SISTEMAS DE INFORMAÇÃO INDUSTRIAIS ORIENTADOS AO CONTROLO: PERSPECTIVAS METODOLÓGICAS PARA TECNOLOGIAS RECONFIGURÁVEIS[*]

Ricardo J. Machado
Dept. Sistemas de Informação
Escola de Engenharia – Univ. do Minho
Campus de Azurém, Guimarães

João M. Fernandes
Dept. Informática
Escola de Engenharia – Univ. do Minho
Campus de Gualtar, Braga

Henrique D. Santos
Dept. Sistemas de Informação
Escola de Engenharia – Univ. do Minho
Campus de Azurém, Guimarães

Resumo: Este artigo pretende descrever, de uma forma sumária, a metodologia proposta para suportar o desenvolvimento de aplicações tempo-real embebidas, capazes de suportar computacionalmente sistemas de informação industriais.

1. Introdução

As capacidades computacionais industriais afectam directamente o desenvolvimento tecnológico e a competitividade económica de um país. O controlo industrial está a tornar-se cada vez mais complexo, uma vez que, para garantir que os produtos fabricados cumpram, escrupulosamente, as especificações de qualidade, é necessário controlar e monitorizar os processos e os equipamentos industriais de uma forma cada vez mais apertada.

O controlo, monitorização e supervisão de processos industriais tem exigido o investimento em soluções tecnológicas cada vez mais baseadas em sistemas tempo-real embebidos, especialmente desenvolvidos para interligarem inteligentemente os equipamentos de fabrico aos sistemas de informação de gestão da produção, da qualidade e da manutenção. Os *sistemas de informação industriais orientados ao controlo* (***i**ndustrial **c**ontrol-based **i**nformation **s**ystems - ICIS*) têm, assim, como objectivo principal a gestão do fluxo de informação entre os níveis CIM (***c**omputer **i**ntegrated **m**anufacturing*) [Waldner 1992] inferiores e os superiores (fig. 1).

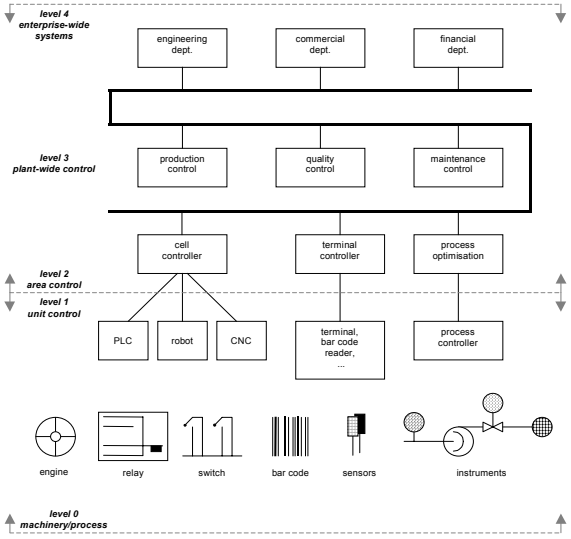

Fig. 1: Níveis CIM numa organização industrial.

No contexto da fig. 1, parece evidente a necessidade de soluções tecnológicas para a interligação facilitada dos nível CIM inferiores (0, 1 e 2) e superiores (3 e 4) [Ranky 1990]. Estas soluções devem recorrer a sistemas baseados em computador (***c**omputer-**b**ased **s**ystems - CBS*) tipicamente tempo-real embebidos e, eventualmente, distribuídos, capazes de suportar computacionalmente os ICIS,

[*] Trabalho parcialmente financiado pela FCT - FUNDAÇÃO DA CIÊNCIA E TECNOLOGIA, no âmbito do projecto PRAXIS/P/EEI/10155/1998, *Sistemas Embebidos Reconfiguráveis: Metodologias de Desenvolvimento para Aplicações Tempo-Real.*

perfeitamente complementares aos sistemas de informação de gestão (***m**anagement **i**nformation **s**ystems* - *MIS*) dentro das organizações industriais [Scholz-Reiter 1992]. O conjunto integrado MIS + ICIS é a solução para as promessas que as abordagens ERP (***e**nterprise **r**esources **p**lanning*) e POS (***p**lant **o**perations **s**ystems*) têm feito no sentido de fornecer uma plataforma aplicacional que integre, de uma forma unificada, o controlo e a gestão de toda a informação organizacional [Lipro 1999].

