Model Assets: Um padrão de arquitetura de utilização de técnicas de MDE para geração de artefatos

Paulo Artur de Sousa Duarte[1], Vanessa. S. Viana[2], Rossana M. C. Andrade[1], Fernando A.M. Trinta[1], José. M. Soares[2]

Universidade Federal do Ceará (UFC) – Fortaleza, CE– Brasil
[1]MDCC – Mestrado e Doutorado em Ciência da Computação
[2] DETI – Departamento de Engenharia de Teleinformática
{pauloduarte, vanessaviana, rossana, fernandotrinta}@great.ufc.br, marques@ufc.br

Abstract. *The pattern presents an approach found in some papers that aims to unite the concepts of Model-Driven Engineering (MDE) and Software Product Lines (SPL) to facilitate the development of pervasive or ubiquitous applications. This union occurs with adding, during the stages of the development cycle of an SPL, MDE techniques so that entailed in the creation of artifacts in the form of metamodels. Such artifacts would then be using for the product development of the line.*

Resumo. *O padrão apresentado mostra uma abordagem encontrada em alguns trabalhos que visa unir os conceitos de Model-Driven Engineering (MDE) e Linhas de Produto de Software (LPS) para facilitar o desenvolvimento de aplicações ubíquas ou pervasivas. Tal junção ocorre com a adição, durante as etapas do ciclo de desenvolvimento de uma LPS, de técnicas de MDE de forma que acarretasse na geração de artefatos na forma de metamodelos. Tais artefatos, então, seriam utilizando para o desenvolvimento dos produtos específicos da linha.*

1. Introdução

Uma área de desenvolvimento emergente nos tempos recentes, tanto na academia quanto na indústria, tem sido o desenvolvimento de aplicações ubíquas. Sendo a computação ubíqua, na visão de [Araujo, 2003], a junção dos conceitos de computação pervasiva (aplicações dinâmicas "invisíveis" para o usuário) e móvel (aplicações para dispositivos móveis), como mostrado na Figura 1, a computação ubíqua teria todos os desafios existentes no desenvolvimento de aplicações para ambos.

O conceito de computação ubíqua surgiu com Mark Weiser em que ele retrata a computação ubíqua como a ideia de um novo paradigma, no qual os computadores estarão "embutidos" nos elementos usados pelos seres humanos [Weiser, 1991]. Pode-se também identificar a ideia de transparência da computação ao usuário, como foi relatada no trabalho de Yau [Yau et al., 2002].

Dentre os principais desafios e dificuldades encontradas no desenvolvimento de software tradicional e ubíquo, pode-se citar questões envolvendo a fragmentação do software entre os mais diferentes dispositivos, plataformas, e versões de sistemas operacionais existentes. A questão da fragmentação é um problema atual no que tange ao desenvolvimento para aplicações móveis [Kuusela,

2011]. Também pode-se citar o desenvolvimento de novas arquiteturas que levam em consideração as vantagens e as limitações intrísecas que esse tipo de aplicação teria.

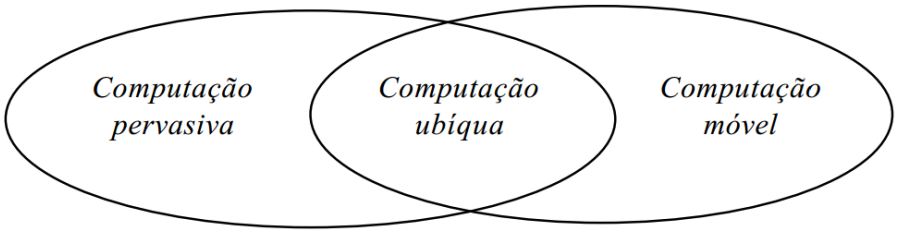

Figura 1: Relação existente entre os conceitos de computação ubíqua, pervasiva e móvel (Araújo, 2003)

Uma solução utilizada em alguns projetos para a questão do desenvolvimento multi-plataforma foi a utilização de linhas de produto de software [Quinton *et al*, 2011]. Segundo a definição do SEI (*Software Engineering Institute*), uma Linha de Produto de Software (LPS) pode ser descrita como "um conjunto de sistemas de software que compartilham uma gama de características comuns e gerenciadas para satisfazer um conjunto de necessidades específicas de um segmento em particular do mercado e que são desenvolvidos de um mesmo conjunto de *core assets* de uma maneira pré-definida." [Northrop, 2002].

