Transforms: Um Ambiente de Apoio a Modelagem e Execução de Processos de Software Dirigido por Modelos

Bruno C. da Silva[1,2], Ana Patrícia F. Magalhães[2], Rita Suzana P. Maciel[3], Narciso Martins[2], Leandro Nogueira[2] e João C. Queiroz[2]

[1]Instituto de Informática – UFRGS – Porto Alegre / RS, Brasil

[2]Faculdade Ruy Barbosa – Salvador / BA, Brasil

[3]Universidade Federal da Bahia (UFBA) – Salvador / BA, Brasil

{brunocarreiro,anapatriciamagalhaes,ritasuzana}@gmail.com

Abstract. *This paper presents the Transforms tool which provides a supporting environment for process modeling and enactment according to MDA. Using the Transforms tool it is possible to define traditional software process activities and artifacts as well the necessary transformation chain of an MDA process.*

Resumo. *Este artigo apresenta a ferramenta Transforms que provê um ambiente de apoio a modelagem e execução de processos de software de acordo com a MDA. Com a Transforms é possível definir atividades e artefatos inerentes a um processo de software tradicional bem como a cadeia de transformações exigida em um processo MDA.*

1. Introdução

Um processo de software pode ser visto como um conjunto atividades e resultados associados que conduzem a um produto de software [Sommerville 2006]. Organizações definem seus processos de software para guiar seus desenvolvedores na execução de tais atividades. No entanto, a falta de uma terminologia consistente e padronizada para especificação de processos de software e a existência de notações distintas incluindo linguagem natural afeta negativamente a compreensão e comunicação do processo entre as partes envolvidas. Nesse contexto, a evolução do processo torna-se mais difícil e a execução do mesmo ainda mais distante da especificação. Consequentemente, a descrição do processo se torna obsoleta e custosa para manter. Sendo assim, pode prejudicar a produtividade do time e a qualidade do produto de software [Fuggeta 2000; Sommerville 2006].

Diversas linguagens foram propostas especialmente para modelagem de processos (PML – *Process Modeling Languages*), tais como APEL [Dami 1998], PROMENADE [Franch 1999], E^3 [Jaccheri 1999], SPEM (*Software and Systems Process Engineering Metamodel*) [OMG 2008], entre outras. O SPEM é um padrão do OMG (*Object Management Group*) que define um metamodelo baseado no MOF (*Meta Object Facility*) e um perfil UML para especificação de processos de software.

Outra proposta que foca na produtividade e qualidade do processo de desenvolvimento de software é a *Model-Driven Architecture* (MDA) [OMG 2003].

Diferentemente de modelos de processos de desenvolvimento tradicionais (por exemplo, RUP – *Rational Unified Process*), um processo MDA requer a seleção de metamodelos e regras de mapeamento para montagem da cadeia de transformação que produz modelos e código [Maciel 2009]. Entretanto, ferramentas MDA existentes atualmente estão particularmente focadas na definição e execução de transformações para produzirem código e artefatos de implantação a partir de modelos (AndroMDA[1], oAW[2], entre outras). Assim, demais atividades do processo de software não são consideradas, tais como controle e acompanhamento de tarefas, atribuição de papéis ao time, criação/visualização de artefatos de todo tipo (não somente modelos), entre outras. Além disso, em vários momentos, modelos precisam ser manipulados manualmente (ou semi-automaticamente) como, por exemplo, marcar diagramas antes de aplicar uma transformação, aplicar estereótipos etc. Assim, o acompanhamento sistemático de definições do processo, além da cadeia de transformação, torna-se prejudicado em um processo MDA.

Este trabalho apresenta a ferramenta *Transforms* que tem como principal objetivo a modelagem e apoio à execução de processos de software MDA. Inicialmente a *Transforms* foi desenvolvida como uma ferramenta para transformação de modelos a fim de apoiar um processo de desenvolvimento MDA específico [Silva 2006]. Atualmente, a *Transforms* vai além de uma ferramenta de transformação, oferecendo dois ambientes: um para edição de processos conforme os padrões SPEM 2.0 e MDA; e o outro para execução de processos especificados pelo editor. A seção 2 descreve esses dois ambientes em detalhe. A seção 3 apresenta as considerações finais e os trabalhos futuros.

2. Ambientes *Transforms*

A *Transforms* foi desenvolvida levando em consideração o metamodelo conceitual SPEM/MDA definido em [Maciel 2009]. Este metamodelo baseia-se em conceitos do SPEM 2.0 e faz uso de especializações para atender o contexto da MDA. Tal extensão foi necessária para permitir a definição de elementos e relações específicas de um processo MDA, como suas fases (CIM, PIM, PSM), metamodelos e regras de transformação.