O desenvolvimento e a implementação tecnológica do suporte computacional deste tipo de sistemas de informação exige, no entanto, que sejam tratadas várias questões metodológicas e arquitecturais [Machado *et al.* 1999]. Desta forma, tem vindo a assistir-se à tentativa de conjugar, ao nível do *hardware*, a flexibilização das primitivas de controlo com a optimização do desempenho, recorrendo às recentes tecnologias das arquitecturas de processamento (re)configuráveis para implementar partes dos CBS para controlo industrial, juntamente com uma abordagem de integração coerente do desenvolvimento conjunto das componentes de *hardware* e de *software* (co-projecto *hardware*/*software*).

O domínio da computação tempo-real apresenta-se, assim, fundamental para cada vez mais áreas aplicacionais como, por exemplo, as telecomunicações, os sistemas aviónicos, os sistemas de defesa e o, já referido, controlo industrial, o que significa que as infra-estruturas do mundo moderno dependem cada vez mais dos CBS tempo-real embebidos. A dependência crescente deste tipo de sistemas exige um perfeito domínio metodológico e tecnológico no desenvolvimento de sistemas cada vez menos convencionais, tendencialmente abertos e distribuídos

2. Co-Projecto Hardware/Software

O co-projecto *hardware*/*software* tem promovido a fertilização cruzada entre os domínios do *hardware* e do *software*, permitindo uma unificação semântica dos conceitos relevantes na modelação ao nível do sistema, a utilização da abstracção de dados (orientação ao objecto) no projecto de *hardware* digital e o recurso a modelos executáveis para avaliar o sistema nas fases iniciais do seu desenvolvimento [Machado *et al.* 2000]. A investigação em fertilização cruzada entre os dois domínios tem dado excelentes resultados e deve continuar no sentido de promover a prototipagem virtual (totalmente em *software*) dos sistemas, apostando cada vez mais na incorporação da abordagem operacional e dos métodos em espiral nas metodologias de desenvolvimento de suporte ao co-projecto *hardware*/*software* de CBS tempo-real embebidos [Fernandes 2000].

Os problemas da partição *hardware*/*software* e do escalonamento global das funcionalidades dos sistemas *hardware*/*software* estão longe de estar consensualmente resolvidos, uma vez que é necessário tratar questões complexas como a sincronização do *software* pseudo-concorrente com o *hardware* inerentemente paralelo, conjuntamente com a minimização dos custos de comunicação entre as várias partições tecnologicamente diversas [Yen *et al.* 1996]. Estes problemas agravam-se quando se está perante uma classe de sistemas como os tempo-real embebidos, porque surgem requisitos não funcionais adicionais que limitam e condicionam fortemente as decisões de concepção.

3. Arquitecturas Reconfiguráveis

Com o surgimento de tecnologias que possibilitam a reconfiguração dinâmica (em tempo de execução) de estruturas de *hardware*, aqueles problemas começam a ser "intratáveis", uma vez que, quer a nível da co-partição, quer a nível do escalonamento global, surge agora uma nova dimensão que elimina, quase que por completo, as anteriores diferenças funcionais efectivas entre o *hardware* e o *software*, alterando radicalmente a forma de realizar a estimação dos recursos, crucial para realizar uma partição *hardware*/*software* de uma forma consciente e sustentada. Assim, justifica-se uma aposta clara na investigação em massa nestes domínios de trabalho para que o co-projecto *hardware*/*software* possa beneficiar rapidamente da utilização de arquitecturas alvo que recorram exaustivamente à nova classe ISP (***i**n-**s**ystem **p**rogramming*) de componentes CPLD (***c**omplex **p**rogrammable **l**ogic **d**evices*) e FPGA (***f**ield-**p**rogrammable **g**ate **a**rrays*), tornando possível a substituição das primitivas de *software* que estão no caminho crítico por primitivas de *hardware* (implementadas multiplexando-se no tempo os recursos físicos disponíveis através da sua reconfiguração dinâmica), o que permite que se cumpram, com o mínimo de *hardware*, os índices de desempenho temporais exigidos. Com a tecnologia ISP os componentes FPGA têm atingido posições de destaque na implementação de sistemas computacionais

reconfiguráveis, em contraste com o seu tradicional papel na substituição provisória e precária dos dispendiosos MPGAs (***m**ask-**p**rogrammable **g**ate **a**rrays*) [Hutchings *et al.* 1995].

