A Summary of Toolboxes Scoping MDWE Front-Ends

Jean T. Piagetti, Maurício El Uri, Fábio Paulo Basso

[1]Universidade Federal do Pampa
(UNIPAMPA) – Alegrete, RS – Brazil

{mauricioeluri,jeanpiagetti}@gmail.com, fabiobasso@unipampa.edu.br

Abstract. *This paper characterizes 27 studies, published between 2005 and 2017, scoping the development of web front-ends through toolboxes for Model-Driven Web Engineering (MDWE). It addresses Domain Specific Languages (DSLs), embedded with some design techniques, devoted to help in the various phases of software development. Moreover, it also contextualizes model-based tools for rapid prototyping, with evolutive prototyping and source code generation, and highligths those that speed-up the analysis process through automated design. Our conclusion is that only six of these proposals have been reported to be used in industry, thus a limitation found in the literature of the area that hinders the understanding of the acceptance criteria of these tools in contexts of software factories.*

Resumo. *Este artigo caracteriza 27 estudos, publicados entre 2005 e 2017, endereçando o desenvolvimento de web front-ends por meio de ferramentas para Model-Driven Web Engineering (MDWE). Ele aborda Linguagens Específicas de Domínio (DSLs), com o uso de técnicas de design, propostas para auxiliar em muitas fases de desenvolvimento de software. Também aborda quais destas ferramentas são caracterizadas para prototipagem rápida de aplicações, com prototipação evolutiva e geração de código fonte, que tem como objetivo acelerar o processo de de análise por meio de design automático. Nossa conclusão é que apenas seis destas propostas têm sido relatadas com aplicação na indústria, assim uma limitação encontrada na literatura da área que atrapalha a compreensão dos critérios de aceitação destas ferramentas em contextos de fábricas de software.*

1. Introdução

Temos visto cada vez mais esforços para tornar o conteúdo presente na internet acessível aos usuários finais. Porém, mesmo com todos estes esforços, poucos programadores e equipes de desenvolvimento realmente se preocupam em tornar seus websites acessíveis em múltiplas plataformas [Brambilla and Fraternali 2014, Basso et al. 2017]. Por exemplo, além do conhecimento técnico necessário para desenvolvimento de sistemas de informação [Basso et al. 2016], desafios para incorporar em aplicativos as novidades do mercado incluem, principalmente, a dificuldade das equipes de desenvolvimento de cumprir com o cronograma em iterações de um processo de desenvolvimento cada vez mais curtas [Basso et al. 2015]. Por conta disso, importantes características de um sistema web como acessibilidade, facilidade de uso e navegabilidade acabam ficando de lado em prol de cumprir com o escasso tempo de mercado.

Neste sentido, ferramentas para Prototipação Rápida de Aplicações (RAP) [Elkoutbi et al. 2006, Planas et al. 2009] podem contribuir para superar este desafio. Algumas destas ferramentas permitem, por meio de modelos, a geração de sistemas conforme múltiplas plataformas [Basso et al. 2016]. Para tanto, Model-Driven Engineering (MDE) [Kent 2002] é um paradigma para o desenvolvimento de software baseado em transformações de modelos, e vem sendo implementado com algumas técnicas de design que contribuem para a RAP.

Em processos típicos baseados em MDE, as transformações de modelo devem receber um modelo altamente detalhado para gerar partes funcionais de aplicativos [Schmidt 2006]. Para gerar o código-fonte completo [Kelly and Tolvanen 2008], várias partes de um design de aplicativo são detalhadas em DSLs (Domain Specific Languages) [Voelter 2009] e/ ou decoradas com anotações adicionadas aos elementos de modelo representados com a UML (Unified Modeling Language) [Elkoutbi et al. 2006], uma linguagem de modelagem de propósito geral usada em projetos arquiteturais de alguns sistemas web. Em qualquer caso, isso torna a construção do software dependente de tarefas de design.

