Desenvolvimento de sistemas embarcados com suporte a tempo-real seguindo o Projeto de Sistemas Orientados a Aplicação

Danillo Santos, Roberto Matos, Rafael Cancian, Antônio Augusto Fröhlich

[1]Laboratório de Integração Software Hardware
Universidade Federal de Santa Catarina
UFSC/CTC/LISHA P.O.BOX 476, Florianópolis – SC – Brasil

{danillo,roberto,cancian,guto}@lisha.ufsc.br

Abstract. *Embedded systems design gets more complex as result of low cost provided by the technology advance, that allows the development of more complex embedded applications, like multimedia processing in cell phones. Most common development strategies are not suitable for the design of real-time and extremmely adapted applications. Application-oriented system design (AOSD) strategy has shown efficient to embedded systems software design. The present work shows a way to improve AOSD strategy hardware generation, including real-time requirements support.*

Resumo. *O projeto de sistemas embarcados torna-se complexo devido o baixo custo proporcionado pelo avanço da tecnologia de fabricação do hardware, que permite o desenvolvimento de aplicações embarcadas complexas, como processamento multimídia em telefones celulares. Porém, as estratégias de projeto existentes não são adequadas para o desenvolvimento de aplicações com requisitos de tempo real e extremamente adaptadas. O desenvolvimento de sistemas orientados a aplicação (AOSD) mostrou-se eficiente para o desenvolvimento de software para sistemas embarcados. Este artigo mostra uma maneira de melhorar a geração de hardware, incluindo suporte a requisitos de tempo-real, utilizando a estratégia AOSD.*

1. Introdução

Sistemas embarcados estão presentes em nosso cotidiano, desde sistemas de controle de freio em automóveis a eletrodomésticos inteligentes em nossas casas. O avanço da tecnologia de fabricação do hardware e a diminuição dos custos de produção possibilitou o desenvolvimento de complexas aplicações embarcadas. Aplicações multimídia como exibição de vídeos em celulares e câmeras digitais são exemplos destas aplicações. No entanto, a complexidade destas aplicações reflete-se no processo de desenvolvimento destes sistemas. A evolução das ferramentas de CAD (*Computer Aided-design*) tem auxiliado este processo, porém as estratégias de desenvolvimento existentes nem sempre resultam na melhor solução possível. A falta de estratégias e ferramentas de desenvolvimento que auxiliem a especificação de aspectos de tempo-real como: previsibilidade, confiabilidade, desempenho e compactação motivam este trabalho [Farines et al. 2000] (muitos sistemas de tempo-real são embarcados, o que implica recursos reduzidos de processamento, memória e consumo de energia).

Polpeta em [Polpeta and Fröhlich 2005] apresenta uma maneira de gerar hardware utilizando o projeto de sistemas orientados à aplicação proposto por Fröhlich em [Fröhlich 2001]. Esta estratégia permite que o sistema operacional (assim como todo o software) e o hardware sejam adaptados à aplicação. Isto é obtido reusando componentes de hardware e software. Este trabalho visa utilizar a estratégia AOSD no desenvolvimento de componentes de hardware, criando nestes recursos de suporte a tempo-real. A seção 2, à seguir, mostra uma visão geral da estratégia AOSD, que utilizaremos no desenvolvimento de componentes de hardware. A seção 3 mostra como será feito o desenvolvimento de componentes de hardware seguindo a estratégia AOSD. A seção 4, por fim, conclui o trabalho.

2. Visão geral da estratégia AOSD

A metodologia de desenvolvimento de sistemas orientados a aplicação (AOSD) provê meios para obtenção de componentes de software pertencentes ao sistema através de um processo de engenharia de domínio. Entidades identificadas em um domínio são então organizadas em famílias de abstrações, seguindo a análise de variabilidade presente no *Projeto baseado em famílias*, proposto por Parnas em [Parnas 1976].

Abstrações que incorporam detalhes do ambiente em que se encontram, tem pequenas chances de serem reusadas em cenários diferentes. A dependência de hardware é reduzida em AOSD utilizando o conceito de separação de aspectos proposto por Gregor Kiczales em [Kiczales 1997], no processo de decomposição do domínio. Este conceito possibilita a identificação de variações de cenário, que ao invés de serem modeladas como novos membros de uma família, define um aspecto do cenário.

O desenvolvimento de sistemas orientados à aplicação propõe um processo de engenharia de domínio que modela componentes de software utilizando principalmente três construções: famílias de abstrações independentes de cenários, adaptadores de cenários e interfaces infladas.

Famílias de abstrações independentes de cenário são identificadas durante a fase de decomposição do domínio. Abstrações são identificadas a partir de entidades significativas do domínio e agrupadas em famílias, de acordo com suas características comuns.

Adaptadores de cenário são utilizados para resolver as dependências de cenário [Fröhlich and Schroder-Preikschat 2002]. Estas devem ser identificadas como aspectos durante a decomposição do domínio, mantendo as abstrações independentes de cenários. Os adaptadores de cenários são utilizados para aplicar os aspectos de cenários nas abstrações de maneira transparente.

Interfaces Infladas possuem as funções de todos os membros de uma família, resultando em uma visão única da família como se esta fosse um "super-componente". Isto possibilita que o desenvolvedor da aplicação escreva a aplicação com base em uma interface concisa e bem conhecida, adiando a decisão de qual membro da família deve ser usado até o momento em que o sistema é gerado. Esse membro será então agregado ao SO em tempo de compilação.