O processo de produção de uma LPS tem, além do gerenciamento organizacional, duas outras atividades essenciais: a Engenharia de Domínio, na qual ocorre o desenvolvimento dos artefatos, e a Engenharia de Aplicação, onde ocorre o desenvolvimento dos produtos usando os artefatos produzidos na etapa anterior. [Northrop, 2002]. A Figura 2 demonstra como seria dividida as etapas e o que ocorre exatamente em cada parte do processo [Pohl et al, 2007].

Uma outra abordagem também utilizada para o desenvolvimento de aplicações ubíquas é a *Model-Driven Engineering* (engenharia orientada a modelos), mais conhecida pela sigla MDE. A MDE surgiu como uma abordagem para enfrentar alguns dos principais problemas do desenvolvimento de softwares, como a complexidade das plataformas e as limitações de linguagens como Java ou C# em solucionar tais entraves [Schmidt, 2006]. Outros problemas atacados pela MDE são as dificuldades em se manter documentações e modelagens atualizadas e condizentes com o código e dificuldades em relação ao reuso do conhecimento. E, entre as vantagens encontradas nesta abordagem, pode-se citar a produtividade, a manutenibilidade e a interoperabilidade [Lucrédio, 2009].

Dentre os elementos que compõem a MDE, podemos citar dois focos principais: a construção de modelos de alto nível, normalmente usando linguagens de modelagem de domínio específico e a transformação, por meio de ferramentas de transformação e geradores de código [Schmidt, 2006]. No processo de criação de softwares da MDE, deve-se construir modelos de alto nível que representem fielmente as especificações requeridas do software a ser desenvolvido. Uma vez construído, o código é gerado automaticamente a partir de tais modelos em uma ação que é denominada transformação.

Seguindo este raciocínio, alguns estudos mencionados na seção Usos Conhecidos buscaram métodos de realizar a junção dos conceitos de LPS aos de MDE, buscando unir as características e qualidades de ambas as abordagens. Normalmente, tal junção é feita com a ideia de que os artefatos gerados na etapa de Engenharia de Domínio fossem um, ou mais, metamodelos que pudessem ser trabalhados e modelados na etapa de Engenharia de Aplicação.