Neste contexto, nosso objetivo é descrever 13 anos de pesquisa no design de front-end web focando em Model-Driven Web-Engineering (MDWE). MDWE inclui desde abordagens para arquitetura de redes de computadores ao desenvolvimento de aplicativos web [Brambilla and Fraternali 2014]. Assim, focamos este estudo nas DSLs propostas para design de front-ends web e nas abordagens que automatizam o design por meio de transformações de modelos [Basso et al. 2017]. Para tal, um protocolo de pesquisa foi elaborado, e leva em consideração os aspectos que englobam o conjunto dos elementos do MDWE. No entanto, por motivos de espaço, o protocolo do estudo foi omitido deste relato.

A seguir apresenta-se o resultados do nosso mapeamento. Inicialmente, a Seção 2 apresenta um embasamento teórico. A Seção 3 apresenta nossos achados deste mapeamento. Finalmente, a Seção 4 apresenta nossas conclusões e trabalhos futuros.

2. Embasamento Teórico

No desenvolvimento de sistemas de informações da web, os front-ends da web como o layout são compostos pelos componentes para GUI (Graphical User Interface) e por diagramas comportamentais [Nunes and Schwabe 2006]. Para permitir a geração de código-fonte completo com uma abordagem MDWE, esses modelos são decorados com a semântica para ações do usuário, fluxos de tela e lógica de negócios. Assim, usando-se uma DSL apropriada, é possível abstrair detalhes de implementação, focando na especificação de semântica em modelos que formalizam o conhecimento sobre requisitos de software.

O MDWE permite o desenvolvimento de sistemas de informação para múltiplas plataformas por meio de transformações de modelos [Basso et al. 2016]. Outro emprego comum, por exemplo, é no desenvolvimento de serviços que integram múltiplos sistemas (mashups), usando uma arquitetura orientada a serviço (SOA) e também micro-serviços [de Vrieze et al. 2011]. Por fim, outro bom exemplo está na automação de processos de negócios, realizada por meio de modelos representados com BPMN [Pillat et al. 2015].

Uma ampla variedade de abordagens para MDWE foram propostas como produtos derivados de pesquisa desde 2000 [Ceri et al. 2000]. Atualmente, a falta de um mapeamento das propostas existentes gera uma incerteza sobre o que cada ferramenta oferece para a indústria de software, bem como quais são as lacunas de pesquisa. Assim, este estudo apresenta um mapeamento de propostas para MDWE que introduzem elementos representacionais para o design de GUIs que levam à geração de protótipos. Este mapeamento é novo, já que estudos existentes focam em plataformas específicas como AJAX [Mesbah and Van Deursen 2008], com análise de propostas e desafios sobre frameworks mais utilizados. Diferentemente, nosso estudo foca em tecnologia existente para gerar código específico de plataforma por meio de modelos independentes delas.

3. O Estado da Arte em Design de Web Front-Ends

A Tabela 1 apresenta os artigos encontrados em ordem cronológica. A seguir, descrevemos algumas propostas de design e prototipação consideradas nas abordagens para o MDWE publicadas entre 2013 e 2017.

3.1. Propostas Entre 2005 e 2012

Algumas propostas para o MDWE usam a UML como linguagem de especificação (S01, S03-S05) permitindo que Designers especifiquem modelos em várias camadas (também conhecido como *multiview design*). Este modelos representam componentes para Interfaces Gráficas de Usuário (GUI), fluxos, lógica de negócios, informações estruturais para bancos de dados. Exemplos de elementos projetados dentro de modelos UML são diagramas de casos de uso, diagramas de classes, diagramas de sequência e atividade, todos designados com tags de Perfis UML simples. Em S02 e S07 é apresentado o refinamento de modelos independentes de plataforma (como diagramas de classes anotados, casos de uso, diagramas de colaboração e diagramas de atividades) para específicos de tecnologias web. Além disso, é possível usar esses modelos para gerar sistemas de informações da Web de várias camadas e transformá-lo para abstrações de nível inferior que mapeiam elementos anteriores para o código-fonte.

Em termos representacionais, pode-se citar o estudo S06, que apresenta uma proposta de arquitetura para aplicações AJAX, chamada SPIAR. Esta arquitetura foi baseada na análise de vários frameworks AJAX e configurações levando em conta os problemas de design e limitações do mesmo. Detalhes como propriedades arquiteturais, interatividade com o usuário, latência percebia pelo usuário e desempenho da rede também foram levados em conta, assim como vários outros pontos referentes à qualidade do software.