A ferramenta *Transforms* possui dois módulos com recursos complementares representados pelos componentes *EditorDeProcessos* e *ExecutorDeProcessos* no diagrama ilustrado na figura 1. O primeiro módulo é o de edição de processos que compreende tanto a modelagem visual (*EditoresDiagramas*) quanto a sua definição estrutural e hierárquica (*EditoresEstruturaHierarquica*). O segundo módulo oferece o ambiente para execução acompanhada de processos definidos previamente pelo editor. Os processos ficam armazenados em um repositório (um banco de dados relacional) e são acessíveis através da *InterfaceDeAcessoADados* provida pelo componente *ControleDePersistência*. O subcomponente do executor - *ProjectExplorer* - é o responsável pela acompanhamento e execução de um projeto seguindo um processo definido pelo editor e recuperado do repositório. O subcomponente *ProfileApplication* realiza a aplicação automática de perfil UML nos diagramas editados pelo ambiente da *Transforms*.

[1] AndroMDA.org Home - http://www.andromda.org/

[2] Official openArchitectureWare Home page - http://www.openarchitectureware.org/

Os módulos foram projetados como um produto RCP (*Rich Client Platform*) sobre a plataforma do Eclipse. Dessa maneira, foi possível reusar o seu arcabouço gráfico assim como diversos *plugins* disponíveis para ele, tais como: ATL (*Atlas Transformation Language*), para edição e execução de transformações modelo-a-modelo; Mofscript, para edição e execução de transformações modelo-a-texto; GMF (*Graphical Modeling Framework*), para criação e geração dos editores gráficos dos diagramas customizados conforme o metamodelo conceitual SPEM/MDA; e UML2Tools, para modelagem de artefatos UML integrada ao executor.

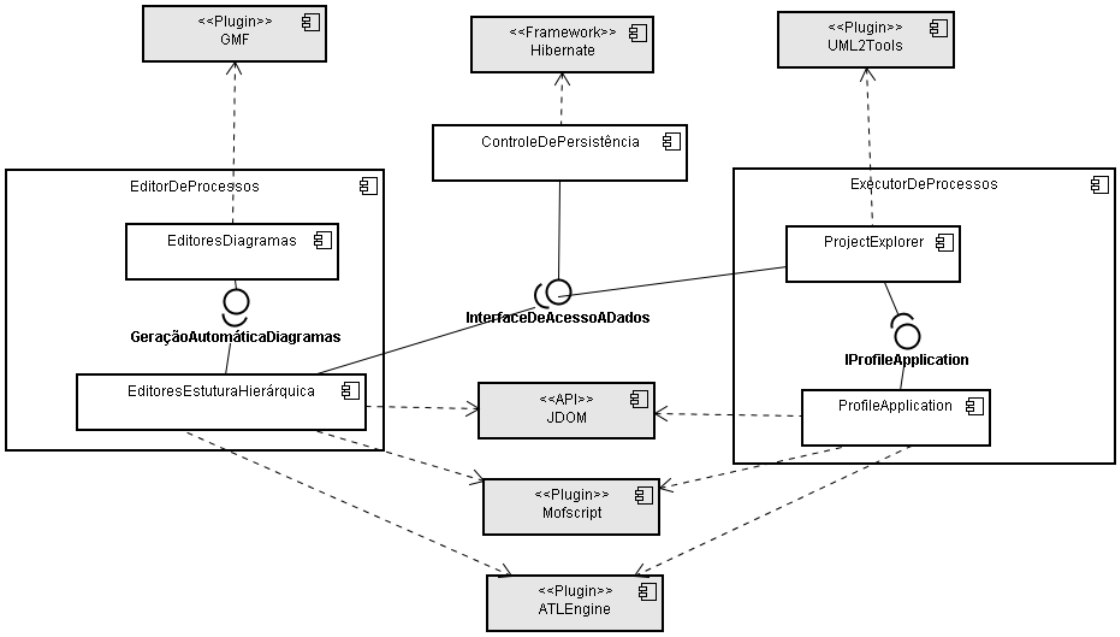

Figura 1 – Arquitetura geral da ferramenta *Transforms*

O uso do *plugin* UML2Tools oferece um recurso integrado ao ambiente do Transforms, embora não seja restrito a ele. A utilização de engenhos de transformação como os do ATL e Mofscript permite que modelos criados em outras ferramentas também possam ser usados em processos na *Transforms*, provendo uma interoperabilidade com outras ferramentas de modelagem utilizando o formato XMI (*XML Metadata Interchange*).