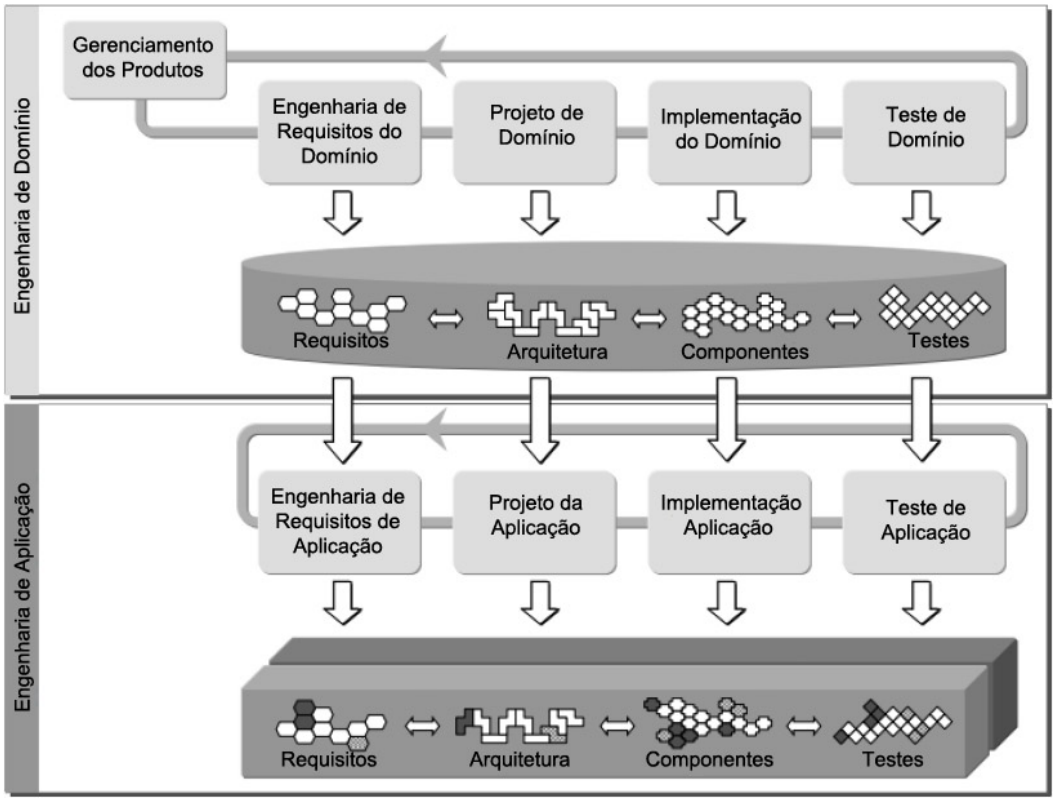

Figura 2: Ciclo de desenvolvimento de uma Linha de Produto de Software [adaptado de Pohl et al, 2007]

2. Contexto

Dentre os principais desafios encontrados no desenvolvimento de aplicações ubíquas pode-se citar a necessidade de criar aplicações lidando com limitações como a volatidade, a capacidade dos dispositivos e a heterogeneidade, a qual impõe barreiras à interoperabilidade das aplicações [Lima, 2011].

Uma abordagem para enfrentar tal desafio seria a adoção do conceito de linhas de produtos de software (LPS) ao desenvolvimento do projeto, de maneira a sistematizar a produção de famílias de softwares com características similiares. No entanto, o ciclo de produção de uma LPS não é totalmente automatizado.

3. Problema

Dentre as questões pendentes aos desenvolvedores, pode-se citar o fato de que toda a Engenharia de Aplicação deve ser feita tendo como objetivo o desenvolvimento de cada produto *per si*, com base nos artefatos que foram gerados na etapa da engenharia de domínio. Deste modo, um desafio atual seria a busca de métodos para automatizar ainda mais tal processo.

Visando essaa necessidade de uma maior automação do processo de Engenharia de Aplicação no contexto da utilização de Linhas de Produtos de Software, qual seria o método que pudesse proporcionar uma solução adequada para tal problema?

4. Solução

Uma solução encontrada em alguns casos foi a integração de técnicas de MDE em determinadas etapas de uma Linha de Produto de Software Ubíqua para facilitar a criação e utilização de artefatos. As inserções das técnicas de MDE seriam realizadas nas seguintes etapas:

- Na Engenharia de Domínio, com o intuito de gerar um conjunto de artefatos na forma de metamodelos;
- Na Engenharia de Aplicação, utilizando os metamodelos gerados na etapa anterior para a geração automática dos códigos das famílias de aplicações ubíquas, com as suas respectivas diferenças.

A divisão do processo da linha de produto de software em duas etapas distintas, Engenharia de Domínio e Engenharia de Aplicação, favorece a implantação dos conceitos de MDE e da própria implementação destes.

Isto é apresentado de acordo com seguinte método:

a. A etapa de Engenharia de Domínio passa a focar no desenvolvimento de um metamodelo, ou conjunto de metamodelos, com todas as características que devem ser inerentes à família de software ubíquo a ser desenvolvida.
b. Tais conjuntos de metamodelos funcionam como os artefatos que são utilizados na etapa seguinte.
c. Na Engenharia de Aplicação, cada aplicação ubíqua é modelada em alto nível de acordo com as especificações de sua plataforma, contexto, versão do sistema operacional, entre outros.
d. Essa modelagem ocorre em alto nível, utilizando os metamodelos previamente criados e, uma vez finalizada a modelagem, o código é gerado automaticamente, encerrando a produção do software.