Outra característica das propostas publicadas entre 2005 e 2012 é o foco na geração de código. Ou seja, nas pesquisas deste intervalo valorizava-se a geração de código por meio de modelos, sendo esta uma novidade. Por exemplo, tendo percebido que lojas que vendem produtos on-line geralmente precisam manter blogs para que seus consumidores possam discutir sobre os itens que fazem parte de seu catálogo, ou mesmo para manter uma facilitada comunicação com o consumidor, autores em S14 desenvolveram uma DSL para suprir esta demanda. Utilizando o Blojsom para geração dos blogs, esta DSL é composta por vários modelos, cada um com sua responsabilidade: estilização, conteúdo, navegação, etc. Apesar de ser capaz de gerar blogs inteiros, necessitando de poucos acabamentos pela equipe de desenvolvimento, esta DSL foi proposta em 2010, em

Tabela 1. Propostas para design e geração de front-end web no MDWE

Id	Título	Abordagem	RAP Approach
S01	Vanderdonckt, J. A mda-compliant environment for developing user interfaces of information systems. In 17th international conference on Advanced Information Systems Engineering	Manual Design	Mockup Model to Code
S02	Nunes, D. A. and Schwabe, D.Rapid prototyping of web applications combiningdomain specific languages and model driven design. In 6th international conferenceon Web engineering	Manual Design	Mockup Model to Code
S03	Elkoutbi, M., Khriss, I., and Keller, R. K. Automated prototyping of user interfaces based on uml scenarios. Automated Software Eng.	Manual Design	Use Case Model to Mockup Model
S04	Kavaldjian, S. A model-driven approach to generating user interfaces. In The 6th Joint Meeting on European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering: companion papers	Manual Design	Mockup Model to Code
S05	Souza, V. E. S., Falbo, R. D. A., and Guizzardi, G. A uml profile for modeling framework-based web information systems. In 12th International Workshop on Exploring Modelling Methods in Systems Analysis and Design EMMSAD2007	Manual Design	Multi-layered Model to Code
S06	Mesbah, A. and Van Deursen, A. A component-and push-based architectural style for ajax applications. Journal of Systems and Software	Manual Design	Runtime MDWE
S07	Melia, S., Gomez, J., Perez, S., and Diaz, O. A model-driven development for gwt-based rich internet applications with ooh4ria. In Web Engineering, 2008.	Manual Design	Mockup Model to Code
S08	Sadat-Mohtasham, S. H. and Ghorbani, A. A. A language for high-level description of adaptive web systems. Journal of Systems and Software	Automated Design	Multi-Layered Model to Code
S09	Yang, F., Gupta, N., Botev, C., Churchill, E., Levchenko, G., and Shanmugasundaram, J. Wysiwyg development of data driven web applications. Proc. VLDB Endowment.	Manual Design	Mockup Design
S10	Ginzburg, J., Rossi, D. D. G., and Urbieta, M. Oblivious integration of volatile functionality in web application interfaces. Journal of Web Engineering	Manual Design	Mockup to Code
S11	Chavarriaga, E. and Macías, J A. A model-driven approach to building modern semantic web-based user interfaces. Advances in Engineering Software	Manual Design	Mockup to Code
S12	De Vrieze, P., Xu, L., Bouguettaya, A., Yang, J., and Chen, J. Building enterprise mashups. Future Generation Computer Systems	Manual Design	Multi-Layered Model to Code
