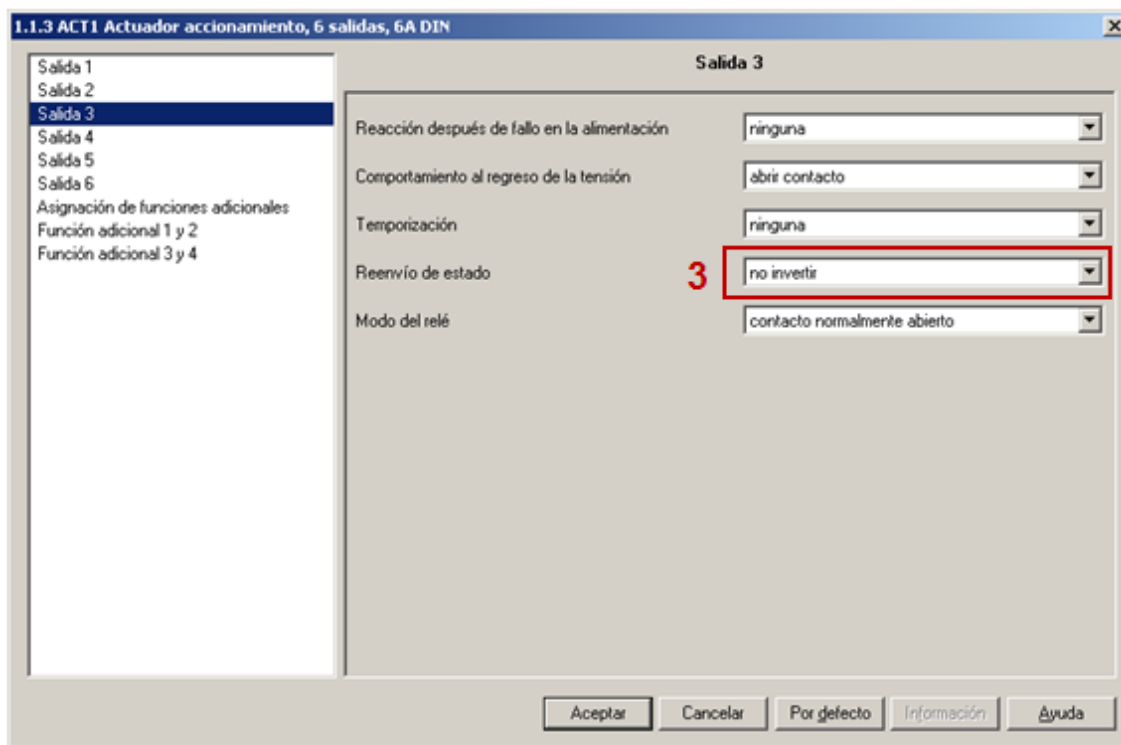


Figura 5.45. Análisis de los parámetros para el dispositivo "Actuador de 6 salidas (1)" (III).

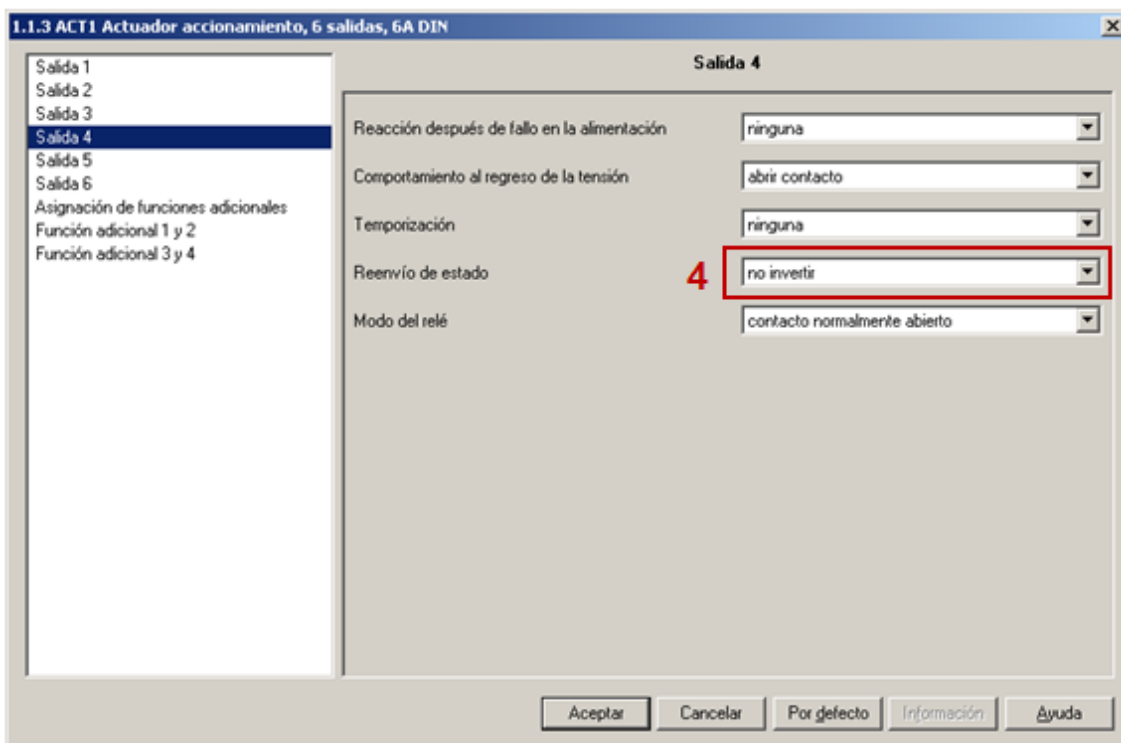


Figura 5.46. Análisis de los parámetros para el dispositivo "Actuador de 6 salidas (1)" (IV).



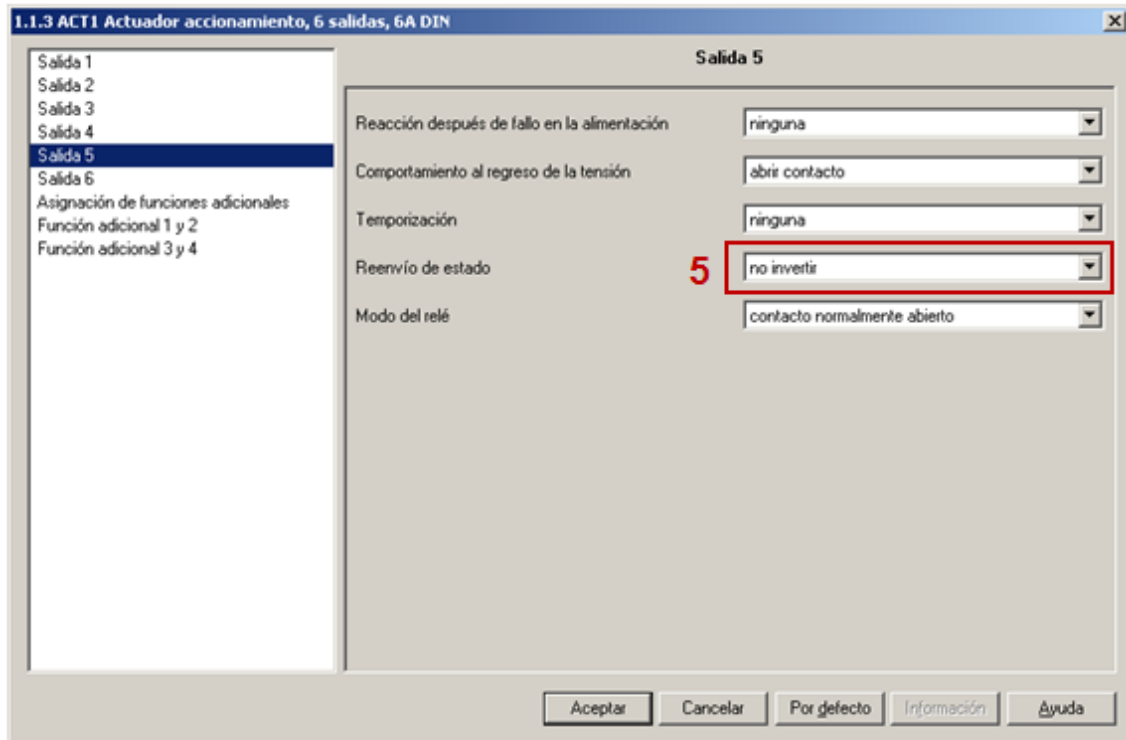


Figura 5.47. Análisis de los parámetros para el dispositivo "Actuador de 6 salidas (1)" (V).

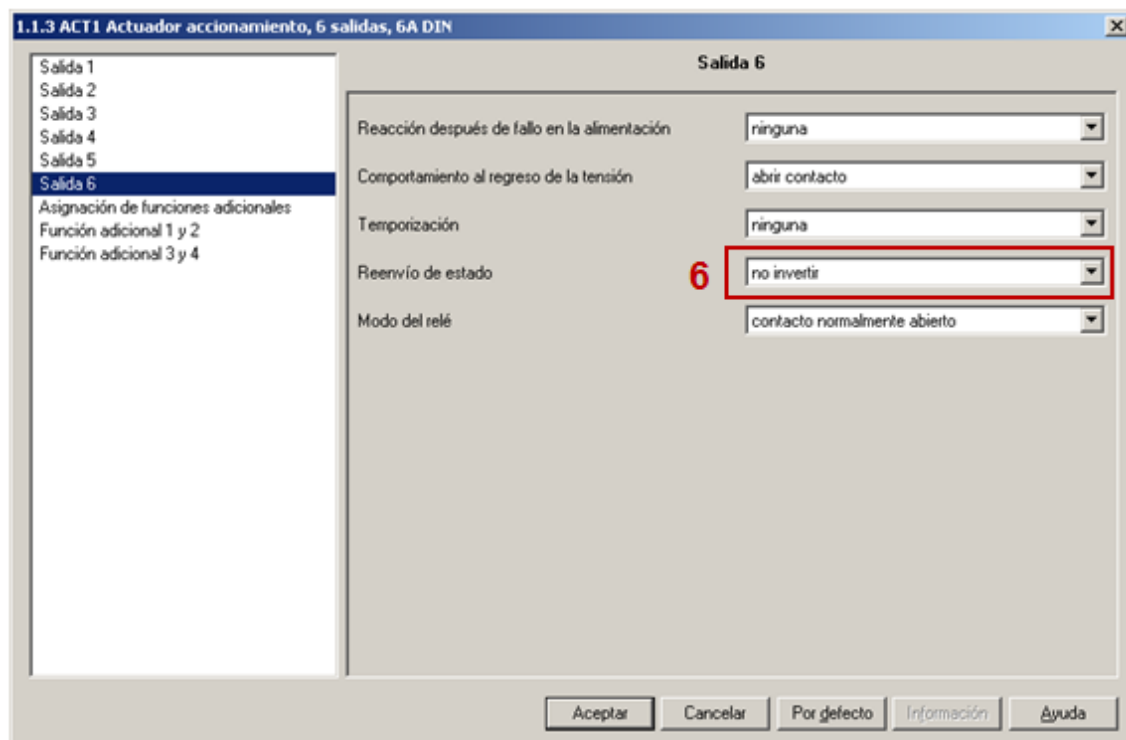


Figura 5.48. Análisis de los parámetros para el dispositivo "Actuador de 6 salidas (1)" (VI).

Numero	Nombre	Valor	Orden aparición
1	Rückmeldung	1	3
2	Rückmeldung	1	6
3	Rückmeldung	1	1
4	Rückmeldung	1	4
5	Rückmeldung	1	2
6	Rückmeldung	1	5

Tabla 5.13. Resumen de los parámetros para el dispositivo “Actuador de 6 salidas (1)”.

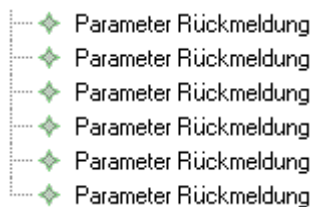


Figura 5.49. Descripción de los parámetros para el dispositivo “Actuador de 6 salidas (1)” utilizando el DSL propuesto en EMF.

#### 5.3.5.4. Análisis de los parámetros del dispositivo “Actuador de 6 salidas (2)”.

El análisis de los parámetros para este dispositivo es exactamente el mismo que para el dispositivo “Actuador de 6 salidas (1)”, por lo tanto sirva dicho análisis para ilustrar la parametrización del mismo y su implementación mediante el DSL en EMF.

#### 5.3.5.5. Análisis de los parámetros del dispositivo “Teclado una fase”.

En la Figura 5.50. se muestra de manera gráfica que parámetros es necesario cambiar en el dispositivo a partir de su configuración por defecto. En dicha figuras se numeran de forma gráfica dichos parámetros, de tal manera que en la Tabla 5.14. se recogen los datos a introducir en el DSL para programar este dispositivo. En la Figura 5.51. se muestra la descripción de estos parámetros utilizando el DSL.

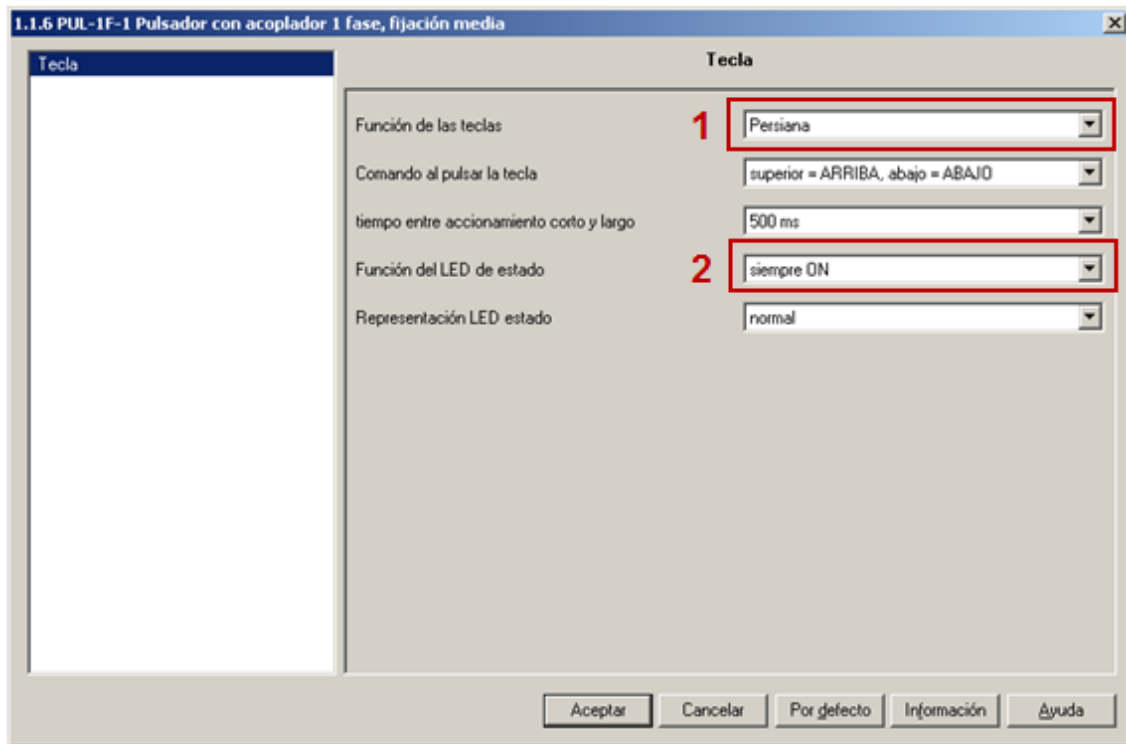


Figura 5.50. Análisis de los parámetros para el dispositivo “Teclado una fase”.

Numero	Nombre	Valor	Orden aparición
1	Grund-Funktion der Wippe links	1	1
2	LED Schalten	0	1

Tabla 5.14. Resumen de los parámetros para el dispositivo “Teclado una fase”.

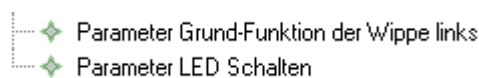


Figura 5.51. Descripción de los parámetros para el dispositivo “Teclado una fase” utilizando el DSL propuesto en EMF.

### 5.3.5.6. Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (1)”.

En las figuras de *Figura 5.52.* a *Figura 5.55.* se muestra de manera gráfica que parámetros es necesario cambiar en el dispositivo a partir de su configuración por defecto. En dichas figuras se numeran de forma gráfica dichos parámetros, de tal manera que en la *Tabla 5.15.* se recogen los datos a introducir en el *DSL* para programar este dispositivo. En la *Figura 5.6.* se muestra la descripción de estos parámetros utilizando el *DSL*.

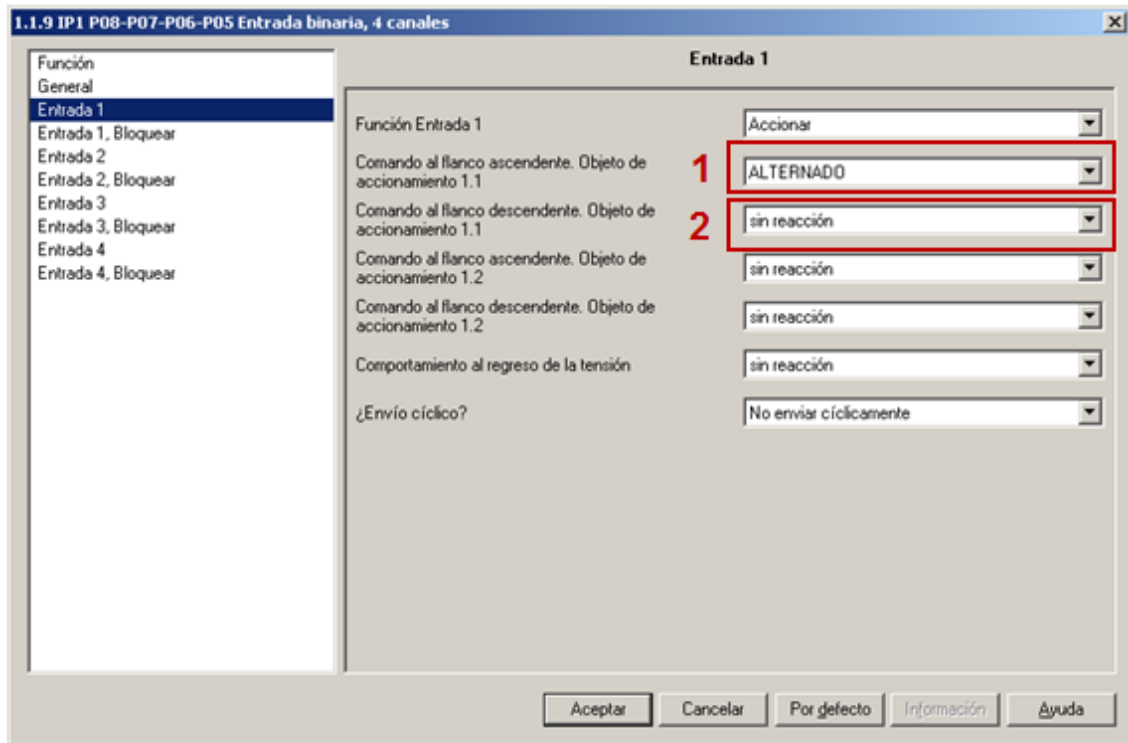


Figura 5.52. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” (I).

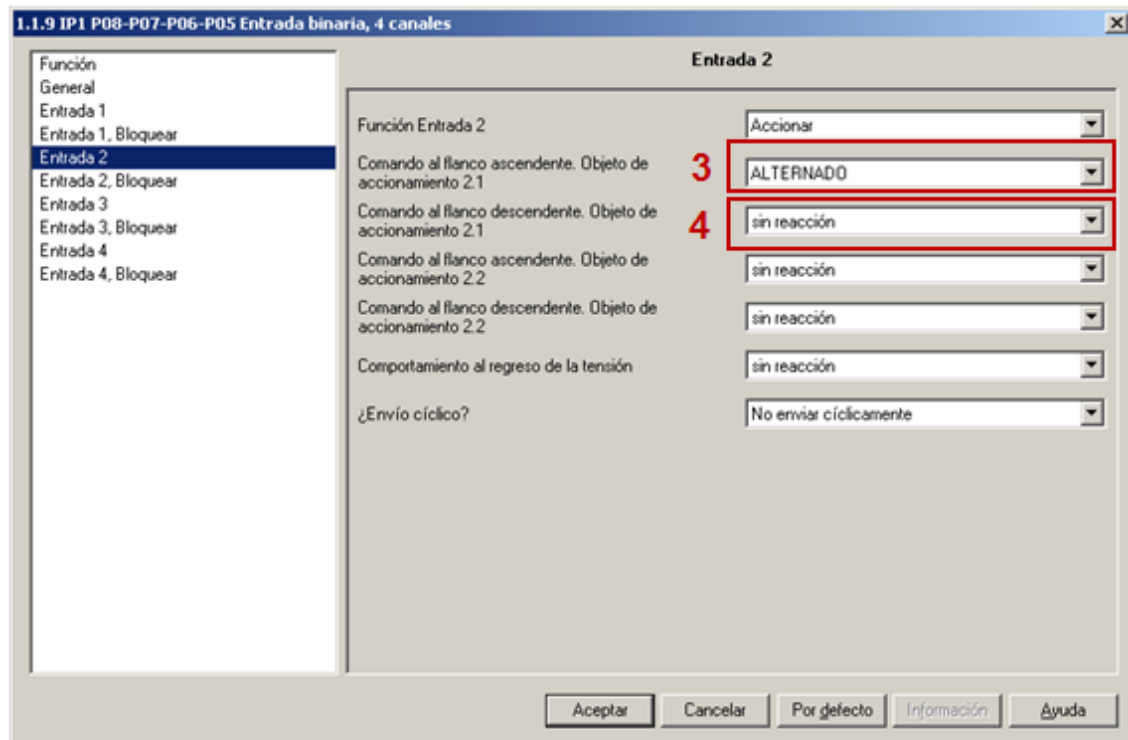


Figura 5.53. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” (II).

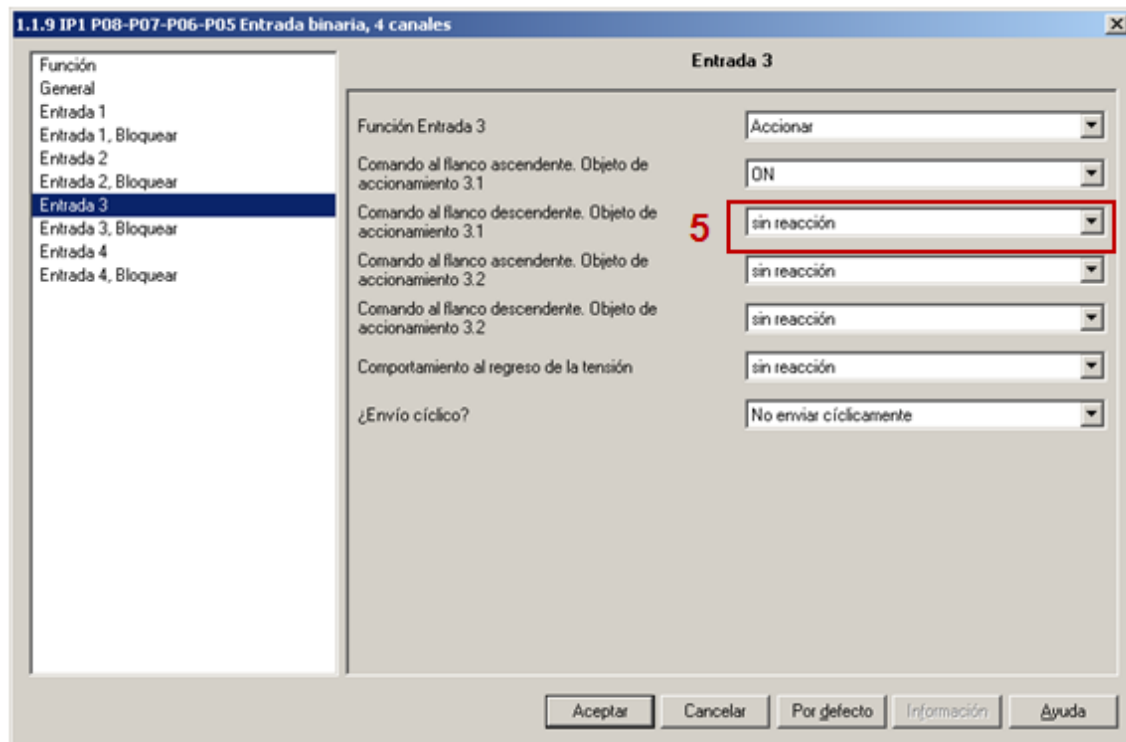


Figura 5.54. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” (III).

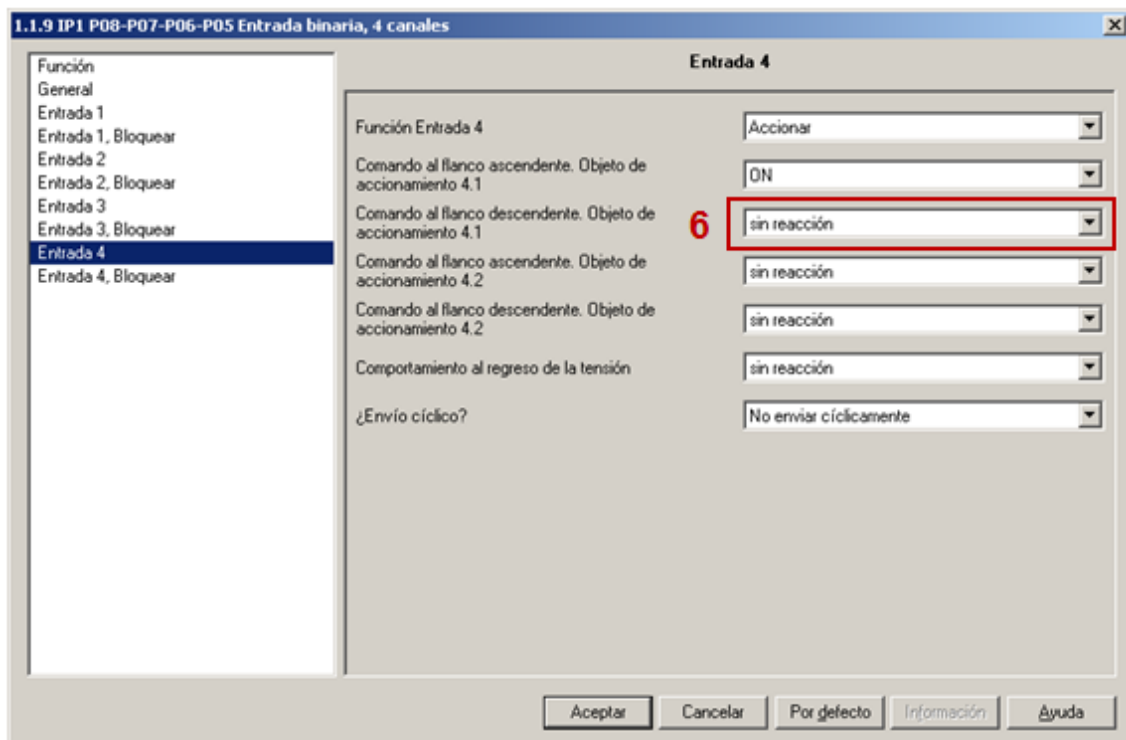


Figura 5.55. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” (IV).

Numero	Nombre	Valor	Orden aparición
1	Befehl_steigendeFlanke_O1	3	6
2	Befehl_fallendeFlanke_O1	0	14
3	Befehl_steigendeFlanke_O1	3	5
4	Befehl_fallendeFlanke_O1	0	9
5	Befehl_fallendeFlanke_O1	0	20
6	Befehl_fallendeFlanke_O1	0	19

Tabla 5.15. Resumen de los parámetros para el dispositivo “Entrada binaria 4 canales (1)”.