A Figura 3 mostra um exemplo simplificado de como é o processo do desenvolvimento completo do software, focando no principal para o padrão: a geração de artefatos no formato de metamodelo. Os demais usos conhecidos utilizam uma visão geral semelhante, embora em alguns casos com alguns refinamentos.

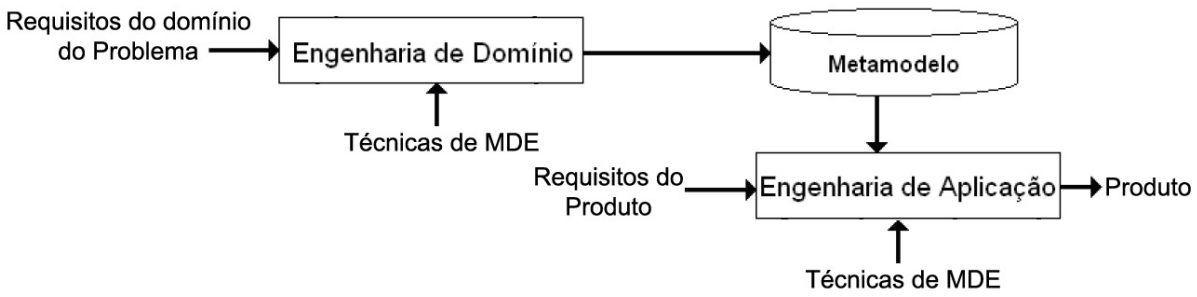

Figura 3. Exemplo do processo de desenvolvimento [adaptado de Santana *et al,* 2009]

5. Forças

Os itens a seguir listam as forças que influenciam positiva ou negativamente na adoção deste padrão no processo de desenvolvimento de uma aplicação.

- **Problemas no Desenvolvimento de Aplicações Ubíquas:** Devido a imensa gama de diversidade de dispositivos e contextos possíveis, o desenvolvimento de aplicações ubíquas trás diversos desafios que, caso não sejam resolvidos adequadamente, podem acarretar em maiores custos de desenvolvimento, perda de produtividade e o problema da fragmentação de software.
- **Alto Custo Inicial:** Tanto a abordagem envolvendo Linhas de Produto de Software como a abordagem MDE tem um custo inicial alto para a empresa para adequar a sua equipe de desenvolvimento para ambos os paradigmas, além do tempo que levará para que esta atinja a curva de aprendizado desejada para um desenvolvimento de sucesso.
- **Custo Decrescente com o Tempo:** Apesar do alto custo inicial, a adoção de ambos conceitos tornam-se gradativamente mais vantajosas, em relação ao desenvolvimento iniciado do zero, à medida que mais produtos da família de software vão sendo desenvolvidos.
- **Abordagens Diferentes Existentes:** Existem trabalhos que mesclam os conceitos de LPS e MDE para enfrentar os problemas de desenvolver aplicações para dispositivos móveis e que não utilizam a ideia deste padrão. Quinton *et al* (2011), por exemplo, utilizou a MDE como técnica para a implementação da estrutura da LPS.

6. Usos conhecidos

Dentre os trabalhos encontrados que utilizam esse padrão de abordagem para o desenvolvimento de aplicações ubíquas ou pervasivas, pode-se destacar os seguintes:

Santana *et al* (2009) utilizam os conceitos de Linhas de Produto de Software e *Model-Driven Engineering* em conjunto para a geração de aplicações ubíquas seguindo a arquitetura apresentada na Figura 4. No caso deste trabalho específico, houve um maior foco no desenvolvimento de uma técnica de DSM (*Domain-Specific Modeling*, Modelagem de Domínio Específico) que permitisse a

construção dos artefatos na forma de metamodelos. Estes metamodelos, re-trabalhados na etapa de engenharia de aplicação e com o auxílio de uma ferramenta de geração de código, poderiam fornecer a base para o desenvolvimento de aplicações ubíquas.

