Generación Automática de Aplicaciones Mixtas Sw/Hw mediante la Integración de Componentes COTS

Cristina Vicente, Ana Toledo, Carlos Fernández, Pedro Sánchez

***Resumen*-- Los grandes avances realizados en el mundo de la electrónica han dado lugar a la proliferación de sistemas mixtos que combinan la flexibilidad de las rutinas Software (Sw) con la velocidad del procesamiento Hardware (Hw). En este artículo se presenta un nuevo enfoque para el desarrollo de este tipo de sistemas mixtos Sw/Hw, que cubre todas las fases de su ciclo de vida. La propuesta que aquí se recoge integra dos de los paradigmas de desarrollo de software más en boga actualmente: el desarrollo de software basado en componentes y la generación automática de software (meta-programación). Si bien este trabajo se centra en el desarrollo de aplicaciones mixtas Sw/Hw en el dominio de los Sistemas de Procesamiento de Información Visual (VIPS), los resultados obtenidos son fácilmente trasladables a otros dominios como los de las aplicaciones de control o de telecomunicación. Como parte de este trabajo se ha desarrollado la herramienta visual IP-CoDER que permite la construcción incremental de VIPS desde el diseño de prototipos funcionales y su particionado en módulos Sw/Hw, hasta la generación automática del código final de la aplicación.**

***Palabras clave*-- Generación Automática de Software, Desarrollo de Software Basado en Componentes, Componentes COTS: Commercial Off-The-Shelf, Sistemas Mixtos Sw/Hw, Sistemas de Procesamiento de Información Visual (VIPS)**.

Este trabajo ha sido parcialmente financiado por el Proyecto Europeo *EFTCOR* (DPI2002-11583-E), y por los proyectos CICYT *COSIVA* (TIC 2000-1765-C03-02) y *ANCLA* (TIC 2003-07804-C05-02).

Los autores son miembros del Grupo de Investigación DSIE de la Universidad Politécnica de Cartagena, Campus Muralla del Mar, Edificio Antigüones, Cartagena, España. Teléfono: +34968326448. Email: Cristina.Vicente@upct.es.

I. INTRODUCCIÓN

Actualmente, son muchos los dispositivos Hw que requieren de una lógica cada vez más compleja y sofisticada. Así, hoy en día encontramos en el mercado desde teléfonos móviles capaces de transmitir vídeo en tiempo real, hasta frigoríficos "inteligentes" capaces de realizar la compra de productos alimentarios a través de Internet. La programación de estos dispositivos suele realizarse utilizando lenguajes propietarios de muy bajo nivel, optimizados para obtener el máximo rendimiento de la plataforma Hw específica empleada en cada producto (microprocesadores, memorias, buses, Digital Signal Processors DSP, Field Programmable Gate Array FPGA, etc).

Los grandes avances realizados en el campo de la electrónica nos permiten contar en la actualidad con Hw cada vez más flexible, potente, pequeño y económico. Sin embargo, algunos sistemas requieren del uso de algoritmos complejos que deben procesar ingentes cantidades de información, a veces bajo estrictos requisitos de tiempo real. En este caso, la implementación exclusivamente Hw de este tipo de sistemas puede resultar desaconsejable, cuando no inviable, por varios motivos. En primer lugar, si bien en términos generales el procesamiento Hw es mucho más rápido que el equivalente Sw, no es sencillo codificar, menos aún de manera optimizada, ciertos algoritmos complejos haciendo uso de lenguajes de tan bajo nivel como los requeridos por este tipo de dispositivos. Es más, codificar y optimizar estos algoritmos puede requerir tanto tiempo que, al ritmo que evoluciona la electrónica, muy probablemente aparecerán en el mercado dispositivos mucho más potentes sobre los que dichas optimizaciones tendrían un impacto prácticamente despreciable. Además, a pesar del abaratamiento de estos dispositivos, la complejidad de ciertos sistemas requiere del uso

de ciertas tecnologías demasiado costosas aún como para resultar viables.