Figura 5.56. Descripción de los parámetros para el dispositivo “Entrada binaria 4 canales (1)” utilizando el DSL propuesto en EMF.

### 5.3.5.7. Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (2)”.

En las figuras de *Figura 5.57.* a *Figura 5.60.* se muestra de manera gráfica que parámetros es necesario cambiar en el dispositivo a partir de su configuración por defecto. En dichas figuras se numeran de forma gráfica dichos parámetros, de tal manera que en la *Tabla 5.16.* se recogen los datos a introducir en el *DSL* para programar este dispositivo. En la *Figura 5.61.* se muestra la descripción de estos parámetros utilizando el *DSL*.

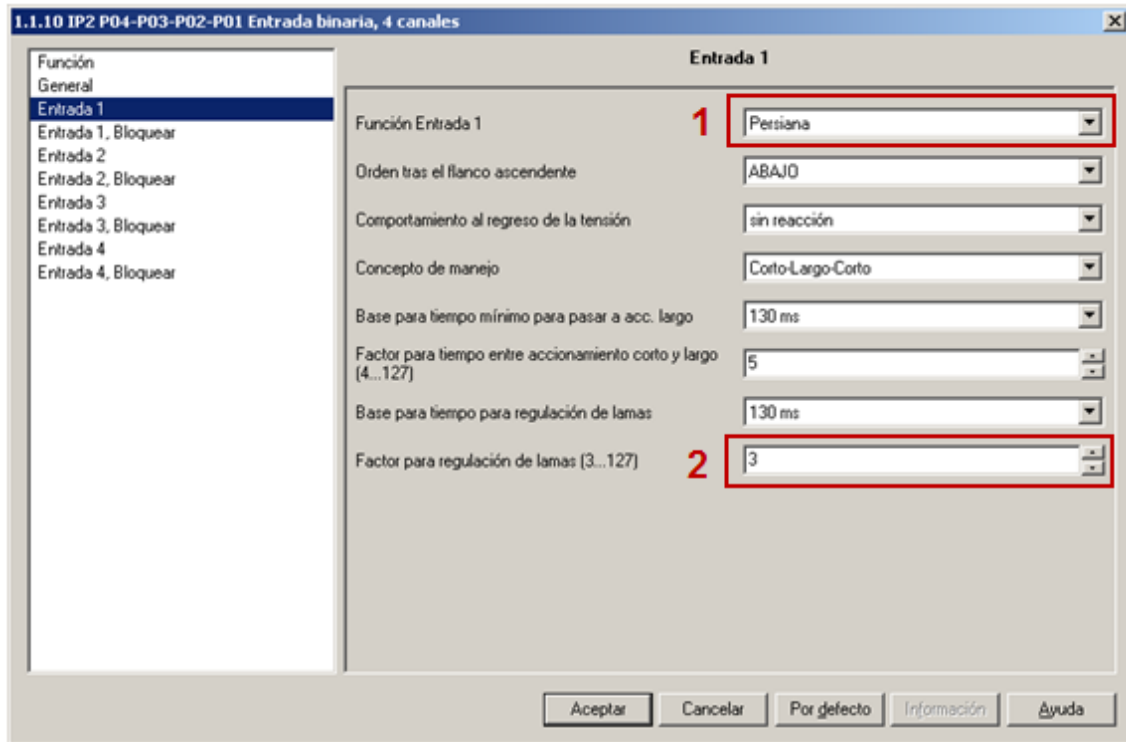


Figura 5.57. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” (I).

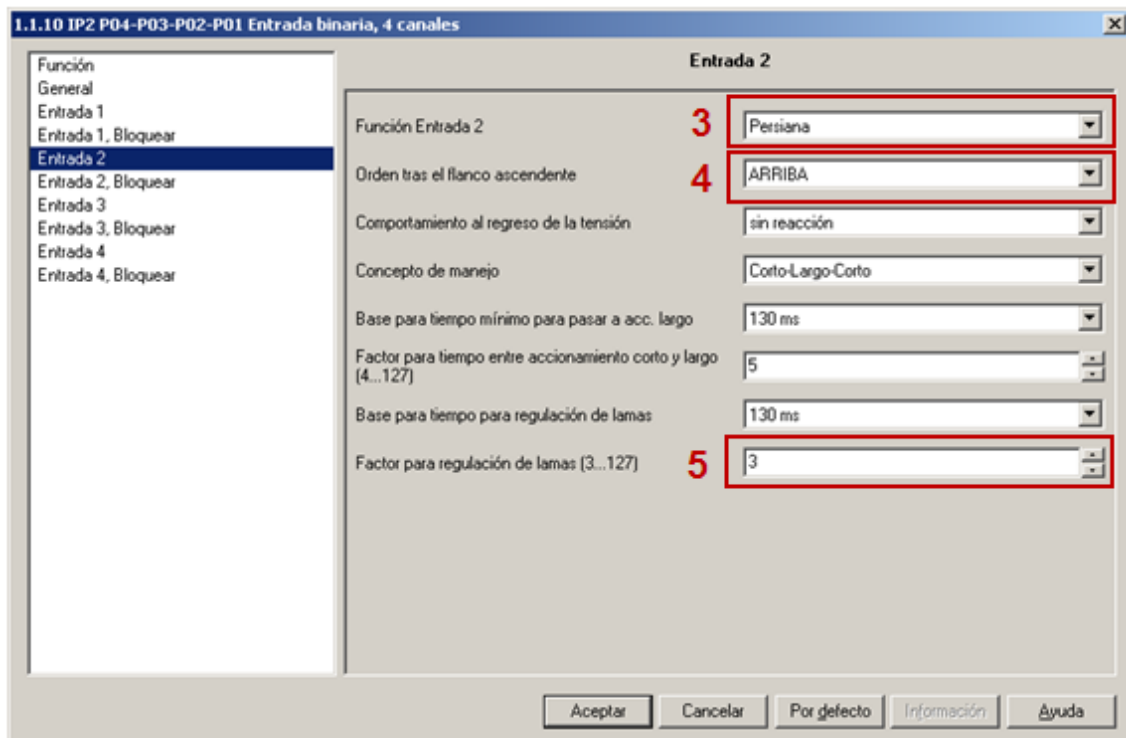


Figura 5.58. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” (II).

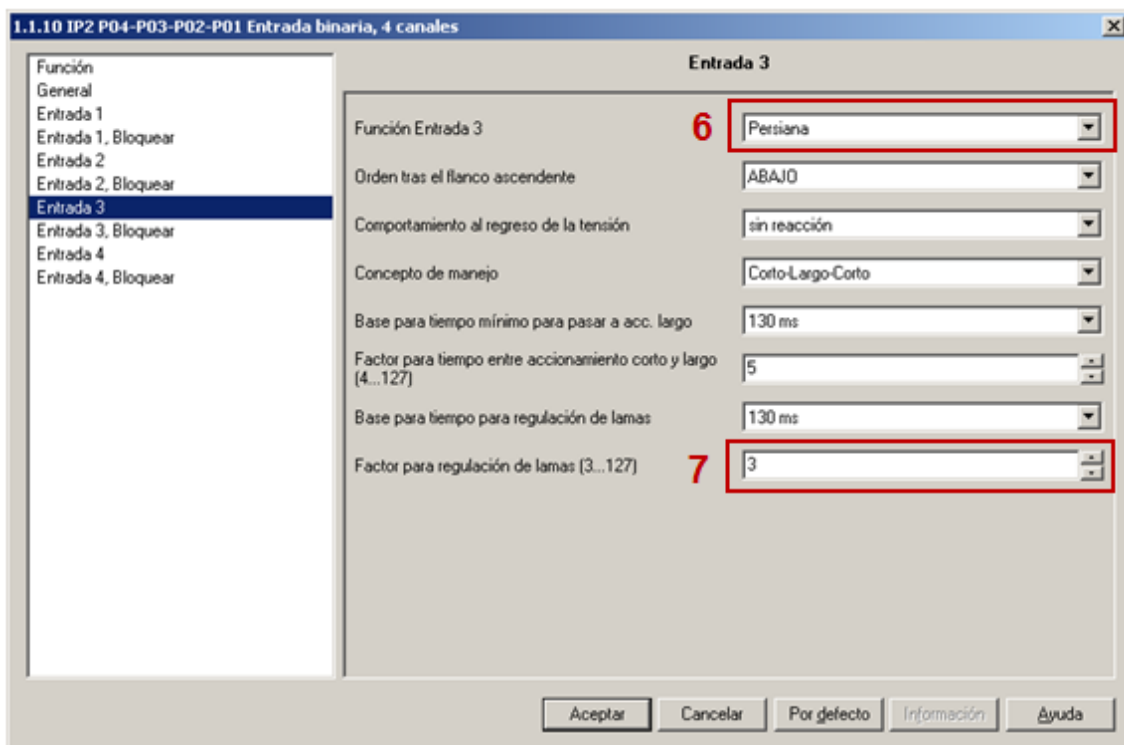


Figura 5.59. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” (III).

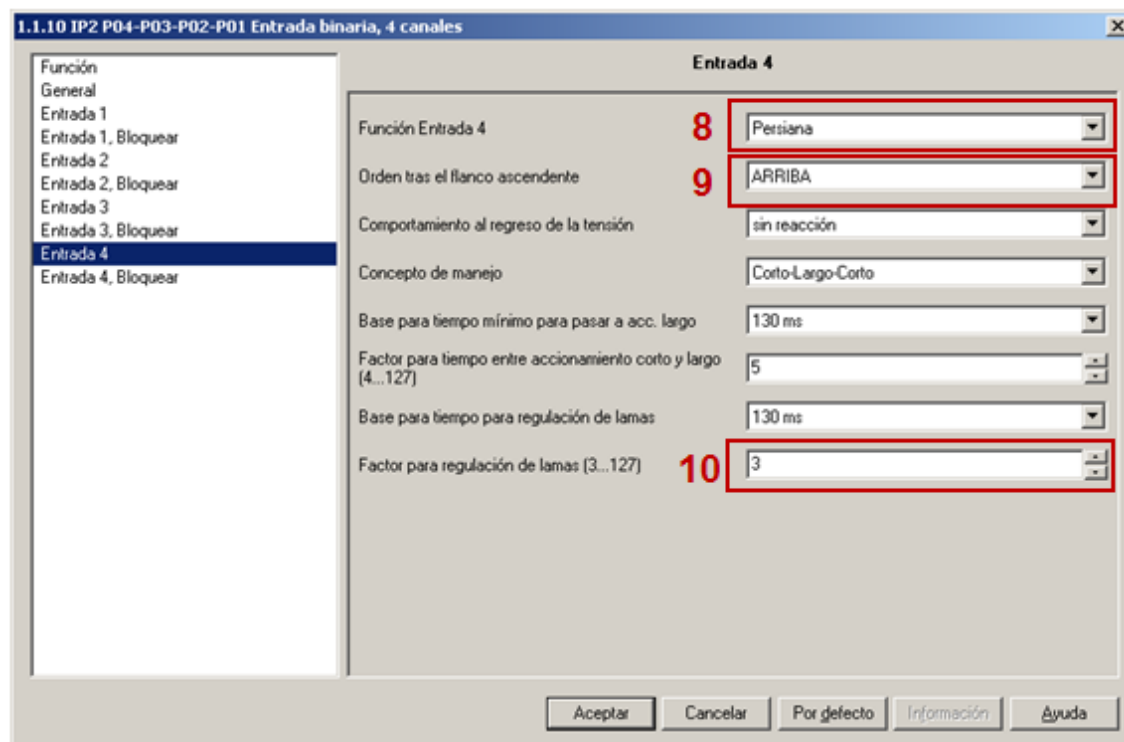


Figura 5.60. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” (IV).



Numero	Nombre	Valor	Orden aparición
1	funktion_k1	3	1
2	Lam_Verst_Faktor	3	6
3	funktion_k2	3	1
4	Befehl_steigendeFlanke	1	5
5	Lam_Verst_Faktor	3	5
6	funktion_k3	3	1
7	Lam_Verst_Faktor	3	7
8	funktion_k4	3	1
9	Befehl_steigendeFlanke	1	8
10	Lam_Verst_Faktor	3	8

Tabla 5.16. Resumen de los parámetros para el dispositivo “Entrada binaria 4 canales (2)”.

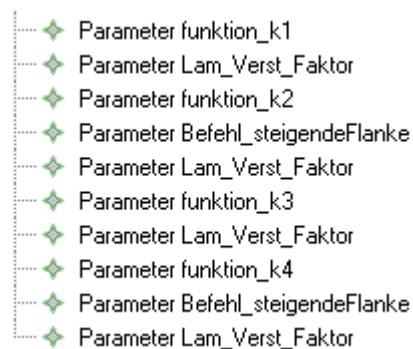


Figura 5.61. Descripción de los parámetros para el dispositivo “Entrada binaria 4 canales (2)” utilizando el DSL propuesto en EMF.

#### 5.3.5.8. Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (3)”.

En las figuras de *Figura 5.62.* a *Figura 5.66.* se muestra de manera gráfica que parámetros es necesario cambiar en el dispositivo a partir de su configuración por defecto. En dichas figuras se numeran de forma gráfica dichos parámetros, de tal manera que en la *Tabla 5.17.* se recogen los datos a introducir en el *DSL* para programar este dispositivo. En la *Figura 5.67.* se muestra la descripción de estos parámetros utilizando el *DSL*.

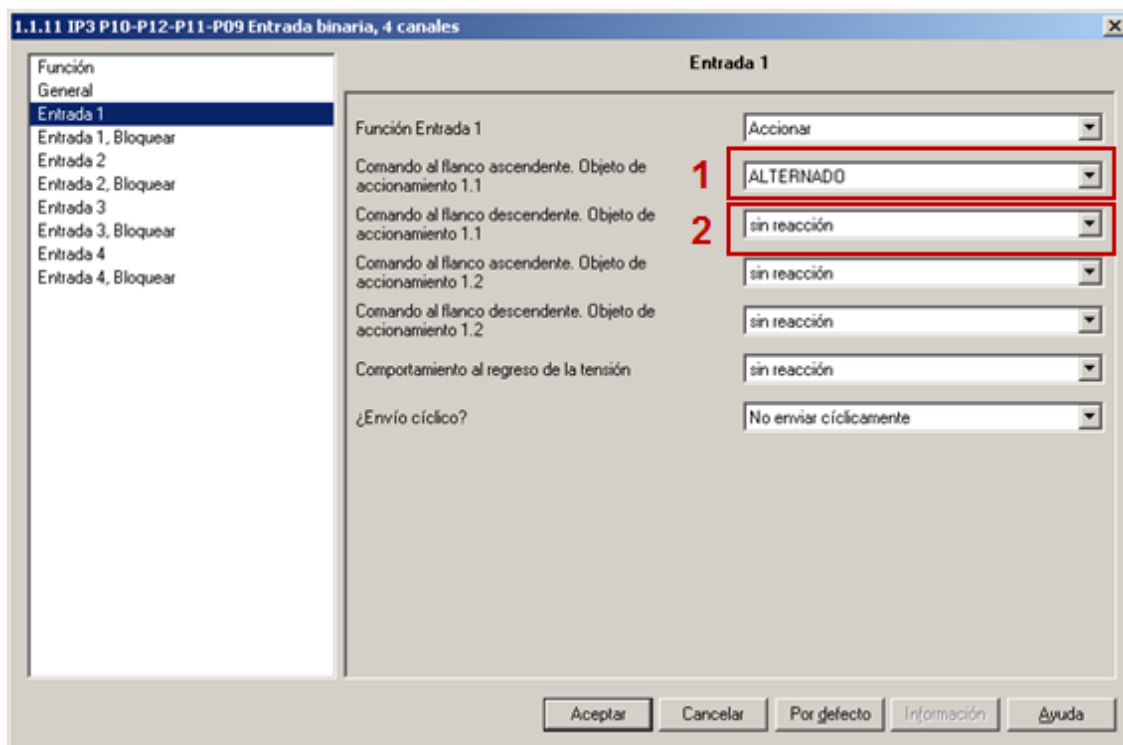


Figura 5.63. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (I).

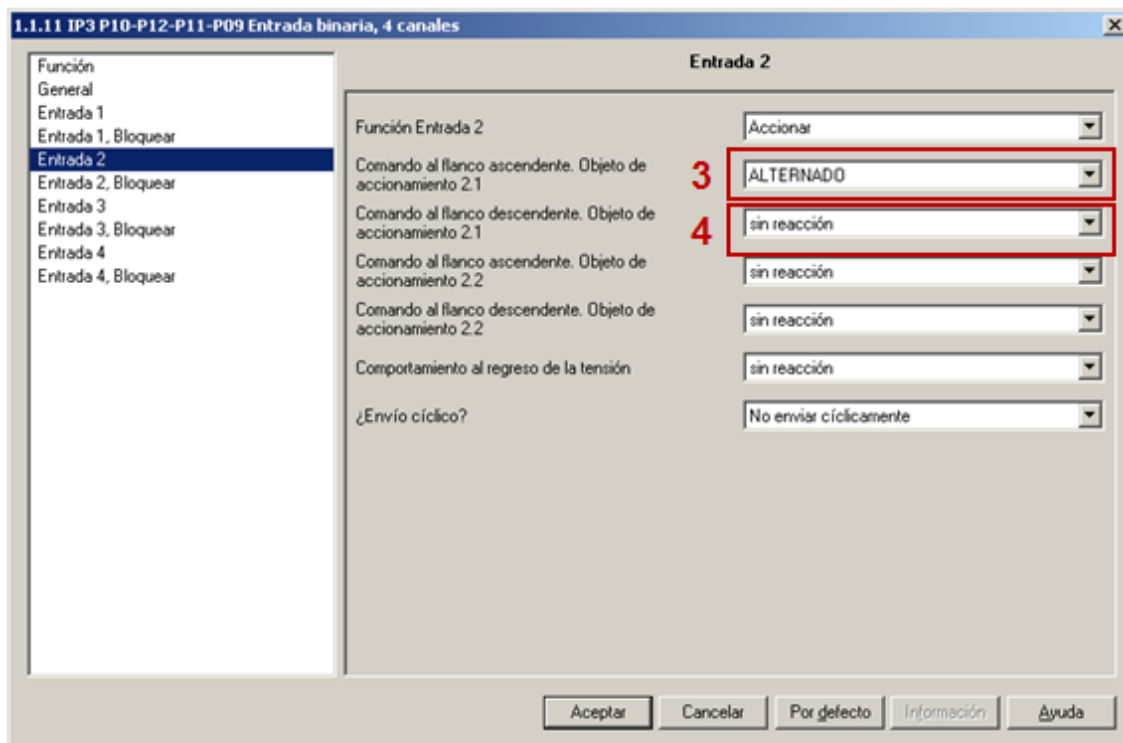


Figura 5.64. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (II).

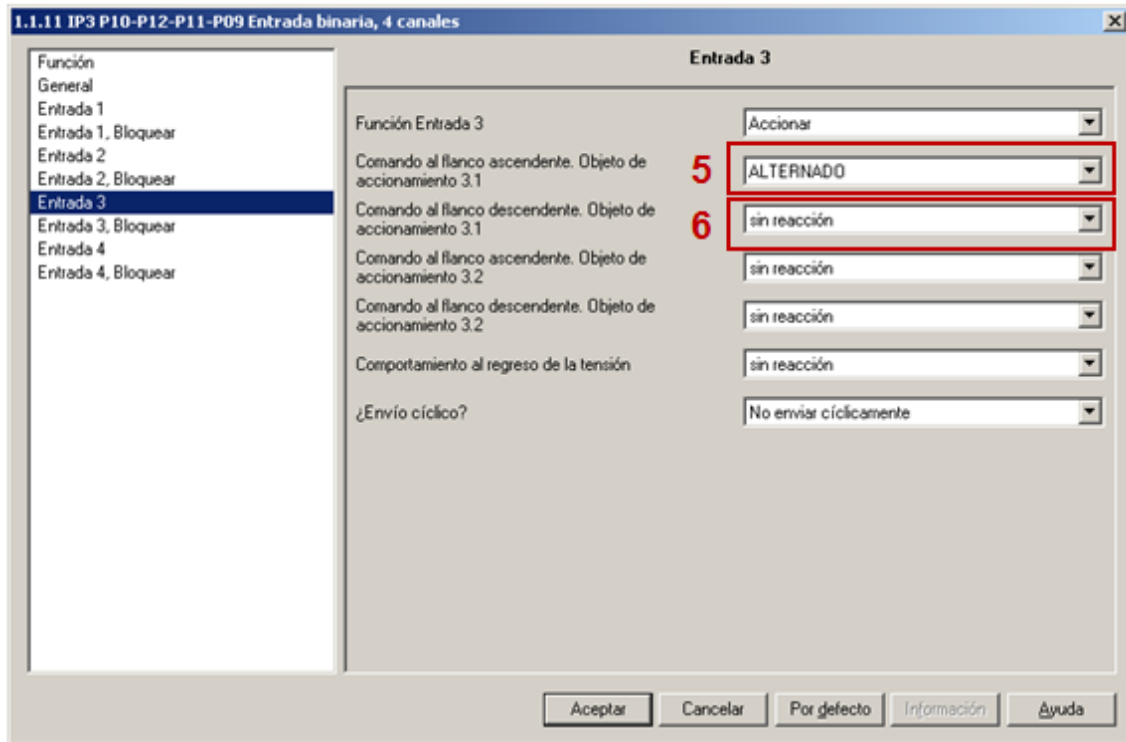


Figura 5.65. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (III).

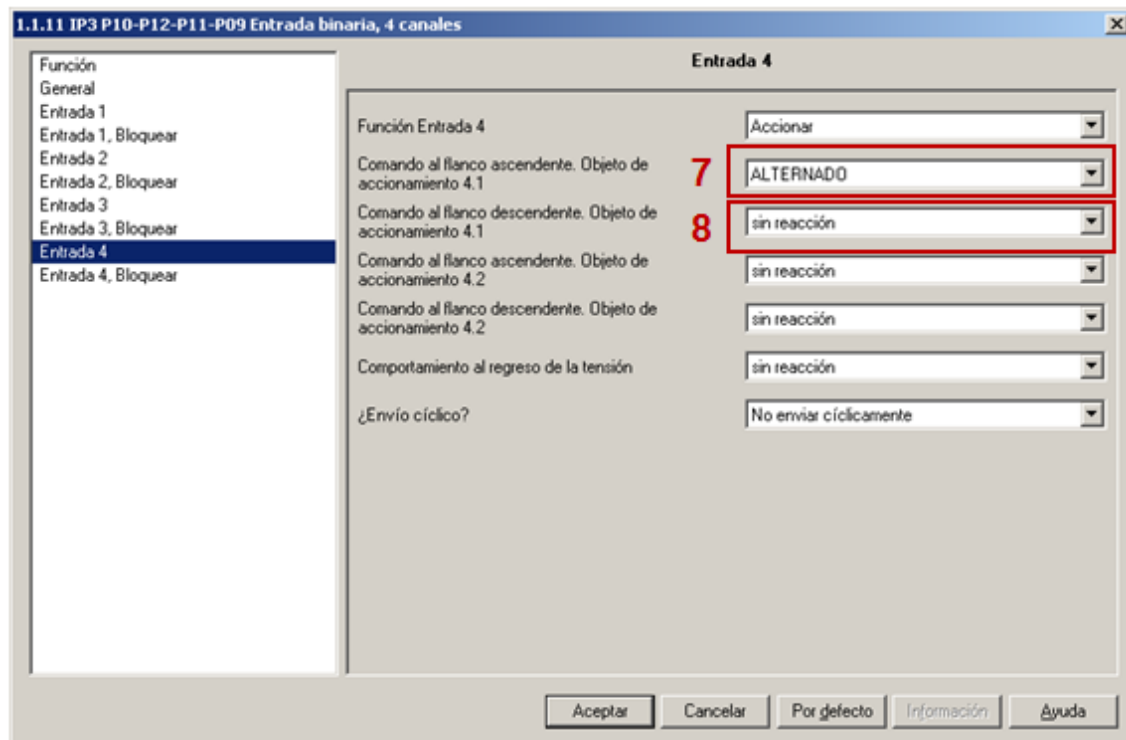


Figura 5.66. Análisis de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” (IV).

Numero	Nombre	Valor	Orden aparición
1	Befehl_steigendeFlanke_O1	3	6
2	Befehl_fallendeFlanke_O1	0	14
3	Befehl_steigendeFlanke_O1	3	5
4	Befehl_fallendeFlanke_O1	0	9
5	Befehl_steigendeFlanke_O1	3	8
6	Befehl_fallendeFlanke_O1	0	20
7	Befehl_steigendeFlanke_O1	3	7
8	Befehl_fallendeFlanke_O1	0	19

Tabla 5.17. Resumen de los parámetros para el dispositivo “Entrada binaria 4 canales (3)”.



Figura 5.67. Descripción de los parámetros para el dispositivo “Entrada binaria 4 canales (3)” utilizando el DSL propuesto en EMF.

#### 5.3.5.9. Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (4)”.

El análisis de los parámetros para este dispositivo es exactamente el mismo que para el dispositivo “Entrada binaria 4 canales (3)”, por lo tanto sirva dicho análisis para ilustrar la parametrización del mismo y su implementación mediante el DSL en EMF.

#### 5.3.5.10. Análisis de los parámetros del dispositivo “Entrada binaria 4 canales (5)”.

El análisis de los parámetros para este dispositivo es exactamente el mismo que para el dispositivo “Entrada binaria 4 canales (2)”, por lo tanto sirva dicho análisis para ilustrar la parametrización del mismo y su implementación mediante el DSL en EMF.

#### 5.3.5.11. Análisis de los parámetros del dispositivo “Pulsador 4 fases”.

Este dispositivo no necesita ser configurado, ya que todos sus parámetros toman los valores por defecto.

### 5.3.6. Análisis de los objetos de comunicación.

En los siguientes sub-apartados se va a analizar individualmente los objetos de comunicación pertenecientes a cada dispositivo del proyecto, así como sus asociaciones a las direcciones de grupo; de tal manera que se pueda modelar mediante el *DSL* la comunicación entre los dispositivos necesaria para obtener la funcionalidad descrita en el *Apartado 5.3.2*.

#### 5.3.6.1. Análisis de los objetos de comunicación del dispositivo “Regulador de fluorescencia”.

En la *Figura 5.68*. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el *DSL* puede verse en la *Figura 5.69*.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
1	Salida 2 / Accionar / Lamp02 Regulada	1/0/2 Iluminación Todas	Falso
		1/1/4 LR02 Salon ON/OFF	Verdadero
2	Salida 3 / Accionar / Bombilla Fluorescente	1/1/8 Bombilla Fluorescente ON/OFF	Verdadero
		1/0/2 Iluminación Todas	Falso
3	Salida 1 / Regular / Lamp01 Regulada	1/1/1 LR01 Dormitorio REGULACIÓN	Verdadero
4	Salida 2 / Regular / Lamp02 Regulada	1/1/5 LR02 Salon REGULACIÓN	Verdadero
5	Salida 3 / Regular / Bombilla Fluorescente	1/1/9 Bombilla Fluorescente REGULACIÓN	Verdadero
6	Salida 1 / Valor luminoso / Lamp01 Regulada	1/1/3 LR01 Dormitorio VALOR	Verdadero
7	Salida 2 / Valor luminoso / Lamp02 Regulada	1/1/7 LR02 Salon VALOR	Verdadero
8	Salida 3 / Valor luminoso / Bombilla Fluorescente	1/1/11 Bombilla Fluorescente VALOR	Verdadero
0	Salida 1 / Accionar / Lamp01 Regulada	1/0/2 Iluminación Todas	Falso
		1/1/0 LR01 Dormitorio ON/OFF	Verdadero
9	Salida 1 / Reenvío de estado / Lamp01 Regulada	1/1/2 LR01 Dormitorio ESTADO	Verdadero
10	Salida 2 / Reenvío de estado / Lamp02 Regulada	1/1/6 LR02 Salon ESTADO	Verdadero
11	Salida 3 / Reenvío de estado / Bombilla Fluorescente	1/1/10 Bombilla Fluorescente ESTADO	Verdadero

Figura 5.68. Objetos de comunicación y asociaciones para el dispositivo “Regulador de fluorescencia”.

