UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS E COMPUTAÇÃO

FORMALIZAÇÃO DE EXPERIMENTOS CONTROLADOS EM ENGENHARIA DE SOFTWARE

Por
MARÍLIA ARANHA FREIRE
Tese de Doutorado

NATAL

Março, 2015

MARILIA ARANHA FREIRE

Formalização de Experimentos Controlados em Engenharia de Software

Esta Tese foi julgada adequada para a obtenção do título de doutor em Ciência da Computação e aprovado em sua forma final pelo Programa de Pós-Graduação em Sistemas e Computação do Departamento de Informática e Matemática Aplicada da Universidade Federal do Rio Grande do Norte.

Prof. Dr Uirá Kulesza – UFRN
Orientador

Profª. Drª. Elizabeth Ferreira Gouvea – UFRN
Vice - Coordenadora do Programa

Banca Examinadora

Prof. Dr Uirá Kulesza – UFRN
Presidente

Prof. Dr. Eduardo Henrique da Silva Aranha – UFRN

Prof. Dr. Fernando Marques Figueira Filho - UFRN

Profª. Drª. Thais Vasconcelos Batista – UFRN

Prof. Dr. Dalton Dario Serey Guerrero – UFCG

Prof. Dr. Sérgio Castelo Branco Soares – UFPE

Março, 2015

UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS E COMPUTAÇÃO

MARÍLIA ARANHA FREIRE

Formalização de Experimentos Controlados em Engenharia de Software

Tese submetida à Coordenação do Programa de Pós-Graduação em Sistemas e Computação, do Centro Ciências Exatas e da Terra, da Universidade Federal do Rio Grande do Norte, como parte dos requisitos para obtenção de título de Doutor em Sistemas e Computação.

Orientador: Uirá Kulesza, Dr.

Coorientador: Eduardo Aranha, Dr.

NATAL

Março, 2015

Catalogação da Publicação na Fonte. UFRN / SISBI / Biblioteca Setorial
Centro de Ciências Exatas e da Terra – CCET.

Freire, Marília Aranha.
Formalização de experimentos controlados em engenharia de software / Marília Aranha Freire. - Natal, 2015.
xiv, 198 f.: il.

Orientador: Prof. Dr. Uirá Kulesza.
Coorientador: Prof. Dr. Eduardo Aranha.

Tese (Doutorado) – Universidade Federal do Rio Grande do Norte. Centro de Ciências Exatas e da Terra. Programa de Pós-Graduação em Sistemas e Computação.

1. Engenharia de software experimental – Tese. 2. Experimentos controlados – Tese. 3. Linguagem específica de domínio – Tese. 4. Engenharia dirigida por modelos – Tese. I. Kulesza, Uirá. II. Aranha, Eduardo. III. Título.

RN/UF/BSE-CCET CDU: 004.41

Agradecimentos

A Deus, inteligência suprema, causa primária de tudo que existe, por todas as bênçãos que recebo diariamente!

Agradeço e dedico este trabalho à minha família que sempre me apoiou, vibrando comigo nos momentos felizes, de conquistas, ou suportando e aliviando meus estresses, nos momentos de apreensão ou doença. Obrigada pela paciência e amor de todos vocês: Clélio, Amarilis, Cynthia, Adriano, Geraldo, Verana, Andrei, Lissa e Luana. Agradeço adicionalmente a todos os membros das minhas famílias Aranha e Freire pelo incentivo e crédito que sempre me deram.

Ao meu orientador, prof. Dr. Uirá Kulesza, por ter me dado o crédito de voltar à pesquisa, pela constante ajuda e conselhos para que eu pudesse seguir por esses longos anos de doutoramento, além da amizade constante. Sua dedicação ao trabalho e seus conhecimentos foram essenciais para a realização deste trabalho.

Ao meu coorientador, prof. Dr. Eduardo Aranha, por ter aceitado fazer parte deste trabalho e por ter contribuindo, e muito, com seus conhecimentos, ajudando sempre com muita paciência e tranquilidade.

A todos os meus amigos do grupo de pesquisa, com muitos dos quais realizei inúmeros trabalhos colaborativos, agradeço pelos momentos de estudo e descontração. Agradeço, em especial, a Fellipe Aleixo, Wanderson Santos, Daniel Alencar, Edmilson Campos e Gustavo Sizílio.