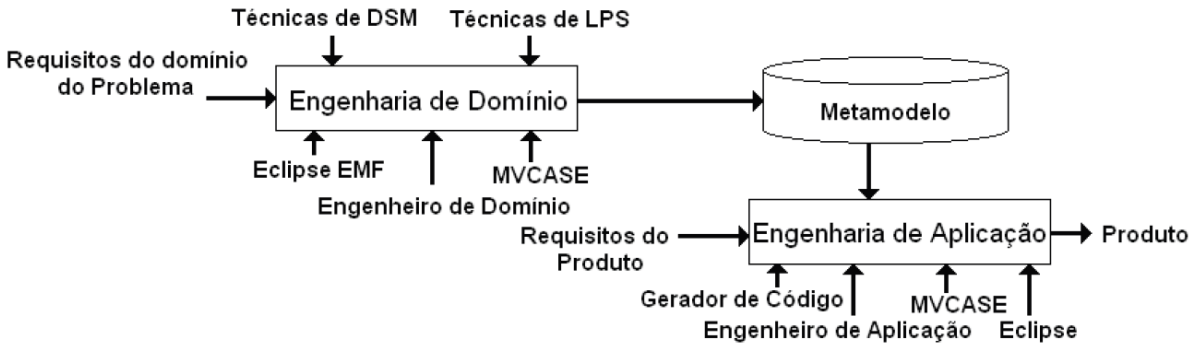

Figura 4. Abordagem do processo de desenvolvimento de Santana *et al* (2009)

Cirilo *et al* (2010), na aplicação *Model-Driven RuchUbi*, utilizam uma arquitetura similar, incluindo os mesmos frameworks e demais tecnologias, com o objetivo de obter uma linha de produtos de software capaz de gerar interfaces ricas adaptativas de aplicações ubíquas interativas, utilizando conceitos de *Rich Internet Applications* (RIA) sensíveis ao contexto. O processo, conforme mostrado na Figura 5, prevê a geração de artefatos na forma de metamodelos. Estes metamodelos forneceriam uma infra-estrutura genérica para auxiliar o desenvolvimento das aplicações na etapa final do desenvolvimento.

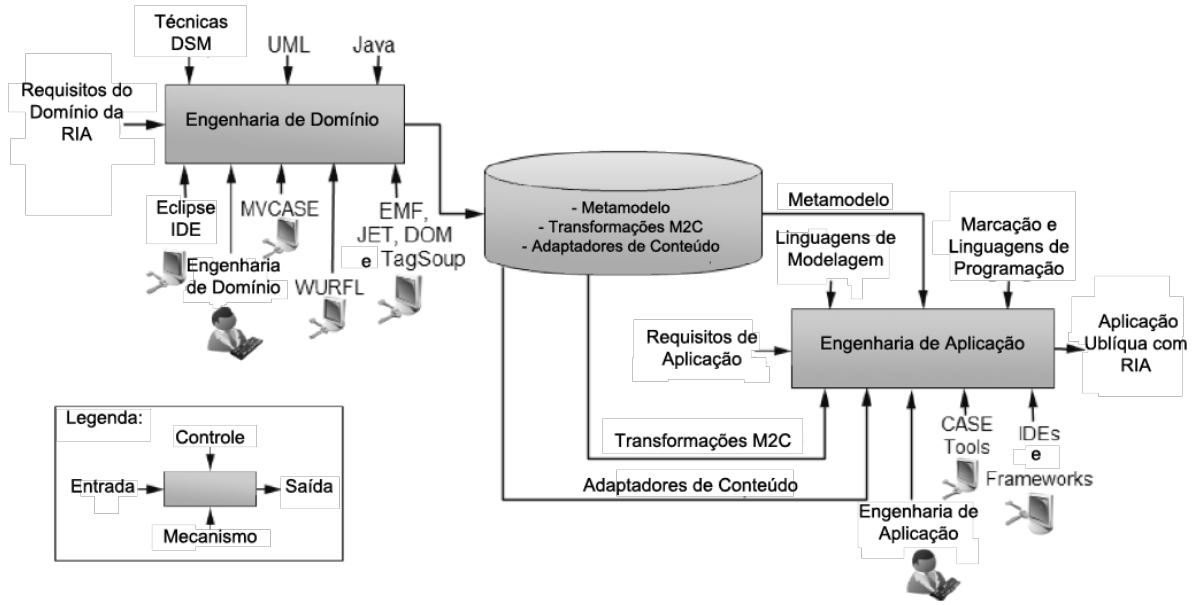

Figura 5. Visão geral do *Model Driven RichUbi* (adaptado de Cirilo *et al*, 2010)