- ◆ Comm Object Regulador fluorescencia - 1
- ◆ Comm Object Regulador fluorescencia - 2
- ◆ Comm Object Regulador fluorescencia - 3
- ◆ Comm Object Regulador fluorescencia - 4
- ◆ Comm Object Regulador fluorescencia - 5
- ◆ Comm Object Regulador fluorescencia - 6
- ◆ Comm Object Regulador fluorescencia - 7
- ◆ Comm Object Regulador fluorescencia - 8
- ◆ Comm Object Regulador fluorescencia - 0
- ◆ Comm Object Regulador fluorescencia - 9
- ◆ Comm Object Regulador fluorescencia - 10
- ◆ Comm Object Regulador fluorescencia - 11

Figura 5.69. Descripción en EMF de los objetos de comunicación para el dispositivo “Regulador de fluorescencia” usando el DSL propuesto.

### 5.3.6.2. Análisis de los objetos de comunicación del dispositivo “Actuador de persianas”.

En la Figura 5.70. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la Figura 5.71.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
12	Output A / Stop Up-Down / Persiana1 Stop	1/0/1 Stop Todas Persianas	Falso
		1/2/1 P01 Persiana1 Dorm STOP	Verdadero
42	Output B / Stop Up-Down / Persiana2 Stop	1/0/1 Stop Todas Persianas	Falso
		1/2/3 P02 Persiana2 Dorm ppal STOP	Verdadero
41	Output B / Move shutter Up-Down / Persiana2 Mover	1/0/0 Mover Todas Persianas	Falso
		1/2/2 P02 Persiana2 Dorm MOV	Verdadero
11	Output A / Move shutter Up-Down / Persiana1 Mover	1/0/0 Mover Todas Persianas	Falso
		1/2/0 P01 Persiana1 Dorm MOV	Verdadero
72	Output C / Stop Up-Down / Persiana3 Stop	1/0/1 Stop Todas Persianas	Falso
		1/2/5 P03 Persiana3 Dorm STOP	Verdadero
71	Output C / Move shutter Up-Down / Persiana3 Mover	1/0/0 Mover Todas Persianas	Falso
		1/2/4 P03 Persiana3 Dorm MOV	Verdadero
102	Output D / Stop Up-Down / Persiana4 Stop	1/0/1 Stop Todas Persianas	Falso
		1/2/7 P04 Persiana4 Dorm STOP	Verdadero
101	Output D / Move shutter Up-Down / Persiana4 Mover	1/0/0 Mover Todas Persianas	Falso
		1/2/6 P04 Persiana4 Dorm MOV	Verdadero

Figura 5.70. Objetos de comunicación y asociaciones para el dispositivo “Actuador de persianas”.

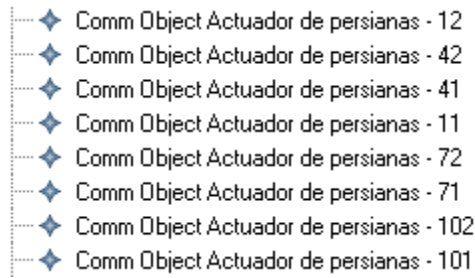


Figura 5.71. Descripción en EMF de los objetos de comunicación para el dispositivo “Actuador de persianas” usando el DSL propuesto.

5.3.6.3. Análisis de los objetos de comunicación del dispositivo “Actuador de 6 salidas (1)”.

En la Figura 5.72. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la Figura 5.73.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
17	Salida 6 / Reenvío de estado / Lamp09 Estado	1/1/25 L09 ESTADO	Verdadero
5	Salida 6 / Accionar / Lamp09	1/0/2 Iluminación Todas	Verdadero
		1/1/24 L09 ON/OFF	Falso
4	Salida 5 / Accionar / Lamp10	1/1/26 L10 ON/OFF	Verdadero
		1/0/2 Iluminación Todas	Falso
16	Salida 5 / Reenvío de estado / Lamp10 Estado	1/1/27 L10 ESTADO	Verdadero

Figura 5.72. Objetos de comunicación y asociaciones para el dispositivo “Actuador de 6 salidas (1)”.

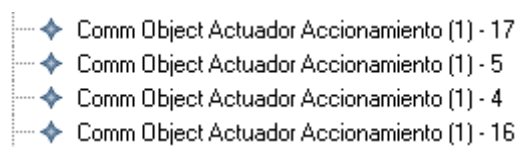


Figura 5.73. Descripción en EMF de los objetos de comunicación para el dispositivo “Actuador de 6 salidas (1)” usando el DSL propuesto.

5.3.6.4. Análisis de los objetos de comunicación del dispositivo “Actuador de 6 salidas (2)”.

En la Figura 5.74. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la Figura 5.75.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
15	Salida 4 / Reenvío de estado / Lamp05 Estado	1/1/17 L05 ESTADO	Verdadero
17	Salida 6 / Reenvío de estado / Lamp03 Estado	1/1/13 L03 Aseo ESTADO	Verdadero
12	Salida 1 / Reenvío de estado / Lamp08 Estado	1/1/23 L08 ESTADO	Verdadero
5	Salida 6 / Accionar / Lamp03	1/0/2 Iluminación Todas	Verdadero
2	Salida 3 / Accionar / Lamp06	1/1/12 L03 Aseo ON/OFF	Falso
		1/0/2 Iluminación Todas	Verdadero
0	Salida 1 / Accionar / Lamp08	1/1/18 L06 ON/OFF	Falso
		1/0/2 Iluminación Todas	Verdadero
3	Salida 4 / Accionar / Lamp05	1/1/22 L08 ON/OFF	Falso
		1/0/2 Iluminación Todas	Verdadero
13	Salida 2 / Reenvío de estado / Lamp07 Estado	1/1/16 L05 ON/OFF	Falso
		1/1/21 L07 ESTADO	Verdadero
1	Salida 2 / Accionar / Lamp07	1/0/2 Iluminación Todas	Verdadero
		1/1/20 L07 ON/OFF	Falso
14	Salida 3 / Reenvío de estado / Lamp06 Estado	1/1/19 L06 ESTADO	Verdadero
4	Salida 5 / Accionar / Lamp04	1/0/2 Iluminación Todas	Verdadero
		1/1/14 L04 ON/OFF	Falso
16	Salida 5 / Reenvío de estado / Lamp04 Estado	1/1/15 L04 ESTADO	Verdadero

Figura 5.74. Objetos de comunicación y asociaciones para el dispositivo “Actuador de 6 salidas (2)”.



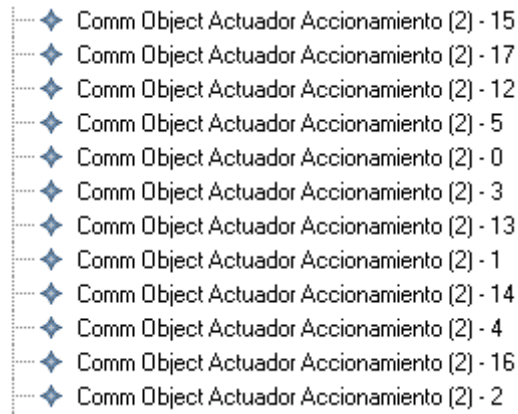


Figura 5.75. Descripción en EMF de los objetos de comunicación para el dispositivo “Actuador de 6 salidas (2)” usando el DSL propuesto.

### 5.3.6.5. Análisis de los objetos de comunicación del dispositivo “Teclado una fase”.

En la Figura 5.76. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la Figura 5.77.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
1	Tecla / Accionamiento largo	1/0/0 Mover Todas Persianas	Verdadero
0	Tecla / Accionamiento corto	1/0/1 Stop Todas Persianas	Verdadero
		1/0/2 Iluminación Todas	Falso

Figura 5.76. Objetos de comunicación y asociaciones para el dispositivo “Teclado una fase”.

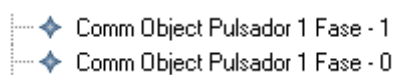


Figura 5.77. Descripción en EMF de los objetos de comunicación para el dispositivo “Teclado una fase” usando el DSL propuesto.

### 5.3.6.6. Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (1)”.

En la Figura 5.78. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la Figura 5.79.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
0	Entrada 1 / Objeto accionamiento 1.1 / Pulsador08	Dirección de grupo: 1/1/4 LR02 Salon ON/OFF	Verdadero
		Dirección de grupo: 1/1/6 LR02 Salon ESTADO	Falso
1	Entrada 2 / Objeto accionamiento 2.1 / Pulsador07	1/1/27 L10 ESTADO	Falso
		1/1/26 L10 ON/OFF	Verdadero

Figura 5.78. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (1)”.

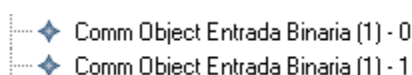


Figura 5.79. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (1)” usando el DSL propuesto.

### 5.3.6.7. Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (2)”.

En la Figura 5.80. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la Figura 5.81.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
2	Entrada 3 / Accionamiento corto / Pulsador02	1/2/1 P01 Persiana1 Dorm STOP	Verdadero
10	Entrada 3 / Accionamiento largo / Pulsador02	1/2/0 P01 Persiana1 Dorm MOV	Verdadero
9	Entrada 2 / Accionamiento largo / Pulsador03	1/2/2 P02 Persiana2 Dorm MOV	Verdadero
1	Entrada 2 / Accionamiento corto / Pulsador03	1/2/3 P02 Persiana2 Dorm ppal STOP	Verdadero
0	Entrada 1 / Accionamiento corto / Pulsador04	1/2/3 P02 Persiana2 Dorm ppal STOP	Verdadero
8	Entrada 1 / Accionamiento largo / Pulsador04	1/2/2 P02 Persiana2 Dorm MOV	Verdadero
3	Entrada 4 / Accionamiento corto / Pulsador01	1/2/1 P01 Persiana1 Dorm STOP	Verdadero
11	Entrada 4 / Accionamiento largo / Pulsador01	1/2/0 P01 Persiana1 Dorm MOV	Verdadero

Figura 5.80. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (2)”.



Figura 5.81. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (2)” usando el DSL propuesto.

5.3.6.8. Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (3)”.

En la Figura 5.82. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la Figura 5.83.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
2	Entrada 3 / Objeto accionamiento 3.1 / Pulsador11	1/1/14 L04 ON/OFF	Verdadero
		1/1/15 L04 ESTADO	Falso
0	Entrada 1 / Objeto accionamiento 1.1 / Pulsador10	1/1/24 L09 ON/OFF	Verdadero
		1/1/25 L09 ESTADO	Falso
1	Entrada 2 / Objeto accionamiento 2.1 / Pulsador12	1/1/12 L03 Aseo ON/OFF	Verdadero
		1/1/13 L03 Aseo ESTADO	Falso
3	Entrada 4 / Objeto accionamiento 4.1 / Pulsador09	1/1/22 L08 ON/OFF	Verdadero
		1/1/23 L08 ESTADO	Falso

Figura 5.82. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (3)”.

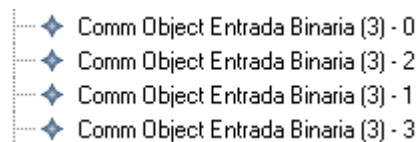


Figura 5.83. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (3)” usando el DSL propuesto.

5.3.6.9. Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (4)”.

En la Figura 5.84. se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la Figura 5.85.

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
2	Entrada 3 / Objeto accionamiento 3.1 / Pulsador14	1/1/16 L05 ON/OFF	Verdadero
		1/1/17 L05 ESTADO	Falso
0	Entrada 1 / Objeto accionamiento 1.1 / Pulsador16	1/1/20 L07 ON/OFF	Verdadero
		1/1/21 L07 ESTADO	Falso
1	Entrada 2 / Objeto accionamiento 2.1 / Pulsador15	1/1/18 L06 ON/OFF	Verdadero
		1/1/19 L06 ESTADO	Falso
3	Entrada 4 / Objeto accionamiento 4.1 / Pulsador13	1/1/0 LR01 Dormitorio ON/OFF	Verdadero
		1/1/2 LR01 Dormitorio ESTADO	Falso

Figura 5.84. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (4)”.

- ◆ Comm Object Entrada Binaria (4) - 2
- ◆ Comm Object Entrada Binaria (4) - 0
- ◆ Comm Object Entrada Binaria (4) - 1
- ◆ Comm Object Entrada Binaria (4) - 3

Figura 5.85. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (4)” usando el DSL propuesto.

### 5.3.6.10. Análisis de los objetos de comunicación del dispositivo “Entrada binaria 4 canales (5)”.

En la *Figura 5.86.* se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la *Figura 5.87.*

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
2	Entrada 3 / Accionamiento corto / Pulsador18	1/2/5 P03 Persiana3 Dorm STOP	Verdadero
10	Entrada 3 / Accionamiento largo / Pulsador18	1/2/4 P03 Persiana3 Dorm MOV	Verdadero
9	Entrada 2 / Accionamiento largo / Pulsador19	1/2/6 P04 Persiana4 Dorm MOV	Verdadero
1	Entrada 2 / Accionamiento corto / Pulsador19	1/2/7 P04 Persiana4 Dorm STOP	Verdadero
0	Entrada 1 / Accionamiento corto / Pulsador20	1/2/7 P04 Persiana4 Dorm STOP	Verdadero
8	Entrada 1 / Accionamiento largo / Pulsador20	1/2/6 P04 Persiana4 Dorm MOV	Verdadero
3	Entrada 4 / Accionamiento corto / Pulsador17	1/2/5 P03 Persiana3 Dorm STOP	Verdadero
11	Entrada 4 / Accionamiento largo / Pulsador17	1/2/4 P03 Persiana3 Dorm MOV	Verdadero

*Figura 5.86. Objetos de comunicación y asociaciones para el dispositivo “Entrada binaria 4 canales (5)”.*



*Figura 5.87. Descripción en EMF de los objetos de comunicación para el dispositivo “Entrada binaria 4 canales (5)” usando el DSL propuesto.*

5.3.6.11. Análisis de los objetos de comunicación del dispositivo “Pulsador 4 fases”.

En la *Figura 5.88.* se detallan los objetos de comunicación del dispositivo en cuestión relevantes para la configuración funcional requerida, y su descripción usando el DSL puede verse en la *Figura 5.89.*

Nº Objeto Comunicación	Correspondiente al parámetro	Asociado a la dirección	Sending
0	Tecla 1 / Accionamiento corto /	1/2/1 P01 Persiana1 Dorm STOP	Verdadero
1	Tecla 2 / Accionamiento corto /	1/2/5 P03 Persiana3 Dorm STOP	Verdadero
2	Tecla 3 / Accionamiento corto /	1/2/3 P02 Persiana2 Dorm ppal STOP	Verdadero
3	Tecla 4 / Accionamiento corto /	1/2/7 P04 Persiana4 Dorm STOP	Verdadero
4	Tecla 1 / Accionamiento largo /	1/2/0 P01 Persiana1 Dorm MOV	Verdadero
5	Tecla 2 / Accionamiento largo /	1/2/4 P03 Persiana3 Dorm MOV	Verdadero
6	Tecla 3 / Accionamiento largo /	1/2/2 P02 Persiana2 Dorm MOV	Verdadero
7	Tecla 4 / Accionamiento largo /	1/2/6 P04 Persiana4 Dorm MOV	Verdadero

Figura 5.88. Objetos de comunicación y asociaciones para el dispositivo “Pulsador 4 fases”.

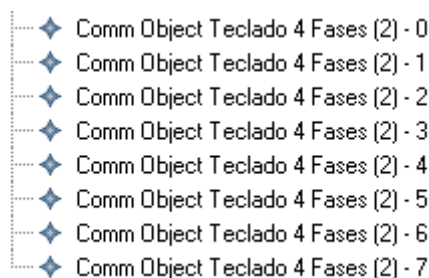


Figura 5.89. Descripción en EMF de los objetos de comunicación para el dispositivo “Pulsador 4 fases” usando el DSL propuesto.

5.3.7. Transformación JET.

El resultado de la transformación en *JET* a partir del proyecto descrito utilizando el *DSL* puede verse en el *ANEXO III.*

5.3.8. Proyecto ETS3 resultado.

Una vez ejecutado el código resultado de la transformación en *ETS3 Makro*, el proyecto de *ETS3* resultante puede apreciarse en la *Figura 8.90.* y la *Figura 8.91.* Aunque no

pueda apreciarse en esas figuras, los dispositivos se configuran como se deseaba y las asociaciones de los objetos de comunicación se hacen como se pretendía. Tras configurar los dispositivos de la maleta *EIB/KNX* usando ese proyecto, se aprecia que el funcionamiento resultante es el buscado; por lo que se puede afirmar que el flujo de diseño propuesto funciona de manera correcta para proyectos relativamente complejos.

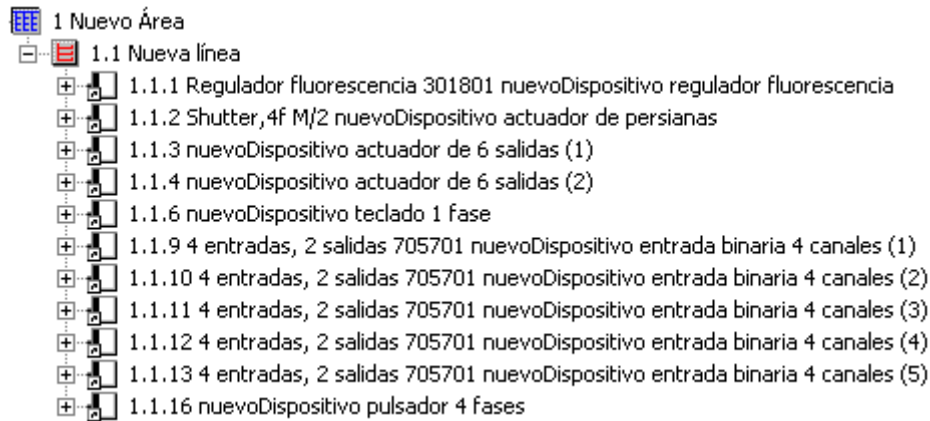


Figura 5.90. Proyecto ETS3 resultante (I).

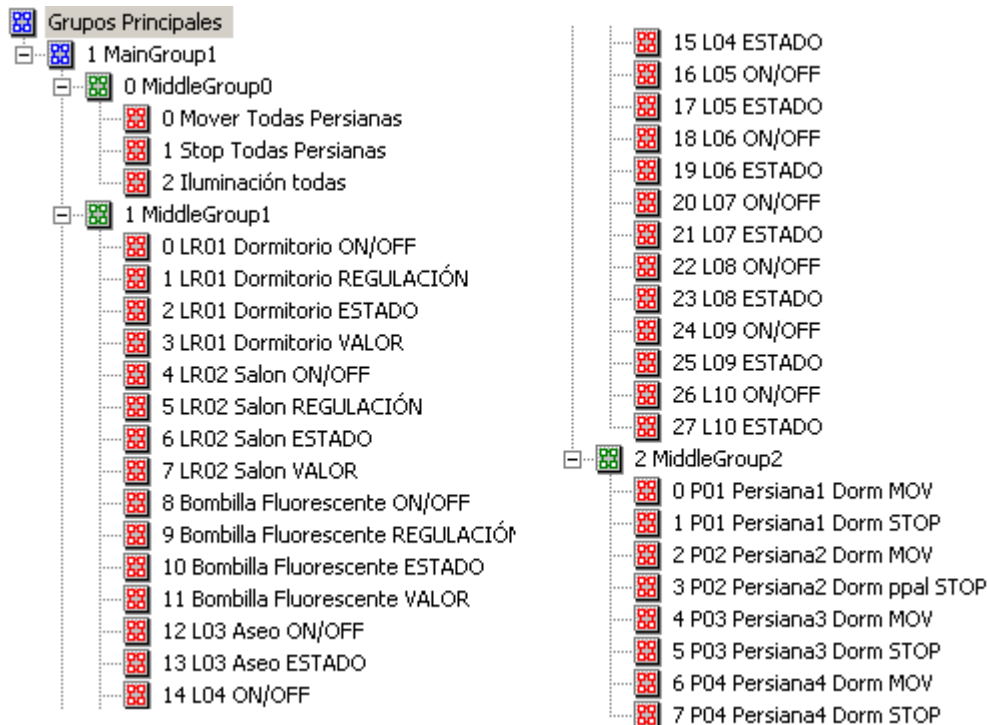


Figura 5.91. Proyecto ETS3 resultante (II).





## ***Conclusiones y trabajos futuros***

---

En este capítulo se van a analizar las aportaciones realizadas por el presente TFM tanto al estado de la técnica como al autor a nivel personal. Además se relatarán las líneas de investigación abiertas tras la realización de los trabajos.



## 6.1. Conclusiones.

Derivadas de la conclusión de los trabajos expuestos a lo largo del presente documento, surgen reflexiones en dos ámbitos: en el ámbito puramente técnico y en lo personal. En el ámbito técnico se puede decir que estos trabajos representan el primer paso para llevar a una implementación práctica los trabajos realizados por el Dr. Jiménez. Esto es relevante debido a que estos trabajos contemplan todos los conceptos de MDA y permite trasladar todas las ventajas de ingeniería dirigida por modelos al desarrollo de aplicaciones domóticas, por lo que aunque el desarrollo en sí no sea plenamente funcional sí que tiene la importancia de servir de prueba de concepto para estos trabajos. En el ámbito personal estos trabajos son el fruto de mucho tiempo y mucho esfuerzo, por lo que su finalización conlleva una gran satisfacción.

## 6.2. Trabajos futuros.

Tras la realización de trabajos de investigación en cualquier área de investigación, siempre surgen nuevos retos y posibilidades de forma derivada. Las líneas de investigación futuras surgidas tras la realización de los trabajos expuestos con anterioridad se detallan a continuación.

### 6.2.1. Implementación de un DSL gráfico para el DSL propuesto.

La línea de actuación más inmediata que surge tras los trabajos realizados en este Trabajo Final de Máster es la implementación de un DSL gráfico que de soporte al DSL propuesto aquí. A este respecto, actualmente la investigadora Paqui Rosique se encuentra actualmente trabajando en una primera versión del mismo que estará funcional en fechas venideras.

Implementar un DSL gráfico es algo que dotaría de una dimensión extra a los desarrollos implementados en este TFM, dado que los dotaría de una entidad propia diferente a la expuesta en el capítulo introductorio. En este sentido, transformaría las herramientas desarrolladas en un entorno de desarrollo para expertos en el dominio complementario a ETS3; de tal forma que permitiría la implementación de proyectos EIB/KNX de manera sistemática y más rápida estos usuarios potenciales. Así, un DSL gráfico correctamente implementado podría aportar a estos usuarios ventajas tales como una simplificación del proceso de desarrollo, una disminución de la probabilidad de cometer errores derivada de la introducción de restricciones mayores que en el entorno ETS3, y un análisis más visual y sencillo de las cuestiones de conectividad entre los diversos dispositivos derivado de la forma gráfica de describir los proyectos.

### **6.2.2. Implementación de las transformaciones entre el DSL propuesto en [Jiménez 09] y el DSL propuesto en este trabajo.**

La segunda línea abierta tras la conclusión de estos trabajos tendría como resultado el dotar de una funcionalidad práctica al uno de los trabajos más importantes realizados sobre la aplicación de los conceptos propuestos en MDE al dominio de la domótica, resultando pues la línea más importante y prometedora. En este sentido, dotar de soporte a un desarrollo tan probado y bien estructurado como el DSL propuesto por el Dr. Jiménez (dan nota de ello recientes publicaciones importantes) es algo que bien merecería el esfuerzo de realizar el estudio y los trabajos necesarios para implementar las transformaciones.

Así, el producto final de esta línea de investigación sería el poder dar soporte a usuarios poco conocedores del dominio domótico para que puedan implementar sus propias configuraciones de instalaciones EIB/KNX; algo que tanto desde el punto de vista de la investigación como del lucro empresarial ofrece un gran potencial. De esta forma, el contar con un DSL gráfico correctamente planteado, bien estructurado, completo y testado para describir instalaciones domóticas EIB/KNX de una manera sencilla y bastante separada del dominio específico de la tecnología es algo muy a tener en cuenta.

### **6.2.3. Realización de desarrollos para trasladar los trabajos a otras plataformas domóticas distintas de KNX/EIB.**

Una línea de trabajo alternativa que surge de la anterior es buscar dar soporte a diversas tecnologías dentro del marco del DSL propuesto por el Dr. Jiménez. Así, implementar metodologías y herramientas que desarrollar implementar proyectos domóticos en otras metodologías a partir de las descripciones realizadas utilizadas el DSL de [Jimenez 09] derivaría en la implementación de una herramienta de desarrollo de sistemas domóticos multi-tecnológica y multi-plataforma. Así, claras ventajas de MDE tales como reutilización y generación automática libre de errores tendrían cabida de forma general en la realización de proyectos domóticos multiplataforma, lo cuál sería un avance importante.

La tecnología más propicia para afrontar esta línea de investigación con solvencia podrían BCU/SDK. “*Free Development Environment for Bus Coupling Units of the European Installation Bus*” de Martin Köegler [Köegler 08] ofrece una nueva perspectiva para afrontar la parte final del proceso de desarrollo de aplicaciones domóticas basado en MDE: utilizar bus coupling units (BCUs) en lugar de dispositivos domóticos comerciales, programando dichas BCUs mediante las herramientas desarrolladas por Köegler. Con la intención de evaluar las posibilidades de estos trabajos, nuestro grupo ya ha realizado un dossier en el cuál se analizan y se proponen estrategias a seguir para conseguir el objetivo propuesto en este subapartado.