O surgimento recente de sistemas reconfiguráveis, baseados na tecnologia de FPGAs ISP (FCCMs – ***F**PGA based **c**ustom **c**omputing **m**achines*), tem levantado questões sobre a eficácia tecnológica das inúmeras propostas já existentes de arquitecturas alvo (no âmbito do co-projecto *hardware*/*software*), nomeadamente ao nível da granulosidade do *hardware* reconfigurável e da proximidade topológica entre os recursos de *hardware* reconfigurável e o CPU [Esteves *et al.* 1997]. As inúmeras questões de base ainda existentes estão na origem da dificuldade em os sistemas reconfiguráveis serem suportados por um verdadeiro paradigma de computação reconfigurável, uma vez que este ainda não oferece uma abordagem perfeitamente sistémica ao projectista de *software* que pretende beneficiar transparentemente das vantagens tecnológicas dos sistemas reconfiguráveis, sem se embrenhar nos intrincados aborrecimentos inerentes às diversas tarefas de síntese de *hardware*, ainda indispensáveis para dotar as arquitecturas reconfiguráveis das necessárias optimizações que as tornam atraentes em termos do seu desempenho relativo [Buell *et al.* 1996].

Desta forma, a computação reconfigurável deve ser considerada como um paradigma especialmente adequado ao desenvolvimento de CBS tempo-real embebidos, uma vez que esta classe de sistemas beneficia enormemente de abordagens que promovem a eficiência na implementação dedicada das funcionalidades, bem como do aumento da capacidade computacional ao nível dos circuitos [Mangione-Smith *et al.* 1997]. No entanto, mesmo neste âmbito mais restrito (uma vez que se exclui, por exemplo, a computação genérica), o sucesso do paradigma da computação reconfigurável exige inovações metodológicas, ao nível do controlo da complexidade e da continuidade dos modelos, no desenvolvimento daquele tipo de sistemas [Fernandes *et al.* 1999]. Uma vez que o desenvolvimento de CBS tempo-real embebidos tem sido progressivamente realizado segundo a abordagem do co-projecto *hardware*/*software*, aquelas inovações passam, necessariamente, pelo casamento das actuais tecnologias ECAD (***e**lectronic **c**omputer **a**ided **d**esign*) e CASE (***c**omputer **a**ided **s**oftware **e**ngineering*) com vista à obtenção de uma verdadeira metodologia EDA (***e**lectronic **d**esign **a**utomation*) que integre metodologicamente a prototipagem virtual e a síntese ao nível do sistema, como forma de lidar com a cada vez maior complexidade existente ao nível da semântica das aplicações, da funcionalidade inerente aos sistemas projectados e da tecnologia de suporte às arquitecturas alvo utilizadas.

4. Três Níveis de Co-Projecto

Tendo em conta as considerações anteriores acerca do co-projecto *hardware*/*software* e das arquitecturas de processamento reconfiguráveis, a grande questão consiste, então, em como obter um ambiente de desenvolvimento para o co-projecto, ao nível do sistema, de CBSs tempo-real embebidos que suportem a implementação de ICIS, de tal forma que os modelos possam ser iterativamente reificados até à implementação final, sem existir a necessidade de efectuar os macro-refinamentos de uma forma manual, com reutilização transparente de módulos de *hardware* e de *software* e que integre adequadamente, ao longo do processo de desenvolvimento, as actividades dos três tipos de engenheiros (de sistemas, de *software* e de *hardware*) necessários para uma correcta execução de projectos desta natureza e com esta complexidade.

Perante este problema, os autores propõem uma "democratização" do co-projecto *hardware*/*software*, colocando-o ao alcance dos três tipos de profissionais de engenharia já referidos e garantindo que não fique, na prática, vedado, aos que não possuem uma forte motivação para as questões de *hardware*. Para isso, é necessário contrariar a forma tradicional de realizar projectos (*top-down*), desacoplando-a em três níveis distintos e organizados segundo um macroprocesso de desenvolvimento do tipo *middle-out* (fig. 2): (1) *nível 1*, em que os engenheiros de *hardware* executam a concepção de arquitecturas alvo; (2) *nível 2*, em que os engenheiros de *software* executam a concepção de módulos funcionais; (3) *nível 3*, em que os engenheiros de sistemas (neste caso, de informação) executam a concepção de soluções finais.