2.1 Editor de Processos MDA

O Editor possibilita a especificação de processos MDA compreendendo a montagem da cadeia de transformações com suas respectivas regras, incluindo a definição da seqüência de tarefas a serem executadas e os artefatos envolvidos em cada etapa do processo, bem como a atribuição de papéis para o time desenvolvedor.

Este módulo contempla editores customizados para: construção de diagramas de classe, designado à definição da estrutura do processo; criação de diagramas de casos de uso, destinado à definição de responsabilidades; elaboração de diagramas de atividades, quando necessário for a especificação do fluxo de realização de tarefas do processo. O ambiente de modelagem destes diagramas disponibiliza paletas montadas de acordo com as definições do metamodelo SPEM/MDA. Por exemplo, no diagrama de classes é possível criar classes que representem Fases do processo MDA tais como CIM, PIM,

PSM ou Código, associar perfis UML às fases, definir tarefas, artefatos de entrada e saída. Analogamente, é possível fazer a especificação textualmente e organizá-la em uma estrutura hierárquica também de acordo com o metamodelo SPEM/MDA. Neste caso, os diagramas são automaticamente gerados a partir da estrutura hierárquica.

A figura 2 ilustra uma tela do editor de processos MDA dividida em quatro regiões. A região (A) apresenta uma estrutura hierárquica com os elementos que compõem o processo em edição (*tasks*, *roles*, *workproducts*, etc.). A região (B) corresponde à área de modelagem visual do processo via diagramas UML, onde demonstra-se um diagrama de classes com definições estruturais da biblioteca de elementos do processo. Na região (C), encontra-se a paleta de opções para o diagrama em edição, que é apresentada de acordo com o metamodelo SPEM/MDA. A região (D) possibilita editar propriedades de um elemento selecionado no diagrama. No exemplo da figura 2, observa-se a disciplina *Levantamento de Dados* composta por quatro tarefas: *Planejar Projetos*, *Levantar Requisitos*, *Construir Protótipo* e *Transformar CIM-PIM*. Esta última tarefa é de responsabilidade do papel *Analista de Negócio* e tem como entrada os artefatos UML *Modelo Conceitual* e *Requisitos CIM*, além da regra de transformação *CIM para PIM* gerando como saída o artefato UML *Modelo PIM*. Por sua vez, a regra *CIM para PIM* (vide estereótipo *TransformationRule*), como sendo uma transformação modelo-a-modelo, pode ser especificada pelo próprio ambiente do Editor de processos através da integração com o plug-in do ATL. Quando as transformações forem modelo-a-texto, há disponível também a integração com o plug-in do Mofcript para criação deste tipo de transformação.