S13	Aquino, N., Vanderdonckt, J., and Pastor, O. Transformation templates: adding flexibility to model-driven engineering of user interfaces. In 2010 ACM Symposium on Applied Computing	Manual Design	Mockup to Code
S14	Díaz, O. and Villoria, F. M. Generating blogs out of product catalogues: An mde approach. Journal of Systems and Software	Manual Design	Mockup to Code
S15	Busch, M., Koch, N., Masi, M., Pugliese, R., and Tiezzi, F. Towards model-driven development of access control policies for web applications. In Workshop on Model-Driven Security	Manual Design	Mockup to Code
S16	Vara, J. M. and Esperanza, M. A framework for model-driven development of information systems: Technical decisions and lessons learned. Journal of Systems and Software	Manual Design	Multi-layered Model to Code
S17	Rivero, J. M., Grigera, J., Rossi, G., Luna, E. R., and Koch, N. Towards agile model-driven web engineering. In IS Olympics: Information Systems in a Diverse World	Manual Design	Mockup to Code
S18	Molina, A. I., Giraldo, W. J., Gallardo, J., Redondo, M. A., Ortega, M., and Garc. Ciat-gui: A mde-compliant environment for developing graphical user interfaces of information systems. Advances in Engineering Software	Manual Design	Multi-Layered Model to Code
S19	Grigera, J., Rivero, J., Luna, E. R., Giacosa, F., and Rossi, G. From requirements to web applications in an agile model-driven approach. In Web Engineering	Manual Design	Mockup to Code
S20	Deufemia, V., D'Souza, C., and Ginige, A. Visually modelling data intensive web applications to assist end-user development. In 6th International Symposium on Visual Information Communication and Interaction	Automated Design	Runtime MDWE
S21	Batory, D., Latimer, E., and Azanza, M. (2013). Teaching model driven engineering from a relational database perspective. In 16th International Conferenceon Model Driven Engineering Languages and Systems	Manual Design	Physical Model to Mockup to Code
S22	Basso, F. P., Pillat, R. M., Frantz, R. Z., and Rooz-Frantz, F. Assisted tasks to generate pre-prototypes for web information systems. In 16th International Conferenceon Enterprise Information Systems.	Automated Design	Domain Model to Mockup to Multi-Layered MVC to Code
S23	Brambilla, M. and Fraternali, P. Large-scale model-driven engineering of web user interaction: The webml and webratio experience. Science of Computer Programming	Manual Design	Multi-layered Model to Code
S24	Basso, F. P., Pillat, R. M., Roos-Frantz, F., and Frantz, R. Z. Combining mde and scrum on the rapid prototyping of web information systems. International Journal of Web Engineering and Technology	Automated Design	Domain Model to Mockup to Multi-Layered MVC to Code
S25	Antonelli, H. L., da Silva, E. A. N., and Fortes, R. P. M. (2015). A model-driven development for creating accessible web menus. Procedia Computer Science	Automated Design	Mockup to Code
S26	Basso, F. P., Pillat, R. M., Oliveira, T. C., Roos-Frantz, F., and Frantz, R. Z. Automated design of multi-layered web information systems. Journal of Systems and Software	Automated Design	Domain Model to Mockup to Multi-Layered MVC to Code
S27	Chavarriaga, E., Jurado, F., and Díez, F. (2017). An approach to build xml-based domain specific languages solutions for client-side web applications. Computer Languages, Systems & Structures	Automated Design	Mockup to Code