Este desacoplamento reflecte a preocupação da metodologia proposta em integrar diferentes fluxos de projecto que, apesar de independentes para os engenheiros de *hardware*, os de *software* e os de sistemas de informação, sejam capazes de comunicar entre si, no sentido em que permitam criar uma

comunidade de desenvolvimento em que em cada um dos três níveis de projecto são implementadas as primitivas de controlo correspondentes às capacidades e funções de cada profissional envolvido. Esta preocupação pode ser caracterizada à luz dos 5 T's de [Madisetti *et al.* 1996]:

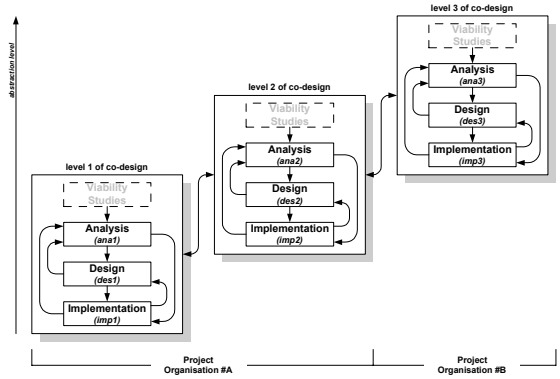

Fig. 2: Macroprocesso da metodologia proposta pelos autores.

(1) *Prazos (timelines)*. A pressão do *time to market*, juntamente com as frequentes alterações de requisitos (e com a consequente alteração nos prazos do projecto) sugerem que as metodologias reduzam o tempo de desenvolvimento e promovam a utilização de tecnologias de rápida implementação, tal como a baseada em componentes do tipo COTS (***c****omponents-****o****ff-****t****he-****s****helf*) [Voas 1998]. Para que seja possível seguir esta recomendação na utilização de arquitecturas de processamento reconfiguráveis, torna-se necessário definir módulos funcionais (construídos a partir de arquitecturas alvo "carregadas" com *software* parametrizável) que implementem, transparentemente, primitivas de controlo de baixo-nível, oferecendo a possibilidade de suportar a reutilização de tecnologia reconfigurável no nível 3 de co-projecto, por parametrização de ***f****unctional-****m****odules-****o****ff-****t****he-****s****helf* (FMOTSs).

(2) *Tarefas (tasks)*. As metodologias utilizadas actualmente ainda são pouco sistemáticas do ponto de vista das várias tarefas a executar, não promovendo um correcto desenvolvimento integrado do *hardware* e do *software* e não fornecendo orientações para uma abordagem baseada na reutilização de componentes disponíveis de projectos anteriores. Esta realidade justifica a necessidade de enquadrar, cuidadosamente, o co-projecto *hardware*/*software* nos três níveis de projecto, definindo políticas diferenciadas de co-projecto para a concepção de arquitecturas alvo (engenheiro de *hardware*), para a concepção de módulos funcionais (engenheiro de *software*) e para a concepção de soluções final (engenheiros de sistemas de informação).

(3) *Ferramentas (tools)*. Há a necessidade de promover a integração de ferramentas, de modo a tornar possível, e de uma forma (semi-)automática, o projecto efectivo ao nível do sistema (*system-level design*) para os engenheiros de sistemas de informação. Esta exigência só se apresenta, no entanto, possível se os outros dois tipos de profissionais forem contemplados com ferramentas de desenvolvimento que suportem directamente os respectivos fluxos de projecto e que permitam a comunicação (semi-)automática entre os três níveis de co-projecto, nomeadamente na manipulação semântica de representações unificadas e na geração automática de código [Machado *et al.* 1997].

(4) *Tecnologia (technology)*. Com as tendências de duplicação da complexidade dos componentes programáveis cada 18 meses e de oferta de tecnologia de ponta em cada vez menores intervalos de tempo, as soluções desenvolvidas à medida (*custom solutions*) apresentam-se interessantes somente durante um período de tempo cada vez menor (muitas vezes inferior ao tempo de desenvolvimento requerido), tal como acontece com os sistemas baseados em ASICs (***a****pplication-****s****pecific* ***i****ntegrated* ***c****ircuits*). Desta forma, é cada vez mais pertinente adoptar metodologias que suportem tecnologias que possibilitem a actualização periódica dos componentes do sistema, de forma a aumentar o seu desempenho global, mas garantindo a mesma funcionalidade. Esta actualização periódica do modelo dos componentes (*model year upgrade*) garante uma quase permanente actualização tecnológica do sistema, recorrendo, mais uma vez, à tecnologia COTS. Do ponto de vista metodológico, esta vantagem deve ser capitalizada realizando o

desacoplamento formal entre os três níveis de co-projecto, o que desobriga cada um dos níveis da necessidade de desenvolver actualizações tecnológicas que não no seu nível.

