Uma Proposta de Processo de Avaliação de Usabilidade para DSLs*

Ildevana Poltronieri Rodrigues[1], **Avelino Francisco Zorzo**[1], **Maicon Bernardino**[2]

[1] Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Av. Ipiranga, 6681, Partenon – Porto Alegre – RS – Brasil

[2] Universidade Federal do Pampa (UNIPAMPA)
Av. Tiarajú, 810, Ibirapuitã – Alegrete, RS – Brasil

ildevana@gmail.com, avelino.zorzo@pucrs.br, bernardino@acm.org

Abstract. *Several studies indicate the lack of well-defined methods to evaluate usability criteria in Domain-Specific Languages (DSL). Therefore, software engineers evaluate experimentally, even subjectively, whether a DSL is easy to use and easy to understand. In this context, this paper presents the proposal of an evaluation process based on the Usa-DSL framework, which was developed to evaluate the usability of DSLs. The purpose of the Usa-DSL process is to help DSL designers and developers to develop, execute, analyze and disseminate the results of assessing those languages, making this task easier and more intuitive.*

Resumo. *Diversos estudos indicam a inexistência de métodos bem definidos para avaliar critérios de usabilidade em linguagens específicas de domínio (Domain-Specific Languages - DSL). Desta forma, engenheiros de software avaliam de forma experimental, até mesmo subjetivamente, se uma DSL é fácil de ser utilizada e de fácil entendimento, por exemplo. Neste contexto, este trabalho apresenta a proposta de um processo de avaliação apoiado no framework Usa-DSL, o qual foi desenvolvido para avaliar a usabilidade de DSL. O Usa-DSL Process tem como principal objetivo auxiliar os projetistas e desenvolvedores de DSLs no desenvolvimento, execução, análise e divulgação dos resultados da avaliação de tais linguagens, tornando essa tarefa mais fácil e intuitiva.*

1. Introdução

A Engenharia de Software (ES) [Sommerville 2001] trata de todas as etapas de concepção de um software, utilizando abordagens sistemáticas na busca do desenvolvimento de softwares eficientes e de qualidade. A área apresenta diferentes métodos e técnicas que podem ser utilizadas no processo de desenvolvimento de software, porém o seu uso depende do tipo de aplicação a ser desenvolvida. Apesar dos métodos e técnicas que vem sendo utilizados, pode-se identificar a existência de uma variedade de linguagens de programação que aliadas à complexidade do seu desenvolvimento, podem apresentar certas dificuldades às empresas no que se refere à modelagem, implementação, avaliação e manutenção dos sistemas, e com isso, comprometer a qualidade dos mesmos. Um dos métodos utilizados para modelar e identificar as características de um domínio de aplicação é a Engenharia de Domínio (*Domain Engineering*) [Pohl et al. 2010]. Esta área é responsável pelo desenvolvimento das aplicações reusáveis, as quais destinam-se a modelagem e identificação de características do domínio de aplicação.

*Agradecimentos: CAPES pela concessão de bolsas de doutorado.

Sendo assim, para minimizar dificuldades inerentes ao desenvolvimento de aplicações, destaca-se a possibilidade de uso do desenvolvimento dirigido por modelos (*Model-Driven Development* - MDD) [Mernik et al. 2005], o qual, a partir de modelos do sistema, permite a geração automática de códigos fonte. Em específico para este artigo, a abordagem de MDD está baseada em Linguagens Específicas de Domínio (*Domain-Specific Languages - DSLs*) [Fowler 2010]. Pode-se destacar que tem se investido em diferentes propósitos para o desenvolvimento de DSLs, os quais variam conforme o domínio da aplicação, como por exemplo, teste de desempenho [Bernardino et al. 2016].

O processo de desenvolvimento de uma DSL é similar ao processo de implementação de outros softwares, no que se refere as atividades de análise, projeto, implementação e integração. Porém, o seu desenvolvimento exige o conhecimento de ferramentas de desenvolvimento, como por exemplo de *Language Workbench* (LW) que pode facilitar a criação e manutenção destas linguagens, mas por outro lado pode tornar o seu desenvolvimento mais oneroso, pois não são comuns aos usuais desenvolvedores.

Visando o esforço necessário para o seu desenvolvimento, torna-se ainda maior a intenção de que elas possam ser cada vez mais utilizadas plenamente. Para tal, acredita-se que as DSLs devam atender a critérios de usabilidade e de satisfação de uso no que se refere à experiência do usuário [Sinha and Smidts 2006]. Para que os usuários de domínios específicos possam utilizar estas linguagens de forma mais independente e com facilidade, é importante preocupar-se com os critérios de usabilidade e ainda com a diversidade de contextos e domínios, levando em conta a satisfação no uso durante a interação destes usuários com os diferentes sistemas.