## ***Bibliografía***

---

# BIBLIOGRAFÍA

- [Alonso 08] Alonso, D.: *“Desarrollo de Software para Robots de Servicio: un enfoque Dirigido por Modelos y Orientado a Componentes”*. Tesis Doctoral, Universidad Politécnica de Cartagena, 2008.
- [ArcStyler 03] Sitio web de iO-Software.  
<http://www.iosoftware.com/>
- [Bold 10] Sitio web de Bold Software.  
<http://www.boldsoft.com/>
- [Dresden 10] Sitio web del proyecto Dresden OCL en Sourceforge.  
<http://dresden-ocl.sourceforge.net/>
- [Eclipse 10a] Sitio web del proyecto Eclipse. Apartado concerniente a la introducción al proyecto.  
<http://www.eclipse.org/home/newcomers.php>
- [EMF 10a] Sitio web del proyecto Eclipse. Apartado concerniente al proyecto EMF.  
<http://www.eclipse.org/modeling/emf/?project=emf#emf>
- [ETS3 07] Konnex Association, 2007.  
Manual de ayuda del programa ETS3.
- [Ganymede 10] Sitio web del proyecto Eclipse. Apartado concerniente al paquete Eclipse Ganymede.  
<http://www.eclipse.org/ganymede/>
- [GEF 10] Sitio web del proyecto Eclipse. Apartado concerniente a GMF.  
<http://www.eclipse.org/gef/>
- [GMF 10a] Sitio web del proyecto Eclipse. Apartado concerniente a GMF.  
[http://wiki.eclipse.org/index.php/Graphical\\_Modeling\\_Framework](http://wiki.eclipse.org/index.php/Graphical_Modeling_Framework)
- [GMF 10b] Sitio web del proyecto Eclipse. Tutorial para el runtime GMF.  
[http://wiki.eclipse.org/GMF\\_Tutorial](http://wiki.eclipse.org/GMF_Tutorial)
- [GMF 10c] Sitio web del proyecto Eclipse. Artículo sobre el tooling GMF.  
<http://www.eclipse.org/articles/Article-Introducing-GMF/article.html>
- [JET 04a] Sitio web del proyecto Eclipse. Tutorial de JET 1.  
[http://www.eclipse.org/articles/Article-JET/jet\\_tutorial1.html](http://www.eclipse.org/articles/Article-JET/jet_tutorial1.html)
- [JET 04b] Sitio web del proyecto Eclipse. Tutorial de JET 2.  
[http://www.eclipse.org/articles/Article-JET2/jet\\_tutorial2.html](http://www.eclipse.org/articles/Article-JET2/jet_tutorial2.html)
- [JET 04c] Sitio web del proyecto Eclipse. Ayuda referente a JET.  
<http://help.eclipse.org/help32/index.jsp?topic=/org.eclipse.emf.doc/tutorials/clibmod/clibmod.html>

- [Jiménez 09] Jiménez-Buendía, M.: *"Desarrollo de sistemas domóticos utilizando un enfoque dirigido por modelos"*. Tesis Doctoral en la Universidad Politécnica de Cartagena, 2009.
- [KNX 10a] Sitio web del consorcio KNX, 2010.  
<http://www.knx.org/es/>
- [KNX 10b] Sitio web del consorcio KNX, 2010; apartado concerniente a ETS3  
<http://www.knx.org/uk/knx-tools/ets/price-order/>
- [Köegler 08] Martin Kögler, 2008: Free Development Environment for Bus Coupling Units of the European Installation Bus; Automations Systems Group, Technical University Vienna
- [Losilla 07a] Losilla, F. et al.: *"Wireless sensor network application development: An architectur e-centric MDE approach"*. Lecture notes in computer science, vol. 4758, pp. 179-194, 2007.
- [Losilla 07b] Losilla, F. et al. *"A WSAN solution for irrigation control from a model driven perspective"*. Wireless Sensor And Actor Networks pp. 35-46, 2007.
- [M2T 10a] Sitio web del proyecto Eclipse. Apartado concerniente a M2T.  
<http://www.eclipse.org/modeling/m2t/>
- [M2T 10b] Sitio web del proyecto Eclipse. Wiki concerniente a M2T.  
<http://wiki.eclipse.org/M2T-JET-FAQ>
- [MagicDraw 09] Sitio web de MagicDraw UML.  
<http://www.magicdraw.com/>
- [Makros 08] Manual de ayuda del programa Makros, 2008.
- [Meseguer 07] Meseguer-Copado, P.J.: *"Programación de redes de sensores inalámbricas para aplicaciones domóticas"*. Proyecto Final de Carrera en la Universidad Politécnica de Cartagena, 2007.
- [MOFScript 10] Sitio web del proyecto Eclipse. Apartado relativo a MOFScript.  
<http://www.eclipse.org/gmt/mofscript/>
- [Muñoz 07] Muñoz J.; Pelechano V.; Cetina C.: *"Implementing a Pervasive Meetings Room: A Model Driven Approach"*. Proceedings of the 3rd International Workshop on Ubiquitous Computing (IWUC 2006), pp. 13 – 20, May 2006.
- [Objecteering 09] Sitio web de Objecteering/UML.  
<http://www.objecteering.com/>
- [OMG 10] Sitio web de OMG referente a MDA, 2010.  
<http://www.omg.org/mda/>
- [Optimal 01] Sitio web de Compuware.  
<http://www.compuware.com/>
- [Ortiz 07] Ortiz, F.J. et al.: *"Experiences using a component-oriented architectural framew*

*ork for robots and its improvement with a MDE approach*". Lecture Notes in Computer Science, vol. 4758, pp. 335-338, 2007.

- [Poole 01] Poole, J. D. : "*Model-Driven Architecture: Vision, Standards And Emerging Technologies*". ECOOP, 2001.
- [Poseidon 09] Sitio web de Gentleware, apartado dedicado a Poseidon for UML.  
<http://www.gentleware.com/new-poseidon-for-uml-8-0.html>
- [Rational 09] Sitio web de IBM, apartado dedicado a Rational Rose.  
<http://www-01.ibm.com/software/awdtools/developer/rose/>
- [Rojo 09] Rojo-Gallardo, J.: "*Diseño, instalación y puesta en marcha de un sistema domótico e instalación de paneles solares térmicos en una vivienda existente*". Proyecto Final de Carrera en la Universidad Politécnica de Cataluña, 2009.
- [Schmidt 06] Schmidt, D.C.: "*Model-Driven Engineering*". IEEE Computer 39, February 2006.
- [UML 05] List of UML tools, Sitio web "Objects by design".  
[http://www.objectsbydesign.com/tools/umltools\\_byCompany.html](http://www.objectsbydesign.com/tools/umltools_byCompany.html)
- [Vicente 07] Vicente-Chicote, C.; Alonso, C.; Chauvel, F.: "*V(3)studio: A component-based architecture description meta-model - Extensions to model component behaviour variability*". Proceedings of the Second International Conference On Software And Data Technologies (ICSOFT 2007), 2007.
- [Visual 10a] Sitio web de Microsoft, apartado relativo a Visual Studio.  
<http://msdn.microsoft.com/es-es/vstudio/2010.aspx>
- [Visual 10b] Sitio web de Microsoft, "Generación de código y plantillas de texto"  
<http://msdn.microsoft.com/es-es/library/bb126445.aspx>
- [Voelter 07] Voelter M., Groher I., "*Product line implementation using aspect-oriented and model-driven software development*". Proceedings of the 11th International Software Product Line Conference (SPLC 2007), pp. 233-242, 2007.
- [Vogel 09] Vogel L.: "*Java Emitter Template (JET) – Tutorial*", 2009  
<http://www.vogella.de/articles/EclipseJET/article.html>



## ***Lista de Abreviaturas***

---

<b>AL</b>	<i>Acopladores de línea</i>
<b>CIM</b>	<i>Computation Independent Model</i>
<b>CSMA/CA</b>	<i>Carrier Sense Multiple Access/Collision Avoidance</i>
<b>DSIE</b>	<i>División de Sistemas e Ingeniería Electrónica</i>
<b>DSL</b>	<i>Domain Specific Language</i>
<b>EIB</b>	<i>European Installation Bus</i>
<b>EIBA</b>	<i>European Installation Bus Association</i>
<b>EIS</b>	<i>EIB Interworking Standard</i>
<b>EMF</b>	<i>Eclipse Modeling Framework</i>
<b>ETS</b>	<i>Engineering Tool Software</i>
<b>GEF</b>	<i>Graphic Editing Framework</i>
<b>GMF</b>	<i>Graphical Modeling Framework</i>
<b>JDT</b>	<i>Java Development Tools</i>
<b>JET</b>	<i>Java Emitter Template</i>
<b>KNX</b>	<i>Konnex</i>
<b>M2M</b>	<i>Model-To-Model</i>
<b>M2T</b>	<i>Model-To-Text</i>
<b>MDA</b>	<i>Model Driven Architecture</i>
<b>MDD</b>	<i>Dirigido por Modelos</i>
<b>MDE</b>	<i>Model Driven Engineering</i>
<b>MOF</b>	<i>Meta Object Facility</i>
<b>OCL</b>	<i>Object Constraint Language</i>
<b>OMG</b>	<i>Object Management Group</i>
<b>PIM</b>	<i>Platform Independent Model</i>
<b>PL</b>	<i>Power Line</i>
<b>PSM</b>	<i>Platform Specific Model</i>
<b>QVT</b>	<i>Query View and Transformation</i>
<b>RC</b>	<i>Contador de ruta</i>
<b>RF</b>	<i>Radio Frequency</i>
<b>SCC</b>	<i>Sistemas centralizados comerciales</i>
<b>SWT</b>	<i>Standard Widget Toolkit</i>
<b>TP</b>	<i>Twisted Pair</i>
<b>UML</b>	<i>Unified Modeling Language</i>



## ***Anexo I. Macros programadas***

---

## ***Macros funcionales***

## Macro añadir dispositivo

```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
,
'Macro para añadir un dispositivo
,
'Busca en la base de datos del ETS3 la plantilla en base al fabricante,
'nombre programa, versión programa
,
,
'Versión 6
'Preparado para plantilla
'-----
'-----
'-----
```

Sub Main

```
'Obtengo el Active Project
Dim Project '[EtcDom.IProject]
Set Project = Application.Project

'-----

'Inicialización de variables del macro AnadirDispositivo
```

'-----

'variables a modificar

Dim AnadirDispositivo\_NombreReferencia  
Dim AnadirDispositivo\_DireccionFisica  
Dim AnadirDispositivo\_Fabricante  
Dim AnadirDispositivo\_no\_pedido  
Dim AnadirDispositivo\_version  
Dim AnadirDispositivo\_nombre\_programa

'variables auxiliares

Dim AnadirDispositivo\_AuxiliarYaAnadido  
Dim AnadirDispositivo\_AuxiliarEncontrado  
Dim AnadirDispositivo\_AuxiliarNuevoDispositivo  
Dim AnadirDispositivo\_AuxiliarNombre  
Dim AnadirDispositivo\_AuxiliarFabricante  
Dim AnadirDispositivo\_AuxiliarReferencia  
Dim AnadirDispositivo\_auxDR  
Dim AnadirDispositivo\_inc  
Dim AnadirDispositivo\_numeroEntities  
Dim AnadirDispositivo\_aux\_entity  
Dim AnadirDispositivo\_aux\_appProg  
Dim AnadirDispositivo\_no\_pedido\_auxiliar  
Dim AnadirDispositivo\_version\_auxiliar  
Dim AnadirDispositivo\_nombre\_programa\_auxiliar

'-----

```
'-----  
'           Macro añadir dispositivo  
'-----  
'-----
```

```
'Valores a las variables del macro AnadirDispositivo
```

```
'-----
```

```
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir  
'No es necesario que sea el nombre completo, basta con una parte del mismo  
'AnadirDispositivo_NombreReferencia = "Actuador de 6 salidas"  
'AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)
```

```
'Poner aquí el No pedido  
AnadirDispositivo_no_pedido = "3280-1A"
```

```
'Poner aquí el Nombre programa  
AnadirDispositivo_nombre_programa = "Detector universal (confort) A00802"
```

```
'Poner aquí la Version  
AnadirDispositivo_version = "2"
```

```
'Poner aquí el fabricante (nombre exacto)  
AnadirDispositivo_Fabricante = "Albrecht Jung"  
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)
```



```

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 22

'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
    for each functionalEntity in manufacturer.FunctionalEntities
        if AnadirDispositivo_AuxiliarYaAnadido = false then
            AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
            if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

                AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

                for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

                    Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

                    for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

                        AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

                        set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

```

```

        if AnadirDispositivo_aux_appProg is nothing then
        else
            Cstr(AnadirDispositivo_aux_appProg.Name)
            Cstr(AnadirDispositivo_aux_appProg.Version)
            AnadirDispositivo_nombre_programa_auxiliar =
            AnadirDispositivo_version_auxiliar =

            if (AnadirDispositivo_nombre_programa =
            AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
            AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false)
            then

                Project.Devices.Add(deviceTemplate)
                AnadirDispositivo_AuxiliarNombre
                Project.Areas.Item(2).Lines.Item(2)
                AnadirDispositivo_DireccionFisica
                Set AnadirDispositivo_AuxiliarNuevoDispositivo =
                AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
                Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
                AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
                AnadirDispositivo_AuxiliarYaAnadido = true

            end if
        end if

        set AnadirDispositivo_aux_appProg = nothing

    Next

```

```
                Next
            end if
        end if
    Next
Next
'-----
'-----
'          FIN Macro añadir dispositivo
'-----
'-----
```

End Sub

## Macro modificar dirección física

```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
',
'Macro para modificar la dirección física de un dispositivo
',
'OJO: Para que "se nos permita" por parte del entorno modificar la dirección
'física el dispositivo ha de estar añadido a una línea
',
'Funcionando OK, pero hay que modificar el macro AnadirDispositivo para que
'los dispositivos se añadan a una línea --> Usar v3 de esta macro!!
',
'Versión para plantilla
'-----
'-----
'-----
```

Sub Main

```
Dim Project '[EtcDom.IProject]
Set Project = Application.Project
```

```
'-----
```

```
'Inicialización de variables del macro ModificarDireccionFisica
```

```
'-----
```

```
'variables que hay que fijar
```

```
Dim ModificarDireccionFisica_nombreDispositivo
```

```
Dim ModificarDireccionFisica_valorDireccion
```

```
'variables auxiliares para el funcionamiento del macro
```

```
Dim ModificarDireccionFisica_auxiliarYaModificado
```

```
Dim ModificarDireccionFisica_auxiliarNombreDispositivo
```

```
Dim ModificarDireccionFisica_auxiliarEncontradoDispositivo
```

```
'-----
```

```
'-----
```

```
' Macro modificar direccion fisica
```

```
'-----
```

```
'-----
```

```
'Valores a las variables del macro ModificarDireccionFisica
```

```
'-----
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
```

```
ModificarDireccionFisica_nombreDispositivo = "nuevoDispositivo entrada binaria"
```

```
ModificarDireccionFisica_nombreDispositivo = lcase (ModificarDireccionFisica_nombreDispositivo)
```

```
'Poner aquí el valor de la nueva dirección
```

```

ModificarDireccionFisica_valorDireccion = 21

'valores iniciales variables auxiliares
ModificarDireccionFisica_auxiliarYaModificado = false

For each Device In Project.Devices
    if ModificarDireccionFisica_auxiliarYaModificado = false then
        ModificarDireccionFisica_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
        ModificarDireccionFisica_auxiliarEncontradoDispositivo =
InStrRev(ModificarDireccionFisica_auxiliarNombreDispositivo,ModificarDireccionFisica_NombreDispositivo)
        if (ModificarDireccionFisica_auxiliarEncontradoDispositivo > 0) then
            Device.Address = ModificarDireccionFisica_valorDireccion

            end if
        end if
    Next

'-----
'-----
'          FIN Macro modificar dirección física
'-----
'-----

End sub

```

## Macro modificar parámetro

```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
'
'Macro para modificar los valores de un parámetro
'Versión preparada para utilizar en la plantilla
'Funcionando OK!
'-----
'-----
'-----
```

Sub Main

```
Dim Project '[EtcDom.IProject]
Set Project = Application.Project

'-----

'Inicialización de variables del macro ModificarParametro
'-----

'variables que hay que fijar
Dim ModificarParametro_nombreDispositivo
```

```

Dim ModificarParametro_nombreParametro
Dim ModificarParametro_valorParametro
Dim ModificarParametro_ordenAparicion

'variables auxiliares para el funcionamiento del macro
Dim ModificarParametro_auxiliarYaModificado
Dim ModificarParametro_auxiliarNombreDispositivo
Dim ModificarParametro_auxiliarNombreParametro
Dim ModificarParametro_auxiliarEncontradoDispositivo
Dim ModificarParametro_auxiliarEncontradoParametro
Dim ModificarParametro_auxiliarInc

'-----
'-----
'           Macro modificar parámetro
'-----
'-----

'Valores a las variables del macro ModificarParametro
'-----

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = "entrada binaria"
ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase(ModificarParametro_nombreParametro)

```



```

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner
el valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 40

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices
    if ModificarParametro_auxiliarYaModificado = false then
        ModificarParametro_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
        ModificarParametro_auxiliarEncontradoDispositivo =
InStrRev(ModificarParametro_auxiliarNombreDispositivo,ModificarParametro_NombreDispositivo)
        if (ModificarParametro_auxiliarEncontradoDispositivo > 0) then
            for each Parameter in Device.DeviceParameters
                if (ModificarParametro_auxiliarYaModificado = false) then
                    ModificarParametro_auxiliarNombreParametro = LCase(CStr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)
                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then
                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1
                            if (ModificarParametro_auxiliarInc =
ModificarParametro_ordenAparicion) then

```

```

Parameter.value = ModificarParametro_valorParametro
ModificarParametro_auxiliarYaModificado = true
end if
end if
end if
end if
Next
end if
end if
Next
'-----
'-----
'          FIN Macro modificar parámetro
'-----
'-----
End sub

```

## Macro añadir dirección de grupo

```
'David Montoro Mouzo; david_montoro@hotmail.com
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
',
'Macro para añadir una dirección de grupo al proyecto
',
'Necesaria para modificar las direcciones de grupo de los objetos de
'comunicaciones
',
'Existe Main + Existe Middle -> OK
'Existe Main + No existe Middle -> OK
'No existe Main + No existe Middle -> OK
',
'Funcionando OK
',
'v2, corregidos fallos en la lógica
',
'Preparada para plantilla
'-----
'-----
'-----
```

Sub Main

```
Dim Project '[EtcDom.IProject]
Set Project = Application.Project

'-----
```

```

'Inicialización de variables del macro AnadirDireccionGrupo
'-----

'variables que hay que fijar
Dim AnadirDireccionGrupo_1erDigito
Dim AnadirDireccionGrupo_2oDigito
Dim AnadirDireccionGrupo_3erDigito
Dim AnadirDireccionGrupo_NombreDireccion

'variables auxiliares para el funcionamiento del macro
Dim AnadirDireccionGrupo_auxiliarEncontradoMainGroup
Dim AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup
Dim AnadirDireccionGrupo_auxiliarMainGroup
Dim AnadirDireccionGrupo_va
Dim AnadirDireccionGrupo_vb
Dim AnadirDireccionGrupo_vc

'Valores a las variables del macro AnadirDireccionGrupo
'-----

AnadirDireccionGrupo_1erDigito = 2
AnadirDireccionGrupo_2oDigito = 4
AnadirDireccionGrupo_3erDigito = 7
AnadirDireccionGrupo_NombreDireccion = "mi prima luisa"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups
  if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
    AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
    Set AnadirDireccionGrupo_va = MainGroup
    For each MiddleGroup in MainGroup.MiddleGroups
      if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
        AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
        Set AnadirDireccionGrupo_vc = MiddleGroup
        MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true
      end if
    next
  end if
Next

  if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
  Set AnadirDireccionGrupo_vb = AnadirDireccionGrupo_va
  AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDi
gito))
  AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
  Set AnadirDireccionGrupo_vc = AnadirDireccionGrupo_vb
  AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
  AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
  AnadirDireccionGrupo_auxiliarYaAnadida = true
end if

  if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida =
false) then
  Set AnadirDireccionGrupo_va = AnadirDireccionGrupo_vb
  Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))

```

```

        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set                               AnadirDireccionGrupo_vb           =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDi
gito))
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set                               AnadirDireccionGrupo_vc           =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true
    end if

```

```

'-----
'-----
'          FIN Macro añadir direccion grupo
'-----
'-----

```

End sub

## Macro enlazar dirección de grupo

```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
'
'Macro para modificar la dirección de grupo de un dispositivo
'
'OJO: La dirección de grupo debe estar ya añadida al proyecto
'
'Funcionando OK
'
'V2, cambia la selección de la dirección de grupo a añadir (antes por nombre,
'ahora por número)
'-----
'-----
'-----
```

Sub Main

```
Dim Project '[EtcDom.IProject]
Set Project = Application.Project
```

```
'-----
```

```
'Inicialización de variables del macro ModificarDireccionGrupo
'-----
```

```
'variables que hay que fijar
```

```
Dim ModificarDireccionGrupo_nombreDispositivo
```

```
Dim ModificarDireccionGrupo_numeroObjetoComuniacion
```

```
Dim ModificarDireccionGrupo_SendDireccionGrupo
```

```
Dim ModificarDireccionGrupo_1
```

```
Dim ModificarDireccionGrupo_2
```

```
Dim ModificarDireccionGrupo_3
```

```
'variables auxiliares para el funcionamiento del macro
```

```
Dim ModificarDireccionGrupo_auxiliarYaModificado
```

```
Dim ModificarDireccionGrupo_auxiliarEncontradoDispositivo
```

```
Dim ModificarDireccionGrupo_auxiliarAs
```

```
Dim ModificarDireccionGrupo_auxiliarDg
```

```
'-----
```

```
'-----
```

```
' Macro modificar dirección grupo
```

```
'-----
```

```
'-----
```

```
'Valores a las variables del macro ModificarDireccionGrupo
```

```
'-----
```



```

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarDireccionGrupo_nombreDispositivo = "actuador"
ModificarDireccionGrupo_nombreDispositivo = lcase (ModificarDireccionGrupo_nombreDispositivo)

'Poner aquí el número de objeto de comunicación a modificar
ModificarDireccionGrupo_numeroObjetoComunicacion = 4

'Poner aquí la dirección de grupo
ModificarDireccionGrupo_1 = 1
ModificarDireccionGrupo_2 = 1
ModificarDireccionGrupo_3 = 1

'Si queremos que sea send o no (booleano = [true, false])
ModificarDireccionGrupo_SendDireccionGrupo = true

'valores iniciales variables auxiliares
ModificarDireccionGrupo_auxiliarYaModificado = false

Set ModificarDireccionGrupo_auxiliarDg =
Project.MainGroups.Item(cint(ModificarDireccionGrupo_1)).MiddleGroups.Item(cint(ModificarDireccionGrupo_2)).GroupAddresses.Item(cint(ModificarDireccionGrupo_3))

For each Device In Project.Devices

    if ModificarDireccionGrupo_auxiliarYaModificado = false then

        ModificarDireccionGrupo_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
        ModificarDireccionGrupo_auxiliarEncontradoDispositivo =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDispositivo, ModificarDireccionGrupo_nombreDispositivo)

```

```

if (ModificarDireccionGrupo_auxiliarEncontradoDispositivo > 0) then
    For Each DeviceCommunicationObject In Device.VisibleDeviceCommunicationObjects
        if DeviceCommunicationObject.CommunicationObjectNumber =
ModificarDireccionGrupo_numeroObjetoComunicacion then
            Set ModificarDireccionGrupo_auxiliarAs =
DeviceCommunicationObject.Associations.Add(ModificarDireccionGrupo_auxiliarDg)
            ModificarDireccionGrupo_auxiliarAs.Send =
ModificarDireccionGrupo_SendDireccionGrupo
        end if
    Next
    ModificarDireccionGrupo_auxiliarYaModificado = true
end if
end if
Next
'-----
'-----
'          FIN Macro modificar dirección grupo
'-----
'-----
End sub

```

## ***Macros auxiliares***

## Macro obtener el nombre de todos los dispositivos en la base de datos

```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
'
'Macro para obtener los dispositivos en la base de datos para cada fabricante
'Necesario crear en el directorio raíz de la unidad C:\ una carpeta con el nombre "Info dispositivos"
'Funcionando OK
'-----
'-----
'-----
```

Sub Main

```
Dim Project '[EtcDom.IProject]
Set Project = Application.Project

'-----

Dim auxDR
Set aux = Project.parent

Dim Directorio
Dim fso
Dim f1
Set fso = CreateObject("Scripting.FileSystemObject")
```

```
For each manufacturer in aux.manufacturers
```

```
    Dim NombreFiltrado
```

```
    NombreFiltrado = Cstr(manufacturer.name)
```

```
    NombreFiltrado = Replace(NombreFiltrado, ".", "_")
```

```
    NombreFiltrado = Replace(NombreFiltrado, ",", "_")
```

```
    NombreFiltrado = Replace(NombreFiltrado, "/", "_")
```

```
    NombreFiltrado = Replace(NombreFiltrado, " ", "_")
```

```
    NombreFiltrado = Replace(NombreFiltrado, ":", "_")
```

```
    NombreFiltrado = Replace(NombreFiltrado, "'", "_")
```

```
    Directorio = "C:\Info dispositivos\manufacturer_" + NombreFiltrado + Cstr(rnd()) + ".txt"
```

```
    Set f1 = fso.CreateTextFile(directorio, True)
```

```
    for each functionalEntity in manufacturer.FunctionalEntities
```

```
        f1.WriteLine(functionalEntity.name)
```

```
    Next
```

```
Next
```

```
End sub
```

## Macro obtener información de los dispositivos agregados al proyecto

```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
'
'Macro para obtener la información relevante de un dispositivo
'Necesario crear en el directorio raíz de la unidad C:\ una carpeta con el nombre "Info dispositivos"
'Funcionando OK
'-----
'-----
'-----
```

Sub Main

```
Dim Project '[EtcDom.IProject]
Set Project = Application.Project

Dim Nombre

Dim Dev

'Msgbox Project.Devices.count
```

```

'Set Dev = Project.Devices.Item(1)

for each device in project.devices

    set dev = device

    'Procesado del nombre y creación del archivo.txt
    '-----

    Nombre = Dev.Name
    'MsgBox Nombre

    NombreFiltrado = Cstr(Nombre)
    NombreFiltrado = Replace(NombreFiltrado, ".", "_")
    NombreFiltrado = Replace(NombreFiltrado, ",", "_")
    NombreFiltrado = Replace(NombreFiltrado, "/", "_")
    NombreFiltrado = Replace(NombreFiltrado, " ", "_")
    NombreFiltrado = Replace(NombreFiltrado, ":", "_")
    NombreFiltrado = Replace(NombreFiltrado, "'", "_")

    'MsgBox NombreFiltrado

    Dim Directorio
    Directorio = "C:\Info dispositivos\"+Cstr(Dev.UniqueName)+"_"+NombreFiltrado+".txt"

    Dim fso

```

```

Dim f1

Set fso = CreateObject("Scripting.FileSystemObject")
Set f1 = fso.CreateTextFile(directorio, True)

'Info general
'-----
f1.WriteLine(Nombre)
f1.WriteLine("-----")
")
f1.WriteLine("-----")
")

f1.WriteLine("")
f1.WriteLine("Proyecto: "+Dev.Parent.Name)
f1.WriteLine("Unique Name: "+Dev.UniqueName)
f1.WriteLine("Nombre: "+Dev.name)
f1.WriteLine("Fabricante: "+Dev.CatalogEntry.Manufacturer.Name)
f1.WriteLine("Entrada de catálogo: "+Dev.CatalogEntry.Name)

    dim tipoDir
    tipoDir = vartype(Dev.IndividualAddress)
    dim valorDir
    valorDir = Dev.IndividualAddress
    if tipoDir =1 then valorDir = "Null"

f1.WriteLine("Dirección individual: "+Cstr(valorDir))

```



```

        tipoDir = vartype(Dev.Address)
        valorDir = Dev.Address
        if tipoDir =1 then valorDir = "Null"
f1.WriteLine("Dirección física: "+Cstr(valorDir))

f1.WriteLine("")
f1.WriteLine("Numero de objetos de comunicación: "+Cstr(Dev.DeviceCommunicationObjects.Count))
f1.WriteLine("Numero de parámetros: "+Cstr(Dev.DeviceParameters.Count))
f1.WriteLine("")
f1.WriteLine("")

for each baggage in dev.CatalogEntry.Baggages 'Esto hay ke estudiarlo!!
    f1.WriteLine("Archivos necesarios")
    f1.WriteLine(baggage.localfiledir)
    f1.WriteLine(baggage.localfilename)

next

f1.WriteLine("")
f1.WriteLine("")

'Lo referente a las projects parts
'-----

f1.WriteLine("(1)PROJECT PARTS")
f1.WriteLine("-----")

```

```

For Each ProjectPart In Dev.ProjectParts
    f1.WriteLine("")
    f1.WriteLine("·NOMBRE: " + ProjectPart.Name + "      ·TIPO: " +ProjectPart.PartType)
Next

```

```

f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")

```

'Lo referente a líneas y áreas

'-----

```

f1.WriteLine("(2)LÍNEAS Y ÁREAS")
f1.WriteLine("-----")

```

```

For Each ProjectPart In Dev.ProjectParts
    f1.WriteLine("")
    f1.WriteLine("·NOMBRE: " + ProjectPart.Name + "      ·TIPO: " +ProjectPart.PartType)
Next

```

```

f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")