Aos amigos do percurso do doutorado, Alessandro, Plácido, Minora, Liliane, Tainá, Strapação e todos os que frequentavam o laboratório de doutorado, compartilhando os percalços e conquistas ao longo desses anos.

Aos meus amigos de uma vida inteira pela amizade verdadeira, pelo constante apoio e pela, sempre exagerada, confiança que depositam no meu desempenho profissional.

À Diretoria de Gestão e Tecnologia da Informação do IFRN, por ter possibilitado o meu afastamento para dedicação exclusiva ao doutorado durante 5

semestres. Agradeço, adicionalmente, o apoio dos colegas que fazem parte da DIATINF.

Aos professores membros da minha banca de proposta de tese, Prof. Dr. Manoel Mendonça e Prof. Dr. Fernando Figueira, pelos excelentes feedbacks que recebi.

Agradeço ao prof. Dr. Andreas Jedlitschka (IESE) e a profa. Dra. Natalia Juristo (UPM), com os quais trabalhei durante o meu período de sanduíche, pelas contribuições recebidas.

Agradeço aos professores do PPGSC da UFRN que contribuíram direta e indiretamente para a realização do meu doutorado.

Agradeço aos funcionários do DIMAp/UFRN pelos serviços prestados sempre com amizade e respeito, em especial agradeço a Rita e Sr. Gaspar.

Agradeço aos professores que aceitaram fazer parte da minha banca de defesa de tese de doutorado, Prof. Dr. Sérgio Soares (UFPE), Prof. Dr. Dalton Guerrero (UFCG), Profa. Dra. Thaís Batista (UFRN) e Prof. Dr. Fernando Figueira (UFRN).

Por fim, agradeço também aos alunos do PPGSC que participaram dos experimentos que fazem parte deste trabalho.

Os homens semeiam na terra o que colherão na vida espiritual: os frutos da sua coragem ou da sua fraqueza.

Allan Kardec.

Resumo

A condução de estudos empíricos é de vital importância para coletar evidências científicas sobre novas tecnologias de software. Neste sentido, nos últimos anos, centenas de experimentos controlados vêm sendo realizados na área da engenharia de software. Um experimento controlado é uma técnica que permite aos cientistas testarem uma hipótese de pesquisa e a relação causa e efeito entre as variáveis envolvidas no ambiente de estudo. Entretanto, o planejamento, execução, análise e empacotamento de um experimento controlado são considerados atividades complexas, custosas e propensas a erros. As poucas ferramentas existentes de apoio ao processo de experimentação auxiliam várias atividades envolvidas em um experimento mas possuem limitações e grande necessidade de melhorias.

Neste contexto, este trabalho propõe: (i) investigar abordagens e ambientes existentes de apoio a formalização e condução de experimentos controlados em ES identificando suas limitações e benefícios; (ii) propor uma linguagem específica de domínio para a formalização de experimentos controlados; e (iii) desenvolver uma abordagem dirigida por modelos que usa a formalização de um experimento para geração de *workflows* customizáveis de apoio à condução de experimentos controlados. O trabalho é avaliado através da condução de: (i) um estudo de viabilidade da abordagem dirigida por modelos através da modelagem de um experimento existente e geração de workflows customizáveis a partir do seu projeto estatístico; (ii) um estudo empírico de análise da expressividade e completude da linguagem específica de domínio proposta através da modelagem de 16 experimentos; (iii) um experimento controlado que investiga a compreensão da linguagem pelos experimentadores; e (iv) um experimento controlado que investiga a usabilidade da linguagem através do seu uso direto na especificação de experimentos. Os resultados obtidos em tais estudos trazem evidências que a abordagem proposta é viável, e que a linguagem tem um bom nível de expressividade e completude. Além disso, as análises mostram que a compreensão do plano experimental escrito na linguagem proposta é mais fácil e mais rápida que quando analisando a especificação de um plano experimental descrito em um artigo. Por fim, a percepção dos experimentadores foi positiva em relação à utilização da linguagem.