Por todo ello, en ocasiones resulta imprescindible abordar la construcción de este tipo de sistemas a partir de un diseño mixto Sw/Hw. De este modo, parte del procesamiento, generalmente el más complejo, se codificará haciendo uso de lenguajes de programación de alto nivel y se ejecutará sobre un procesador convencional (microprocesador). Simultáneamente, ciertos cálculos sencillos y que deban realizarse de manera muy veloz, se ejecutarán en paralelo sobre un dispositivo Hw de propósito específico (por ejemplo, un sensor inteligente) o más general (por ejemplo, una FPGA).

En particular, entre los sistemas de este tipo, nuestro grupo de investigación cuenta con una amplia experiencia en el desarrollo de Sistemas de Procesamiento de Información Visual (VIPS) [1-2]. En este artículo se analizará cómo el uso de técnicas de generación automática de código [3] y de desarrollo de Sw basado en componentes [4], en particular de tipo COTS [5], puede ayudar a diseñar e implementar este tipo de sistemas de manera rápida, sencilla e intuitiva. A pesar de que este trabajo se centra en el dominio específico de los VIPS, la metodología que aquí se presenta es de aplicación para una amplia variedad de sistemas mixtos Sw/Hw, como los sistemas de control en tiempo real, los sistemas de telecomunicación, o las redes de sensores, entre otros.

El resto del artículo se estructura como sigue. En el siguiente apartado se describe la metodología tradicionalmente empleada para el desarrollo de los VIPS y los inconvenientes que de ella se derivan. En el apartado 3 se propone un nuevo enfoque metodológico que permite realizar diseños menos dependientes del Hw y por lo tanto más flexibles y fáciles de reutilizar. Para dar soporte a esta nueva metodología se ha creado la herramienta visual IP-CoDER, que se presenta en el apartado 4. Esta herramienta permite diseñar e implementar nuevos VIPS de manera semiautomática, haciendo uso de una librería de componentes Sw y Hw para procesamiento de imágenes, construida a partir de diversos componentes COTS. Finalmente, en el apartado 5 se recogen algunas conclusiones y líneas de trabajo futuras.

II. SISTEMAS DE PROCESAMIENTO DE INFORMACIÓN VISUAL (VIPS)

En la actualidad, el abanico de aplicaciones que hacen uso de información de tipo visual es amplísimo: sistemas de inspección visual automatizada de productos industriales, aplicaciones de videoconferencia, sistemas de seguridad biométricos basados en la identificación del rostro o de las huellas digitales, sistemas de diagnóstico médico, aplicaciones de compresión de vídeo, etc.

El procesamiento de imágenes es una tarea computacionalmente muy costosa dado el ingente volumen de información contenido en cada imagen y la complejidad de algunos de los algoritmos necesarios para extraer la información relevante en cada caso. Por ello, la implementación de VIPS suele requerir del uso tanto de lenguajes de programación eficientes como de procesadores potentes. En la actualidad, el mercado ofrece diversos productos, tanto Sw como Hw, útiles para el desarrollo de VIPS:

- Librerías de procesamiento de imágenes [6-8] optimizadas para distintas plataformas:
  - Plataformas Sw de propósito general como los procesadores de Intel©, o de propósito específico como algunos DSP.
  - Plataformas Hw de propósito general como las FPGA, o costosas plataformas de propósito específico basadas en DSPs (p.e. los sistemas de Matrox© [7]).
- Herramientas de programación, ya sean generales (por ejemplo, cualquiera de los entornos de desarrollo C/C++) o específicas para el desarrollo de VIPS [9].
- Herramientas de (co-) simulación [10-11].

Sin embargo, ninguno de estos productos cubre por sí sólo todas las fases del desarrollo de los VIPS. Así, a fecha de hoy, este tipo de sistemas se siguen codificando de manera casi artesanal, siguiendo una metodología muy centrada en el Hw, tal y como se describe a continuación.

A. Construcción tradicional de los vips

Tradicionalmente, los VIPS se construyen en torno a un determinado Hw, a pesar de tratarse de sistemas intensivamente Sw. De hecho, la práctica habitual consiste en seleccionar, en primera instancia, la plataforma Hw sobre la que se implementará el VIPS, atendiendo a ciertos requisitos no funcionales como el coste máximo del sistema final o la velocidad mínima a la que deben procesarse las imágenes (ver Fig. 1), para posteriormente, desarrollar la lógica Sw atendiendo a la funcionalidad exigida al sistema.