Alguns estudos indicam a inexistência de métodos ou processos bem definidos para avaliar critérios de usabilidade em DSLs. Em vista disto, Engenheiros de Software avaliam de forma empírica, a facilidade de uso e de entendimento de suas DSLs [Rodrigues et al. 2017]. Com a intenção de auxiliar esses profissionais na avaliação de usabilidade das DSLs, foi proposto um *framework* de avaliação que define diretrizes para o planejamento, execução, análise e publicação dos resultados desta avaliação. Apesar do *framework* auxiliar no que diz respeito ao que deve conter uma avaliação, identificou-se por de um grupo focal a necessidade de que o mesmo fosse orientado por um processo bem definido, a partir de um conjunto de boas práticas que possam ser reproduzidas, contendo fluxos, passos previsíveis e detalhados, papéis, atividades, bem como tarefas que envolvem o uso de *guidelines* e artefatos [Poltronieri et al. 2018].

De forma a descrever este conjunto de práticas, está sendo utilizada a notação de metamodelagem *Software Engineering Process Metamodels* (SPEM). O SPEM é um metamodelo de Engenharia de Processos e uma estrutura conceitual que possibilita fornecer os conceitos necessários para modelar, documentar, apresentar, gerenciar, alterar e executar processos de desenvolvimento [(OMG) 2018]. A partir das definições desta notação foi realizado o mapeamento do Usa-DSL *framework* que servirá de apoio para o planejamento e desenvolvimento do processo denominado, Usa-DSL *Process*. Este processo possui os elementos de alto nível definidos pelo Usa-DSL *framework*, bem como as novas diretrizes e elementos do SPEM que serão definidos ao longo do desenvolvimento do ciclo de vida do Usa-DSL *Process*. Por sua vez, acredita-se que ao atender as recomendações ao utilizar este processo, seja possível aumentar o uso das DSLs desenvolvidas.

Este artigo está organizado da seguinte forma. Na Seção 2 são apresentados trabalhos relacionados. A Seção 3 descreve o *Framework* Usa-DSL. A Seção 4 apresenta a definição do Usa-DSL *Process*. Finalmente, a Seção 5 apresenta a conclusão e os trabalhos futuros.

2. Trabalhos Relacionados

Linguagens Específicas de Domínio buscam gerar alternativas viáveis para soluções existentes, com o propósito de simplificar o trabalho de desenvolvedores e usuários do domínio. Neste sentido, percebe-se que a comunidade que trata de aspectos ligados a conceitos referentes a DSLs tem concentrado seus esforços no desenvolvimento de novas técnicas/métodos, deixando de investigar as integrações de DSLs com outros processos de Engenharia de Software ou avaliar a eficácia das DSLs [Kosar et al. 2010]. O estudo de Kosar destaca que, usualmente os trabalhos apresentam as fases de análise de domínio, projeto e implementação, enquanto avaliação e manutenção são raramente apresentadas, evidenciando a preocupação com a falta de pesquisa na fase de avaliação das DSLs. Além disso, eles apontam que as DSLs raramente são avaliadas por seus usuários finais. No entanto, existem estudos que apresentam algumas avaliações realizadas por esta comunidade, porém tais avaliações geralmente são aplicadas na versão final da DSL quando os problemas encontrados são onerosos e muitas vezes inviáveis de serem resolvidos. Alguns destes estudos foram analisados na Revisão Sistemática da Literatura (RSL) de Rodrigues *et al.* [Rodrigues et al. 2017], no qual foi percebida a inexistência de métodos bem definidos para avaliar critérios de usabilidade em DSLs, levando os Engenheiros de Software a avaliar de forma empírica, até mesmo subjetivamente, se uma DSL é fácil de ser utilizada e de fácil entendimento. As principais pesquisas descritas nesse artigo, e que estão relacionadas a este trabalho, são apresentadas na Tabela 1.