(5) *Talento (talent)*. Para além de todo o trabalho de investigação e desenvolvimento e da aplicação prática em projectos reais de engenharia dos princípios de projecto expostos, é necessário não desprezar a formação e o enquadramento dos três tipos de profissionais exigidos por esta abordagem, garantindo uma efectiva exploração desta nova forma de desenvolvimento que, tentando levar ao extremo o paradigma da prototipagem virtual aplicada ao controlo industrial, pretende chegar a um novo paradigma designado, eventualmente, de *automação virtual*.

A fig. 2 ilustra dois tipos de projectos, cada um executado numa organização distinta: (1) organização #A, que suporta os níveis 1 (engenheiro de *hardware*) e 2 (engenheiro de *software*) de co-projecto, através do fornecimento de FMOTSs prontos a usar; (2) organização #B, que suporta o nível 3 (engenheiro de sistemas de informação), através da configuração e interligação de FMOTSs para fornecer uma solução final que suporte um ICIS. Existe ainda uma terceira classes de organizações: organização #C, que corresponde à organização industrial que recebe a solução final para instalar na sua planta fabril.

5. Modelo do Processo dos Três Níveis de Co-Projecto

A execução de uma tarefa de captura de requisitos ao nível do processo (notar que não é ao nível do produto) da abordagem dos três níveis de co-projecto descritos anteriormente dá origem a um diagrama de casos de uso UML (não apresentado neste artigo) que representa as obrigações e responsabilidades dos três tipos de profissionais envolvidos. A partir deste diagrama é possível obter o diagrama do ciclo de vida das soluções finais (fig. 3).

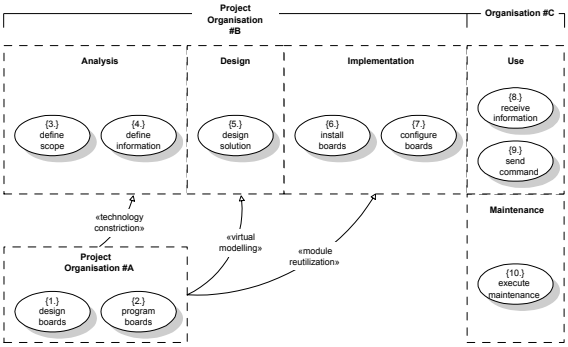

Fig. 3: Diagrama do ciclo de vida das soluções finais.

Neste diagrama aparecem três novas categorias para o estereótipo *UML* do tipo *«relationship»*:

(1) *«technology constriction»*. Este estereótipo restringe o domínio de actuação dos casos de uso *3. definir âmbito* e *4. definir informação* (na captação dos requisitos do utilizador das soluções finais, no âmbito da fase de análise dos projectos executados por uma organização do tipo #B) aos âmbitos de aplicação suportados tecnologicamente pela organização do tipo #A fornecedora. Esta delimitação da natureza das actividades económicas suportadas deve-se ao facto de que cada organização do tipo #A possui um conjunto limitado de arquitecturas alvo, com as suas características próprias em termos de capacidade computacional, e disponibiliza um conjunto de classes funcionais (FMOTSs) bem definidas.

(2) *«virtual modelling»*. Este estereótipo impõe ao caso de uso *5. conceber solução* (na concepção das soluções finais, no âmbito da fase de concepção dos projectos executados por uma organização do tipo #B) a necessidade de manipular modelos virtuais dos FMOTSs, disponibilizados pela organização do tipo #A fornecedora e seleccionados para fazerem parte integrante da implementação final. Esta modelação virtual é somente do ponto de vista das características necessárias para a interligação e parametrização ao nível da solução final, garantindo a continuidade dos modelos durante na passagem dos níveis 1 e 2 de co-projecto para o nível 3, pelo que os FMOTSs, previamente desenvolvidos pela

organização do tipo #A fornecedora, devem ser vistos como "caixas pretas" responsáveis por implementar determinadas funcionalidades. Esta transparência tecnológica deve garantir um bom controlo da complexidade na concepção das soluções finais.