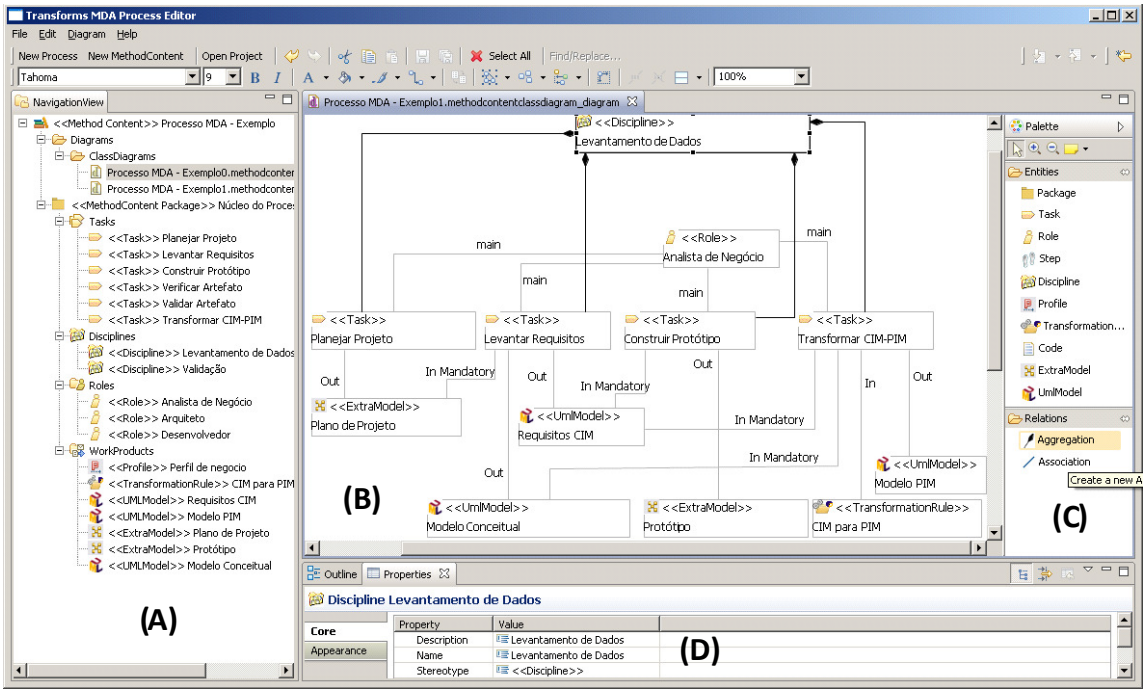

Figura 2 – Editor de processos MDA

2.2 Executor de Processos MDA

O executor de processo provê um ambiente de apoio ao desenvolvimento de software na abordagem MDA seguindo um processo especificado no módulo Editor. Desta forma, o

desenvolvimento de um sistema é iniciado com a seleção de um processo dentre os disponíveis no repositório mantido pelo módulo editor.

A figura 3 ilustra o Executor dividido em três regiões. A região (A) apresenta a estrutura do processo utilizada como base para o desenvolvimento de um projeto de software. Nesta área, o gerente atribui papéis para pessoas da equipe de desenvolvimento e o Executor realiza o controle de acesso para execução de tarefas associadas a cada papel. O processo aberto na figura 3 foi modelado no Editor ilustrado na figura 2. Este processo contém tanto tarefas manuais, como a construção de modelos UML demonstrado na região (B), quanto automáticas, como a execução de uma transformação ATL destacada na região (C). Os perfis UML definidos no processo são aplicados automaticamente pelo Executor à medida que os diagramas são construídos.