Palavras-chave: engenharia de software experimental, experimentos controlados, linguagem específica de domínio, engenharia dirigida por modelos.

Abstract

The conduction of empirical studies is very important to gather scientific evidences of new software technologies. Over the last years, a hundred of controlled experiments have been conducted in the software engineering area. A controlled experiment is a technique that allows researchers to test a research hypothesis and the causal effect analysis among the variables involved in the study environment. However, the planning, execution, analysis and packaging of a controlled experiment are considered work intensive, time consuming and error-prone activities. A few existing supporting tools can help the accomplishment of many of these activities but they still have many limitations and improvement needs.

In this context, this thesis proposes: (i) to investigate existing approaches and environments to support the formalization and conduction of SE controlled experiments by identifying their limitations and benefits; (ii) to propose a domain-specific language (DSL) to formalize the specification of controlled experiments; and (iii) to develop a model-driven approach that can use the experiment specification in the DSL to generate customized workflows to support the execution of controlled experiments.

This work is evaluated through the conduction of: (i) a feasibility study of the model-driven approach through the modeling of a real experiment and the generation of workflows according to its experimental design; (ii) an empirical study that assesses the expressivity and completeness of the domain-specific language through the modeling of 16 existing experiments; (iii) a controlled experiment that investigates the DSL comprehensibility by the experimenters; and (iv) a controlled experiment that investigates the language usability through the specification of experiments. The studies results bring evidences of the approach feasibility, and the expressiveness and completeness of the DSL. In addition, our controlled experiments results show that: (i) the experimental plan comprehension when written in the proposed DSL is easier to understand and faster to specify when compared to the experiment specification described in scientific papers; and (ii) the experimenters' perception was positive when using the DSL.

Keywords: experimental software engineering; controlled experiments; domain-specific language; model-driven engineering.

Sumário

Índice de Figuras

Índice de Tabelas

1. Introdução

A necessidade de transferência de conhecimento da academia para a indústria sempre existiu, desde que novas teorias, metodologias, técnicas e ferramentas são desenvolvidas pela comunidade científica em instituições de pesquisa e são necessárias para aprimorar os processos e a qualidade dos produtos do setor industrial. Segundo Basili (BASILI, 1996), a experimentação é realizada para ajudar-nos a melhor avaliar, prever, entender, controlar e melhorar o produto e o processo de desenvolvimento de software. Nesse sentido, a importância da experimentação na engenharia de software vem crescendo tanto na visão acadêmica quanto na visão da indústria. Para a indústria, a experimentação é muito útil para direcionar a evolução dos ambientes reais de produção, isso porque ela fornece informações importantes sobre os reais benefícios e limitações das tecnologias que estão sendo propostas, permitindo uma tomada de decisão mais precisa quanto à adoção ou não destes novos processos, tecnologias, técnicas e métodos. Na academia, a experimentação é utilizada para compreender os problemas, criar teorias, desenvolver soluções, formular e testar hipóteses, comparar sistematicamente as soluções propostas em relação às soluções existentes, etc. Além disso, a transferência bem sucedida de conhecimento de uma tecnologia geralmente tem levado em torno de 18 anos (JEDLITSCHKA, CIOLKOWSKI, *et al.*, 2007) (REDWINE e RIDDLE, 1985). Portanto, os estudos sistematizados aliados à realização de experimentos pode acelerar este processo e, conseqüentemente, reduzir o tempo de obter os ganhos esperados por inovações científicas produzidas pela academia dentro do contexto de projetos reais sendo desenvolvidos pela indústria.

Neste contexto, a comunidade de engenharia de software vem aumentando a demanda por estudos empíricos mais rigorosos visando obter evidências científicas sobre as tecnologias de software produzidas. Com isso, centenas de experimentos controlados vêm sendo realizados nesta área e esse número vem crescendo nos últimos anos (BASILI, 1996) (BASILI, 2006) (BASILI, 2013) (SJOEBERG, HANNAY, *et al.*, 2005) (WOHLIN, RUNESON, *et al.*, 2012) (JURISTO e MORENO, 2010) (KO, LATOZA e BURNET, 2015). Um experimento controlado é uma técnica que permite cientistas testarem uma hipótese de pesquisa e relacionamento causa e efeito entre as variáveis (dependentes e independentes) envolvidas no ambiente do estudo. Entretanto,

o planejamento, execução, análise e empacotamento de um experimento controlado ainda são atividades complexas, custosas e propensas a erros.