(3) *«module reutilization»*. Este estereótipo garante aos casos de uso *6. instalar boards* e *7. parametrizar boards* (na implementação das soluções finais, no âmbito da fase de implementação dos projectos executados por uma organização do tipo #B) a possibilidade de basear todas as suas actividades na reutilização de módulos previamente desenvolvidos pela organização do tipo #A fornecedora. Esta garantia de reutilização possibilita às organizações do tipo #A beneficiar da economia de escala na amortização dos custos de desenvolvimentos de FMOTSs e oferece às organizações do tipo #B a utilização de FMOTSs largamente testados e experimentados noutras soluções finais, uma vez que são evitados desenvolvimentos tecnológicos à medida das exigências peculiares de cada solução final.

De facto, estes três estereótipos, que estabelecem relações entre casos de uso de organizações do tipo #A e do tipo #B, foram criados para definir a separação formal entre os níveis 2 e 3 de co-projecto, estabelecendo uma fronteira bem definida entre as actividades dos engenheiros de *software* e dos engenheiros de sistemas de informação.

Aplicando a estratégia do *4-step rule set* descrita em [Fernandes *et al.* 2000], é possível obter um diagrama de objectos ao nível do processo (fig. 4). Este diagrama de objectos representa os requisitos das ferramentas de desenvolvimento que devem suportar a abordagem dos três níveis de co-projecto. Neste diagrama existem dois pacotes aplicacionais:

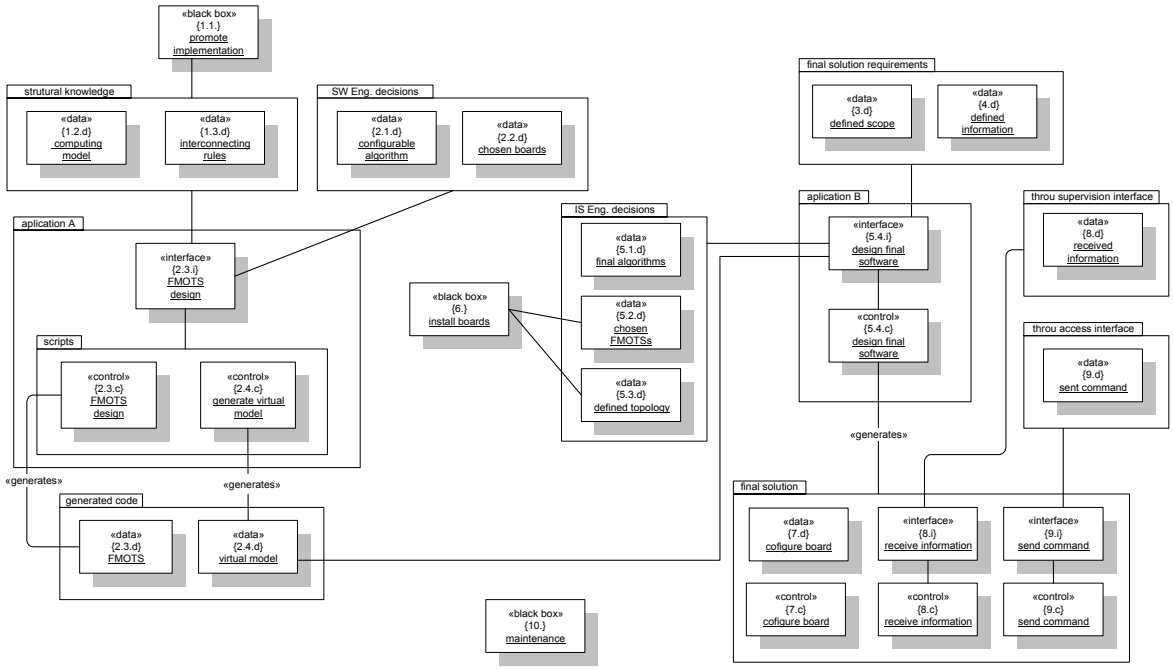

Fig. 4: Diagrama de objectos ao nível do processo.