Dentre os estudos apontados destacam-se as abordagens de Sinha and Smidts [Sinha and Smidts 2006] que utiliza um modelo de medidas para avaliar indicadores de usabilidade de uma DSL. Os autores destacam que as medidas de usabilidade foram definidas para suas aplicações ou mais especificamente para as *interfaces* gráficas destas aplicações. Albuquerque *et al.* [Albuquerque et al. 2015] apresenta um método de avaliação denominado *Cognitive Dimensions Notation* (CDN) que contém 14 dimensões. Tais dimensões serviram para apoiar o desenvolvimento das características de seu trabalho, *i.e.*, expressividade da DSL, que se refere ao quanto a DSL representa o domínio, e concisão da DSL, que se refere a quais termos podem ser excluídos sem comprometer a representatividade de artefatos de domínio. Essas características também foram divididas em métricas como: expressividade, que é composta de dependências ocultas, abstrações, mapeamento de proximidade; e, concisão, que é composta de viscosidade, visibilidade, difusão e operações mentais difíceis.

No trabalho de Teruel *et al.* [Teruel et al. 2014], os autores apresentam a condução de um experimento para avaliar a usabilidade de uma linguagem colaborativa de modelagem de requisitos de sistema. Já Barisic *et al.* [Barisic et al. 2012] propõem uma metodologia que envolve conceitos de usabilidade em processo de desenvolvimento de DSLs e considera que a avaliação deve acontecer desde o início do seu desenvolvimento até a sua finalização.

Desta forma, apesar do estudo de Barisic *et al.* [Barisic et al. 2014] apontar a fun-

damental importância em considerar critérios de qualidade de uso no desenvolvimento de uma DSL, para que essas possam atender as necessidades do usuário final, ainda identifica-se o uso de avaliações *ad-hoc*, como por exemplo no estudo de Kabac *et al.* [Kabáč et al. 2015] que não define o método de avaliação para usabilidade, porém buscam mensurar o aprendizado durante seu estudo experimental ao introduzir uma ferramenta no contexto da indústria. O estudo de Ewais *et al.* [Ewais and De Troyer 2014], apesar de não explicitarem o método de avaliação utilizado pode ser observado que a avaliação foi realizada sobre o ponto de vista de usabilidade e satisfação do usuário, a fim de avaliar se a abordagem criada era apropriada para melhorar a linguagem antes de iniciar o seu desenvolvimento.

Tabela 1. Resumo dos Trabalhos Relacionados

Estudo	Critério de Análise	Método de Avaliação de Usabilidade	Método de Avaliação de Eng. de Soft.
Albuquerque *et al.*	Método de avaliação denominado Cognitive Dimensions Notation(CDN) que contém 14 dimensões	Teste de Usabilidade	Estudo Experimental
Barisic *et al.*	Um modelo cognitivo para linguagens baseadas em cenários de usuários	Teste de Usabilidade	Estudo de Caso
Ewais and Troyer	Uma estratégia para avaliar a usabilidade de uma linguagem antes de ser implementada	Teste de Usabilidade	Estudo Experimental
Barisic *et al.*	Uma metodologia baseada em recomendação que considera técnicas centradas no usuário	Teste de Usabilidade	Não Informado
Sinha and Smidts	Avaliação a partir de quatro heurísticas propostas por Nielsen	Avaliação Heurística	Estudo Experimental
Turel *et al.*	Condução de um experimento para avaliar a usabilidade de uma linguagem colaborativa de modelagem de requisitos de sistema	Teste de Usabilidade	Estudo Experimental
Kabac *et al.*	Mensurar o aprendizado da linguagem durante um estudo experimental	Não Informado	Estudo Experimental
Alonso-Rios *et al.*	Descreve uma taxonomia de usabilidade para suportar diferentes estágios no desenvolvimento de sistemas	Teste de Usabilidade	Estudo Experimental
Kosar *et al.*	Condução de experimento utilizando o *Cognitive Dimension Framework* para comparar a utilização de uma Linguagem Específica de Domínio com uma biblioteca de aplicativos	Teste de Usabilidade / Dimensões Cognitivas	Estudo Experimental
Souza *et al.*	Avaliação de usabilidade da Linguagem Específica de Domínio para Cenários de Simulação Espacial DSL3S e suas ferramentas de apoio	Não Informado	Estudo Experimental
Bačíková *et al.*	Técnicas manuais para avaliar a usabilidade do domínio	Teste de Usabilidade	Estudo Experimental
Barisic *et al.*	Avaliação de usabilidade de Linguagem Específica de domínio USE-ME	Teste de Usabilidade	Estudo Experimental/ Estudo de Caso