```

```
'Lo referente a los objetos de comunicación
```

```
'-----
```

```
f1.WriteLine("(3)OBJETOS DE COMUNICACIÓN")
```

```
f1.WriteLine("-----")
```

```
For Each DeviceCommunicationObject In Dev.VisibleDeviceCommunicationObjects
```

```
    f1.WriteLine("")
```

```
    dim descripcion
```

```
    descripcion = DeviceCommunicationObject.Description
```

```
    if descripcion = "" then descripcion = "Sin descripción"
```

```
        f1.WriteLine("·NOMBRE: " + DeviceCommunicationObject.Name + "    ·NUMERO DE OBJETO: " +  
CStr(DeviceCommunicationObject.CommunicationObjectNumber)+ "    ·FUNCIÓN: " +  
DeviceCommunicationObject.CommunicationObjectFunction+ "    ·DESCRIPCIÓN: " + descripcion )
```

```
        f1.WriteLine("·VISIBLE: " + CStr(DeviceCommunicationObject.visible))
```

```
        f1.WriteLine("·FLAGS COM-LECT-ESCRT-TX: " +  
CStr(DeviceCommunicationObject.CommunicationEnabled) + "-" + CStr(DeviceCommunicationObject.ReadEnabled) +  
"-" + CStr(DeviceCommunicationObject.WriteEnabled) + "-" +  
CStr(DeviceCommunicationObject.TransmissionEnabled))
```

```
        f1.WriteLine("·NÚMERO DE ASOCIACIONES: " + CStr(DeviceCommunicationObject.Associations.count))
```

```
    dim i
```

```

        i=0
        For Each Association in DeviceCommunicationObject.Associations
            i=i+1
            f1.WriteLine("                - Nº Asociación:"+ Cstr(i) +"    - Dirección de grupo: " +
CStr(Association.GroupAddress) + "    - Sending: "+ CStr(Association.Send) )
            Next
        Next

```

```

f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")

```

'Todo lo referente a los parámetros

'-----

```

f1.WriteLine("(4)PARÁMETROS")
f1.WriteLine("  -----")

```

```

        f1.WriteLine("")
        f1.WriteLine("Tipos:")
        f1.WriteLine("* 0 = vbEmpty - Indicates Empty (uninitialized) ")
f1.WriteLine("* 1 = vbNull - Indicates Null (no valid data)")
f1.WriteLine("* 2 = vbInteger - Indicates an integer")

```

```

f1.WriteLine("* 3 = vbLong - Indicates a long integer")
f1.WriteLine("* 4 = vbSingle - Indicates a single-precision floating-point number")
f1.WriteLine("* 5 = vbDouble - Indicates a double-precision floating-point number")
f1.WriteLine("* 6 = vbCurrency - Indicates a currency")
f1.WriteLine("* 7 = vbDate - Indicates a date")
f1.WriteLine("* 8 = vbString - Indicates a string")
f1.WriteLine("* 9 = vbObject - Indicates an automation object")
f1.WriteLine("* 10 = vbError - Indicates an error")
f1.WriteLine("* 11 = vbBoolean - Indicates a boolean")
f1.WriteLine("* 12 = vbVariant - Indicates a variant (used only with arrays of Variants)")
f1.WriteLine("* 13 = vbDataObject - Indicates a data-access object")
f1.WriteLine("* 17 = vbByte - Indicates a byte")
f1.WriteLine("* 8192 = vbArray - Indicates an array")

```

For Each Parameter In Dev.DeviceParameters

```

    f1.WriteLine("")
    dim descripcion1

    descripcion1 = Parameter.Description
    if descripcion1 = "" then descripcion1 = "Sin descripción"

    f1.WriteLine("·NOMBRE: " + Parameter.Name + "    ·DESCRIPCIÓN: " + descripcion1 )

    dim tipo

```

```
        tipo = vartype(Parameter.Value)
        dim valor
        valor = Parameter.Value
        if tipo =1 then valor = "Null"

        f1.WriteLine("·TIPO: "+Cstr(tipo)+"   ·VALOR: " + Cstr(valor)+ "   ·VISIBLE Y VÁLIDO: "
+Cstr(Parameter.visible))
```

```
    Next
```

```
Next
```

```
End Sub
```

## Macro obtener información del dispositivo activo

```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
'
'Macro obtener la info del dispositivo activo
'Necesario crear en el directorio raíz de la unidad C:\ una carpeta con el nombre "Info dispositivos"
'Funcionando OK
'-----
'-----
'-----
```

Sub Main

```
Dim obj
Set obj = GetActiveDomObject()
If obj Is Nothing Then
    MsgBox "Nada seleccionado"
Else

    dim dev
    set dev = obj

'Procesado del nombre y creación del archivo.txt
'-----

Nombre = Dev.Name
'MsgBox Nombre
```

```

NombreFiltrado = Cstr(Nombre)
NombreFiltrado = Replace(NombreFiltrado, ".", "_")
NombreFiltrado = Replace(NombreFiltrado, ",", "_")
NombreFiltrado = Replace(NombreFiltrado, "/", "_")
NombreFiltrado = Replace(NombreFiltrado, " ", "_")
NombreFiltrado = Replace(NombreFiltrado, ":", "_")
NombreFiltrado = Replace(NombreFiltrado, "'", "_")

'MsgBox NombreFiltrado

Dim Directorio
Directorio = "C:\Info dispositivos\"+Cstr(Dev.UniqueName)+"_"+NombreFiltrado+".txt"

Dim fso
Dim f1

Set fso = CreateObject("Scripting.FileSystemObject")
Set f1 = fso.CreateTextFile(directorio, True)

'Info general
'-----
f1.WriteLine(Nombre)
f1.WriteLine("-----")
f1.WriteLine("-----")
f1.WriteLine("")
f1.WriteLine("Proyecto: "+Dev.Parent.Name)
f1.WriteLine("Unique Name: "+Dev.UniqueName)
f1.WriteLine("Nombre: "+Dev.name)
f1.WriteLine("Fabricante: "+Dev.CatalogEntry.Manufacturer.Name)

```



```

f1.WriteLine("Entrada de catálogo: "+Dev.CatalogEntry.Name)

    dim tipoDir
    tipoDir = vartype(Dev.IndividualAddress)
    dim valorDir
    valorDir = Dev.IndividualAddress
    if tipoDir =1 then valorDir = "Null"

f1.WriteLine("Dirección individual: "+Cstr(valorDir))

    tipoDir = vartype(Dev.Address)
    valorDir = Dev.Address
    if tipoDir =1 then valorDir = "Null"
f1.WriteLine("Dirección física: "+Cstr(valorDir))

f1.WriteLine("")
f1.WriteLine("Numero de objetos de comunicación: "+Cstr(Dev.DeviceCommunicationObjects.Count))
f1.WriteLine("Numero de parámetros: "+Cstr(Dev.DeviceParameters.Count))
f1.WriteLine("")
f1.WriteLine("")

f1.WriteLine("")
f1.WriteLine("")

'Lo referente a las projects parts
'-----

```

```

f1.WriteLine("(1)PROJECT PARTS")
f1.WriteLine("  -----")

For Each ProjectPart In Dev.ProjectParts
    f1.WriteLine("")
    f1.WriteLine("·NOMBRE: " + ProjectPart.Name + "      ·TIPO: " +ProjectPart.PartType)
Next

f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")

'Lo referente a líneas y áreas
'-----

f1.WriteLine("(2)LÍNEAS Y ÁREAS")
f1.WriteLine("  -----")

For Each ProjectPart In Dev.ProjectParts
    f1.WriteLine("")
    f1.WriteLine("·NOMBRE: " + ProjectPart.Name + "      ·TIPO: " +ProjectPart.PartType)
Next

f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")

```

'Lo referente a los objetos de comunicación

```
'-----  
f1.WriteLine("(3)OBJETOS DE COMUNICACIÓN")  
f1.WriteLine("-----")
```

For Each DeviceCommunicationObject In Dev.VisibleDeviceCommunicationObjects

```
f1.WriteLine("")  
dim descripcion
```

```
descripcion = DeviceCommunicationObject.Description  
if descripcion = "" then descripcion = "Sin descripción"
```

```
f1.WriteLine("·NOMBRE: " + DeviceCommunicationObject.Name + " ·NUMERO DE OBJETO: " +  
CStr(DeviceCommunicationObject.CommunicationObjectNumber)+ " ·FUNCIÓN: " +  
DeviceCommunicationObject.CommunicationObjectFunction+ " ·DESCRIPCIÓN: " + descripcion )  
f1.WriteLine("·VISIBLE: " + CStr(DeviceCommunicationObject.visible))  
f1.WriteLine("·FLAGS COM-LECT-ESCR-TX: " + CStr(DeviceCommunicationObject.CommunicationEnabled) + "-" +  
CStr(DeviceCommunicationObject.ReadEnabled) + "-" + CStr(DeviceCommunicationObject.WriteEnabled) + "-" +  
CStr(DeviceCommunicationObject.TransmissionEnabled))  
f1.WriteLine("·NÚMERO DE ASOCIACIONES: " + CStr(DeviceCommunicationObject.Associations.count))
```

```
dim i  
i=0
```

For Each Association in DeviceCommunicationObject.Associations

```
i=i+1  
f1.WriteLine(" - Nº Asociación:"+ Cstr(i) + " - Dirección de grupo: " +  
CStr(Association.GroupAddress) + " - Sending: "+ CStr(Association.Send) )
```

Next

Next

```
f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")
f1.WriteLine("")
```

```
'Todo lo referente a los parámetros
```

```
'-----
```

```
f1.WriteLine("(4)PARÁMETROS")
f1.WriteLine("-----")
```

```
    f1.WriteLine("")
    f1.WriteLine("Tipos:")
    f1.WriteLine("* 0 = vbEmpty - Indicates Empty (uninitialized) ")
f1.WriteLine("* 1 = vbNull - Indicates Null (no valid data)")
f1.WriteLine("* 2 = vbInteger - Indicates an integer")
f1.WriteLine("* 3 = vbLong - Indicates a long integer")
f1.WriteLine("* 4 = vbSingle - Indicates a single-precision floating-point number")
f1.WriteLine("* 5 = vbDouble - Indicates a double-precision floating-point number")
f1.WriteLine("* 6 = vbCurrency - Indicates a currency")
f1.WriteLine("* 7 = vbDate - Indicates a date")
f1.WriteLine("* 8 = vbString - Indicates a string")
f1.WriteLine("* 9 = vbObject - Indicates an automation object")
f1.WriteLine("* 10 = vbError - Indicates an error")
```

```

f1.WriteLine("* 11 = vbBoolean - Indicates a boolean")
f1.WriteLine("* 12 = vbVariant - Indicates a variant (used only with arrays of Variants)")
f1.WriteLine("* 13 = vbDataObject - Indicates a data-access object")
f1.WriteLine("* 17 = vbByte - Indicates a byte")
f1.WriteLine("* 8192 = vbArray - Indicates an array")

```

For Each Parameter In Dev.DeviceParameters

```

f1.WriteLine("")
dim descripcion1

```

```

descripcion1 = Parameter.Description
if descripcion1 = "" then descripcion1 = "Sin descripción"

```

```

f1.WriteLine("·NOMBRE: " + Parameter.Name + "      ·DESCRIPCIÓN: " + descripcion1 )

```

```

dim tipo
tipo = vartype(Parameter.Value)
dim valor
valor = Parameter.Value
if tipo =1 then valor = "Null"

```

```

f1.WriteLine("·TIPO: "+Cstr(tipo)+"      ·VALOR: " + Cstr(valor)+ "      ·VISIBLE Y VÁLIDO: "
+Cstr(Parameter.visible))

```

Next

End If

End Sub

```
Function GetActiveDomObject()  
    Dim ActiveWindow    ' [Ets3Pro.IAomWindow] the active window.  
    Dim Control         ' the embedded control  
    Dim Selection       ' selected item  
  
    Set ActiveWindow = Application.ActiveWindow  
    If ActiveWindow Is Nothing Then  
        Set GetActiveDomObject = Nothing  
    Else  
        Set Control = ActiveWindow.Control(0)  
        If Control Is Nothing Then  
            Set GetActiveDomObject = Nothing  
        Else  
            Selection = Control.SelectedItem  
            Set GetActiveDomObject = Control.ItemObject(Selection)  
        End If  
    End If  
End Function
```



## ***Macros replica de proyecto***



```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
'
'Macro para crear un proyecto ejemplo desde cero
'Hay que abrir en ETS3 un proyecto en blanco y ejecutar la macro
'
'Funcionando OK
'-----
'-----
'-----
```

Sub Main

```
'Obtengo el Active Project
Dim Project '[EtcDom.IProject]
Set Project = Application.Project

, .....
, .....
'
' Inicialización de variables
, .....
, .....

'
'Inicialización de variables del macro AnadirDispositivo
```

```

'
'variables a modificar
Dim AnadirDispositivo_NombreReferencia
Dim AnadirDispositivo_DireccionFisica
Dim AnadirDispositivo_Fabricante
'variables auxiliares
Dim AnadirDispositivo_AuxiliarYaAnadido
Dim AnadirDispositivo_AuxiliarEncontrado
Dim AnadirDispositivo_AuxiliarNuevoDispositivo
Dim AnadirDispositivo_AuxiliarNombre
Dim AnadirDispositivo_AuxiliarFabricante
Dim AnadirDispositivo_AuxiliarReferencia
Dim AnadirDispositivo_auxDR
Dim inc
Dim numeroEntitys

'
'Inicialización de variables del macro AnadirDireccionGrupo
'

'variables que hay que fijar
Dim AnadirDireccionGrupo_1erDigito
Dim AnadirDireccionGrupo_2oDigito
Dim AnadirDireccionGrupo_3erDigito
Dim AnadirDireccionGrupo_NombreDireccion

'variables auxiliares para el funcionamiento del macro
Dim AnadirDireccionGrupo_auxiliarEncontradoMainGroup

```

```

Dim AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup
Dim AnadirDireccionGrupo_auxiliarMainGroup
Dim AnadirDireccionGrupo_va
Dim AnadirDireccionGrupo_vb
Dim AnadirDireccionGrupo_vc

,
'Inicialización de variables del macro ModificarParametro
,

'variables que hay que fijar
Dim ModificarParametro_nombreDispositivo
Dim ModificarParametro_nombreParametro
Dim ModificarParametro_valorParametro
Dim ModificarParametro_ordenAparicion

'variables auxiliares para el funcionamiento del macro
Dim ModificarParametro_auxiliarYaModificado
Dim ModificarParametro_auxiliarNombreDispositivo
Dim ModificarParametro_auxiliarNombreParametro
Dim ModificarParametro_auxiliarEncontradoDispositivo
Dim ModificarParametro_auxiliarEncontradoParametro
Dim ModificarParametro_auxiliarInc

,
'Inicialización de variables del macro ModificarDireccionGrupo
,

```

```
'variables que hay que fijar
Dim ModificarDireccionGrupo_nombreDispositivo
Dim ModificarDireccionGrupo_numeroObjetoComuniacion
Dim ModificarDireccionGrupo_NombreDireccionGrupo
Dim ModificarDireccionGrupo_SendDireccionGrupo
```

```
'variables auxiliares para el funcionamiento del macro
Dim ModificarDireccionGrupo_auxiliarYaModificado
Dim ModificarDireccionGrupo_auxiliarNombreDispositivo
Dim ModificarDireccionGrupo_auxiliarEncontradoDispositivo
Dim ModificarDireccionGrupo_auxiliarAs
Dim ModificarDireccionGrupo_auxiliarDg
Dim ModificarDireccionGrupo_auxiliarYaEncontradaDireccion
Dim ModificarDireccionGrupo_auxiliarNombreDireccion
Dim ModificarDireccionGrupo_auxiliarEncontradaDireccion
```

```
, .....
, .....
'
'                                Añadimos los dispositivos      (OK)
, .....
, .....
```

```
'Añadimos Actuador accionamiento 6salidas
'
```

```
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Actuador de 6 salidas"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)
```

```
'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)
```

```
'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 4
```

```
'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
```

```
for each manufacturer in AnadirDispositivo_auxDR.manufacturers
  for each functionalEntity in manufacturer.FunctionalEntities
    if AnadirDispositivo_AuxiliarYaAnadido = false then
      AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
      if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then
        numeroEntities = functionalEntity.FunctionalEntities.count
        for inc=1 to numeroEntities step 1
          AnadirDispositivo_AuxiliarReferencia =
LCase(CStr(FunctionalEntity.FunctionalEntities.Item(Cint(inc)).name))
          AnadirDispositivo_AuxiliarEncontrado =
InStrRev(AnadirDispositivo_AuxiliarReferencia,AnadirDispositivo_NombreReferencia)
          if (AnadirDispositivo_AuxiliarEncontrado > 0) then
```

```

        Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(FunctionalEntity.FunctionalEntities.Item(Cint(inc)).DeviceTemplates.Item(1))
        AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
        Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)
        AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_DireccionFisica
        AnadirDispositivo_AuxiliarYaAnadido = true
    end if
Next
end if
end if
Next
Next
Next
'
'Añadimos Entrada binaria de 4 canales
'
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Entrada binaria, 4-canales"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

```

'Poner aquí la direccion fisica del nuevo dispositivo

AnadirDispositivo\_DireccionFisica = 11

'Variables auxiliares

AnadirDispositivo\_AuxiliarYaAnadido = false

AnadirDispositivo\_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo\_NombreReferencia

Set AnadirDispositivo\_auxDR = Project.parent

for each manufacturer in AnadirDispositivo\_auxDR.manufacturers

  for each functionalEntity in manufacturer.FunctionalEntities

    if AnadirDispositivo\_AuxiliarYaAnadido = false then

      AnadirDispositivo\_AuxiliarFabricante = LCase(CStr(manufacturer.name))

      if (AnadirDispositivo\_AuxiliarFabricante = AnadirDispositivo\_Fabricante) then

        numeroEntities = functionalEntity.FunctionalEntities.count

        for inc=1 to numeroEntities step 1

          AnadirDispositivo\_AuxiliarReferencia =

LCase(CStr(FunctionalEntity.FunctionalEntities.Item(Cint(inc)).name))

          AnadirDispositivo\_AuxiliarEncontrado =

InStrRev(AnadirDispositivo\_AuxiliarReferencia,AnadirDispositivo\_NombreReferencia)

          if (AnadirDispositivo\_AuxiliarEncontrado > 0) then

            Set AnadirDispositivo\_AuxiliarNuevoDispositivo =

Project.Devices.Add(FunctionalEntity.FunctionalEntities.Item(Cint(inc)).DeviceTemplates.Item(1))

            AnadirDispositivo\_AuxiliarNuevoDispositivo.Name =

AnadirDispositivo\_AuxiliarNombre

            Set AnadirDispositivo\_AuxiliarNuevoDispositivo.line =

Project.Areas.Item(2).Lines.Item(2)

            AnadirDispositivo\_AuxiliarNuevoDispositivo.Address =

AnadirDispositivo\_DireccionFisica

            AnadirDispositivo\_AuxiliarYaAnadido = true

```

end if
Next
end if
end if
Next
Next
```

```

.....
.....
'
Añadimos las direcciones de grupo (OK)
.....
.....
```

```

'
'Añadimos la 1.1.1. L04 ON/OFF
'
```

```
AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 1
AnadirDireccionGrupo_NombreDireccion = "L04 ON/OFF"
```

```
'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```



```

if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

    AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
    Set AnadirDireccionGrupo_va = MainGroup

    For each MiddleGroup in MainGroup.MiddleGroups

        if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

            AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
            Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

        end if

    next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDi
gito))

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida =
false) then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDi
gito))

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

```

```
,  
'Añadimos la 1.1.2.    L04 ESTADO  
,
```

```
AnadirDireccionGrupo_1erDigito = 1  
AnadirDireccionGrupo_2oDigito = 1  
AnadirDireccionGrupo_3erDigito = 2  
AnadirDireccionGrupo_NombreDireccion = "L04 ESTADO"
```

```
'valores iniciales variables auxiliares  
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false  
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false  
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true  
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true  
                Set AnadirDireccionGrupo_vc =
```

```
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)  
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito  
                AnadirDireccionGrupo_auxiliarYaAnadida = true
```

```

                end if
            next
        end if

    Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDi
gito))
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida =
false) then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    end if
end if

```

```

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDi
gito))
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

, .....
, .....
'
                                Configuramos los dispositivos (parámetros)
, .....
, .....

,
'1.1.4. ACT2 Actuador accionamiento, 6 salidas, 6A DIN
'"Reenvío de estado" -> No invertir (OK)
'Rückmeldung [2] = 0
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = "Actuador de 6 salidas"
ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase(ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner
el valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 2

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = LCase(CStr(Device.Name))

        ModificarParametro_auxiliarEncontradoDispositivo =
InStrRev(ModificarParametro_auxiliarNombreDispositivo,ModificarParametro_NombreDispositivo)

        if (ModificarParametro_auxiliarEncontradoDispositivo > 0) then

            for each Parameter in Device.DeviceParameters

```

```

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc =
ModificarParametro_ordenAparicion) then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    Next

end if

```

```
end if
```

```
Next
```

```
,
```

```
'1.1.11. Entrada binaria, 4 canales
```

```
'"comando al flanco ascendente. Objeto de accionamiento 3.1" -> ALTERNADO (OK)
```

```
'Befehl_steigendeFlanke_01 [13] = 3
```

```
,
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
```

```
ModificarParametro_nombreDispositivo = "Entrada binaria, 4-canales"
```

```
ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
```

```
'Poner aquí el nombre EXACTO del parámetro
```

```
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
```

```
ModificarParametro_nombreParametro = lcase(ModificarParametro_nombreParametro)
```

```
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,
```

```
'en vbs no hay disitinción en tipos de variables en su definición)
```

```
ModificarParametro_valorParametro = 3
```

```
'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
```

```
ModificarParametro_ordenAparicion = 13
```

```
'valores iniciales variables auxiliares
```

```
ModificarParametro_auxiliarYaModificado = false
```



```

ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = LCase(CStr(Device.Name))

        ModificarParametro_auxiliarEncontradoDispositivo =
InStrRev(ModificarParametro_auxiliarNombreDispositivo, ModificarParametro_NombreDispositivo)

        if (ModificarParametro_auxiliarEncontradoDispositivo > 0) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

```

```

ModificarParametro_ordenAparicion) then
    if (ModificarParametro_auxiliarInc =
        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true
    end if
    end if
end if
end if
Next
end if
end if
Next
'1.1.11. Entrada binaria, 4 canales
'"comando al flanco descendente. Objeto de accionamiento 3.1" -> Sin reaccion (0) (OK)
'Befehl_fallendeFlanke_01 [40]=0
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

```

```

ModificarParametro_nombreDispositivo = "entrada binaria"
ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase(ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner
el valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 40

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = LCase(CStr(Device.Name))

        ModificarParametro_auxiliarEncontradoDispositivo =
InStrRev(ModificarParametro_auxiliarNombreDispositivo,ModificarParametro_NombreDispositivo)

        if (ModificarParametro_auxiliarEncontradoDispositivo > 0) then

```

```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc =
ModificarParametro_ordenAparicion) then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

```

```

                Next
            end if
        end if

Next

, .....
, .....
'
                                Enlazamos las direcciones de grupo
, .....
, .....

,

'(s) Actuador accionamiento, obj 4, "salida 5, accionador"
'1.1.1.    L04 ON/OFF
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarDireccionGrupo_nombreDispositivo = "Actuador de 6 salidas"
ModificarDireccionGrupo_nombreDispositivo = lcase (ModificarDireccionGrupo_nombreDispositivo)

'Poner aquí el número de objeto de comunicación a modificar
ModificarDireccionGrupo_numeroObjetoComunicacion = 4

'Poner aquí el nombre EXACTO de la dirección de grupo (ya añadida) a la que quiero vincular el objeto de
comunicación
ModificarDireccionGrupo_NombreDireccionGrupo = "L04 ON/OFF"

```

```

ModificarDireccionGrupo_NombreDireccionGrupo = Lcase(ModificarDireccionGrupo_NombreDireccionGrupo)

'Si queremos que sea send o no (booleano = [true, false])
ModificarDireccionGrupo_SendDireccionGrupo = true

'valores iniciales variables auxiliares
ModificarDireccionGrupo_auxiliarYaModificado = false
ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false

For each GroupAddress in Project.GroupAddresses

    if (ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false) then

        ModificarDireccionGrupo_auxiliarNombreDireccion = LCase(Cstr(GroupAddress.Name))
        ModificarDireccionGrupo_auxiliarEncontradaDireccion =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDireccion, ModificarDireccionGrupo_NombreDireccionGrupo)

        if (ModificarDireccionGrupo_auxiliarEncontradaDireccion > 0) then

            if
(Len(ModificarDireccionGrupo_auxiliarNombreDireccion)=Len(ModificarDireccionGrupo_NombreDireccionGrupo)) then

                Set ModificarDireccionGrupo_auxiliarDg = GroupAddress
                ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = true

            end if

        end if

    end if

end if

```

```

        end if
    Next

    For each Device In Project.Devices

        if ModificarDireccionGrupo_auxiliarYaModificado = false then

            ModificarDireccionGrupo_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
            ModificarDireccionGrupo_auxiliarEncontradoDispositivo =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDispositivo, ModificarDireccionGrupo_NombreDispositivo)

            if (ModificarDireccionGrupo_auxiliarEncontradoDispositivo > 0) then

                For Each DeviceCommunicationObject In Device.VisibleDeviceCommunicationObjects

                    if DeviceCommunicationObject.CommunicationObjectNumber =
ModificarDireccionGrupo_numeroObjetoComunicacion then

                        Set ModificarDireccionGrupo_auxiliarAs =
DeviceCommunicationObject.Associations.Add(ModificarDireccionGrupo_auxiliarDg)
                        ModificarDireccionGrupo_auxiliarAs.Send =
ModificarDireccionGrupo_SendDireccionGrupo

                    end if

                Next

                ModificarDireccionGrupo_auxiliarYaModificado = true
            end if
        end if
    Next

```

```

        end if

    end if

Next

,
'(s) Entrada binaria, obj 2, "Entrada 3 objeto accionamiento"
'1.1.1.    L04 ON/OFF
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarDireccionGrupo_nombreDispositivo = "Entrada binaria"
ModificarDireccionGrupo_nombreDispositivo = lcase (ModificarDireccionGrupo_nombreDispositivo)

'Poner aquí el número de objeto de comunicación a modificar
ModificarDireccionGrupo_numeroObjetoComunicacion = 2

'Poner aquí el nombre EXACTO de la dirección de grupo (ya añadida) a la que quiero vincular el objeto de
comunicación
ModificarDireccionGrupo_NombreDireccionGrupo = "L04 ON/OFF"
ModificarDireccionGrupo_NombreDireccionGrupo = Lcase(ModificarDireccionGrupo_NombreDireccionGrupo)

'Si queremos que sea send o no (booleano = [true, false])
ModificarDireccionGrupo_SendDireccionGrupo = true

'valores iniciales variables auxiliares

```



```
ModificarDireccionGrupo_auxiliarYaModificado = false
ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false
```

```
For each GroupAddress in Project.GroupAddresses
```

```
    if (ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false) then
```

```
        ModificarDireccionGrupo_auxiliarNombreDireccion = LCase(Cstr(GroupAddress.Name))
```

```
        ModificarDireccionGrupo_auxiliarEncontradaDireccion =
```

```
InStrRev(ModificarDireccionGrupo_auxiliarNombreDireccion, ModificarDireccionGrupo_NombreDireccionGrupo)
```

```
        if (ModificarDireccionGrupo_auxiliarEncontradaDireccion > 0) then
```

```
            if
```

```
(Len(ModificarDireccionGrupo_auxiliarNombreDireccion)=Len(ModificarDireccionGrupo_NombreDireccionGrupo)) then
```

```
                Set ModificarDireccionGrupo_auxiliarDg = GroupAddress
```

```
                ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = true
```

```
            end if
```

```
        end if
```

```
    end if
```

```
Next
```

```
For each Device In Project.Devices
```

```

if ModificarDireccionGrupo_auxiliarYaModificado = false then

    ModificarDireccionGrupo_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
    ModificarDireccionGrupo_auxiliarEncontradoDispositivo =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDispositivo,ModificarDireccionGrupo_NombreDispositivo)

    if (ModificarDireccionGrupo_auxiliarEncontradoDispositivo > 0) then

        For Each DeviceCommunicationObject In Device.VisibleDeviceCommunicationObjects

            if DeviceCommunicationObject.CommunicationObjectNumber =
ModificarDireccionGrupo_numeroObjetoComunicacion then