Para possibilitar que os componentes e o sistema operacional sejam portáveis para a maioria das arquiteturas, um sistema projetado de acordo com AOSD utiliza mediadores de hardware [Polpeta and Fröhlich 2004]. A idéia principal deste artefato de portabili-

dade é manter um *contrato de interface* entre o sistema operacional e o hardware. Cada componente de hardware é acessado através de seu próprio mediador, o que possibilita a portabilidade das abstrações que o usam, sem criar dependências desnecessárias. Mediadores são metaprogramados estaticamente e se "dissolvem" nas abstrações do sistema assim que o contrato de interface é firmado. Em outras palavras, um mediador de hardware provê as funcionalidades do componente de hardware correspondente através de uma interface orientada ao sistema operacional.

No contexto de dispositivos de lógica programável (*Programable Logic Devices* PLD) onde IPs normalmente são implementados utilizando linguagens de descrição de hardware como VHDL e Verilog, mediadores de hardware podem inferir IPs que sejam realmente necessários para compor o hardware do sistema e algumas de suas características. Estes IPs são identificados assim que o mediador de hardware é instanciado pela aplicação, logo, o hardware do sistema terá apenas os componentes necessários para suportar a aplicação [Polpeta and Fröhlich 2005] sendo desenvolvida.

3. Componentes de hardware segundo AOSD

Mediadores de hardware possuem propriedades configuráveis [Polpeta and Fröhlich 2004] que auxiliam na adaptação dos componentes de hardware que irão compor a arquitetura desejada. A utilização do projeto de sistemas orientados à aplicação no desenvolvimento dos componentes de hardware torna possível a alteração do comportamento destes componentes baseada no código da aplicação e a aplicação de adaptadores de cenários ao hardware.

Dentre as possíveis propriedades de um mediador de hardware está o atendimento a requisitos de tempo-real. Um componente de hardware correspondente a um mediador com esta propriedade suporta aplicações de tempo-real. O uso do paralelismo intrínseco do hardware e o aumento do previsibilidade do sistema (conseguido utilizando componentes de hardware específicos) facilitam o atendimento de requisitos de tempo-real.

Além das propriedades de tempo-real, outras também podem ser implementadas em hardware, como a geração de códigos CRC em dispositivos de comunicação (UART, Ethernet, etc.) e o acesso exclusivo à áreas de memória compartilhadas. Com isso pode-se aumentar o desempenho geral do sistema e permitir uma melhor exploração do espaço de projeto.

A implementação dessas características em componentes de hardware seria possível se as linguagens de descrição de hardware possuíssem características avançadas, como metaprogramação estática e orientação a objetos, presentes em linguagens de programação como C++. Linguagens de descrição de hardware, como VHDL e Verilog, permitem ao desenvolvedor a especificação de campos configuráveis nas entidades (*generics* em VHDL). Porém, a parametrização e a configuração existentes nessas linguagens não permitem a aplicação de técnicas utilizadas em AOSD, como orientação a objetos e aspectos.

Nossa proposta é utilizar características do pré-compilador C++ para modificar arquivos VHDL existentes, gerando descrições de processos (estruturas VHDL) diferentes de acordo com a aplicação. Isso permitirá realizar a configuração dos componentes de hardware (arquivos de descrição) durante o processo de compilação, baseado no código da aplicaçcão. Isto não é possível simplesmente com as estruturas presentes em lingua-

gens de descrição de hardware. Isto nos permitirá conseguir a adaptação do hardware à aplicação como é feito com o software.

4. Conclusão

Esse trabalho foi motivado pela crescente complexidade no desenvolvimento de sistemas embarcados e as deficiências das estratégias de desenvolvimento existentes. Apresentamos a estratégia de desenvolvimento AOSD, desenvolvimento de sistemas orientados a aplicação, que permite a geração do hardware e do software adaptados à aplicação.

O uso de artefatos e conceitos de AOSD, possibilitou a inferência de componentes de hardware específicos a partir do código da aplicação (mediadores de hardware). Isto reduz o tempo de desenvolvimento e auxilia o desenvolvimento de um sistema com o menor número de componentes possível.

Propomos neste artigo uma estratégia para geração de componentes de hardware configurados durante a compilação do software da aplicação. Isso será realizado utilizando conceitos presentes em AOSD como orientação a objetos e aspectos, que permitem a definição de propriedades dos componentes de hardware de acordo com os requisitos da aplicação. Isto possibilitará a obtenção de arquiteturas "enxutas" com componentes adaptados ao sistema alvo.

5. Agradecimentos

Agradecemos à FINEP (Financiadora de Estudos e Projetos) que financiou parcialmente este trabalho, processo n.: 01.04.0903.00.

Referências Bibliográficas

Farines, J. M., Fraga, J. d. S., and Oliveira, R. S. d. (2000). *Sistemas de Tempo Real*. Escola de Computação 2000, 1 edition.

Fröhlich, A. A. (2001). *Application-Oriented Operating Systems*. Sankt Augustin: GMD - Forschungszentrum Informationstechnik, 1 edition.

Fröhlich, A. A. and Schroder-Preikschat, W. (2002). Scenario adapters: Efficiently adapting components. *In Proceedings of the 4th Wolrd Multiconference on Systemics, Cybernetics and Informatics*.

Kiczales, Gregor, e. a. (1997). Aspect-oriented programming. *In Proceedings of the European Conference on Object-oriented Programming*, 1241:220–242.

Parnas, D. L. (1976). On the design and development of program families. *IEEE Transactions on Software Engineering*, 2:1–9.

Polpeta, F. V. and Fröhlich, A. A. (2004). Hardware mediators: a portability artifact for component-based systems. *In: Proceedings of the International Conference on Embedded and Ubiquitous Computing*, 3207:271–280.

Polpeta, F. V. and Fröhlich, A. A. (2005). On the automatic generation of soc-based embedded systems. *In: Proceedings of the 10th IEEE International Conference on Emerging Technologies and Factory Automation*.