Após a RSL, outros trabalhos foram identificados em uma busca exploratória de forma *ad-hoc*, como de Souza *et al.* [de Sousa and da Silva 2018] que relata um experimento com a iniciativa de avaliar a usabilidade da Linguagem Específica de Domínio para Cenários de Simulação Espacial DSL3S e suas ferramentas de apoio. Apesar dos autores informarem que realizaram uma avaliação de usabilidade, eles não descrevem o tipo de avaliação e o seu protocolo. Ainda identificou-se o estudo de Bačíková *et al.* [Bačíková et al. 2017], em que apresenta técnicas manuais para avaliar a usabilidade do domínio, no qual foram projetadas seis técnicas de avaliação qualitativa e uma técnica de avaliação formal baseada no questionário para avaliar usabilidade *System Usability Scale* (SUS). Neste trabalho os autores demonstram o uso das técnicas no domínio de música, sendo que duas das técnicas foram usadas para verificar experimentalmente o impacto de aspectos de usabilidade do domínio. Outro estudo importante foi apresentado por Alonso-Rios *et al.* [Alonso-Ríos et al. 2009] que descrevem uma taxonomia de usabilidade. Essa proposta de taxonomia ajudou a apoiar o desenvolvimento do modelo Usa-DSL, uma vez

que muitos atributos mostrados na taxonomia dos autores, sob a ótica da usabilidade do sistema, foram tratados em nossa proposta.

Mais recentemente identificou-se o *framework* conceitual para avaliação de usabilidade de DSL de Barisic [Barisic 2017] denominado USE-ME (*Usability Software Engineering Modelling Environment*). Esse estudo cita a RSL de Rodrigues *et al.* [Rodrigues et al. 2017], bem como apresentada e destaca que uma DSL mal projetada pode trazer danos e diminuir a produtividade, quando comparado a uma alternativa existente. Descreve, ainda, que o estado atual referente a prática em Engenharia de Linguagem de Software (*Software Language Engineering*) negligencia a usabilidade de DSLs.

3. Usa-DSL: *Framework* para Avaliação de Usabilidade de DSLs

Pensando na necessidade de avaliar linguagens para domínio específicos, buscou-se na literatura por meio da execução de uma Revisão Sistemática da Literatura (RSL), diferentes métodos e *frameworks* que realizassem avaliação de usabilidade. Desta forma, a partir dos resultados da RSL, foi proposto o *Framework* para avaliação de usabilidade de DSLs. Denominado Usa-DSL [Poltronieri et al. 2018], sendo o termo "Usa" referente à *Usability* (Usabilidade), enquanto que DSL à *Domain-Specific Language* (Linguagem Específica de Domínio). O Usa-DSL *Framework* possui uma estrutura que é composta por etapas, fases e atividades.

A estrutura do *framework* é um legado do processo de ciclo de vida de projeto [Stone et al. 2005], no qual estão dispostas *Steps*, *Phases* e *Activities* como pode ser visto na Figura 1. Os *Steps* estão definidas em 11 áreas de concentração. Já as *Phases* são compostas por 4 ciclos de execução. As *Activities* são formadas por um conjunto de 32 conceitos que são distribuídas entre as *Phases*. A estrutura do Usa-DSL *framework* foi planejada para ser utilizada a partir da necessidade de cada avaliação. Este foi avaliado através de entrevista e um grupo focal envolvendo participantes com experiência em IHC e Engenharia de Software. As avaliações tiveram como objetivo apresentar o *framework*, e obter a opinião dos participantes quanto a sua clareza, facilidade no uso e entendimento. A partir destas avaliações pode-se concluir, que tal *framework* necessitava de um amparo sistemático para sua execução. Desta forma, para sistematizar seu uso está sendo desenvolvido o Usa-DSL *Process* para orientar as diretrizes previstas nas phases, steps and activities do *framework*.