um momento que a tecnologia de blogs ainda era imatura, porém Blojsom já está defasado, o que acaba dificultando a utilização da DSL pela comunidade atualmente. Portanto, para terem longevidade, DSLs devem ser construídas de modo agnóstico às plataformas.

3.2. Prototipação Rápida de Aplicações

Trabalhos S02, S05, S13, S14, S17, S22-S24 e S26 permitem especificar a semântica para lógica, habilitando a geração de código fonte completo para todas as camadas de aplicações. Essas contribuições são importantes porque geram protótipos funcionais de sistemas de informações da web que são testáveis pelo desenvolvedor.

Nesse sentido, estudos S17 e S25 afirmam que não é possível garantir que os protótipos gerados através da UML atendam às necessidades do cliente. Eles sugerem o desenvolvimento orientado a mockup, com transformações iniciadas a partir de mockups como o ponto chave para melhorar o feedback do cliente em fases preliminares de um

processo de software. Uma vez que se verifica a aceitação de um determinado protótipo funcional, isto dá uma confiança maior do que validar requisitos usando protótipos de papel. Isto não significa que o protótipo apresentado é o produto final de uma Sprint. Logo, o mesmo deve ser refinado ou em código ou em modelo ao longo da execução de uma Sprint.

Apesar de muitas propostas permitirem a geração de protótipos funcionais por meio de modelos, poucas são adequadas para a prototipação rápida de aplicações. Então há uma diferença entre ferramentas aptas à prototipação e outras aptas para a prototipação rápida. Por exemplo, no estudo S17 os autores anotam manualmente os mockups enquanto propostas recentes usam técnicas de design automatizadas para acelerar a produção de modelos. Da mesma forma, o estudo S11 propôs a geração de front-ends da web em modelos usando algumas técnicas de design automatizado, mas como front-ends da web e não são anotados com a semântica que permite a geração de protótipos totalmente implementados. O estudo S20 também gera modelos MVC através de mockups anotados. Apesar de ser caracterizado como rápido, os modelos são projetados em uma abordagem descartável do MDWE (chamada de throwaway prototyping), e portanto não usado em todo o processo de desenvolvimento de software por meio de refinamentos.

A seguir, apresenta-se de que modo os trabalhos mais recentes resolvem esta limitação.

3.3. Propostas Recentes

Essas propostas publicadas entre 2005 e 2008 aplicam uma abordagem simples de design. Ou seja, elas repassam a responsabilidade para detalhamento do modelo para o usuário final. Adotando uma abordagem de detalhamento de cima para baixo (detalha-se um modelo abstrato e transforma-se este num modelo específico de plataforma), é possível modelar semântica de ações em componentes GUI de sistemas de informação na web. Por exemplo, modelos conceituais são altamente detalhados com semântica de ações antes de gerar um protótipo testável. A mesma abordagem manual para refinamento de modelos também é encontrada em abordagens mais recentes como S17 e S21.

Tal abordagem é ruim para execução em metodologias ágeis. Assim, somente quando o modelo de entrada é validado para transformações, é possível transformá-lo em código-fonte. Propostas mais recentes pós 2007 são tentativas de usar as práticas que requerem tetes mais imediatos, como técnicas de design automatizado (S08, S18, S25) e combinações entre MDWE e Agile (S17, S19, S22 e S24). Estas últimas sugerirem que o Engenheiro comece gerando Mockups (modelos de GUI anotados projetados com um DSM em vez de UML).

O estudo S08, publicado em 2008, apresenta um exemplo de suporte ferramental para apoio ao design automatizado de GUI. Trata-se de uma proposta para sistemas adaptativos voltados para web numa abordagem interativa, que permite montar as páginas web na medida que o usuário vai navegando na aplicação. Isso é conhecido na literatura como prototipação evolutiva de software e permite ao desenvolvedor direcionar o contexto da aplicação conforme os requisitos são descobertos. Além disso, a DSL foi projetada com base na análise de sistemas web-adaptativos com ênfase em engenharia de software baseada em linhas de produto. Trata-se de uma contribuição que permite ao designer especificar um mockup e definir um contexto, através de um UML Profile, fazendo

o mapeamento da interação num contexto adaptativo.

Para visualizar e modificar protótipos gerados a partir do MDWE, o estudo S18 apresenta a ferramenta denominada CIAT-GUI. O diferencial do suporte ferramental é que isto permite testar modelos de sistemas de informação em diferentes níveis de abstração de projeto. Esses níveis exigem o uso de técnicas de design automatizadas, portanto auxiliando na execução de prototipação evolutiva do software.

A esse respeito, podemos destacar também uma recente pesquisa experimental S21. Com o objetivo de permitir que alunos não experientes automatizem o design de aplicativos web, os autores consideraram uma perspectiva diferente para o design: perspectiva do banco de dados em que os modelos são, de fato, tabelas e relacionamentos. Esta pesquisa mostrou que os alunos produzem melhores resultados se usando ferramentas simples (por exemplo, transformações de modelos codificados em Java). Isso permite que pessoas não experientes automatizem tarefas de design. Além disso, a pesquisa sugere que automatizar o design de protótipos é viável no contexto de fábricas de software, já que faltam especialistas no MDE.