(1) *Package aplicação A*. Este *package* deve suportar o nível 2 de co-projecto através da disponibilização de: (i) um ambiente com uma HMI (*2.3.i desenvolver FMOTS*) capaz de, em organizações do tipo #A, auxiliar as tarefas de projecto do engenheiro de *software*, detentor de conhecimento estrutural das arquitecturas alvo (*1.2.d modelo de computação* e *1.3.d regras de interligação*), a partir das suas duas grandes pré-decisões (*2.1.d algoritmo parametrizável* e *2.2.d boards escolhidos*); (ii) um "motor" que (semi-)automatize o desenvolvimento de FMOTSs (*2.3.c desenvolver FMOTS*) e a geração do código final (*2.3.d FMOTS*) para síntese e implementação na arquitectura alvo; (iii) um "motor" que (semi-)automatize a geração de modelos virtuais (*2.4.c gerar modelo virtual*) para que, no ambiente correspondente ao *package aplicação B*, seja possível parametrizar e interligar, de uma forma tecnologicamente transparente, os FMOTSs previamente desenvolvidos.

(2) *Package aplicação B*. Este *package* deve suportar o nível 3 de co-projecto através da disponibilização de: (i) um ambiente com uma HMI (*5.4.i desenvolver software final*) capaz de, em organizações do tipo #B, auxiliar as tarefas de projecto do engenheiro de sistemas de informação, conhecedor dos requisitos da solução final (*3.d âmbito definido* e *4.d informação definida*), a partir das suas três grandes pré-decisões (*5.1.d algoritmos finais*, *5.2.d FMOTSs escolhidos* e *5.3.d topologia definida*); (ii) um "motor" que (semi-)automatize a geração das soluções finais (*5.4.c desenvolver software final*).

A fig. 5 ilustra o ambiente EDA global da abordagem dos três níveis de co-projecto, com as ferramentas ECAD, CASE e CAE (***c**omputer **a**ided **e**ngineering*) em cascata suportando os três tipos de engenheiros no projecto de soluções finais.

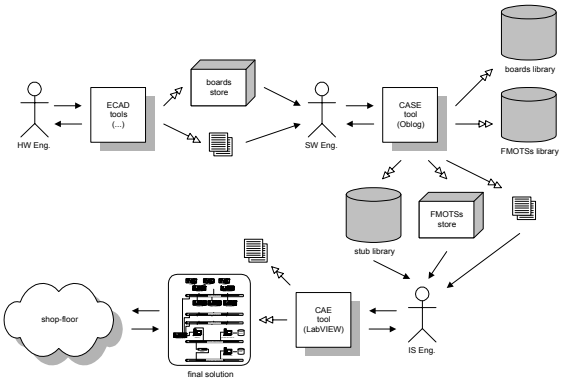

Fig. 5: Global three-level co-design environment.

6. Conclusões

A abordagem proposta é baseada num forte investimento em técnicas de especificação e desenvolvimento orientados por objectos [Machado *et al.* 1998].

No que diz respeito às metodologias de especificação, é imperioso garantir uma correcta abordagem ao nível: (1) da linguagem, através da especificação explícita dos requisitos temporais e de modelos de passagem do tempo e do suporte ao tratamento de excepções, à modelação do ambiente, à abordagem multi-vista e à semântica operacional dos meta-modelos, o que não se tem verificado de uma forma generalizada; (2) do controlo da complexidade, através do suporte a formalismos gráficos e à abordagem *bottom-up*, o que já é uma preocupação frequente nas metodologias existentes; (3) da continuidade dos modelos, através da integração das representações e incorporação de refinamentos, ao longo das várias fases de desenvolvimento, e da extensão ao suporte de *software* e *hardware*, o que se apresenta como fundamental para suportar o paradigma do co-projecto *hardware*/*software* para CBS tempo-real embebidos.

No âmbito das metodologias de desenvolvimento, é importante migrar, de uma forma pragmática, para uma solução que integre as vantagens da capacidade semântica dos métodos orientados por objectos com os modelos que suportem adequadamente as vantagens operacionais das abordagens transformacionais, aumentando, assim, a possibilidade de automatizar cada vez mais o processo de desenvolvimento. Isto não significa que as ferramentas automáticas são a panaceia para os problemas existentes no desenvolvimento de sistemas tempo-real embebidos, mas antes que elas podem executar as tarefas que não exijam um esforço criativo, libertando assim o projectista para resolver problemas mais críticos e que exijam uma criatividade ainda não automatizável na tomada de decisões.