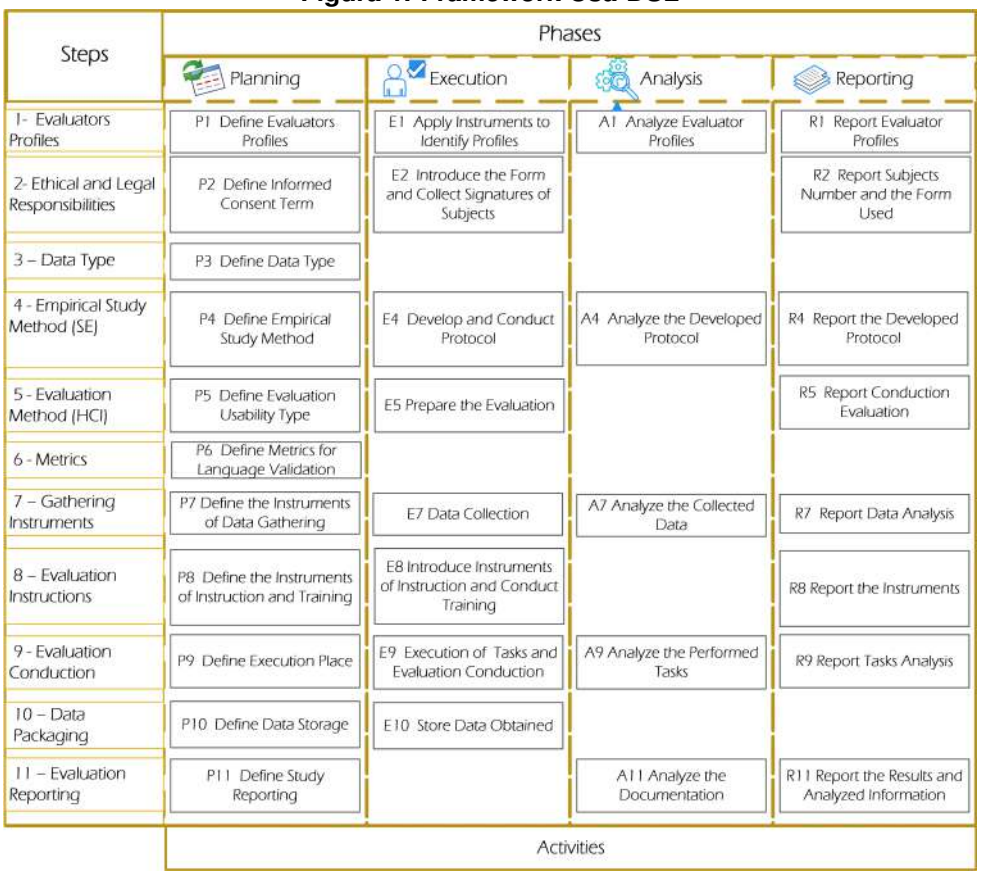
4. Usa-DSL *Process*

Para o desenvolvimento deste processo será utilizado o metamodelo *Software Process Engineering MetaModel* (SPEM) [(OMG) 2018], uma linguagem para modelagem de processos desenvolvida pela *Object Management Group* (OMG) com o apoio da ferramenta *EPF Composer* [Foundation 2018] desenvolvida pelo projeto *open source* do Eclipse *Foundation*, esta ferramenta é utilizada na elaboração, customização e publicação de processos. O Usa-DSL *Process* está sendo desenvolvido com base nos conceitos do *Framework* Usa-DSL, ou seja, seguindo a sua estrutura de *Phases*, *Steps* e *Activities*.

O Usa-DSL *Process* foi pensado tendo como princípios básicos:

(i) Auxiliar na elaboração da avaliação de usabilidade de forma prática e eficiente;
(ii) Auxiliar o desenvolvimento da DSL por meio de *feedback* contínuo, promovendo melhoria contínua da linguagem;

Figura 1. Framework Usa-DSL



(iii) Foco na avaliação centrada no usuário [Nielsen and Molich 1990] durante todas as fases de construção, minimizando retrabalho e desuso da DSL;
(iv) Tornar a avaliação de DSLs mais fácil e produtiva, procurando alcançar critérios de usabilidade e de satisfação de uso [Sinha and Smidts 2006].

Seu intuito principal é nortear as questões "Quem fará o quê, quando e como?" [Pressman 2010], sempre de forma iterativa [Stone et al. 2005] e guiado para o desenvolvimento de avaliações formais com facilidade e rapidez. O seu principal objetivo é fornecer os procedimentos para realização da avaliação no que se refere à usabilidade de DSLs, verificando por meio de métodos, técnicas e atividades o quanto estas linguagens atendem à questões de qualidade de uso. O Usa-DSL *Process* trata da elaboração das avaliações de usabilidade envolvendo os métodos *Heuristic Evaluation* e *Usability Testing* [Nielsen 1993], com a intenção de sistematizar e desenvolver as avaliações de uma forma produtiva e com menos custo.