O desenvolvimento de DSLs para modelagem de elementos específicos para acessibilidade ainda desperta o interesse na pesquisa. Por exemplo, o estudo S25 propõe uma DSL que utiliza os pontos de referência do WCAG 2.0 (Web Content Acessibility Guidelines). Chamada de AMenu - Acessible Menu, esta DSL é capaz de gerar toda a estrutura de menus acessíveis para a web, facilitando assim o refinamento do modelo e a manutenção da aplicação por meio de geração de protótipos. A DSL apresentada é muito simples, é textual e apresenta integração com muitas das IDE's utilizadas atualmente pelos desenvolvedores, inclusive o Eclipse.

Enquanto as demais ferramentas foram construídas utilizando Eclipse-EMF ou UML Profiles, o trabalho mais recente (S27) trata de uma DSL textual programada utilizando Javascript. Ela foi criada para o desenvolvimento de sistemas web, sendo dividida em vários modelos, onde cada um é responsável por determinada camada de aplicação. Baseada em XML, ela é capaz de gerar todo código HTML3, CSS3 e Javascript necessário, sendo a geração de Javascript um dos diferenciais da ferramenta. Portanto, exceto pela novidade ser o desenvolvimento da própria DSL, ela permite ao modelador o uso da mesma semântica de ações já introduzida pelos trabalhos anteriores.

3.4. Aplicação em Contextos da Indústria

Outro ponto que investigamos foi como estas tecnologias vem sendo utilizadas na indústria. Encontramos que, dentre 27 artigos, apenas seis das pesquisas apresentam estudos de viabilidade das ferramentas em contextos industriais: S09, S20, S22-S24, S26. Cabe ressaltar que os estudos 22-24 e 26 apresentam relatos de aplicação em mais de um contexto industrial. Ou seja, são relatados em projetos de software de vários domínios, cujas tecnologias utilizadas para implementação também variam em termos de APIs, padrões arquiteturais e de codificação. Portanto, são poucos os casos em que o objetivo de se obter independência de plataformas é comprovadamente observado em projetos de software executados por meio de MDWE.

Essa observação é um tanto paradoxal para este tipo de pesquisa. Em geral, as abordagens de MDWE são motivadas sobre o guarda-chuva de desenvolvimento multiplataforma (independente delas), mas a maioria dos relatos (24) não corroboram com

sua comprovada efetividade na prática como uma solução deste problema. Um dos motivos por esta pequena amostra, i.e., de três relatos de contextos envolvendo desenvolvimento multi-plataforma, pode ser o fato de que artigos tratando de DSLs oferecem uma contribuição focada em termos de abstração, e não de prática. Para remover esta limitação, que nos impede de realizar uma conclusão definitiva sobre a efetividade das DSLs na prática, é preciso realizar uma segunda revisão do tipo snowballing. Por meio dela, espera-se encontrar os trabalhos dos mesmos autores dos 27 artigos relatados, mas que apresentem estudos observacionais complementares.

4. Conclusões e Trabalho Futuros

Este artigo apresentou um mapeamento de literatura em MDWE front-ends, caracterizando de forma resumida o total de 27 propostas. Por meio deste mapeamento, encontramos que a pesquisa dirigida para características estruturais de DSLs, como design de componentes que embutem semântica de ações neste contexto, é madura. Ou seja, em termos de representatividade, as DSLs existentes cobrem, se não todas, grande parte das necessidades para design de componentes de web front-ends das plataformas de desenvolvimento atuais. No entanto, as abordagens para design automatizado ainda estão no seu início. Portanto, pretende-se ampliar este estudo para os seguintes complementos:

1) Um consenso observado nos estudos é que DSLs para representação de front-ends web são essenciais para desenvolvimento rápido de aplicações em multi-plataformas. Além disso, DSLs construídas com base em design automático permitem capturar as necessidades do cliente nas fases preliminares de desenvolvimento de software, o que é importante para a execução da disciplina da Engenharia de Software denominada Verificação e Validação (V&V). Assim, em trabalhos futuros pretende-se explorar técnicas de V&V alinhadas com design automático, avaliando também se ferramentas de nosso grupo de pesquisa como a MockupToME [Basso et al. 2016], contribuem para a execução de atividades nessa disciplina;