La primera consecuencia negativa derivada del hecho de anteponer la selección del Hw a la implementación del Sw, suele ser la adquisición de plataformas sobredimensionadas, tanto en coste como en capacidad de procesamiento, haciendo válido el dicho “más vale que sobre”. Además, la elección y adquisición de una plataforma Hw como primer paso en la construcción de un VIPS, suele imponer el uso de determinados controladores (*drivers*) y librerías de procesamiento de imágenes optimizadas para dicha plataforma (ver Fig. 1). Sin embargo, quizá la limitación más importante resultado de la subordinación del Sw al Hw, sea la dificultad de conseguir diseños flexibles y reutilizables. Como consecuencia, cada nuevo VIPS se construye prácticamente desde cero, aunque su lógica sea casi idéntica a la de otros sistemas desarrollados previamente, pero implementados sobre otras plataformas. Una de las razones que podrían justificar la concepción Hw de los VIPS, es que este tipo de aplicaciones suelen implementarlas ingenieros en automatización y control con experiencia como programadores, pero con escasa formación en Ingeniería del Software. Así, en la mayoría de los casos, las fases de análisis y diseño o bien se obvian, o bien se realizan de forma poco sistemática sin seguir una metodología formal de desarrollo de Sw.

En cuanto a la fase de implementación, ésta suele realizarse en dos etapas. La primera de ellas consiste en desarrollar un prototipo Sw sobre el que testar distintos algoritmos de procesamiento de imágenes, seleccionados generalmente de manera empírica (prueba y error), hasta dar con la combinación que permita cumplir los requisitos funcionales. Este prototipo suele programarse haciendo uso de herramientas de muy alto nivel (p.e. Matlab) que facilitan el desarrollo rápido de aplicaciones pero que no son especialmente eficientes. En la mayoría de los casos, una vez validado este prototipo inicial se desecha, siendo necesario re-implementar cada algoritmo seleccionado, bien en su versión Sw (haciendo

uso de librerías Sw optimizadas), bien en su versión Hw (utilizando un lenguaje de descripción de Hw, p.e. VHDL[1]).

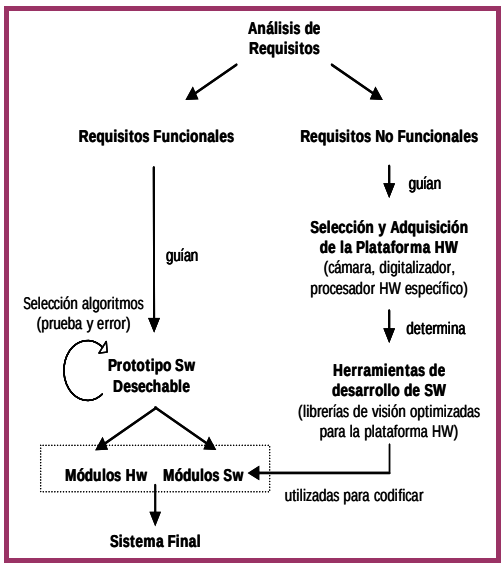

Figura 1. Esquema de las fases de desarrollo de un VIPS siguiendo el enfoque tradicional.

III. UNA NUEVA METODOLOGÍA DE DESARROLLO

Con el fin de solucionar los problemas derivados de la concepción Hw de los VIPS, se propone una nueva metodología que parte de los requisitos funcionales del sistema y permite retrasar la selección y adquisición de un determinado Hw hasta la fase final de implementación. La propuesta metodológica que aquí se presenta persigue el desarrollo de sistemas más flexibles, cuyos diseños sean reutilizables y fácilmente adaptables a distintas plataformas. Esta flexibilidad sólo es posible si se afronta la construcción de estos sistemas mixtos poniendo el énfasis en su componente Sw.

Como se describe a continuación, la construcción de VIPS suele requerir explorar una gran cantidad de configuraciones: distintas combinaciones de algoritmos de procesamiento de imágenes, diferentes particiones Sw/Hw de la funcionalidad y diversas plataformas sobre las que realizar la co-simulación e implementación final del sistema. Así, muy probablemente, durante el desarrollo de estos sistemas deberemos volver atrás en el diseño para reconsiderar alguna de las decisiones adoptadas. Por ello, la metodología que proponemos no consiste en una secuencia lineal de tareas sino en un proceso en espiral que permite refinar el sistema de manera iterativa e incremental hasta llegar a la versión final.