O Usa-DSL *Process* possui uma estrutura composto por conteúdo de método (*Method Content*) e processo (*Process*). O *Method Content* trata de uma base de conhecimento, no qual são armazenadas as informações que serão consumidas pelo processo. As informações armazenadas no *Method Content* estão organizadas em *Work Products*, *Profiles*, *Tasks* e *Steps*. Já o *Process* trata da organização dos elementos do *Method Content*, relacionando este elementos em sequência parcialmente ordenados e de forma personalizada para cada projeto específico. O *Process* possui elementos estruturais básicos como: *Phases*, *Activities*, *Profiles Use*, *Tasks Use* e *Work Products Use*. O Usa-DSL assim como

o SPEM, contém uma categoria de agrupamento de elementos que se destina aos *Guidelines*, ou seja, *templates* de documentos que serão consumidos, gerados ou alterados durante todo o ciclo de vida. Estes *templates* serão utilizados tanto pelo conteúdo de método como pelo processo, eles são descritos de uma forma única. O Usa-DSL *Process* possui diversos artefatos deste tipo que poderão ser utilizados durante a execução de uma tarefa e/ou do processo gerado, são eles: *Informed Consent Term*, *Profile Questionnaire*, *Opinion Questionnaire*, *Glossary*, *Documentation*, *DSL Guide*, *Usage Scenario*, *Trainning Documentation DSL* e *Study Protocol*. Os demais elementos que compõem cada um destes conceitos serão descritos em detalhes na Seção 4.1.

4.1. *Method Content* - Usa-DSL *Process*

O *Method Content* que compõem o Usa-DSL *Process* está organizado conforme os elementos da notação SPEM, porém mapeados para os elementos do Usa-DSL *Framework*:

Work Product - são os arquivos que serão consumidos ou gerados durante a execução de uma atividade.

Profile - é um elemento do *Method Content* que define os papéis desempenhados dentro do processo. Os *Profiles* são usados para definir quem executa cada *Task*, bem como para definir os responsáveis por um conjunto de *Work Products*. O Usa-DSL *Process* define oito (8) *Profiles*, os quais foram agrupados em três (3) *Profile Sets*, no qual foram reunidos por tipo de execução, são eles:

- *Process Executors:* este conjunto de *Profiles* reúne os usuários do Usa-DSL *Process*, ou seja, quem planeja e conduz as avaliações. Estes *Profiles* fazem parte do grupo que projeta, desenvolve e possui interesse em avaliar a DSL, são eles: *DSL Analyst*, *DSL Developer* e *DSL Tester*;
- *Usability Evaluation Subjects:* este conjunto de *Profiles* reúne os *Domain Analyst*, *Domain Developer*, *Domain Tester* e *End User*, ou seja, as pessoas que serão convidadas a participar da avaliação. Aqueles que contribuirão com os desenvolvedores da DSL, apontando as melhorias ou correções que devem ser realizadas para que se alcance um experiência mais agradável para o usuário;
- *Heurístic Evaluation Subjects:* este *Profile* é composto apenas pelo *Expert* em IHC, ou seja, o especialista em avaliação heurística. Estes participantes tem como objetivo avaliar a DSL e contribuir com o grupo de desenvolvedores a partir dos erros e grau de severidade apontados durante a avaliação, buscando melhorar a usabilidade e experiência do usuário antes de apresentar a DSL para os usuários finais.

Task - descreve uma unidade de trabalho com propósito claro, a ser atribuída a uma *Activity* e um *Profile* específico, com a finalidade de atingir uma meta bem definida. São fornecidas explicações completas de como executar o trabalho para atingir determinada meta, bem como o trabalho que deve ser feito e os *Guidelines* que serão utilizados durante a tarefa. As *Tasks* que compõem o Usa-DSL *Process* serão apresentadas no contexto de uma *Activity* ao qual ela pertence.

Step - é uma forma flexível de definir diferentes agrupamentos para categorias de conteúdo. No Usa-DSL *Process* um *Step* é definido pelo elemento *Discipline* que é derivado de um elemento *Category* do SPEM. Um *Step* é um agrupamento de trabalho

com base na similaridade de preocupações e cooperação do esforço de trabalho, ou seja, é uma coleção de *Tasks* relacionadas a determinadas *Activities* e compostas por um conjunto de perfis denominados *Role Sets*, por exemplo, um *Evaluation Instructions* é um *Step* que agrupará todas as *Tasks* e *Profiles* que fazem parte desta área de preocupação.

4.2. *Process* - Usa-DSL *Process*

O *Process* do Usa-DSL apresenta o ciclo de vida do processo de avaliação de usabilidade, por meio dos elementos derivados do SPEM. Este processo é composto pelos elementos: *Phase*, *Activity*, *Profile Use*, *Task Use* e *Work Product Use*, os quais serão descritos a seguir:

Phase no Usa-DSL *Process* é um elemento do tipo *Activity* no SPEM, no qual define a divisão de trabalho, ou seja, este elemento organiza o processo em um espaço de tempo, no qual as *Activities* serão executadas. As *Phases* são orientados por *Steps* que determinam quais *Activities* serão executadas e em que momento do processo. O Usa-DSL *Process* é composto por quatro *Phases*: *Planning*, *Execution*, *Analysis* e *Reporting*. É importante enfatizar que cada *Phase* do processo é concluída por um marco, ou seja, um conjunto de atividades e artefatos gerados pelo grupo que realiza a avaliação a ser entregue caracterizando o encerramento da *Phase* corrente;

Activities no Usa-DSL *Process* é um elemento do tipo *Activity* no SPEM, e compõem uma divisão de trabalho que possui um conjunto de *Tasks*, as quais definem unidades básicas de trabalho dentro de um processo, bem como o processo em si. Ela representa uma unidade geral de trabalho que pode ser atribuída a um *Profile Use* específico, bem como receber entradas e produzir saídas. Uma *Activity* representa um elemento de agrupamento para outros elementos de divisão de trabalho, tais como: *Tasks Use*, *Profiles Use* e *Work Product Use*. No caso do Usa-DSL *Process*, as *Activities* agrupam as *Tasks* referentes a um determinado *Step* dentro de um espaço de tempo de uma *Phase* no ciclo de vida do processo. Por este motivo, cada *Activity* possui uma identificação composta pela letra inicial da *Phase* e o número do *Step*.

Profile Use é um elemento do tipo *Role Use* no SPEM, este elemento representa um executor de uma *Activity* ou um participante da *Activity*. Os *Profiles Use* são perfis que estão definidos no *Profiles* da base de conhecimento, portanto já possuem funções definidas. Nesta etapa do processo eles apenas serão relacionadas a *Task* que devem executar.

Task Use é um elemento de divisão de trabalho que representa uma *Task* que está sendo utilizada por um determinado *Profile* no contexto de uma *Activity* específica. Pode-se exemplificar por meio da tarefa *P1a Choose the Profile Evaluator*, a qual está relacionada a *P1 - Define Evaluators Profiles Activity* e esta sendo executada pelo avaliador da DSL, por exemplo *Profile DSL Developer*, em que neste caso estará realizando o planejamento da avaliação por meio do Usa-DSL *Process*.

Work Products Use são os artefatos definidos no *Work Product* e que serão utilizados durante a execução do processo. Estes artefatos são produtos ou documentos de trabalho que podem ser consumidos, gerados ou alterados durante a execução de uma *Task* específica dentro de uma *Activity*. Por exemplo, o *Informed Consent Term* é um artefato de termo de consentimento livre esclarecido consumido durante a tarefa *P2a Select the Informed Consent Term* e será alterado na tarefa *E2c Colleted Signature of the Sub-*

ject no momento que o participante assinar o termo, a partir desta assinatura inclui-se um dado que não possuía ao artefato quando este foi adicionado a base de conhecimento do processo.

5. Considerações Finais e Trabalhos Futuros

Atualmente tem se identificado um aumento no desenvolvimento de DSLs, porém percebe-se que o maior investimento dos desenvolvedores ainda se concentra nas fases de análise de domínio, projeto e implementação, negligenciando as fases de avaliação ou até mesmo avaliando de forma subjetiva. Entretanto, para que as DSLs se tornem mais atrativas e com maior qualidade de uso, a literatura sugere que seja dedicado um maior esforço na avaliação de usabilidade destas DSLs [Quiñones and Rusu 2017]. Sendo assim, o Usa-DSL *Process* está sendo desenvolvido para que possa auxiliar os engenheiros de linguagem com tais avaliações, buscando minimizar as dificuldades inerentes a utilização das aplicações construídas e com a preocupação que elas não caiam em desuso. Com o uso do Usa-DSL *Process*, busca-se facilitar o planejamento, execução e análise da avaliação de DSLs, oferecendo informações de forma clara e objetivas para que a equipe de avaliação (analista, desenvolvedor, testador) possa planejar e avaliar a DSL.

Referente aos trabalhos futuros, está sendo descrito e modelado o Usa-DSL *Process* utilizando a notação SPEM e a ferramenta *EPF Composer*. Nesta etapa se está adicionando informações e materiais informativos do Usa-DSL, bem como os *Guidelines* e artefatos para que então seja apresentado o ciclo de vida completo do processo e sua aplicação. Após a conclusão do desenvolvimento do Usa-DSL *Process* estão prevista as avaliações deste processo, no qual os potenciais usuários utilizarão este processo para realizar a avaliação de suas DSLs.

Referências

Albuquerque, D., Cafeo, B., Garcia, A., Barbosa, S., Abrahão, S., and Ribeiro, A. (2015). Quantifying usability of domain-specific languages: An empirical study on software maintenance. *Journal of Systems and Software*, 101:245 – 259.