2) A literatura carece de estudos de viabilidade do MDWE conduzidos em contextos de Fábricas de Software [Basso et al. 2017]. Ao longo de nossa investigação, percebeu-se que poucos trabalhos discutiram como a indústria recebeu estas tecnologias. Isso é um tanto decepcionante, já que tais pesquisas são motivadas em problemas da indústria de software. Assim, como seguimento dos estudos empíricos discutidos em [Basso et al. 2015], pretende-se estudar como diferentes atributos de qualidade, na transferência de tecnologia de MDWE, afetam a aceitação destes produtos no mercado, e;

3) Por fim, deve-se ampliar o relato deste mapeamento. Por motivos de formato do relato, e como forma de caracterização da área de pesquisa, pôde-se apenas conceituar as propostas existentes. Assim, ficou faltando elementos importantes de um mapeamento sistemático como o protocolo de pesquisa, questões de pesquisa e as suas respostas, bem como uma análise comparativa das características estruturais das 27 DSLs encontradas. Portanto, como próximo trabalho, pretende-se apresentar esta parte complementar de nosso estudo.

Referências

[Basso et al. 2017] Basso, F. P., Oliveira, T. C., Werner, C. M., and Becker, L. B. (2017). Building the foundations for 'mde as service'. *IET Software*, 11:195–206.

[Basso et al. 2016] Basso, F. P., Pillat, R. M., Oliveira, T. C., Roos-Frantz, F., and Frantz, R. Z. (2016). Automated design of multi-layered web information systems. *Journal of Systems and Software*, 117:612 – 637.

[Basso et al. 2015] Basso, F. P., Pillat, R. M., Roos-Frantz, F., and Frantz, R. Z. (2015). Combining mde and scrum on the rapid prototyping of web information systems. *International Journal of Web Engineering and Technology*, 10(3):214–244.

[Brambilla and Fraternali 2014] Brambilla, M. and Fraternali, P. (2014). Large-scale model-driven engineering of web user interaction: The webml and webratio experience. *Science of Computer Programming*, 89, Part B:71 – 87. Special issue on Success Stories in Model Driven Engineering.

[Ceri et al. 2000] Ceri, S., Fraternali, P., and Bongio, A. (2000). Web modeling language (webml): a modeling language for designing web sites. *Computer Networks*, 33(1-6):137–157.

[de Vrieze et al. 2011] de Vrieze, P., Xu, L., Bouguettaya, A., Yang, J., and Chen, J. (2011). Building enterprise mashups. *Future Generation Computer Systems*, 27(5):637 – 642.

[Elkoutbi et al. 2006] Elkoutbi, M., Khriss, I., and Keller, R. K. (2006). Automated prototyping of user interfaces based on uml scenarios. *Automated Software Engg.*, 13(1):5–40.

[Kelly and Tolvanen 2008] Kelly, S. and Tolvanen, J.-P. (2008). *Domain Specific Modeling: Enabling Full Code Generation*. IEEE Computer Society - John Wiley & Sons.

[Kent 2002] Kent, S. (2002). Model driven engineering. In *Integrated Formal Methods*, pages 286–298.

[Mesbah and Van Deursen 2008] Mesbah, A. and Van Deursen, A. (2008). A component- and push-based architectural style for ajax applications. *Journal of Systems and Software*, 81(12):2194–2209.

[Nunes and Schwabe 2006] Nunes, D. A. and Schwabe, D. (2006). Rapid prototyping of web applications combining domain specific languages and model driven design. In *6th international conference on Web engineering*, pages 153–160.

[Pillat et al. 2015] Pillat, R. M., Oliveira, T. C., Alencar, P. S., and Cowan, D. D. (2015). BPMNt: A BPMN extension for specifying software process tailoring. *Information and Software Technology*, 57(0):95 – 115.

[Planas et al. 2009] Planas, E., Cabot, J., and Gómez, C. (2009). Verifying action semantics specifications in uml behavioral models. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5565 LNCS:125–140.

[Schmidt 2006] Schmidt, D. C. (2006). Guest editor's introduction: Model-driven engineering. *IEEE Computer*, 39(2):25–31.

[Voelter 2009] Voelter, M. (2009). Best practices for dsls and model-driven development. *Journal of Object Technology*, 8(6):79–102.