                Set ModificarDireccionGrupo_auxiliarAs =
DeviceCommunicationObject.Associations.Add(ModificarDireccionGrupo_auxiliarDg)
                ModificarDireccionGrupo_auxiliarAs.Send =
ModificarDireccionGrupo_SendDireccionGrupo

            end if

        Next

        ModificarDireccionGrupo_auxiliarYaModificado = true

    end if

end if

Next

Next

```

```
,  
'(s) Actuador accionamiento, obj 16, "reenvio estado"  
'1.1.2.    L04 ESTADO  
,
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)  
ModificarDireccionGrupo_nombreDispositivo = "Actuador de 6 salidas"  
ModificarDireccionGrupo_nombreDispositivo = lcase (ModificarDireccionGrupo_nombreDispositivo)
```

```
'Poner aquí el número de objeto de comunicación a modificar  
ModificarDireccionGrupo_numeroObjetoComunicacion = 16
```

```
'Poner aquí el nombre EXACTO de la dirección de grupo (ya añadida) a la que quiero vincular el objeto de  
comunicación
```

```
ModificarDireccionGrupo_NombreDireccionGrupo = "L04 ESTADO"  
ModificarDireccionGrupo_NombreDireccionGrupo = Lcase(ModificarDireccionGrupo_NombreDireccionGrupo)
```

```
'Si queremos que sea send o no (booleano = [true, false])  
ModificarDireccionGrupo_SendDireccionGrupo = true
```

```
'valores iniciales variables auxiliares  
ModificarDireccionGrupo_auxiliarYaModificado = false  
ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false
```

```
For each GroupAddress in Project.GroupAddresses
```

```

    if (ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false) then

        ModificarDireccionGrupo_auxiliarNombreDireccion = LCase(CStr(GroupAddress.Name))
        ModificarDireccionGrupo_auxiliarEncontradaDireccion =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDireccion, ModificarDireccionGrupo_NombreDireccionGrupo)

        if (ModificarDireccionGrupo_auxiliarEncontradaDireccion > 0) then

            if
(Len(ModificarDireccionGrupo_auxiliarNombreDireccion)=Len(ModificarDireccionGrupo_NombreDireccionGrupo)) then

                Set ModificarDireccionGrupo_auxiliarDg = GroupAddress
                ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = true

            end if

        end if

    end if

Next

For each Device In Project.Devices

    if ModificarDireccionGrupo_auxiliarYaModificado = false then

        ModificarDireccionGrupo_auxiliarNombreDispositivo = LCase(CStr(Device.Name))

```

```

        ModificarDireccionGrupo_auxiliarEncontradoDispositivo =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDispositivo,ModificarDireccionGrupo_NombreDispositivo)

        if (ModificarDireccionGrupo_auxiliarEncontradoDispositivo > 0) then

                For Each DeviceCommunicationObject In Device.VisibleDeviceCommunicationObjects

                        if DeviceCommunicationObject.CommunicationObjectNumber =
ModificarDireccionGrupo_numeroObjetoComunicacion then

                                Set ModificarDireccionGrupo_auxiliarAs =
DeviceCommunicationObject.Associations.Add(ModificarDireccionGrupo_auxiliarDg)
                                ModificarDireccionGrupo_auxiliarAs.Send =
ModificarDireccionGrupo_SendDireccionGrupo

                                end if

                        Next

                        ModificarDireccionGrupo_auxiliarYaModificado = true

                end if

        end if

Next

,
'(ns) Entrada binaria, obj2, "objeto accionamiento"

```

'1.1.2. L04 ESTADO

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarDireccionGrupo\_nombreDispositivo = "Entrada binaria"

ModificarDireccionGrupo\_nombreDispositivo = lcase (ModificarDireccionGrupo\_nombreDispositivo)

'Poner aquí el número de objeto de comunicación a modificar

ModificarDireccionGrupo\_numeroObjetoComunicacion = 2

'Poner aquí el nombre EXACTO de la dirección de grupo (ya añadida) a la que quiero vincular el objeto de comunicación

ModificarDireccionGrupo\_NombreDireccionGrupo = "L04 ESTADO"

ModificarDireccionGrupo\_NombreDireccionGrupo = Lcase(ModificarDireccionGrupo\_NombreDireccionGrupo)

'Si queremos que sea send o no (booleano = [true, false])

ModificarDireccionGrupo\_SendDireccionGrupo = false

'valores iniciales variables auxiliares

ModificarDireccionGrupo\_auxiliarYaModificado = false

ModificarDireccionGrupo\_auxiliarYaEncontradaDireccion = false

For each GroupAddress in Project.GroupAddresses

    if (ModificarDireccionGrupo\_auxiliarYaEncontradaDireccion = false) then

        ModificarDireccionGrupo\_auxiliarNombreDireccion = LCase(Cstr(GroupAddress.Name))

        ModificarDireccionGrupo\_auxiliarEncontradaDireccion =

InStrRev(ModificarDireccionGrupo\_auxiliarNombreDireccion, ModificarDireccionGrupo\_NombreDireccionGrupo)

```

        if (ModificarDireccionGrupo_auxiliarEncontradaDireccion > 0) then
            if
(Len(ModificarDireccionGrupo_auxiliarNombreDireccion)=Len(ModificarDireccionGrupo_NombreDireccionGrupo)) then
                Set ModificarDireccionGrupo_auxiliarDg = GroupAddress
                ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = true
            end if
        end if

    end if
Next

For each Device In Project.Devices

    if ModificarDireccionGrupo_auxiliarYaModificado = false then

        ModificarDireccionGrupo_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
        ModificarDireccionGrupo_auxiliarEncontradoDispositivo =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDispositivo,ModificarDireccionGrupo_NombreDispositivo)

        if (ModificarDireccionGrupo_auxiliarEncontradoDispositivo > 0) then

            For Each DeviceCommunicationObject In Device.VisibleDeviceCommunicationObjects

```

```
        if DeviceCommunicationObject.CommunicationObjectNumber =  
ModificarDireccionGrupo_numeroObjetoComunicacion then  
  
            Set ModificarDireccionGrupo_auxiliarAs =  
DeviceCommunicationObject.Associations.Add(ModificarDireccionGrupo_auxiliarDg)  
            ModificarDireccionGrupo_auxiliarAs.Send =  
ModificarDireccionGrupo_SendDireccionGrupo  
  
        end if  
  
    Next  
  
    ModificarDireccionGrupo_auxiliarYaModificado = true  
  
end if  
  
end if  
  
Next  
  
End Sub
```



## ***Anexo II. Plantilla JET***

---



```

'
'Inicialización de variables del macro AnadirDispositivo
'

'variables a modificar
Dim AnadirDispositivo_NombreReferencia
Dim AnadirDispositivo_DireccionFisica
Dim AnadirDispositivo_Fabricante
'variables auxiliares
Dim AnadirDispositivo_AuxiliarYaAnadido
Dim AnadirDispositivo_AuxiliarEncontrado
Dim AnadirDispositivo_AuxiliarNuevoDispositivo
Dim AnadirDispositivo_AuxiliarNombre
Dim AnadirDispositivo_AuxiliarFabricante
Dim AnadirDispositivo_AuxiliarReferencia
Dim AnadirDispositivo_auxDR
Dim inc
Dim numeroEntitys

'
'Inicialización de variables del macro AnadirDireccionGrupo
'

'variables que hay que fijar
Dim AnadirDireccionGrupo_1erDigito
Dim AnadirDireccionGrupo_2oDigito
Dim AnadirDireccionGrupo_3erDigito
Dim AnadirDireccionGrupo_NombreDireccion

'variables auxiliares para el funcionamiento del macro
Dim AnadirDireccionGrupo_auxiliarEncontradoMainGroup
Dim AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup
Dim AnadirDireccionGrupo_auxiliarMainGroup
Dim AnadirDireccionGrupo_va
Dim AnadirDireccionGrupo_vb
Dim AnadirDireccionGrupo_vc

'
'Inicialización de variables del macro ModificarParametro
'

```

```

'variables que hay que fijar
Dim ModificarParametro_nombreDispositivo
Dim ModificarParametro_nombreParametro
Dim ModificarParametro_valorParametro
Dim ModificarParametro_ordenAparicion

'variables auxiliares para el funcionamiento del macro
Dim ModificarParametro_auxiliarYaModificado
Dim ModificarParametro_auxiliarNombreDispositivo
Dim ModificarParametro_auxiliarNombreParametro
Dim ModificarParametro_auxiliarEncontradoDispositivo
Dim ModificarParametro_auxiliarEncontradoParametro
Dim ModificarParametro_auxiliarInc

,
'Inicialización de variables del macro ModificarDireccionGrupo
,

'variables que hay que fijar
Dim ModificarDireccionGrupo_nombreDispositivo
Dim ModificarDireccionGrupo_numeroObjetoComuniacion
Dim ModificarDireccionGrupo_NombreDireccionGrupo
Dim ModificarDireccionGrupo_SendDireccionGrupo

'variables auxiliares para el funcionamiento del macro
Dim ModificarDireccionGrupo_auxiliarYaModificado
Dim ModificarDireccionGrupo_auxiliarNombreDispositivo
Dim ModificarDireccionGrupo_auxiliarEncontradoDispositivo
Dim ModificarDireccionGrupo_auxiliarAs
Dim ModificarDireccionGrupo_auxiliarDg
Dim ModificarDireccionGrupo_auxiliarYaEncontradaDireccion
Dim ModificarDireccionGrupo_auxiliarNombreDireccion
Dim ModificarDireccionGrupo_auxiliarEncontradaDireccion

<!--
PASO 1
Parte de la plantilla para añadir los dispositivos
-->

, .....
, .....
,
Se añaden los dispositivos

```

```

.....
.....
<c:iterate select = "//BusDevice" var = "c"> <%-- iterador para añadir los dispositivos al proyecto --%>
'
'      INICIO DISPOSITIVO <c:get select = "upper-case($c/@name)"/>
'
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = <c:get select = "lower-case($c/@name)"/>
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = <c:get select = "lower-case($c/@manufacturer)"/>
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
<c:setVariable select = "($c/@adress)" var="DevAddress"/>

'Variables auxiliares (NO TOCAR)
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
  for each functionalEntity in manufacturer.FunctionalEntities
    if AnadirDispositivo_AuxiliarYaAnadido = false then
      AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
      if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then
        numeroEntities = functionalEntity.FunctionalEntities.count
        for inc=1 to numeroEntities step 1
          AnadirDispositivo_AuxiliarReferencia =
LCase(CStr(FunctionalEntity.FunctionalEntities.Item(Cint(inc)).name))
          AnadirDispositivo_AuxiliarEncontrado =
InStrRev(AnadirDispositivo_AuxiliarReferencia,AnadirDispositivo_NombreReferencia)
          if (AnadirDispositivo_AuxiliarEncontrado > 0) then
            Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(FunctionalEntity.FunctionalEntities.Item(Cint(inc)).DeviceTemplates.Item(1))
            AnadirDispositivo_AuxiliarNuevoDispositivo.Name = AnadirDispositivo_AuxiliarNombre
            Set AnadirDispositivo_AuxiliarNuevoDispositivo.line = Project.Areas.Item(2).Lines.Item(2)
            AnadirDispositivo_AuxiliarNuevoDispositivo.Address = AnadirDispositivo_DireccionFisica
            AnadirDispositivo_AuxiliarYaAnadido = true

```

```

                                end if
                            Next
                        end if
                    end if
                Next
            Next
        '
        '      FIN DISPOSITIVO      <c:get select = "upper-case($c/@name)"/>
        '
</c:iterate> <%-- FIN iterador para añadir los dispositivos al proyecto --%>

<%--
PASO 2
Parte de la plantilla para añadir las direcciones de grupo
--%>
'
'                                     Se añaden las direcciones de grupo
'
<c:iterate select = "//Association" var = "d"> <%-- iterador para añadir las direcciones de grupo --%>
'
' Añadimos la <c:get select = "$d/@MainGroup"/><c:get select = "$d/@SubGroup"/><c:get select = "$d/@GroupAddress"/> <c:get select =
"$d/@name"/>
'

AnadirDireccionGrupo_1erDigito = <c:get select = "$d/@MainGroup"/>
AnadirDireccionGrupo_2oDigito = <c:get select = "$d/@SubGroup"/>
AnadirDireccionGrupo_3erDigito = <c:get select = "$d/@GroupAddress"/>
AnadirDireccionGrupo_NombreDireccion = "<c:get select = "$d/@name"/>"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true

```

```

Set AnadirDireccionGrupo_va = MainGroup

For each MiddleGroup in MainGroup.MiddleGroups

    if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

        AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
        Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito))
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito))
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

```





```

For each Device In Project.Devices
    if ModificarParametro_auxiliarYaModificado = false then
        ModificarParametro_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
        ModificarParametro_auxiliarEncontradoDispositivo =
InStrRev(ModificarParametro_auxiliarNombreDispositivo, ModificarParametro_NombreDispositivo)
        if (ModificarParametro_auxiliarEncontradoDispositivo > 0) then
            for each Parameter in Device.DeviceParameters
                if (ModificarParametro_auxiliarYaModificado = false) then
                    ModificarParametro_auxiliarNombreParametro = LCase(CStr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)
                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then
                        if (Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro))
then
                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1
                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion) then
                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true
                            end if
                        end if
                    end if
                end if
            end if
        end if
    Next
end if
end if

```

Next

```
</c:iterate> <%-- FIN iterador 2 para modificar los parámetros --%>
</c:iterate> <%-- FIN iterador 1 para modificar los parámetros --%>

<%--
PASO 4
Parte de la plantilla para enlazar las direcciones de grupo
--%>
'
.....
'
                                Se enlazan las direcciones de grupo
'
.....
<c:iterate select = "//Association" var = "g"> <%-- Iterador en Association para asociar direcciones de grupo--%>
    <c:setVariable select="$g/sourceAssociation" var = "gs"/>
    <c:setVariable select="$g/targetAssociation" var = "gt"/>
    <c:setVariable select="$gs/.." var = "gsp"/>
    <c:setVariable select="$gt/.." var = "gtp"/>
<%--
sourceAssociation
--%>
'
' (s) <c:get select = "$gsp/@name"/>, obj <c:get select = "$gs/@id"/>, "<c:get select = "$gs/@name"/>"
' <c:get select = "$g/@MainGroup"/>/<c:get select = "$g/@SubGroup"/>/<c:get select = "$g/@GroupAddress"/> <c:get select = "$g/@name"/>
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarDireccionGrupo_nombreDispositivo = "<c:get select = "$gsp/@name"/>"
ModificarDireccionGrupo_nombreDispositivo = lcase (ModificarDireccionGrupo_nombreDispositivo)

'Poner aquí el número de objeto de comunicación a modificar
ModificarDireccionGrupo_numeroObjetoComunicacion = <c:get select = "$gs/@id"/>

'Poner aquí el nombre EXACTO de la dirección de grupo (ya añadida) a la que quiero vincular el objeto de comunicación
ModificarDireccionGrupo_NombreDireccionGrupo = "<c:get select = "$g/@name"/>"
ModificarDireccionGrupo_NombreDireccionGrupo = Lcase (ModificarDireccionGrupo_NombreDireccionGrupo)

'Si queremos que sea send o no (booleano = [true, false])
ModificarDireccionGrupo_SendDireccionGrupo = true <%--¿? como es sourceAssociation lo pongo a true ¿?--%>

'valores iniciales variables auxiliares
ModificarDireccionGrupo_auxiliarYaModificado = false
```

```

ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false

For each GroupAddress in Project.GroupAddresses
    if (ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false) then
        ModificarDireccionGrupo_auxiliarNombreDireccion = LCase(CStr(GroupAddress.Name))
        ModificarDireccionGrupo_auxiliarEncontradaDireccion =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDireccion, ModificarDireccionGrupo_NombreDireccionGrupo)
        if (ModificarDireccionGrupo_auxiliarEncontradaDireccion > 0) then
            if (Len(ModificarDireccionGrupo_auxiliarNombreDireccion)=Len(ModificarDireccionGrupo_NombreDireccionGrupo)) then
                Set ModificarDireccionGrupo_auxiliarDg = GroupAddress
                ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = true
            end if
        end if
    end if
Next

For each Device In Project.Devices
    if ModificarDireccionGrupo_auxiliarYaModificado = false then
        ModificarDireccionGrupo_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
        ModificarDireccionGrupo_auxiliarEncontradoDispositivo =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDispositivo, ModificarDireccionGrupo_NombreDispositivo)
        if (ModificarDireccionGrupo_auxiliarEncontradoDispositivo > 0) then
            For Each DeviceCommunicationObject In Device.VisibleDeviceCommunicationObjects
                if DeviceCommunicationObject.CommunicationObjectNumber =
ModificarDireccionGrupo_numeroObjetoComunicacion then
                    Set ModificarDireccionGrupo_auxiliarAs =
DeviceCommunicationObject.Associations.Add(ModificarDireccionGrupo_auxiliarDg)

```

```
ModificarDireccionGrupo_auxiliarAs.Send = ModificarDireccionGrupo_SendDireccionGrupo
```

```
end if
```

```
Next
```

```
ModificarDireccionGrupo_auxiliarYaModificado = true
```

```
end if
```

```
end if
```

```
Next
```

```
<%--
```

```
targetAssociation
```

```
--%>
```

```
,
```

```
'(ns) <c:get select = "$gtp/@name"/>, obj <c:get select = "$gt/@id"/>, "<c:get select = "$gt/@name"/>"
```

```
'<c:get select = "$g/@MainGroup"/>/<c:get select = "$g/@SubGroup"/>/<c:get select = "$g/@GroupAddress"/> <c:get select = "$g/@name"/>
```

```
,
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
```

```
ModificarDireccionGrupo_nombreDispositivo = "<c:get select = "$gtp/@name"/>"<%--¿? ojo, hay que poner el nombre del dispositivo al que pertenece la dirección ¿? -- ../@lang selecciona el atributo lang del padre del nodo contextual--%>
```

```
ModificarDireccionGrupo_nombreDispositivo = lcase (ModificarDireccionGrupo_nombreDispositivo)
```

```
'Poner aquí el número de objeto de comunicación a modificar
```

```
ModificarDireccionGrupo_numeroObjetoComunicacion = <c:get select = "$gt/@id"/>
```

```
'Poner aquí el nombre EXACTO de la dirección de grupo (ya añadida) a la que quiero vincular el objeto de comunicación
```

```
ModificarDireccionGrupo_NombreDireccionGrupo = "<c:get select = "$g/@name"/>"
```

```
ModificarDireccionGrupo_NombreDireccionGrupo = Lcase (ModificarDireccionGrupo_NombreDireccionGrupo)
```

```
'Si queremos que sea send o no (booleano = [true, false])
```

```
ModificarDireccionGrupo_SendDireccionGrupo = false <%--¿? como es targetAssociation lo pongo a false ¿?--%>
```

```
'valores iniciales variables auxiliares
```

```
ModificarDireccionGrupo_auxiliarYaModificado = false
```

```
ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false
```

```
For each GroupAddress in Project.GroupAddresses
```

```

if (ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = false) then

    ModificarDireccionGrupo_auxiliarNombreDireccion = LCase(CStr(GroupAddress.Name))
    ModificarDireccionGrupo_auxiliarEncontradaDireccion =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDireccion, ModificarDireccionGrupo_NombreDireccionGrupo)

    if (ModificarDireccionGrupo_auxiliarEncontradaDireccion > 0) then

        if (Len(ModificarDireccionGrupo_auxiliarNombreDireccion)=Len(ModificarDireccionGrupo_NombreDireccionGrupo)) then

            Set ModificarDireccionGrupo_auxiliarDg = GroupAddress
            ModificarDireccionGrupo_auxiliarYaEncontradaDireccion = true

        end if

    end if

end if

Next

For each Device In Project.Devices

    if ModificarDireccionGrupo_auxiliarYaModificado = false then

        ModificarDireccionGrupo_auxiliarNombreDispositivo = LCase(CStr(Device.Name))
        ModificarDireccionGrupo_auxiliarEncontradoDispositivo =
InStrRev(ModificarDireccionGrupo_auxiliarNombreDispositivo, ModificarDireccionGrupo_NombreDispositivo)

        if (ModificarDireccionGrupo_auxiliarEncontradoDispositivo > 0) then

            For Each DeviceCommunicationObject In Device.VisibleDeviceCommunicationObjects

                if DeviceCommunicationObject.CommunicationObjectNumber =
ModificarDireccionGrupo_numeroObjetoComunicacion then

                    Set ModificarDireccionGrupo_auxiliarAs =
DeviceCommunicationObject.Associations.Add(ModificarDireccionGrupo_auxiliarDg)
                    ModificarDireccionGrupo_auxiliarAs.Send = ModificarDireccionGrupo_SendDireccionGrupo

                end if

            end if

        end if

    end if

end if

```

```
                Next
                ModificarDireccionGrupo_auxiliarYaModificado = true
            end if
        end if
    Next
    </c:iterate> <%-- FIN iterador en Association para asociar direcciones de grupo --%>
</c:iterate> <%-- FIN iterador de los proyectos --%>
```

## ***Anexo III. Código generado con la transformación***

---

```
'David Montoro Mouzo
'2010
'Universidad Politécnica de Cartagena
'División de sistemas e ingeniería electrónica (DSIE)
'-----
'
'Macro para crear el proyecto ProyectoMaleta
'Hay que abrir en ETS3 un proyecto en blanco y ejecutar la macro
'
'-----
'-----
'-----
```

Sub Main

```
'Se obtiene el Active Project
Dim Project '[EtcDom.IProject]
Set Project = Application.Project

, .....
, .....
'
' Inicialización de variables
, .....
, .....

'
'Inicialización de variables del macro AnadirDispositivo
'
```



```

'variables a modificar
Dim AnadirDispositivo_NombreReferencia
Dim AnadirDispositivo_DireccionFisica
Dim AnadirDispositivo_Fabricante
Dim AnadirDispositivo_no_pedido
Dim AnadirDispositivo_version
Dim AnadirDispositivo_nombre_programa

'variables auxiliares
Dim AnadirDispositivo_AuxiliarYaAnadido
Dim AnadirDispositivo_AuxiliarEncontrado
Dim AnadirDispositivo_AuxiliarNuevoDispositivo
Dim AnadirDispositivo_AuxiliarNombre
Dim AnadirDispositivo_AuxiliarFabricante
Dim AnadirDispositivo_AuxiliarReferencia
Dim AnadirDispositivo_auxDR
Dim AnadirDispositivo_inc
Dim AnadirDispositivo_numeroEntities
Dim AnadirDispositivo_aux_entity
Dim AnadirDispositivo_aux_appProg
Dim AnadirDispositivo_no_pedido_auxiliar
Dim AnadirDispositivo_version_auxiliar
Dim AnadirDispositivo_nombre_programa_auxiliar

,

'Inicialización de variables del macro AnadirDireccionGrupo
,

'variables que hay que fijar

```

```

Dim AnadirDireccionGrupo_1erDigito
Dim AnadirDireccionGrupo_2oDigito
Dim AnadirDireccionGrupo_3erDigito
Dim AnadirDireccionGrupo_NombreDireccion

'variables auxiliares para el funcionamiento del macro
Dim AnadirDireccionGrupo_auxiliarEncontradoMainGroup
Dim AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup
Dim AnadirDireccionGrupo_auxiliarMainGroup
Dim AnadirDireccionGrupo_va
Dim AnadirDireccionGrupo_vb
Dim AnadirDireccionGrupo_vc

,

'Inicialización de variables del macro ModificarParametro
,

'variables que hay que fijar
Dim ModificarParametro_nombreDispositivo
Dim ModificarParametro_nombreParametro
Dim ModificarParametro_valorParametro
Dim ModificarParametro_ordenAparicion

'variables auxiliares para el funcionamiento del macro
Dim ModificarParametro_auxiliarYaModificado
Dim ModificarParametro_auxiliarNombreDispositivo
Dim ModificarParametro_auxiliarNombreParametro
Dim ModificarParametro_auxiliarEncontradoDispositivo

```

```

Dim ModificarParametro_auxiliarEncontradoParametro
Dim ModificarParametro_auxiliarInc

'
'Inicialización de variables del macro ModificarDireccionGrupo
'

'variables que hay que fijar
Dim ModificarDireccionGrupo_nombreDispositivo
Dim ModificarDireccionGrupo_numeroObjetoComuniacion
Dim ModificarDireccionGrupo_NombreDireccionGrupo
Dim ModificarDireccionGrupo_SendDireccionGrupo

'variables auxiliares para el funcionamiento del macro
Dim ModificarDireccionGrupo_auxiliarYaModificado
Dim ModificarDireccionGrupo_auxiliarNombreDispositivo
Dim ModificarDireccionGrupo_auxiliarEncontradoDispositivo
Dim ModificarDireccionGrupo_auxiliarAs
Dim ModificarDireccionGrupo_auxiliarDg
Dim ModificarDireccionGrupo_auxiliarYaEncontradaDireccion
Dim ModificarDireccionGrupo_auxiliarNombreDireccion
Dim ModificarDireccionGrupo_auxiliarEncontradaDireccion

.....
.....
'
'                               Se añaden los dispositivos
'
.....
.....

```

```

'
'      INICIO DISPOSITIVO      ACTUADOR 6 SALIDAS
'
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Actuador de 6 salidas (1)"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el No pedido
AnadirDispositivo_no_pedido = "2136.6REG"

'Poner aquí el Nombre programa
AnadirDispositivo_nombre_programa = "Accionamiento con reenvío, func. lógica, temporiza"

'Poner aquí la Version
AnadirDispositivo_version = "1"

'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 3

'Variables auxiliares

```

```

AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
  for each functionalEntity in manufacturer.FunctionalEntities
    if AnadirDispositivo_AuxiliarYaAnadido = false then
      AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
      if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

        AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

        for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

          Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

          for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

            AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

            set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

            if AnadirDispositivo_aux_appProg is nothing then

              else

```

```

AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)
AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)
AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_DireccionFisica
AnadirDispositivo_AuxiliarYaAnadido = true

end if
end if

set AnadirDispositivo_aux_appProg = nothing

Next
Next
end if
end if
Next
Next

```

```
'  
' FIN DISPOSITIVO ACTUADOR 6 SALIDAS  
'
```

```
'  
' INICIO DISPOSITIVO ACTUADOR 6 SALIDAS  
'
```

```
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir  
'No es necesario que sea el nombre completo, basta con una parte del mismo  
AnadirDispositivo_NombreReferencia = "Actuador de 6 salidas (2)"  
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)
```

```
'Poner aquí el No pedido  
AnadirDispositivo_no_pedido = "2136.6REG"
```

```
'Poner aquí el Nombre programa  
AnadirDispositivo_nombre_programa = "Accionamiento con reenvío, func. lógica, temporiza"
```

```
'Poner aquí la Version  
AnadirDispositivo_version = "1"
```

```
'Poner aquí el fabricante (nombre exacto)  
AnadirDispositivo_Fabricante = "Albrecht Jung"
```

```

AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 4

'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
    for each functionalEntity in manufacturer.FunctionalEntities
        if AnadirDispositivo_AuxiliarYaAnadido = false then
            AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
            if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

                AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

                for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

                    Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

                    for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

                        AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

```



```

set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

if AnadirDispositivo_aux_appProg is nothing then

else
    AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
    AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

    if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

        Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)
        AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
        Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)
        AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_DireccionFisica
        AnadirDispositivo_AuxiliarYaAnadido = true

    end if
end if

set AnadirDispositivo_aux_appProg = nothing

Next

```

```

Next
    end if
end if
Next
Next

'
'   FIN DISPOSITIVO   ACTUADOR 6 SALIDAS
'

'
'   INICIO DISPOSITIVO   TECLADO DE 1 FASE
'

'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Teclado 1 fase"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el No pedido
AnadirDispositivo_no_pedido = "2071.02LED"

'Poner aquí el Nombre programa
AnadirDispositivo_nombre_programa = "Accionar, regular, persiana 105601"