Além disso, a comunidade vem discutindo como melhor apoiar a aplicação de experimentos controlados na área. Estes estudos vêm propondo guias para relatar experimentos (JEDLITSCHKA, CIOLKOWSKI e PFAHL, 2008), guias práticos para a condução de experimentos (KO, LATOZA e BURNET, 2015), *frameworks* conceituais para guiar a replicação de experimentos (MENDONÇA, MALDONADO, *et al.*, 2008), e ambientes/ferramentas para apoiar a condução e replicação destes experimentos (FREIRE, ALENCAR, *et al.*, 2013). Embora estes estudos tragam percepções e resultados para a condução de experimentos controlados, poucos deles (TRAVASSOS, SANTOS, *et al.*, 2008) (CARTAXO, COSTA, *et al.*, 2012) (GARCIA, HOHN, *et al.*, 2008) (SIY e WU, 2009) (SCATALON, GARCIA, & CORREIA e M, 2011) propõem como formalizar o planejamento, execução e análise de experimentos controlados. Além disso, as abordagens existentes que visam formalizar a especificação de experimentos controlados – tais como linguagens específicas de domínio e ontologias - não foram devidamente avaliadas. Portanto, este tema ainda requer profundas pesquisas, entre elas, na formalização dos experimentos. Como benefícios, a formalização de um experimento, ao especificar uma terminologia única, possibilita a troca de informações entre *stakeholders* (projetista do experimento, estatísticos, especialistas do domínio) e entre a comunidade de pesquisadores, facilita a replicação destes estudos e a realização de meta-análise, pois a falta de padronização é um obstáculo para a integração de conhecimento de um estudo isolado em uma base de conhecimento. Adicionalmente, a formalização pode servir de base para o desenvolvimento de diferentes ambientes de suporte a experimentação.

Para superar estes desafios, este trabalho propõe uma linguagem específica de domínio para apoiar a formalização do planejamento de um experimento (experimento controlado ou quasi-experimento) em engenharia de software. A linguagem permite a definição de aspectos do plano experimental, tais como, objetivos, questões e hipóteses de pesquisa, projeto experimental, variáveis dependentes, fatores, procedimento de coleta de dados, entre outros. Como consequência da formalização, o trabalho também apresenta uma abordagem dirigida por modelos para a geração de workflows customizados para cada participante do experimento. Estes workflows guiarão os participantes na execução dos tratamentos investigados no experimento obedecendo o

projeto experimental. Um ambiente de execução de workflows que guia a execução do experimento e permite o seu monitoramento é também definido.

É importante destacar que este trabalho está inserido num contexto mais amplo, onde uma infraestrutura experimental vem sendo desenvolvida com o intuito de dar apoio a uma rede de experimentação em engenharia de software. Este projeto maior é colaborativo e conta com a participação de diversas instituições, entre elas, UFRN, UFPE e PUC-RIO.

1.1. Problema Abordado

Apesar dos benefícios proporcionados por estudos conduzidos pela engenharia de software experimental, a realização destes estudos leva tempo e é ainda complexa pelo fato de não existirem ambientes especializados para auxiliar na especificação, avaliação, execução e replicação de tais estudos. Uma das razões para isso é a escassez de ambientes computadorizados para apoiar a especificação e execução de tais estudos experimentais. Estes ambientes podem ajudar no apoio ao planejamento assim como no acompanhamento das atividades envolvidas em um experimento, dando mais sistemática a sua execução, e facilitando a sua realização em larga escala.