Oliveira *et al* (2009) apresentam uma proposta em unir os conceitos existentes na MDA (*Model-Driven Architecture*, arquitetura orientada a modelos) com os conceitos de uma linha de produto de software visando o desenvolvimento de aplicações ubíquas. A MDA é um conjunto de padrões organizados pela OMG (*Object Management Group*) com o objetivo de padronizar conceitos e tecnologias para o desenvolvimento de softwares orientados a modelos. Entre os conceitos criados estão os níveis de abstrações de um modelo, dentre os quais pode-se citar o PIM (*Plataform Independent Model) e o PSM (Plataform-Specific Model).*

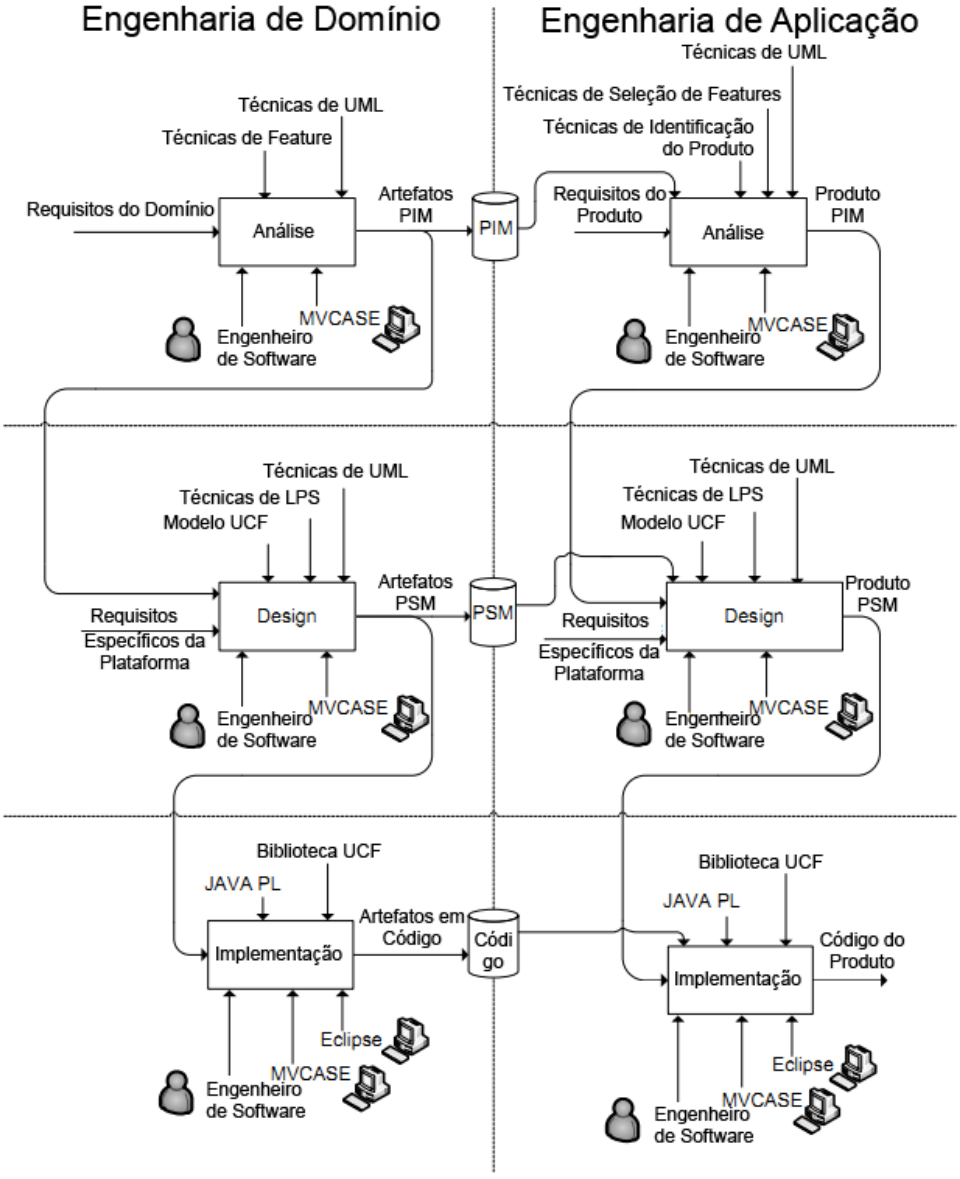

Figura 6. Arquitetura de Oliveira *et al* (adaptado de 2009)

O trabalho apresenta uma abordagem mais incremental, dividida em três etapas de Engenharia de Domínio e de Aplicação com um decrescente nível de abstração, começando da análise do

problema, passando pelo design dele e finalizando com a implementação do código em si. A cada etapa, nas quais vão se acrescentando mais detalhes, menor é o nível de abstração. Assim, na primeira etapa do ciclo, a análise, os artefatos são PIMs. Na etapa do design, onde já entrariam requisitos específicos da plataforma, os artefatos são PSMs. E, na etapa final, os artefatos já estão na forma de código-fonte tradicional. Uma visão geral da abordagem é mostrada na Figura 6.

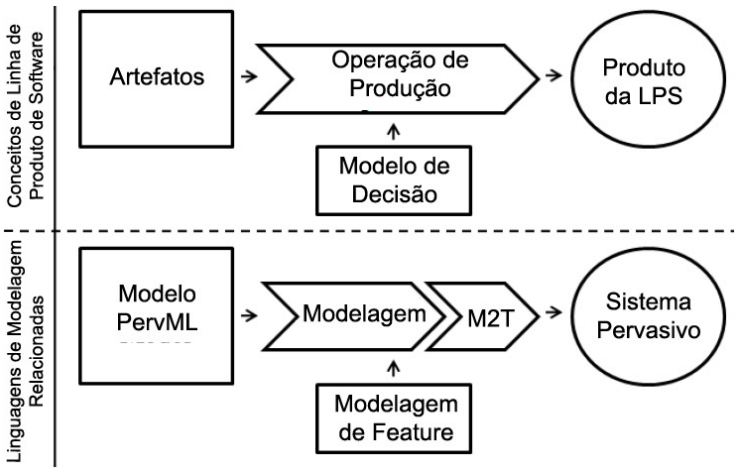

Figura 7. A abordagem de desenvolvimento de Cetina *et al* (adaptado de 2008)