'Poner aquí la Version

```

```
AnadirDispositivo_version = "1"
```

```
'Poner aquí el fabricante (nombre exacto)
```

```
AnadirDispositivo_Fabricante = "Albrecht Jung"
```

```
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)
```

```
'Poner aquí la direccion fisica del nuevo dispositivo
```

```
AnadirDispositivo_DireccionFisica = 6
```

```
'Variables auxiliares
```

```
AnadirDispositivo_AuxiliarYaAnadido = false
```

```
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
```

```
Set AnadirDispositivo_auxDR = Project.parent
```

```
set AnadirDispositivo_aux_appProg = nothing
```

```
for each manufacturer in AnadirDispositivo_auxDR.manufacturers
```

```
  for each functionalEntity in manufacturer.FunctionalEntities
```

```
    if AnadirDispositivo_AuxiliarYaAnadido = false then
```

```
      AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
```

```
      if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then
```

```
        AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count
```

```
        for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1
```

```
          Set AnadirDispositivo_aux_entity =
```

```
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))
```

```

        for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

            AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

            set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

            if AnadirDispositivo_aux_appProg is nothing then
            else
                AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
                AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

                if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

                    Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)
                    AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
                    Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)
                    AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_DireccionFisica
                    AnadirDispositivo_AuxiliarYaAnadido = true

                end if
            end if
        end for

```

```

end if

set AnadirDispositivo_aux_appProg = nothing

Next
Next
end if
end if
Next
Next

'
' FIN DISPOSITIVO TECLADO DE 1 FASE
'

'
' INICIO DISPOSITIVO ENTRADA BINARA, 4-CANALES
'

'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Entrada binaria 4 canales (1)"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el No pedido
AnadirDispositivo_no_pedido = "2076-4T"

```

```

'Poner aquí el Nombre programa
AnadirDispositivo_nombre_programa = "4 entradas, 2 salidas 705701"

'Poner aquí la Version
AnadirDispositivo_version = "1"

'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 9

'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
    for each functionalEntity in manufacturer.FunctionalEntities
        if AnadirDispositivo_AuxiliarYaAnadido = false then
            AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
            if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

                AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

```

```

for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

    Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

    for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

        AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

        set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

        if AnadirDispositivo_aux_appProg is nothing then

            else
                AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
                AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

                if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

                    Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)
                    AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
                    Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)

```

```

AnadirDispositivo_DireccionFisica
                                AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
                                AnadirDispositivo_AuxiliarYaAnadido = true
                                end if
                                end if
                                set AnadirDispositivo_aux_appProg = nothing
                                Next
                                Next
                                end if
                                end if
                                Next
Next
'
'   FIN DISPOSITIVO   ENTRADA BINARA, 4-CANALES
'
'
'   INICIO DISPOSITIVO   ENTRADA BINARA, 4-CANALES
'
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Entrada binaria 4 canales (2)"

```



```
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)
```

```
'Poner aquí el No pedido  
AnadirDispositivo_no_pedido = "2076-4T"
```

```
'Poner aquí el Nombre programa  
AnadirDispositivo_nombre_programa = "4 entradas, 2 salidas 705701"
```

```
'Poner aquí la Version  
AnadirDispositivo_version = "1"
```

```
'Poner aquí el fabricante (nombre exacto)  
AnadirDispositivo_Fabricante = "Albrecht Jung"  
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)
```

```
'Poner aquí la direccion fisica del nuevo dispositivo  
AnadirDispositivo_DireccionFisica = 10
```

```
'Variables auxiliares  
AnadirDispositivo_AuxiliarYaAnadido = false  
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia  
Set AnadirDispositivo_auxDR = Project.parent  
set AnadirDispositivo_aux_appProg = nothing
```

```
for each manufacturer in AnadirDispositivo_auxDR.manufacturers  
  for each functionalEntity in manufacturer.FunctionalEntities  
    if AnadirDispositivo_AuxiliarYaAnadido = false then
```

```

AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

    AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

    for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

        Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

        for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

            AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

            set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

            if AnadirDispositivo_aux_appProg is nothing then

                else
                    AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
                    AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

                    if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

```



```

'      INICIO DISPOSITIVO      ENTRADA BINARA, 4-CANALES
,

'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Entrada binaria 4 canales (3)"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el No pedido
AnadirDispositivo_no_pedido = "2076-4T"

'Poner aquí el Nombre programa
AnadirDispositivo_nombre_programa = "4 entradas, 2 salidas 705701"

'Poner aquí la Version
AnadirDispositivo_version = "1"

'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 11

'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent

```

```

set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
  for each functionalEntity in manufacturer.FunctionalEntities
    if AnadirDispositivo_AuxiliarYaAnadido = false then
      AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
      if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

        AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

        for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

          Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

          for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

            AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

            set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

            if AnadirDispositivo_aux_appProg is nothing then

            else
              AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
              AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

```

```

        if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

                Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)
                AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
                Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)
                AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_DireccionFisica
                AnadirDispositivo_AuxiliarYaAnadido = true

        end if
    end if

    set AnadirDispositivo_aux_appProg = nothing

        Next
            Next
                end if
            end if
        Next
    Next

    '
    '     FIN DISPOSITIVO  ENTRADA BINARA, 4-CANALES

```

,

,

' INICIO DISPOSITIVO ENTRADA BINARA, 4-CANALES

,

'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir  
'No es necesario que sea el nombre completo, basta con una parte del mismo  
AnadirDispositivo\_NombreReferencia = "Entrada binaria 4 canales (4)"  
AnadirDispositivo\_NombreReferencia = LCase(AnadirDispositivo\_NombreReferencia)

'Poner aquí el No pedido  
AnadirDispositivo\_no\_pedido = "2076-4T"

'Poner aquí el Nombre programa  
AnadirDispositivo\_nombre\_programa = "4 entradas, 2 salidas 705701"

'Poner aquí la Version  
AnadirDispositivo\_version = "1"

'Poner aquí el fabricante (nombre exacto)  
AnadirDispositivo\_Fabricante = "Albrecht Jung"  
AnadirDispositivo\_Fabricante = LCase(AnadirDispositivo\_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo  
AnadirDispositivo\_DireccionFisica = 12

```

'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
  for each functionalEntity in manufacturer.FunctionalEntities
    if AnadirDispositivo_AuxiliarYaAnadido = false then
      AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
      if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

        AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

        for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

          Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

          for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

            AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

            set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

            if AnadirDispositivo_aux_appProg is nothing then

```



```

else
    AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
    AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

    if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

        Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)
        AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
        Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)
        AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_DireccionFisica
        AnadirDispositivo_AuxiliarYaAnadido = true

    end if
end if

set AnadirDispositivo_aux_appProg = nothing

Next
Next
end if
end if
Next

```

Next

```
'  
' FIN DISPOSITIVO ENTRADA BINARA, 4-CANALES  
'
```

```
'  
' INICIO DISPOSITIVO ENTRADA BINARA, 4-CANALES  
'
```

```
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir  
'No es necesario que sea el nombre completo, basta con una parte del mismo  
AnadirDispositivo_NombreReferencia = "Entrada binaria 4 canales (5)"  
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)
```

```
'Poner aquí el No pedido  
AnadirDispositivo_no_pedido = "2076-4T"
```

```
'Poner aquí el Nombre programa  
AnadirDispositivo_nombre_programa = "4 entradas, 2 salidas 705701"
```

```
'Poner aquí la Version  
AnadirDispositivo_version = "1"
```

```
'Poner aquí el fabricante (nombre exacto)
```

```

AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 13

'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
    for each functionalEntity in manufacturer.FunctionalEntities
        if AnadirDispositivo_AuxiliarYaAnadido = false then
            AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
            if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

                AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

                for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

                    Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

                    for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

                        AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

```

```

set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

if AnadirDispositivo_aux_appProg is nothing then

else
    AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
    AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

    if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

        Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)
        AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
        Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)
        AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_DireccionFisica
        AnadirDispositivo_AuxiliarYaAnadido = true

    end if
end if

set AnadirDispositivo_aux_appProg = nothing

```

```

Next
Next
end if
end if
Next
Next

```

```

'
'   FIN DISPOSITIVO   ENTRADA BINARA, 4-CANALES
'

```

```

'
'   INICIO DISPOSITIVO   PULSADOR, 4 FASES
'

```

```

'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Pulsador 4 fases"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

```

```

'Poner aquí el No pedido
AnadirDispositivo_no_pedido = "2074NABS"

```

```

'Poner aquí el Nombre programa
AnadirDispositivo_nombre_programa = "Persiana 102C01"

```

```
'Poner aquí la Version
AnadirDispositivo_version = "1"
```

```
'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)
```

```
'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 16
```

```
'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing
```

```
for each manufacturer in AnadirDispositivo_auxDR.manufacturers
  for each functionalEntity in manufacturer.FunctionalEntities
    if AnadirDispositivo_AuxiliarYaAnadido = false then
      AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
      if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

          AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

          for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

              Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))
```

```

        for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

            AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

            set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

            if AnadirDispositivo_aux_appProg is nothing then

            else
                AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
                AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

                if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

                    Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)
                    AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre
                    Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
Project.Areas.Item(2).Lines.Item(2)
                    AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_DireccionFisica
                    AnadirDispositivo_AuxiliarYaAnadido = true

```

```

                                end if
                                end if

                                set AnadirDispositivo_aux_appProg = nothing

                                Next
                                Next
                                end if
                                end if
                                Next
Next
'
'
'   FIN DISPOSITIVO   PULSADOR, 4 FASES
'
'
'
'   INICIO DISPOSITIVO   ACTUADOR DE PERSIANAS
'
'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir
'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Actuador de persianas"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el No pedido

```



```

AnadirDispositivo_no_pedido = "GH Q631 0064 R0111"

'Poner aquí el Nombre programa
AnadirDispositivo_nombre_programa = "Shutter,4f M/2"

'Poner aquí la Version
AnadirDispositivo_version = "32"

'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "ABB"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 2

'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers
    for each functionalEntity in manufacturer.FunctionalEntities
        if AnadirDispositivo_AuxiliarYaAnadido = false then
            AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
            if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

                AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

```

```

for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

    Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

    for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

        AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

        set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

        if AnadirDispositivo_aux_appProg is nothing then

            else

                AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)

                AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

                if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

                    Set AnadirDispositivo_AuxiliarNuevoDispositivo =
Project.Devices.Add(deviceTemplate)

                    AnadirDispositivo_AuxiliarNuevoDispositivo.Name =
AnadirDispositivo_AuxiliarNombre

```

```

Project.Areas.Item(2).Lines.Item(2)
AnadirDispositivo_DireccionFisica

Set AnadirDispositivo_AuxiliarNuevoDispositivo.line =
AnadirDispositivo_AuxiliarNuevoDispositivo.Address =
AnadirDispositivo_AuxiliarYaAnadido = true

end if
end if

set AnadirDispositivo_aux_appProg = nothing

Next
Next
end if
end if
Next
Next

'
' FIN DISPOSITIVO ACTUADOR DE PERSIANAS
'

'
' INICIO DISPOSITIVO REGULADOR FLUORESCENCIA
'

'Poner aquí el nombre (nombre del catalogo) del dispositivo que quiero añadir

```

```

'No es necesario que sea el nombre completo, basta con una parte del mismo
AnadirDispositivo_NombreReferencia = "Regulador fluorescencia"
AnadirDispositivo_NombreReferencia = LCase(AnadirDispositivo_NombreReferencia)

'Poner aquí el No pedido
AnadirDispositivo_no_pedido = "2193REG"

'Poner aquí el Nombre programa
AnadirDispositivo_nombre_programa = "Regulador fluorescencia 301801"

'Poner aquí la Version
AnadirDispositivo_version = "1"

'Poner aquí el fabricante (nombre exacto)
AnadirDispositivo_Fabricante = "Albrecht Jung"
AnadirDispositivo_Fabricante = LCase(AnadirDispositivo_Fabricante)

'Poner aquí la direccion fisica del nuevo dispositivo
AnadirDispositivo_DireccionFisica = 1

'Variables auxiliares
AnadirDispositivo_AuxiliarYaAnadido = false
AnadirDispositivo_AuxiliarNombre = "nuevoDispositivo "+AnadirDispositivo_NombreReferencia
Set AnadirDispositivo_auxDR = Project.parent
set AnadirDispositivo_aux_appProg = nothing

for each manufacturer in AnadirDispositivo_auxDR.manufacturers

```

```

for each functionalEntity in manufacturer.FunctionalEntities
  if AnadirDispositivo_AuxiliarYaAnadido = false then
    AnadirDispositivo_AuxiliarFabricante = LCase(CStr(manufacturer.name))
    if (AnadirDispositivo_AuxiliarFabricante = AnadirDispositivo_Fabricante) then

      AnadirDispositivo_numeroEntities = functionalEntity.FunctionalEntities.count

      for AnadirDispositivo_inc=1 to AnadirDispositivo_numeroEntities step 1

        Set AnadirDispositivo_aux_entity =
functionalEntity.functionalEntities.Item(cint(AnadirDispositivo_inc))

        for each deviceTemplate in AnadirDispositivo_aux_entity.deviceTemplates

          AnadirDispositivo_no_pedido_auxiliar =
Cstr(deviceTemplate.catalogEntry.OrderNumber)

          set AnadirDispositivo_aux_appProg = deviceTemplate.ApplicationProgram

          if AnadirDispositivo_aux_appProg is nothing then

            else
              AnadirDispositivo_nombre_programa_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Name)
              AnadirDispositivo_version_auxiliar =
Cstr(AnadirDispositivo_aux_appProg.Version)

              if (AnadirDispositivo_nombre_programa =
AnadirDispositivo_nombre_programa_auxiliar AND AnadirDispositivo_version = AnadirDispositivo_version_auxiliar AND
AnadirDispositivo_no_pedido_auxiliar = AnadirDispositivo_no_pedido AND AnadirDispositivo_AuxiliarYaAnadido = false) then

```



```

.....
.....
'
                                Se añaden las direcciones de grupo
.....
.....

'
'Añadimos la 1/0/0 Mover Todas Persianas
'

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 0
AnadirDireccionGrupo_3erDigito = 0
AnadirDireccionGrupo_NombreDireccion = "Mover Todas Persianas"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

```

```

For each MiddleGroup in MainGroup.MiddleGroups

    if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

        AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
        Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

```



```

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/0/1 Stop Todas Persianas
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 0
AnadirDireccionGrupo_3erDigito = 1
AnadirDireccionGrupo_NombreDireccion = "Stop Todas Persianas"

```

```

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

```

```

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```
end if
```

```
,
```

```
'Añadimos la 1/0/2 Iluminación todas
```

```
'
```

```
AnadirDireccionGrupo_1erDigito = 1
```

```
AnadirDireccionGrupo_2oDigito = 0
```

```
AnadirDireccionGrupo_3erDigito = 2
```

```
AnadirDireccionGrupo_NombreDireccion = "Iluminación todas"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
```

```
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
```

```
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
```

```
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
```

```

                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

```

```

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/0 LR01 Dormitorio ON/OFF
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 0
AnadirDireccionGrupo_NombreDireccion = "LR01 Dormitorio ON/OFF"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

,

'Añadimos la 1/1/1 LR01 Dormitorio REGULACIÓN

```



,

```
AnadirDireccionGrupo_1erDigito = 1  
AnadirDireccionGrupo_2oDigito = 1  
AnadirDireccionGrupo_3erDigito = 1  
AnadirDireccionGrupo_NombreDireccion = "LR01 Dormitorio REGULACIÓN"
```

```
'valores iniciales variables auxiliares  
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false  
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false  
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true  
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true  
                Set AnadirDireccionGrupo_vc =  
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)  
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito  
                AnadirDireccionGrupo_auxiliarYaAnadida = true
```

```
            end if
```

```

        next
    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/2 LR01 Dormitorio ESTADO
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 2
AnadirDireccionGrupo_NombreDireccion = "LR01 Dormitorio ESTADO"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

```

```

    For each MiddleGroup in MainGroup.MiddleGroups

        if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

            AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
            Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

        end if

    next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/3 LR01 Dormitorio VALOR
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 3
AnadirDireccionGrupo_NombreDireccion = "LR01 Dormitorio VALOR"

```

```

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

```

```

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```
end if
```

```
,
```

```
'Añadimos la 1/1/4 LR02 Salon ON/OFF'
```

```
,
```

```
AnadirDireccionGrupo_1erDigito = 1
```

```
AnadirDireccionGrupo_2oDigito = 1
```

```
AnadirDireccionGrupo_3erDigito = 4
```

```
AnadirDireccionGrupo_NombreDireccion = "LR02 Salon ON/OFF"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
```

```
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
```

```
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
```

```
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
```



```

                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

                end if

                next

            end if

        Next

        if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

            Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
            AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

            Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

            if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

```

```

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/5 LR02 Salon REGULACIÓN
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 5
AnadirDireccionGrupo_NombreDireccion = "LR02 Salon REGULACIÓN"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

,

'Añadimos la 1/1/6 LR02 Salon ESTADO

```

,

```
AnadirDireccionGrupo_1erDigito = 1  
AnadirDireccionGrupo_2oDigito = 1  
AnadirDireccionGrupo_3erDigito = 6  
AnadirDireccionGrupo_NombreDireccion = "LR02 Salon ESTADO"
```

```
'valores iniciales variables auxiliares  
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false  
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false  
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true  
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true  
                Set AnadirDireccionGrupo_vc =  
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)  
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito  
                AnadirDireccionGrupo_auxiliarYaAnadida = true
```

```
            end if
```

```

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/7 LR02 Salon VALOR
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 7
AnadirDireccionGrupo_NombreDireccion = "LR02 Salon VALOR"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

```

```

    For each MiddleGroup in MainGroup.MiddleGroups

        if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

            AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
            Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

        end if

    next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

```



```

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/8 Bombilla Fluorescente ON/OFF
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 8
AnadirDireccionGrupo_NombreDireccion = "Bombilla Fluorescente ON/OFF "

```

```

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

```

```

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true
    end if

```

```
end if
```

```
,
```

```
'Añadimos la 1/1/9 Bombilla Fluorescente REGULACIÓN'
```

```
,
```

```
AnadirDireccionGrupo_1erDigito = 1
```

```
AnadirDireccionGrupo_2oDigito = 1
```

```
AnadirDireccionGrupo_3erDigito = 9
```

```
AnadirDireccionGrupo_NombreDireccion = "Bombilla Fluorescente REGULACIÓN"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
```

```
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
```

```
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
```

```
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
```

```

                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

```

```

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/10 Bombilla Fluorescente ESTADO
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 10
AnadirDireccionGrupo_NombreDireccion = "Bombilla Fluorescente ESTADO"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

,
'Añadimos la 1/1/11 Bombilla Fluorescente VALOR

```



,

```
AnadirDireccionGrupo_1erDigito = 1  
AnadirDireccionGrupo_2oDigito = 1  
AnadirDireccionGrupo_3erDigito = 11  
AnadirDireccionGrupo_NombreDireccion = "Bombilla Fluorescente VALOR"
```

```
'valores iniciales variables auxiliares  
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false  
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false  
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true  
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true  
                Set AnadirDireccionGrupo_vc =  
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)  
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito  
                AnadirDireccionGrupo_auxiliarYaAnadida = true
```

```
            end if
```

```

        next
    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/12 L03 Aseo ON/OFF
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 12
AnadirDireccionGrupo_NombreDireccion = "L03 Aseo ON/OFF"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

```

```

    For each MiddleGroup in MainGroup.MiddleGroups

        if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

            AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
            Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

        end if

    next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/13 L03 Aseo ESTADO
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 13
AnadirDireccionGrupo_NombreDireccion = "L03 Aseo ESTADO"

```

```

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

```

```

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```
end if
```

```
,
```

```
'Añadimos la 1/1/14 L04 ON/OFF'
```

```
,
```

```
AnadirDireccionGrupo_1erDigito = 1
```

```
AnadirDireccionGrupo_2oDigito = 1
```

```
AnadirDireccionGrupo_3erDigito = 14
```

```
AnadirDireccionGrupo_NombreDireccion = "L04 ON/OFF"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
```

```
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
```

```
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
```

```
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
```



```

                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

                end if

                next

            end if

        Next

        if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

            Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
            AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

            Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

            if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

```

```

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/15 L04 ESTADO
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 15
AnadirDireccionGrupo_NombreDireccion = "L04 ESTADO"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    
```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

,
'Añadimos la 1/1/16 L05 ON/OFF

```

```
'  
AnadirDireccionGrupo_1erDigito = 1  
AnadirDireccionGrupo_2oDigito = 1  
AnadirDireccionGrupo_3erDigito = 16  
AnadirDireccionGrupo_NombreDireccion = "L05 ON/OFF"
```

```
'valores iniciales variables auxiliares  
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false  
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false  
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true  
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true  
                Set AnadirDireccionGrupo_vc =  
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)  
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito  
                AnadirDireccionGrupo_auxiliarYaAnadida = true
```

```
            end if
```

```

        next
    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/17 L05 ESTADO
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 17
AnadirDireccionGrupo_NombreDireccion = "L05 ESTADO"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

```

```

For each MiddleGroup in MainGroup.MiddleGroups

    if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

        AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
        Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

```



```

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/18 L06 ON/OFF
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 18
AnadirDireccionGrupo_NombreDireccion = "L06 ON/OFF"

```

```

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

```

```

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```
end if
```

```
,
```

```
'Añadimos la 1/1/19 L06 ESTADO
```

```
,
```

```
AnadirDireccionGrupo_1erDigito = 1
```

```
AnadirDireccionGrupo_2oDigito = 1
```

```
AnadirDireccionGrupo_3erDigito = 19
```

```
AnadirDireccionGrupo_NombreDireccion = "L06 ESTADO"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
```

```
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
```

```
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
```

```
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
```

```

                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

```

```

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/20 L07 ON/OFF
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 20
AnadirDireccionGrupo_NombreDireccion = "L07 ON/OFF"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    
```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

,
'Añadimos la 1/1/21 L07 ESTADO

```



```
'  
AnadirDireccionGrupo_1erDigito = 1  
AnadirDireccionGrupo_2oDigito = 1  
AnadirDireccionGrupo_3erDigito = 21  
AnadirDireccionGrupo_NombreDireccion = "L07 ESTADO"
```

```
'valores iniciales variables auxiliares  
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false  
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false  
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true  
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true  
                Set AnadirDireccionGrupo_vc =  
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)  
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito  
                AnadirDireccionGrupo_auxiliarYaAnadida = true
```

```
            end if
```

```

        next
    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/22 L08 ON/OFF
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 22
AnadirDireccionGrupo_NombreDireccion = "L08 ON/OFF"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

```

```

For each MiddleGroup in MainGroup.MiddleGroups

    if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

        AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
        Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/23 L08 ESTADO
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 23
AnadirDireccionGrupo_NombreDireccion = "L08 ESTADO"

```

```

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

```

```

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```
end if
```

```
,
```

```
'Añadimos la 1/1/24 L09 ON/OFF'
```

```
,
```

```
AnadirDireccionGrupo_1erDigito = 1
```

```
AnadirDireccionGrupo_2oDigito = 1
```

```
AnadirDireccionGrupo_3erDigito = 24
```

```
AnadirDireccionGrupo_NombreDireccion = "L09 ON/OFF"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
```

```
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
```

```
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
```

```
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
```



```

                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

                end if

                next

            end if

        Next

        if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

            Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
            AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

            Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

            if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

```

```

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/25 L09 ESTADO
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 25
AnadirDireccionGrupo_NombreDireccion = "L09 ESTADO"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

,
'Añadimos la 1/1/26 L10 ON/OFF

```

,

```
AnadirDireccionGrupo_1erDigito = 1  
AnadirDireccionGrupo_2oDigito = 1  
AnadirDireccionGrupo_3erDigito = 26  
AnadirDireccionGrupo_NombreDireccion = "L10 ON/OFF"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false  
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false  
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true  
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true  
                Set AnadirDireccionGrupo_vc =  
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)  
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito  
                AnadirDireccionGrupo_auxiliarYaAnadida = true
```

```
            end if
```

```

        next
    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/1/27 L10 ESTADO
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 1
AnadirDireccionGrupo_3erDigito = 27
AnadirDireccionGrupo_NombreDireccion = "L10 ESTADO"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

```

```

    For each MiddleGroup in MainGroup.MiddleGroups

        if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

            AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
            Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

        end if

    next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

```



```

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/2/0 P01 Persiana1 Dorm MOV
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 2
AnadirDireccionGrupo_3erDigito = 0
AnadirDireccionGrupo_NombreDireccion = "P01 Persiana1 Dorm MOV"

```

```

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

```

```

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```
end if
```

```
,
```

```
'Añadimos la 1/2/1 P01 Persiana1 Dorm STOP
```

```
,
```

```
AnadirDireccionGrupo_1erDigito = 1
```

```
AnadirDireccionGrupo_2oDigito = 2
```

```
AnadirDireccionGrupo_3erDigito = 1
```

```
AnadirDireccionGrupo_NombreDireccion = "P01 Persiana1 Dorm STOP"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
```

```
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
```

```
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
```

```
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
```

```

                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

```

```

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/2/2 P02 Persiana2 Dorm MOV
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 2
AnadirDireccionGrupo_3erDigito = 2
AnadirDireccionGrupo_NombreDireccion = "P02 Persiana2 Dorm MOV"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    ,
    'Añadimos la 1/2/3 P02 Persiana2 Dorm ppal STOP

```



```
'  
AnadirDireccionGrupo_1erDigito = 1  
AnadirDireccionGrupo_2oDigito = 2  
AnadirDireccionGrupo_3erDigito = 3  
AnadirDireccionGrupo_NombreDireccion = "P02 Persiana2 Dorm ppal STOP"
```

```
'valores iniciales variables auxiliares  
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false  
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false  
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true  
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true  
                Set AnadirDireccionGrupo_vc =  
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)  
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito  
                AnadirDireccionGrupo_auxiliarYaAnadida = true
```

```
            end if
```

```

        next
    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then
        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/2/4 P03 Persiana3 Dorm MOV
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 2
AnadirDireccionGrupo_3erDigito = 4
AnadirDireccionGrupo_NombreDireccion = "P03 Persiana3 Dorm MOV"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

```

```

    For each MiddleGroup in MainGroup.MiddleGroups

        if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

            AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
            Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
            AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
            AnadirDireccionGrupo_auxiliarYaAnadida = true

        end if

    next

end if

Next

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/2/5 P03 Persiana3 Dorm STOP
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 2
AnadirDireccionGrupo_3erDigito = 5
AnadirDireccionGrupo_NombreDireccion = "P03 Persiana3 Dorm STOP"

```

```

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

```

```

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
    AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

    Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
    AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
    Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
    AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
    Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
    AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

    AnadirDireccionGrupo_auxiliarYaAnadida = true

```

```
end if
```

```
,
```

```
'Añadimos la 1/2/6 P04 Persiana4 Dorm MOV
```

```
,
```

```
AnadirDireccionGrupo_1erDigito = 1
```

```
AnadirDireccionGrupo_2oDigito = 2
```

```
AnadirDireccionGrupo_3erDigito = 6
```

```
AnadirDireccionGrupo_NombreDireccion = "P04 Persiana4 Dorm MOV"
```

```
'valores iniciales variables auxiliares
```

```
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
```

```
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
```

```
AnadirDireccionGrupo_auxiliarYaAnadida = false
```

```
For each MainGroup In Project.MainGroups
```

```
    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then
```

```
        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
```

```
        Set AnadirDireccionGrupo_va = MainGroup
```

```
        For each MiddleGroup in MainGroup.MiddleGroups
```

```
            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then
```

```
                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
```



```

                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

```

```

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

end if

,
'Añadimos la 1/2/7 P04 Persiana4 Dorm STOP
,

AnadirDireccionGrupo_1erDigito = 1
AnadirDireccionGrupo_2oDigito = 2
AnadirDireccionGrupo_3erDigito = 7
AnadirDireccionGrupo_NombreDireccion = "P04 Persiana4 Dorm STOP"

'valores iniciales variables auxiliares
AnadirDireccionGrupo_auxiliarEncontradoMainGroup = false
AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = false
AnadirDireccionGrupo_auxiliarYaAnadida = false

```