A falta de ambientes de suporte tem dificultado a execução de experimentos com profissionais da indústria usando ferramentas de desenvolvimento (SJøBERG, ANDA, *et al.*, 2002), e invalidado resultados devido à problemas de planejamento ou condução do experimento (ACCIOLY, 2012). Os desafios são ainda maiores quando considerando experimentos distribuídos (BUDGEN, KITCHENHAM, *et al.*, 2013), replicação de experimentos (SOLARI, 2013) ou meta-análise (CIOLKOWSKI, 2013), onde informações detalhadas devem estar disponíveis sobre os experimentos executados previamente. Além disso, algumas pesquisam apontam que muitos estudos experimentais apresentam resultados difíceis de interpretar e aplicar na prática, que vão desde o envolvimento de participantes não representativos até a omissão de detalhes metodológicos chave sobre a configuração do experimento (KO, LATOZA e BURNET, 2015). Uma das razões para a falta de ambientes é a ausência de um modelo conceitual que represente o planejamento experimental. Em geral, a maioria das informações à respeito de um experimento controlado é descrita em linguagem natural, resultando em ambiguidade, inconsistência e falta de informação(CIOLKOWSKI, 2013).

A experiência prática na condução de experimentos controlados evidencia alguns problemas recorrentes que são mencionados na literatura (ARISHOLM, SJøBERG, *et al.*, 2002) (HOCHSTEIN, NAKAMURA, *et al.*, 2008) (SJøBERG, ANDA, *et al.*, 2002) (STOLEE e ELBAUM, 2010): (i) muitos experimentos usam "papel & lápis" para coletar os dados relativos às variáveis de resposta; (ii) alguns estudos usam ferramentas existentes adaptadas para gerenciar e executar experimentos; ou (iii) desenvolvem novas ferramentas específicas para auxiliar a execução do experimento em questão.

A utilização da coleta de dados manual (papel & lápis) durante a execução de um experimento já é bem discutida na literatura e seus problemas são claros (ARISHOLM, SJøBERG, *et al.*, 2002) (HOCHSTEIN, NAKAMURA, *et al.*, 2008) (SJøBERG, ANDA, *et al.*, 2002) (TRAVASSOS, SANTOS, *et al.*, 2008): é uma atividade propensa à erros, pode gerar imprecisão nos dados e, ainda, demanda muito mais tempo para coleta e análise dos dados do experimento.

A utilização de ferramentas já existentes adaptadas para o contexto da experimentação pode auxiliar a execução do experimento, mas também gera alguns problemas, tais como: necessidade de adaptar os procedimentos do experimento aos procedimentos da ferramenta (atividades e terminologia), *features* desnecessárias já existentes na ferramenta podem confundir os participantes e, além disso, exige que haja um treinamento mais detalhado sobre o uso da ferramenta, o que demanda tempo na execução do experimento.

Accioly (ACCIOLY, 2012), por exemplo, descreve a realização de cinco experimentos controlados para avaliar duas diferentes técnicas de *design* de testes em linhas de produtos de software. O estudo aponta problemas que representam ameaças para o sucesso de execuções do experimento como por exemplo: alto consumo de tempo, imprecisão nos dados gerados, dificuldade de monitoramento e dificuldades para replicação. A primeira execução do experimento foi realizada usando "papel e lápis", o que dificultou a coleta e análise dos dados. O desenvolvimento de uma nova ferramenta de suporte à execução do experimento também foi adotado em (ACCIOLY, 2012). Esta opção permite o desenvolvimento do ambiente de apoio de acordo com as necessidades do experimento, porém, é uma opção que aumenta o consumo de tempo, geralmente é específica para o domínio do problema sendo tratado e, em muitos casos, não é uma solução reutilizável.

Além disso, os relatos em (ACCIOLY, 2012) enfatizam que a possibilidade de monitoramento online do experimento poderia evitar problemas de entendimento por

parte dos participantes, pois habilita o pesquisador a acompanhar as atividades sendo realizadas e os artefatos sendo gerados de forma antecipada, ainda durante a fase de execução, pois a descoberta de problemas de entendimento que causam invalidação nos dados coletados apenas na fase de análise inviabiliza a utilização dos mesmos e pode aumentar o viés do estudo em questão.

Existe, portanto, a necessidade de um modelo padrão para especificar experimentos controlados, além de ferramentas de apoio a todo o processo de experimentação.

1.2. Limitações dos Trabalhos Atuais

Embora o número de experimentos controlados na engenharia de software tenha aumentado nos últimos anos, ainda é muito limitado o número de abordagens propostas para apoiar a realização deste tipo de estudo, através da sua formalização. Tal carência é evidenciada através dos resultados de uma revisão sistemática conduzida como parte desta tese de doutorado (Capítulo 3).