Cetina *et al* (2008) apresenta uma abordagem utilizando LPS, baseadas em conceitos da MDE, para o desenvolvimento de aplicações pervasivas dinamicamente adaptativas. Conforme mostrado na Figura 7, os metamodelos são desenvolvidos com base na PervML, uma linguagem de domínio específico (DSL, de *Domain-Specific Language*) voltada para a construção de aplicações pervasivas [Serral *et al*, 2008]. Embora não mostre uma visão geral da arquitetura, a Figura 7 mostra o que pode ser visto basicamente como a etapa de engenharia de aplicação, sendo os artefatos utilizados para criação dos produtos, modelos e/ou metamodelos criados utilizando o PervML. Enquanto, o desenvolvimento do produto englobaria a modelagem dos artefatos e a sua posterior transformação (M2T, *Model-To-Text*) em um sistema pervasivo, que é o produto final da linha.

7. Glossário

A seguir, segue a definição de alguns termos dos trabalhos mencionados nos Usos Conhecidos:

- ***Domain-Specific Modeling (DSM):*** Paradigma de modelagem focado no desenvolvimento de metalinguagens e metamodelos mais específicos para um determinado domínio de aplicação.
- ***Model-To-Text* (M2T):** Método de transformação de modelos no qual um código-fonte é gerado a partir de um modelo. Também conhecido como M2C (*Model-To-Code*).
- ***Plataform Independent Model (PIM):*** Seguindo a abordagem da MDA, é o modelo que está em um nível de abstração que envolve elementos que ainda independem da plataforma de implementação. [Lucrédio, 2009].