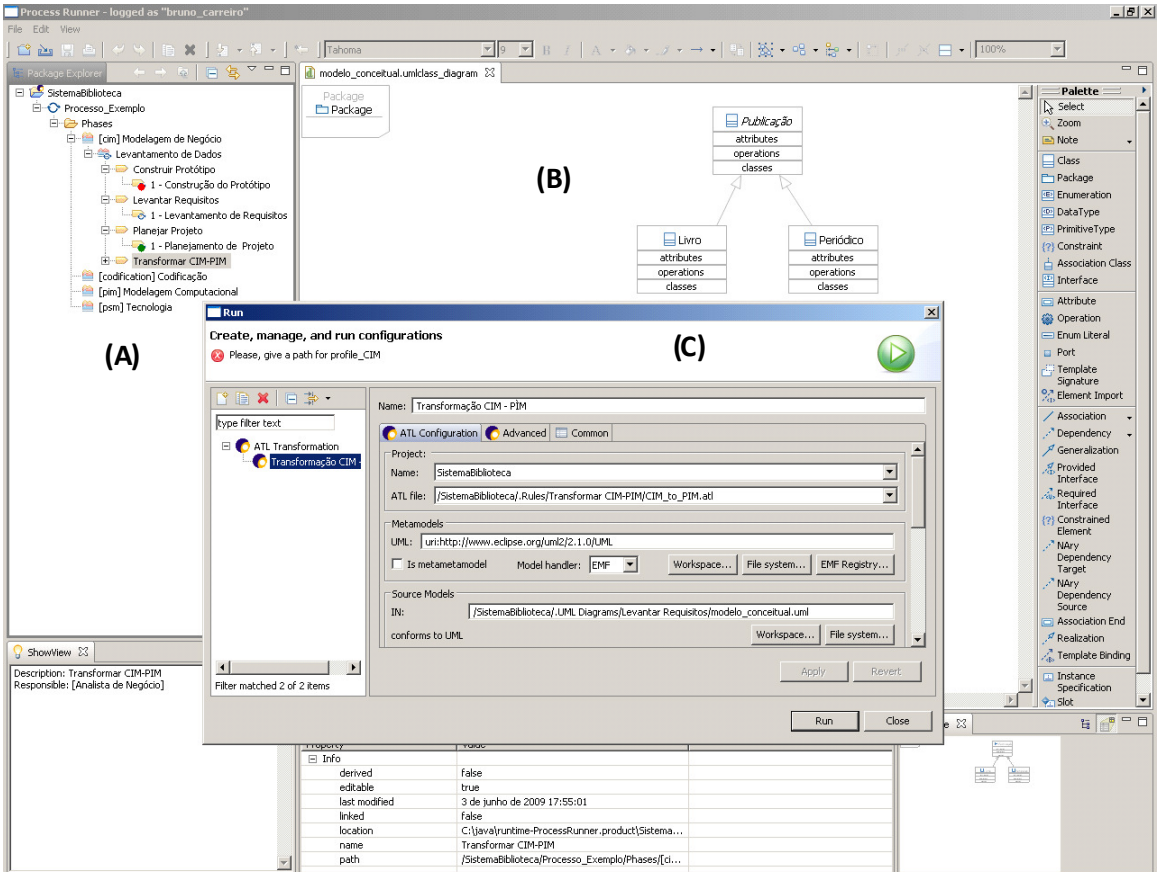

Figura 3. Executor de processos MDA

3. Considerações Finais e Trabalhos Futuros

Dois estudos de caso foram realizados a fim de verificar a aplicabilidade da estratégia de apoio adotada pela ferramenta *Transforms*. No primeiro estudo, a ferramenta foi utilizada para modelar e executar um processo para desenvolvimento de serviços específicos de *middleware* [Maciel 2009]. O segundo foi conduzido em conjunto com a Companhia de Processamento de Dados do Estado da Bahia (PRODEB) e envolveu a modelagem do processo MDA PRODEB para desenvolvimento de aplicações web. Através destes dois estudos de caso foi possível verificar que a *Transforms* oferece um meio de definir explicitamente elementos de processo MDA (CIM, PIM, PSM, metamodelos e regras de transformações), além de definições

tradicionais de processos de software tais como atividades, papéis, artefatos, fluxo de trabalho, entre outras. Além disso, foi possível também apoiar a execução de tais definições de processo através do Executor. Apesar da maioria das ferramentas MDA possuírem um foco direcionado à execução da cadeia de transformações de modelos, ainda assim oferecem pouca flexibilidade na customização da mesma. A ferramenta *Transforms* torna possível que os elementos de um processo MDA sejam definidos explicitamente como parte do processo de software o qual estão inseridos. Adicionalmente, permite-se que a cadeia de transformação seja customizada de acordo com as necessidades da organização em cada contexto específico.

Como trabalhos futuros estão sendo previstas novas funcionalidades para os módulos de edição e execução de processos como, por exemplo: aperfeiçoar requisitos de usabilidade dos dois ambientes; implementar controle de versão dos artefatos do processo; e possibilitar a geração de testes dirigido por modelos. A ferramenta *Transforms* encontra-se disponível livremente para *download* em http://transforms.frb.br. Neste endereço, é possível encontrar também instruções de instalação e uso.

Agradecimentos

Trabalho financiado pela Fundação de Amparo a Pesquisa do Estado da Bahia (Fapesb) sob o projeto n° 8694/2006. Agradecemos também aos profissionais da PRODEB e aos demais colaboradores que participaram do projeto: Alabê Duarte e Alliam Buarque.

Referências

Dami, S.; Estublier, J. and Amiour, M. (1998) APEL: a Graphical Yet Executable Formalism for Process Modeling. *Automated Soft. Eng*. Vol 5, January, pp. 61-96.

Fuggetta, A. (2000). "Software Process: A Roadmap". In: Proceedings of the Intl. Conference on the Future of Software Engineering. ACM, New York, NY, p. 25-34.

Franch, X. and Ribó, J. M. (1999). "Using UML for Modelling the Static Part of a Software Process". In: UML99 - Beyond the Standard Second International Conference, Fort Collins CO, USA.

Jaccheri, M. L.; Baldi M. and Divitini M. (1999). "Evaluating the requirements of software process modeling languages and systems". In: Process Support for Distributed Team-based Software Development. In: *PDTSD99*, Orlando, Florida, pages 570-578, August.

Maciel, R; Silva, B.; Magalhães, A. P. and Rosa, N. (2009). "An Approach to Model-Driven Development Process Specification". In: 11th Intl. Conference on Enterprise Information Systems, Milan, Italy, May.

OMG (2003). MDA Guide. Version 1.0.1 (omg/2003-06-01).

OMG (2008). Software Process Engineering Metamodel Specification, Version 2.0, (formal/08-04-01).

Silva, B.; Maciel, R. and Mascarenhas, L. (2006). "Transforms: Uma Ferramenta MDA/EDOC para Desenvolvimento de Serviços Específicos de Middleware". In: Brazilian Symp. on Software Engineering – Tools session. Florianópolis. p. 19-24.

Sommerville, I. (2006). Software Engineering. 8th edition, Pearson Education.