Travassos et al. (2008) (TRAVASSOS, SANTOS, *et al.*, 2008) apresentam o ambiente experimental eSEE (*experimentation environment to support large-scale experimentation*). Este ambiente propõe o gerenciamento de vários tipos de estudos empíricos na engenharia de software. Seu modelo conceitual é composto de três níveis de organização do conhecimento sobre o processo de experimentação: (i) nível meta – lida com o conhecimento comum a qualquer tipo de estudo experimental; (ii) nível de configuração – conhecimento específico para cada tipo de estudo experimental; e (iii) nível de instância – lida com o conhecimento para um estudo experimental específico. O ambiente possui um protótipo e um conjunto inicial de ferramentas para popular a infraestrutura sendo construída. O trabalho restringe-se a detalhar a abordagem proposta, não explora a comparação com outras abordagens e não apresenta nenhum estudo empírico realizado para avaliar a proposta apresentada.

Sjøberg et al. (2002) (SJøBERG, ANDA, *et al.*, 2002) apresentam SESE (*Simula Experiment Support Environment*), um ambiente baseado na web que apoia o gerenciamento de participantes de um experimento, captura o tempo gasto durante o experimento, habilita a coleta de artefatos, e monitora as atividades dos participantes. O participante é guiado através das atividades de um processo de software e durante esta execução, artefatos são coletados. As atividades relativas ao experimento são

executadas como parte das atividades de rotina do desenvolvedor. Este processo não pode ser adaptado de acordo com o experimento sendo executado. Um estudo para conduzir a replicação de um experimento controlado através da utilização da ferramenta é apresentado em (ARISHOLM, SJøBERG, *et al.*, 2002), embora não tenha produzido evidências, o estudo trouxe lições aprendidas que os autores sugerem como futuras melhorias para o ambiente.

Hochstein et al. (2008) (HOCHSTEIN, NAKAMURA, *et al.*, 2008) descrevem um *framework* como um conjunto integrado de ferramentas para apoiar experimentos em engenharia de software, o *Experiment Manager Framework*. O *framework* foi aplicado para experimentos de computação em alta performance (*high performance computing* - HPC). O ambiente orienta as etapas do desenvolvimento de um software e aplica heurísticas para inferir as atividades dos programadores. O estudo apresenta uma breve comparação com outras abordagens existentes e cita a realização de diversos utilizações do ambiente. Os autores afirmam ainda não ter realizado nenhum experimento controlado que possa trazer evidências dos benefícios esperados.

ExpTool (NETO, GARCIA, *et al.*, 2014) é uma ferramenta de apoio à condução de experimentos controlados cujo foco é o empacotamento do experimento. A ferramenta propõe-se a apoiar as etapas iniciais do experimento, assim como a operação do mesmo. Para a análise e interpretação, a ferramenta permite o registro de informações e arquivos obtidos externamente. Os autores afirmam que pretendem utilizar ontologia como mecanismo de formalização em trabalhos futuros. Nenhum estudo empírico de utilização da ferramenta foi apresentado.

Os trabalhos de pesquisa mencionados acima propõem ferramentas de apoio à condução do experimento. Por outro lado, alguns trabalhos focalizam na formalização do experimento mas não oferecem ferramentas de apoio à execução e análise. Garcia et al. (GARCIA, HOHN, *et al.*, 2008) apresentam uma ontologia para especificação de experimentos controlados chamada *experOntology*. O objetivo deles é formalizar o plano de um experimento através do empacotamento do conhecimento envolvido. É possível modelar o projeto do experimento, associando, manualmente, tratamentos aos participantes. Não é apresentado nenhum estudo empírico realizado para avaliar a ontologia ou que traga evidências dos benefícios de apoio aos pesquisadores.

Siy e WU (SIY e WU, 2009) também apresentam uma ontologia, porém seu foco é o projeto experimental. A proposta é aplicar restrições para verificar que tipos de ameaças à validade o plano modelado apresenta. Inicialmente eles implementaram apenas checagem de dois tipos de ameaças. A proposta deles foi aplicada à uma família de experimentos mas não há nenhum trabalho de avaliação que traga evidências sobre os benefícios da proposta.