- ***Plataform-Specific Model (PSM):*** Seguindo a abordagem da MDA, é o modelo que está em um nível de abstração no qual já são envolvidos elementos específicos que particularizam uma determinada plataforma de implementação. [Lucrédio, 2009].
- ***Rich Internet Applications (RIA):*** São aplicações Web que são executadas diretamente no browser e que não necessitam de instalação prévia.

Agradecimentos

Agradecemos a Rute Castro e a Filipe Correia por terem agido como *shepherd* durante o processo de escrita deste padrão.

Referências

Araujo, R. B. (2003). *Computação Ubíqua: Princípios, Tecnologias e Desafios*. Em: Simpósio Brasileiro de Redes de Computadores. (Org.). Computação Ubíqua: Princípios, Tecnologias e Desafios. 1ª ed. pg. 45-115. Natal, RN. Disponível em: < http://www.professordiovani.com.br/rw/monografia_araujo.pdf>. Acesso em: 26 de janeiro de 2013.

Cetina, C.; Fons, J.; Pelechano, V.; (2008) "Applying Software Product Lines to Build Autonomic Pervasive Systems". Software Product Line Conference. SPLC '08. 12th International , vol., no., pp.117-126, 8-12 Setembro, 2008

Cirilo, C. E.; Prado, A. F.; Souza, W. (2010) "Model Driven RichUbi - A Model Driven Process for Building Rich Interfaces of Context-Sensitive Ubiquitous Applications". Special Interest Group on Design of Communication, SIGDOC.

Kuusela, J. (2012). "How variation changes when an embedded product ceases to be embedded?". Em: Proceedings of the WICSA/ECSA 2012 Companion Volume (WICSA/ECSA '12). ACM, New York, NY, USA, 147-150.

Lima, F. F. P.; (2011) "SysSu - Um Sistema de Suporte para Computação Ubíqua". Dissertação de Mestrado, Departamento de Computação, Universidade Federal do Ceará, Fortaleza, CE.

Lucrédio, D.; (2009) "Uma Abordagem Orientada a Modelos para Reutilização de Software". Tese de Doutorado, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, SP.

Northrop, L.; (2002) "SEI´s Software Product Line Tenets". IEEE Software.

Oliveira, R. P.; Prado, A. F.; Souza, W.; Biajiz, M. (2009) "Development based on MDA, of Ubiquitous Applications Domain Product Lines". IEEE/ACIS International Conference on Computer and Information Science.

Pohl, K.; Böckle, G.; Linden, F.J. van der; (2005) "Software Product Line Engineering: Foundations, Principles and Techniques". Ed. Springer-Verlag New York, Inc.

Quinton, C., Mosser, S., Parra, C., e Duchien L. (2011). "Using multiple feature models to design applications for mobile phones". Em: Proceedings of the 15th International Software Product Line

Conference, Volume 2 (SPLC '11). ACM, New York, NY, USA.

Santana, E. F. Z.; Oliveira, R. P.; Prado, A. F.; Souza, W.; Biajiz, M. (2009) “Modelagem Específica de Domínio em Linhas de Produto de Software na Computação Ubíqua”. Simpósio Brasileiro de Componentes, Arquiteturas e Reutilização de Software, SBCARS.

Schmidt, D. C.. (2006). “Guest Editor's Introduction: Model-Driven Engineering”. Computer vol.39, no.2, pg. 25- 31, Fev. 2006.

Serral, E.; Valderas, P.; Pelechano, V. (2008) “A Model Driven Development Method for Developing Context-Aware Pervasive Systems”. Proceedings of the 5th international conference on Ubiquitous Intelligence and Computing (UIC '08), pg. 662-676. Ed. Springer-Verlag, Berlin, Heidelberg.

Weiser, M. (1991), “The Computer for the 21st Century”. Scientific American, vol.265, no.3, Setembro., pp.94-104

Yau, S.S., Yu, W., Karim, F., (2002). “Development of situation-aware application software for ubiquitous computing environments”. Proceedings of the 26th Annual International Computer Software and Applications Conference. COMPSAC 2002. p.p. 233 – 238