Alonso-Ríos, D., Vázquez-García, A., Mosqueira-Rey, E., and Moret-Bonillo, V. (2009). Usability: A Critical Analysis and a Taxonomy. *International Journal of Human–Computer Interaction*, 26(1):53–74.

Barisic, A. (2017). *Usability Evaluation of Domain-Specific Languages*. PhD thesis, Universidade Nova Lisboa, Lisboa, Portugal.

Barisic, A., Amaral, V., Goulão, M., and Aguiar, A. (2014). Introducing usability concerns early in the dsl development cycle: Flowsl experience report. In *17th International Conference on Model Driven Engineering Languages and Systems*.

Barisic, A., Amaral, V., and Goulão, M. (2012). Usability evaluation of domain-specific languages. In *Quality of Information and Communications Tech.*, pages 342–347.

Bačíková, M., Galko, L., and Hvizdová, E. (2017). Manual techniques for evaluating domain usability. In *14th Intern. Scientific Conference on Informatics*, pages 24–30.

Bernardino, M., Zorzo, A., and Rodrigues, E. (2016). Canopus: A Domain-Specific Language for Modeling Performance Testing. In *9th International Conference on Software Testing, Verification and Validation (ICST)*, pages 157–167. IEEE.

de Sousa, L. M. and da Silva, A. R. (2018). Usability evaluation of the domain specific language for spatial simulation scenarios. *Cogent Engineering*, 5(1):1436889.

Ewais, A. B. and De Troyer, O. (2014). A usability evaluation of graphical modelling languages for authoring adaptive 3d virtual learning environments. In *6th International Conference on Computer Supported Education*, pages 459–466.

Foundation, E. (2018). Eclipse Process Framework Project - EPF Composer.

Fowler, M. (2010). *Domain Specific Languages*. Addison-Wesley Professional, 1 edition.

Kabáč, M., Volanschi, N., and Consel, C. (2015). An evaluation of the diasuite toolset by professional developers: Learning cost and usability. In *6th Workshop on Evaluation and Usability of Programming Languages and Tools*, pages 9–16. ACM.

Kosar, T., Oliveira, N., Mernik, M., Pereira, M. J. V., Črepinšek, M., da Cruz, D., and Henriques, P. R. (2010). Comparing general-purpose and domain-specific languages: An empirical study. *Computer Science and Information Systems*, (14):247–264.

Mernik, M., Heering, J., and Sloane, A. M. (2005). When and how to develop domain-specific languages. *ACM Computing Surveys*, 37(4):316–344.

Nielsen, J. (1993). *Usability Engineering*. Morgan Kaufmann, San Francisco, USA.

Nielsen, J. and Molich, R. (1990). Heuristic evaluation of user interfaces. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 249–256. ACM.

(OMG), O. M. G. (2018). Software and Systems Process Engineering Metamodel Specification - SPEM.

Pohl, K., Böckle, G., and van der Linden, F. (2010). *Software Product Lines Engineering: Foundations, Principles, and Techniques*. Springer - Verlag.

Poltronieri, I., Zorzo, A. F., Bernardino, M., and de Borba Campos, M. (2018). Usa-DSL: Usability Evaluation Framework for Domain-specific Languages. In *33rd Annual ACM Symposium on Applied Computing*, SAC '18, pages 2013–2021. ACM.

Pressman, R. (2010). *Software Engineering: A Practitioner's Approach*. McGraw-Hill.

Quiñones, D. and Rusu, C. (2017). How to develop usability heuristics: A systematic literature review. *Computer Standards & Interfaces*, 53:89–122.

Rodrigues, I., Campos, M., and Zorzo, A. (2017). Usability evaluation of domain-specific languages: a systematic literature review. In *19th International Conference on Human-Computer Interaction*, pages 522–534.

Sinha, A. c. and Smidts, C. d. (2006). An experimental evaluation of a higher-ordered-typed-functional specification-based test-generation technique. *Empirical Software Engineering*, 11(2):173–202.

Sommerville, I. (2001). *Software Engineering*. Addison-Wesley, Boston, MA, USA.

Stone, D., Jarrett, C., Woodroffe, M., and Minocha, S. (2005). *User Interface Design and Evaluation*. Interactive Technologies. Elsevier Science.

Teruel, M. A., Navarro, E., López-Jaquero, V., Montero, F., and González, P. (2014). A cscw requirements engineering case tool: Development and usability evaluation. *Information and Software Technology*, 56(8):922 – 949.