A pesar de que esta propuesta parte de algunos conceptos relativamente abstractos, propios de una metodología de desarrollo de Sw, fundamentalmente se pretende ofrecer una solución pragmática que dé soporte a todo el ciclo de vida de este tipo de productos, cubriendo las lagunas que tradicionalmente aparecen entre el diseño (abstracto) y la implementación (concreta).

A continuación se describe esta nueva metodología indicando, para cada una de las etapas identificadas (ver Fig. 2), qué tipo de herramientas pueden resultar útiles y si éstas existen o no actualmente en el mercado.

A. Prototipado funcional y simulación

La construcción y evaluación de prototipos es una forma rápida y económica de validar los requisitos de un sistema [12]. En concreto, durante esta primera fase se persigue validar los requisitos funciones del VIPS, creando un prototipo ejecutable que permita seleccionar los algoritmos de procesamiento de imágenes más adecuados en cuanto a su fiabilidad y robustez.

El prototipado funcional que se propone, a diferencia del realizado siguiendo el enfoque tradicional, es evolutivo y totalmente independiente de la plataforma sobre la que finalmente se implementen los algoritmos seleccionados.

Así, dado que no hay restricciones relativas al Hw empleado, este primer prototipo podrá implementarse haciendo uso de cualquiera de las librerías de procesamiento de imágenes disponibles en el mercado [6-8], incluso de aquellas optimizadas para una determinada plataforma, sea o no la utilizada finalmente. Dado que la eficiencia no es un factor determinante en este punto, podrá emplearse cualquier lenguaje de programación (C/C++, Matlab, Java, etc), siendo preferible uno de alto nivel que permita construir el prototipo lo más rápidamente posible y modificarlo de manera sencilla, si fuera necesario.

B. Particionado Sw/Hw y Co-simulación

Esta fase consiste en realizar una partición Sw/Hw de los algoritmos seleccionados durante el prototipado funcional, identificando cuáles de ellos se van a implementar como funciones Sw y cuáles como bloques Hw. El resultado de esta partición será un prototipo mixto Sw/Hw (co-prototipo) que se validará haciendo uso de técnicas de co-simulación.

A priori cualquier partición es viable, si bien debe procurarse reducir al máximo el intercambio de información entre elementos Sw y Hw para evitar retardos innecesarios y problemas de sincronización. Por lo general, resulta más útil realizar primero todo el procesamiento Hw (algoritmos sencillos de bajo nivel que acondicionen la imagen justo después de su adquisición), para posteriormente realizar el procesamiento Sw (algoritmos de procesamiento de alto nivel por lo general más complejos); de este modo, sólo será necesario realizar un único intercambio de datos Hw-Sw.

Una vez decidida esta partición es necesario seleccionar una plataforma que proporcione los bloques Hw necesarios, o al menos la funcionalidad para poder construirlos. Nótese que esta elección de una plataforma no implica necesariamente su adquisición ya que, en muchos casos, los fabricantes de estos dispositivos proporcionan modelos Sw lógicamente equivalentes sobre los que es posible simular el comportamiento del Hw de manera virtual. Este es el caso, por ejemplo, de la herramienta System Generator [13] que ofrece modelos Sw Simulink© [10] de las FPGA de Xillinx.

En cuanto al particionado, existen varias herramientas que permiten realizar este proceso de manera (semi-) automática [14]. Sin embargo, éstas suelen requerir especificar el prototipo inicial haciendo uso de la misma herramienta. Obviamente esto contraviene la filosofía de nuestra propuesta con la que se intenta promover la construcción de diseños independientes tanto de las herramientas de desarrollo como del Hw específico.

El mercado también ofrece varias herramientas de co-simulación con las que validar la viabilidad del co-prototipo (correcta sincronización Hw/Sw, tiempos de ejecución, ocu-

[1] VHDL: siglas de VHSIC-HDL (Very high speed integrated circuit Hardware Description Language).