Referências

[Buell *et al.* 1996] Buell, Duncan A. and Jeffrey M. Arnold, Walter J. Kleinfelder, *SPLASH 2: FPGAs in a Custom Computing Machine*, IEEE Computer Science Press, 1996.

[Esteves *et al.* 1997] Esteves, António and João Fernandes, Alberto Proença, *EDGAR: A Platform for Hardware/Software CoDesign*, Embedded System Applications, C. Baron, J. Geffroy (Ed.), Kluwer Academic Publishers, 1997.

[Fernandes *et al.* 1999] Fernandes, João and Ricardo Machado, Henrique Santos, *A UML-based Approach for Modeling Industrial Control Applications*, 2nd OMG/IEEE/ACM Int. Conf. on the Unified Modeling Language - UML'99, E.U.A., October, 1999.

[Fernandes 2000] Fernandes, João Miguel, *MIDAS: Metodologia Orientada ao Objecto para Desenvolvimento de Sistemas Embebidos*. Tese de Doutoramento, Universidade do Minho, 2000.

[Fernandes *et al.* 2000] Fernandes, João and Ricardo Machado, Henrique Santos, *Modeling Industrial Embedded Systems with UML*. 8th IEEE/IFIP/ACM Int. Workshop on Hardware/Software Co-Design - CODES'2000, E.U.A., ACM Press, May, 2000.

[Hutchings *et al.* 1995] Hutchings, Brad L. and Michael J. Wirthlin, *Implementation Approaches for Reconfigurable Logic Applications*, Field- -Programmable Logic Applications, LNCS 975, Springer Verlag, 1995.

[Lipro 1999] *The New Productivity Factor*, LIPRO Holding AG, 1999.

[Machado *et al.* 1997] Machado, Ricardo and João Fernandes, Alberto Proença, *SOFHIA: A CAD Environment to Design Digital Control Systems*, Hardware Description Languages and Their Applications: Specification, Modelling, Verification and Synthesis of Microelectronic Systems, C. Kloos, E. Cerny (Ed.), Chapman & Hall, 1997.

[Machado *et al.* 1998] Machado, Ricardo and João Fernandes, Alberto Proença, *An Object-Oriented Model for Rapid-Prototyping of Data Path/Control Systems - A Case Study*. 9th IFAC/IFIP Symp. on Information Control in Manufacturing – INCOM'98, France, June, 1998.

[Machado *et al.* 1999] Machado, Ricardo and João Fernandes, Henrique Santos, *Architectural and Methodological Concerns for Industrial Real-Time Applications: An Hardware/Software Co-Design Approach*, Workshop on Architectural Aspects in Specification and Design - AASD'99, Portugal, December, 1999.

[Machado *et al.* 2000] Machado, Ricardo and João Fernandes, António Esteves, Henrique Santos, *An Evolutionary Approach to the Use of Petri Net based Models: From Parallel Controllers to HW/SW Co-Design*, Hardware Design and Petri Nets, A. Yakovlev, L. Gomes e L. Lavagno (Ed.), Kluwer Academic Publishers, 2000.

[Madisetti *et al.* 1996] Madisetti, Vijay K. and Mark A. Richards, *Advances in Rapid Prototyping of Digital Systems*, IEEE Design & Test of Computers, Fall, 1996.

[Mangione-Smith *et al.* 1997] Mangione-Smith, William and B. Hutchings, D. Andrews, A. DeHon, C. Ebeling, R. Hartenstein, O. Mencer, J. Morris, K. Palem, V. Prasanna, H. Spaanenburg, *Seeking Solutions for Configurable Computing*, IEEE Computer, December, 1997.

[Ranky 1990] Ranky, Paul G., *Computer Networks for World Class CIM Systems*, CIMware Limited, 1990.

[Scholz-Reiter 1992] Scholz-Reiter, B., *CIM Interfaces: Concepts, Standards and Problems of Interfaces in Computer Integrated Manufacturing*, Chapman & Hall, 1992.

[Voas 1998] Voas, Jeffrey, *The Challenges of Using COTS Software in Component-Based Development*, IEEE Computer, June, 1998.

[Waldner 1992] Waldner, Jean-Baptiste, *CIM: Principles of Computer-Integrated Manufacturing*, John Wiley & Sons, 1992.

[Yen *et al.* 1996] Yen, Ti-Yen and Wayne Wolf, *Hardware-Software Co-Synthesis of Distributed Embedded Systems*, Kluwer Academic Publishers, 1996.