```

For each MainGroup In Project.MainGroups

    if (MainGroup.Address = AnadirDireccionGrupo_1erDigito) then

        AnadirDireccionGrupo_auxiliarEncontradoMainGroup = true
        Set AnadirDireccionGrupo_va = MainGroup

        For each MiddleGroup in MainGroup.MiddleGroups

            if MiddleGroup.Address = AnadirDireccionGrupo_2oDigito then

                AnadirDireccionGrupo_auxiliarEncontradoMiddleGroup = true
                Set AnadirDireccionGrupo_vc =
MiddleGroup.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
                AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
                AnadirDireccionGrupo_auxiliarYaAnadida = true

            end if

        next

    end if

Next

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=true AND AnadirDireccionGrupo_auxiliarYaAnadida = false) then

        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)

```

```

        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito

        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito
        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

    if (AnadirDireccionGrupo_auxiliarEncontradoMainGroup=false AND AnadirDireccionGrupo_auxiliarYaAnadida = false)
then

        Set AnadirDireccionGrupo_va =
Project.MainGroups.Add(AnadirDireccionGrupo_1erDigito,"MainGroup"+CStr(AnadirDireccionGrupo_1erDigito))
        AnadirDireccionGrupo_va.Address = AnadirDireccionGrupo_1erDigito
        Set AnadirDireccionGrupo_vb =
AnadirDireccionGrupo_va.MiddleGroups.Add(AnadirDireccionGrupo_2oDigito,"MiddleGroup"+CStr(AnadirDireccionGrupo_2oDigito)
)
        AnadirDireccionGrupo_vb.Address=AnadirDireccionGrupo_2oDigito
        Set AnadirDireccionGrupo_vc =
AnadirDireccionGrupo_vb.GroupAddresses.Add(AnadirDireccionGrupo_3erDigito,AnadirDireccionGrupo_NombreDireccion)
        AnadirDireccionGrupo_vc.Address=AnadirDireccionGrupo_3erDigito

        AnadirDireccionGrupo_auxiliarYaAnadida = true

    end if

```

```
.....  
.....  
'                               Se configuran los dispositivos (parámetros)  
.....  
.....
```

```
'  
'Disp: Actuador 6 salidas  
'Descrip:  
'Rückmeldung [3] = 1  
'
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)  
ModificarParametro_nombreDispositivo = 3  
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
```

```
'Poner aquí el nombre EXACTO del parámetro  
ModificarParametro_nombreParametro = "Rückmeldung"  
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)
```

```
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el  
valor-,  
'en vbs no hay disitinción en tipos de variables en su definición)  
ModificarParametro_valorParametro = 1
```

```
'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)  
ModificarParametro_ordenAparicion = 3
```

```

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

```

```

then
    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Actuador 6 salidas
'Descrip:
'Rückmeldung [6] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 3
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 6

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```



```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Actuador 6 salidas

'Descrip:

'Rückmeldung [1] = 1

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 3

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Rückmeldung"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay distinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 1

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```

```

                end if
            end if

        end if

    Next

end if

end if

Next

'Disp: Actuador 6 salidas
'Descrip:
'Rückmeldung [4] = 1
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 3
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)

```

```

ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 4

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

```

```

then
    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

'Disp: Actuador 6 salidas
'Descrip:
'Rückmeldung [2] = 1

```

```

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 3
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 2

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

```

```

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    Next

end if

```



```

        end if

Next

'
'Disp: Actuador 6 salidas
'Descrip:
'Rückmeldung [5] = 1
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 3
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 5

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

```

```

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

```

```

                                end if
                            end if
                        end if
                    end if
                end if
            Next
        end if
    end if

Next

    '
    'Disp: Actuador 6 salidas
    'Descrip:
    'Rückmeldung [3] = 1
    '

    'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
    ModificarParametro_nombreDispositivo = 4
    'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

    'Poner aquí el nombre EXACTO del parámetro
    ModificarParametro_nombreParametro = "Rückmeldung"

```

```

ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 3

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

```

```

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then
            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1
                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true
                end if
            end if
        end if
    end if
Next
end if
end if
Next

```

```

'Disp: Actuador 6 salidas
'Descrip:
'Rückmeldung [6] = 1
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 4
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1
'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 6
'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0
For each Device In Project.Devices
    if ModificarParametro_auxiliarYaModificado = false then
        ModificarParametro_auxiliarNombreDispositivo = Device.Address

```

```

if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then
    for each Parameter in Device.DeviceParameters
        if (ModificarParametro_auxiliarYaModificado = false) then
            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then
                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if
            end if
        end if
    end if
end if

```

```

                end if
            Next
        end if
    end if

Next

,
'Disp: Actuador 6 salidas
'Descrip:
'Rückmeldung [1] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 4
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

```



```

ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

```

```

then
    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Actuador 6 salidas
'Descrip:
'Rückmeldung [4] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 4
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 4

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```

```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Actuador 6 salidas

'Descrip:

'Rückmeldung [2] = 1

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 4

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Rückmeldung"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay distinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 2

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```

```

                end if
            end if

        end if

    Next

end if

end if

Next

'Disp: Actuador 6 salidas
'Descrip:
'Rückmeldung [5] = 1
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 4
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Rückmeldung"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)

```

```

ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 5

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

```



```

then
    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

'
'Disp: Teclado de 1 fase
'Descrip:
'Grund-Funktion der Wippe links [1] = 1

```

```

'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 6
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Grund-Funktion der Wippe links"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

```

```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

Next

```

```

        end if

    end if

Next

'
'Disp: Teclado de 1 fase
'Descrip:
'LED Schalten [1] = 0
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 6
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "LED Schalten"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares

```

```

ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

```

```

Parameter.value = ModificarParametro_valorParametro
ModificarParametro_auxiliarYaModificado = true

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke_01 [6] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 9
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 6

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```

```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```



Next

,

'Disp: Entrada binara, 4-canales

'Descrip:

'Befehl\_fallendeFlanke\_01 [14] = 0

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 9

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Befehl\_fallendeFlanke\_01"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay disitinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 14

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```

```

                end if
            end if

        end if

    Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke_01 [5] = 3
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 9
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)

```

```

ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 5

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

```

```

then
    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_fallendeFlanke_01 [9] = 0

```

```

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 9
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 9

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

```

```

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    Next

end if

```

```

        end if

Next

'
'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_fallendeFlanke_01 [20] = 0
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 9
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 20

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

```



```

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

```



```

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 19

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```



```

'Disp: Entrada binara, 4-canales
'Descrip:
'funktion_k1 [1] = 3
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 10
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "funktion_k1"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

```

```

if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then
    for each Parameter in Device.DeviceParameters
        if (ModificarParametro_auxiliarYaModificado = false) then
            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then
                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if
            end if
        end if
    end if
end if

```

```

                end if

            Next

        end if

    end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'Lam_Verst_Faktor [6] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 10
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Lam_Verst_Faktor"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

```

```

ModificarParametro_ordenAparicion = 6

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

```



```

then
    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'funktion_k2 [1] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 10
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "funktion_k2"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```

```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Entrada binaria, 4-canales

'Descrip:

'Befehl\_steigendeFlanke [5] = 1

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 10

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Befehl\_steigendeFlanke"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay distinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 5

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```

```

                end if
            end if

        end if

    Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Lam_Verst_Faktor [5] = 3
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 10
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
'
'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Lam_Verst_Faktor"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)
'
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)

```

```

ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 5

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

```

```

then
    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

'
'Disp: Entrada binara, 4-canales
'Descrip:
'funktion_k3 [1] = 3
'

```



```

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 10
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "funktion_k3"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

```

```

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    Next

end if

```

```
end if
```

```
Next
```

```
,
```

```
'Disp: Entrada binara, 4-canales
```

```
'Descrip:
```

```
'Lam_Verst_Faktor [7] = 3
```

```
,
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
```

```
ModificarParametro_nombreDispositivo = 10
```

```
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
```

```
'Poner aquí el nombre EXACTO del parámetro
```

```
ModificarParametro_nombreParametro = "Lam_Verst_Faktor"
```

```
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)
```

```
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,
```

```
'en vbs no hay disitinción en tipos de variables en su definición)
```

```
ModificarParametro_valorParametro = 3
```

```
'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
```

```
ModificarParametro_ordenAparicion = 7
```

```
'valores iniciales variables auxiliares
```

```
ModificarParametro_auxiliarYaModificado = false
```

```
ModificarParametro_auxiliarInc = 0
```

```

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

```

```

                                end if
                            end if
                        end if
                    end if
                end if
            Next
        end if

        end if

        end if

    end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'funktion_k4 [1] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 10
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "funktion_k4"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

```

```

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```

```

                                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

                                end if

                                end if

                                end if

                                end if

                                end if

                                Next

                                end if

                                end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:

```

```

'Befehl_steigendeFlanke [8] = 1
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 10
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 8

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

```



```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

```

```

                Next
            end if
        end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'Lam_Verst_Faktor [8] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 10
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Lam_Verst_Faktor"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 8

```

```

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

```

```

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true
    end if
end if
end if
end if
Next
end if
end if
Next
,
'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke_01 [6] = 3
,
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 11
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 6

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```

```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Entrada binara, 4-canales

'Descrip:

'Befehl\_fallendeFlanke\_01 [14] = 0

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 11

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Befehl\_fallendeFlanke\_01"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay disitinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 14

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```



```

                end if
            end if

        end if

    Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke_01 [5] = 3
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 11
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)

```

```

ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 5

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

```

```

then
    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_fallendeFlanke_01 [9] = 0

```

```

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 11
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 9

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

```

```

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    Next

end if

```

```

        end if

Next

'
'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke_01 [8] = 3
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 11
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 8

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

```

```

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

```





```

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 20

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```

```

                                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

                                end if

                                end if

                                end if

                                end if

                                Next

                                end if

                                end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:

```

```

'Befehl_steigendeFlanke_01 [7] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 11
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 7

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

```

```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

```

```

                Next
            end if
        end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_fallendeFlanke_01 [19] = 0
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 11
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 19

```

```

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

```

```

Parameter.value = ModificarParametro_valorParametro
ModificarParametro_auxiliarYaModificado = true

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke_01 [6] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 12
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 6

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```



```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Entrada binara, 4-canales

'Descrip:

'Befehl\_fallendeFlanke\_01 [14] = 0

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 12

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Befehl\_fallendeFlanke\_01"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay disitinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 14

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```

```

                end if
            end if

        end if

    Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke_01 [5] = 3
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 12
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)

```

```

ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 5

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

```

```

then
    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_fallendeFlanke_01 [9] = 0

```

```

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 12
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 9

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

```

```

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    Next

end if

```



```
end if
```

```
Next
```

```
,
```

```
'Disp: Entrada binara, 4-canales
```

```
'Descrip:
```

```
'Befehl_steigendeFlanke_01 [8] = 3
```

```
,
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
```

```
ModificarParametro_nombreDispositivo = 12
```

```
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
```

```
'Poner aquí el nombre EXACTO del parámetro
```

```
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
```

```
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)
```

```
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,
```

```
'en vbs no hay disitinción en tipos de variables en su definición)
```

```
ModificarParametro_valorParametro = 3
```

```
'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
```

```
ModificarParametro_ordenAparicion = 8
```

```
'valores iniciales variables auxiliares
```

```
ModificarParametro_auxiliarYaModificado = false
```

```
ModificarParametro_auxiliarInc = 0
```

```

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

```

```

                                end if
                            end if
                        end if
                    end if
                end if
            Next
        end if
    end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_fallendeFlanke_01 [20] = 0
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 12
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

```

```

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 20

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```

```

                                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

                                end if

                                end if

                                end if

                                end if

                                Next

                                end if

                                end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:

```

```

'Befehl_steigendeFlanke_01 [7] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 12
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 7

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

```

```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

```

```

                Next
            end if
        end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_fallendeFlanke_01 [19] = 0
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 12
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_fallendeFlanke_01"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 19

```



```

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

```

```

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

    end if

end if

Next

end if

end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'funktion_k1 [1] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 13
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "funktion_k1"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```

```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Entrada binara, 4-canales

'Descrip:

'Lam\_Verst\_Faktor [6] = 3

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 13

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Lam\_Verst\_Faktor"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay disitinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 6

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```

```

                end if
            end if

        end if

    Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'funktion_k2 [1] = 3
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 13
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "funktion_k2"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)

```

```

ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

```



```

then
    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke [5] = 1

```

```

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 13
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Befehl_steigendeFlanke"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 5

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

```

```

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    Next

end if

```

```
end if
```

```
Next
```

```
,
```

```
'Disp: Entrada binara, 4-canales
```

```
'Descrip:
```

```
'Lam_Verst_Faktor [5] = 3
```

```
,
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
```

```
ModificarParametro_nombreDispositivo = 13
```

```
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
```

```
'Poner aquí el nombre EXACTO del parámetro
```

```
ModificarParametro_nombreParametro = "Lam_Verst_Faktor"
```

```
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)
```

```
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,
```

```
'en vbs no hay disitinción en tipos de variables en su definición)
```

```
ModificarParametro_valorParametro = 3
```

```
'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
```

```
ModificarParametro_ordenAparicion = 5
```

```
'valores iniciales variables auxiliares
```

```
ModificarParametro_auxiliarYaModificado = false
```

```
ModificarParametro_auxiliarInc = 0
```

```

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

```

```

                                end if
                            end if
                        end if
                    end if
                end if
            Next
        end if
    end if

Next

    '
    'Disp: Entrada binara, 4-canales
    'Descrip:
    'funktion_k3 [1] = 3
    '
    'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
    ModificarParametro_nombreDispositivo = 13
    'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

    'Poner aquí el nombre EXACTO del parámetro
    ModificarParametro_nombreParametro = "funktion_k3"
    ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

```

```

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```





```

'Lam_Verst_Faktor [7] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 13
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Lam_Verst_Faktor"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 7

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

```

```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

```

```

                Next
            end if
        end if

Next

,
'Disp: Entrada binara, 4-canales
'Descrip:
'funktion_k4 [1] = 3
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 13
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "funktion_k4"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

```

```

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

```

```

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

        end if

        end if

    end if

Next

end if

end if

Next

'Disp: Entrada binara, 4-canales
'Descrip:
'Befehl_steigendeFlanke [8] = 1
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 13
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

```

```

ModificarParametro_nombreParametro = "Befehl_steigendeFlanke"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 8

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

```

```

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then
            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1
                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true
                end if
            end if
        end if
    end if
Next
end if
end if
Next

```

```

'
'Disp: Entrada binara, 4-canales
'Descrip:
'Lam_Verst_Faktor [8] = 3
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 13
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Lam_Verst_Faktor"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 8

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

```



```

ModificarParametro_auxiliarNombreDispositivo = Device.Address

if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

    for each Parameter in Device.DeviceParameters

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    end if

```

```

                end if
            Next
        end if
    end if
Next

,
'Disp: Actuador de persianas
'Descrip:
'Zeit für aut. Zurücksetzen [1] = 20
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 2
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Zeit für aut. Zurücksetzen"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

```

```

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 20

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```

```

                                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

                                end if

                                end if

                                end if

                                end if

                                Next

                                end if

                                end if

Next

,
'Disp: Actuador de persianas
'Descrip:

```

```

'Betriebsart [2] = 1
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 2
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Betriebsart"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 2

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

```

```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

```

```

                Next
            end if
        end if

Next

,
'Disp: Actuador de persianas
'Descrip:
'Gesamtverfahrzeit [3] = 6
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 2
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Gesamtverfahrzeit"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 6

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 3

```

```

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

```



```

Parameter.value = ModificarParametro_valorParametro
ModificarParametro_auxiliarYaModificado = true

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Actuador de persianas
'Descrip:
'Betriebsart [4] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 2
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

```

```

ModificarParametro_nombreParametro = "Betriebsart"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 4

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

```

```

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then
            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1
                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true
                end if
            end if
        end if
    end if
Next
end if
end if
Next

```

```

'
'Disp: Actuador de persianas
'Descrip:
'Gesamtverfahrzeit [2] = 6
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 2
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Gesamtverfahrzeit"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 6

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 2

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

```

```

ModificarParametro_auxiliarNombreDispositivo = Device.Address

if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

    for each Parameter in Device.DeviceParameters

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    end if

```

```

                end if
            Next
        end if
    end if

Next

,
'Disp: Actuador de persianas
'Descrip:
'Betriebsart [1] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 2
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Betriebsart"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

```

```

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

```

```

then
    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

    end if
end if

    end if

Next

    end if

end if

Next

,
'Disp: Actuador de persianas
'Descrip:
'Gesamtverfahrzeit [5] = 6
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 2

```



```

'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Gesamtverfahrzeit"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 6

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 5

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

```

```

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Actuador de persianas

'Descrip:

'Betriebsart [3] = 1

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 2

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Betriebsart"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay distinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 3

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

                    end if

                end if

            end if

        end for

    end if

end if

```

```

                end if
            end if

        end if

    Next

end if

end if

Next

Next

'
'Disp: Actuador de persianas
'Descrip:
'Gesamtverfahrzeit [7] = 6
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 2
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Gesamtverfahrzeit"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,

```

```

'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 6

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 7

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```



```

'Descrip:
'Grundhelligkeit_Kanal_1    [1] = 3
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Grundhelligkeit_Kanal_1    "
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 3

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

```



```

if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then
    for each Parameter in Device.DeviceParameters
        if (ModificarParametro_auxiliarYaModificado = false) then
            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then
                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1
                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true
                    end if
                end if
            end if
        end if
    end if
end if

```

```

                Next
            end if
        end if

Next
,
'Disp: Regulador fluorescencia
'Descrip:
'Wert_Busausfall [1] = 2
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Wert_Busausfall"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 2

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

```

```

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

```

```

then
    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Regulador fluorescencia
'Descrip:
'Einschaltheelligkeit_Kanal_1 [1] = 6
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Einschaltheiligkeit_Kanal_1"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 6

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```

```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Regulador fluorescencia

'Descrip:

'Faktor\_Dimmschritt\_Kanal\_1 [1] = 50

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 1

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Faktor\_Dimmschritt\_Kanal\_1"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay distinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 50

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 1

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```



```

                end if
            end if

        end if

    Next

end if

end if

Next

'Disp: Regulador fluorescencia
'Descrip:
'Ausg_1_Zeit [1] = 1
'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Ausg_1_Zeit"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)

```

```

ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

```

```

then
    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

'
'Disp: Regulador fluorescencia
'Descrip:
'Ausg_1_Rück [1] = 1
'

```

```

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Ausc_1_Rück"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

```

```

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    Next

end if

```

```
end if
```

```
Next
```

```
,
```

```
'Disp: Regulador fluorescencia
```

```
'Descrip:
```

```
'Grundhelligkeit_Kanal_2 [1] = 3
```

```
,
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
```

```
ModificarParametro_nombreDispositivo = 1
```

```
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
```

```
'Poner aquí el nombre EXACTO del parámetro
```

```
ModificarParametro_nombreParametro = "Grundhelligkeit_Kanal_2 "
```

```
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)
```

```
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,
```

```
'en vbs no hay disitinción en tipos de variables en su definición)
```

```
ModificarParametro_valorParametro = 3
```

```
'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
```

```
ModificarParametro_ordenAparicion = 1
```

```
'valores iniciales variables auxiliares
```

```
ModificarParametro_auxiliarYaModificado = false
```

```
ModificarParametro_auxiliarInc = 0
```

```

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                                Parameter.value = ModificarParametro_valorParametro
                                ModificarParametro_auxiliarYaModificado = true

```

```

end if
end if
end if
Next
end if
end if

Next

'Disp: Regulador fluorescencia
'Descrip:
'Einschaltheelligkeit_Kanal_2 [1] = 6
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Einschaltheelligkeit_Kanal_2"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

```



```

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 6

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```

```

        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                Parameter.value = ModificarParametro_valorParametro
                ModificarParametro_auxiliarYaModificado = true

        end if

        end if

        end if

        Next

        end if

        end if

Next

,
'Disp: Regulador fluorescencia
'Descrip:

```

```

'Faktor_Dimmschritt_Kanal_2 [1] = 50
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Faktor_Dimmschritt_Kanal_2"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 50

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

```

```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

```

```

                Next
            end if
        end if

Next

,
'Disp: Regulador fluorescencia
'Descrip:
'Ausg_2_Zeit [1] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Ausg_2_Zeit"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

```

```

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

```

```

Parameter.value = ModificarParametro_valorParametro
ModificarParametro_auxiliarYaModificado = true

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Regulador fluorescencia
'Descrip:
'Ausg_2_Rück [1] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

```

```

ModificarParametro_nombreParametro = "Ausc_2_Rück"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

```



```

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then
            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1
                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true
                end if
            end if
        end if
    end if
Next
end if
end if
Next

```

```

,
'Disp: Regulador fluorescencia
'Descrip:
'Grundhelligkeit_Kanal_3 [1] = 0
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Grundhelligkeit_Kanal_3 "
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 0

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

```

```

ModificarParametro_auxiliarNombreDispositivo = Device.Address

if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

    for each Parameter in Device.DeviceParameters

        if (ModificarParametro_auxiliarYaModificado = false) then

            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if

            end if

        end if

    end if

```

```

                end if
            Next
        end if
    end if

Next

'
'Disp: Regulador fluorescencia
'Descrip:
'Wert_Busausfall [3] = 2
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Wert_Busausfall"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 2

```

```

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 3

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

```

```

then
    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

    end if

end if

Next

end if

end if

Next

'
'Disp: Regulador fluorescencia
'Descrip:
'Einschaltheelligkeit_Kanal_3 [1] = 6
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1

```

```

'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Einschaltheiligkeit_Kanal_3"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay distinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 6

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_nombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

```

```

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```



Next

,

'Disp: Regulador fluorescencia

'Descrip:

'Faktor\_Dimmschritt\_Kanal\_3 [1] = 50

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 1

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "Faktor\_Dimmschritt\_Kanal\_3"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay distinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 50

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 1

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

                    end if

                end if

            end if

        end for

    end if

end if

```

```

                end if
            end if

        end if

    Next

end if

end if

```

```
Next
```

```
,
```

```
'Disp: Regulador fluorescencia
```

```
'Descrip:
```

```
'Ausc_3_Zeit [1] = 1
```

```
,
```

```
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
```

```
ModificarParametro_nombreDispositivo = 1
```

```
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)
```

```
'Poner aquí el nombre EXACTO del parámetro
```

```
ModificarParametro_nombreParametro = "Ausc_3_Zeit"
```

```
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)
```

```
'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,
```

```

'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

```



```

'
'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "Ausc_3_Rück"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

```

```

for each Parameter in Device.DeviceParameters

    if (ModificarParametro_auxiliarYaModificado = false) then

        ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

end if

Next

```

```

        end if

    end if

Next

,
'Disp: Regulador fluorescencia
'Descrip:
'SoftEIN_JA_NEIN [3] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "SoftEIN_JA_NEIN"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 3

'valores iniciales variables auxiliares

```



```

ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                            if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

```

```

Parameter.value = ModificarParametro_valorParametro
ModificarParametro_auxiliarYaModificado = true

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Regulador fluorescencia
'Descrip:
'SoftAUS_JA_NEIN [3] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "SoftAUS_JA_NEIN"

```

```

ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 3

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

```

```

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then
            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1
                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true
                end if
            end if
        end if
    end if
end if
Next
end if
end if
Next

```

```

'Disp: Regulador fluorescencia
'Descrip:
'SoftEIN_JA_NEIN [1] = 1
'

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "SoftEIN_JA_NEIN"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

```

```

if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then
    for each Parameter in Device.DeviceParameters
        if (ModificarParametro_auxiliarYaModificado = false) then
            ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
            ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

            if (ModificarParametro_auxiliarEncontradoParametro > 0) then
                if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then
                    ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)
then
                        Parameter.value = ModificarParametro_valorParametro
                        ModificarParametro_auxiliarYaModificado = true

                    end if

                end if
            end if
        end if
    end if
end if

```

```

                end if

            Next

        end if

    end if

Next

,
'Disp: Regulador fluorescencia
'Descrip:
'SoftAUS_JA_NEIN [1] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "SoftAUS_JA_NEIN"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

```

```

ModificarParametro_ordenAparicion = 1

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                    ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                    if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                        if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                            ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

```



```

then
    if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

        Parameter.value = ModificarParametro_valorParametro
        ModificarParametro_auxiliarYaModificado = true

    end if

end if

end if

end if

Next

end if

end if

Next

,
'Disp: Regulador fluorescencia
'Descrip:
'SoftEIN_JA_NEIN [2] = 1
,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)
ModificarParametro_nombreDispositivo = 1
'ModificarParametro_nombreDispositivo = lcase (ModificarParametro_nombreDispositivo)

```

```

'Poner aquí el nombre EXACTO del parámetro
ModificarParametro_nombreParametro = "SoftEIN_JA_NEIN"
ModificarParametro_nombreParametro = lcase (ModificarParametro_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el
valor-,
'en vbs no hay disitinción en tipos de variables en su definición)
ModificarParametro_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)
ModificarParametro_ordenAparicion = 2

'valores iniciales variables auxiliares
ModificarParametro_auxiliarYaModificado = false
ModificarParametro_auxiliarInc = 0

For each Device In Project.Devices

    if ModificarParametro_auxiliarYaModificado = false then

        ModificarParametro_auxiliarNombreDispositivo = Device.Address

        if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

            for each Parameter in Device.DeviceParameters

                if (ModificarParametro_auxiliarYaModificado = false) then

                    ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))

```

```

        ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro,ModificarParametro_NombreParametro)

        if (ModificarParametro_auxiliarEncontradoParametro > 0) then

            if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                    Parameter.value = ModificarParametro_valorParametro
                    ModificarParametro_auxiliarYaModificado = true

                end if

            end if

        end if

    end if

Next

end if

end if

```

Next

,

'Disp: Regulador fluorescencia

'Descrip:

'SoftAUS\_JA\_NEIN [2] = 1

,

'Poner aquí el nombre del dispositivo (puede ser exacto o solo una parte de él)

ModificarParametro\_nombreDispositivo = 1

'ModificarParametro\_nombreDispositivo = lcase (ModificarParametro\_nombreDispositivo)

'Poner aquí el nombre EXACTO del parámetro

ModificarParametro\_nombreParametro = "SoftAUS\_JA\_NEIN"

ModificarParametro\_nombreParametro = lcase (ModificarParametro\_nombreParametro)

'Poner aquí el valor del parámetro (da igual si es un número o un string -no hay que tocar nada, solo poner el valor-,

'en vbs no hay distinción en tipos de variables en su definición)

ModificarParametro\_valorParametro = 1

'Poner aquí el orden de aparición de la variable que queremos modificar (importante!!)

ModificarParametro\_ordenAparicion = 2

'valores iniciales variables auxiliares

ModificarParametro\_auxiliarYaModificado = false

ModificarParametro\_auxiliarInc = 0

For each Device In Project.Devices

```

if ModificarParametro_auxiliarYaModificado = false then

    ModificarParametro_auxiliarNombreDispositivo = Device.Address

    if (ModificarParametro_auxiliarNombreDispositivo = ModificarParametro_NombreDispositivo) then

        for each Parameter in Device.DeviceParameters

            if (ModificarParametro_auxiliarYaModificado = false) then

                ModificarParametro_auxiliarNombreParametro = LCase(Cstr(Parameter.Name))
                ModificarParametro_auxiliarEncontradoParametro =
InStrRev(ModificarParametro_auxiliarNombreParametro, ModificarParametro_NombreParametro)

                if (ModificarParametro_auxiliarEncontradoParametro > 0) then

                    if
(Len(ModificarParametro_NombreParametro)=Len(ModificarParametro_auxiliarNombreParametro)) then

                        ModificarParametro_auxiliarInc = ModificarParametro_auxiliarInc +1

                        if (ModificarParametro_auxiliarInc = ModificarParametro_ordenAparicion)

then

                            Parameter.value = ModificarParametro_valorParametro
                            ModificarParametro_auxiliarYaModificado = true

                        end if

```

```
                end if
            end if
        end if
    Next
end if
end if
Next
end sub
```

## ***Anexo IV. Maleta EIB/KNX***

---





# Vivienda EIB/KNX