Cartaxo et al. (CARTAXO, COSTA, *et al.*, 2012) apresentam uma linguagem específica de domínio (DSL) gráfica para modelagem de um plano experimental. A partir da modelagem, é gerado uma descrição do plano em um arquivo PDF. A proposta não dá suporte à execução do plano e não foi avaliada de forma empírica.

Embora os trabalhos de pesquisa existentes ofereçam propostas de apoio a execução de atividades envolvidas em um experimento controlado, elas possuem limitações e potencial para futuras melhorias. Em geral, os trabalhos que apresentam soluções para apoio à condução do experimento não explicitam seus mecanismos para a formalização de experimentos, o que dificulta a troca de informações entre os pesquisadores, assim como a replicação dos experimentos. Por outro lado, os trabalhos que propõem mecanismos de formalização não apresentam ferramentas de apoio à condução do experimento. Além disso, os trabalhos atuais não foram avaliados de forma através da condução de estudos empíricos que tragam evidências dos benefícios apresentados, nem tampouco oferecem suporte automatizado para a execução de experimentos de acordo com o seu projeto estatístico.

1.3. Objetivos

O objetivo geral desta tese de doutorado é propor uma abordagem que promova a formalização de experimentos controlados em engenharia de software. Em particular, é proposta uma linguagem específica de domínio para especificação de experimentos, assim como a adoção de técnicas de engenharia dirigida por modelo para gerar workflows customizados para a execução de um experimento. O trabalho tem os seguintes objetivos específicos:

- Investigar o estado da arte já proposto para o apoio à condução de experimentos em engenharia de software;

- Definir um conjunto de diretrizes para a concepção e implementação de uma linguagem para dar suporte à especificação de experimentos de engenharia de software;
- Definir e modelar aspectos experimentais que farão parte da linguagem específica de domínio;
- Verificar à viabilidade do uso de tecnologias orientadas à modelos para transformar as especificações de experimentos em workflows executáveis;
- Modelar diferentes estudos, utilizando a linguagem desenvolvida para avaliar a sua expressividade e completude;
- Avaliar a linguagem específica de domínio proposta através da condução de experimentos controlados no que diz respeito a compreensão de planos experimentais especificados na linguagem em relação à especificação em linguagem natural, assim como avaliar a utilização da linguagem na especificação de um plano experimental, destacando suas vantagens e limitações.

1.4. Questões de Pesquisa

O desenvolvimento do trabalho é norteado por três questões de pesquisa principais, as quais estão descritas a seguir. Visando responder tais questões, são conduzidos estudos que gerem evidências que permitam a formulação de respostas, mesmo que vinculadas a contextos específicos.

Questão 1: Quais ferramentas vêm sendo propostas para apoiar à realização de experimentos controlados em engenharia de software? Quais são os benefícios e limitações destas ferramentas?

Questão 2: Qual a viabilidade do uso de uma DSL (*domain-specific language*) para geração de workflows de atividades de participantes de um experimento seguindo seu respectivo design estatístico?

Questão 3: Quão completa e expressiva é a linguagem específica de domínio proposta para a formalização de experimentos?

Questão 4: Quais são os benefícios e limitações da abordagem proposta em relação ao uso de apenas linguagem natural para especificação de experimentos?

1.5. Metodologia

Com vistas a responder às questões de pesquisas apresentadas anteriormente, e atingir os objetivos do trabalho, foram definidas as seguintes etapas para o trabalho:

• Realização de pesquisa bibliográfica na área e o levantamento da fundamentação teórica necessária ao desenvolvimento do trabalho;

• Realização de uma revisão sistemática de trabalhos científicos, com o intuito de conhecer as abordagens já propostas para apoio automatizado à realização de experimentos controlados em engenharia de software;

• Definição e implementação de uma abordagem baseada em linguagem específica de domínios e desenvolvimento dirigido por modelos para customização de *workflows* de apoio à realização de experimentos em ES;

• Realização de um estudo exploratório para avaliar a expressividade e a viabilidade da linguagem proposta;

• Realização de avaliações empíricas visando avaliar os **benefícios** e **limitações** da abordagem proposta;

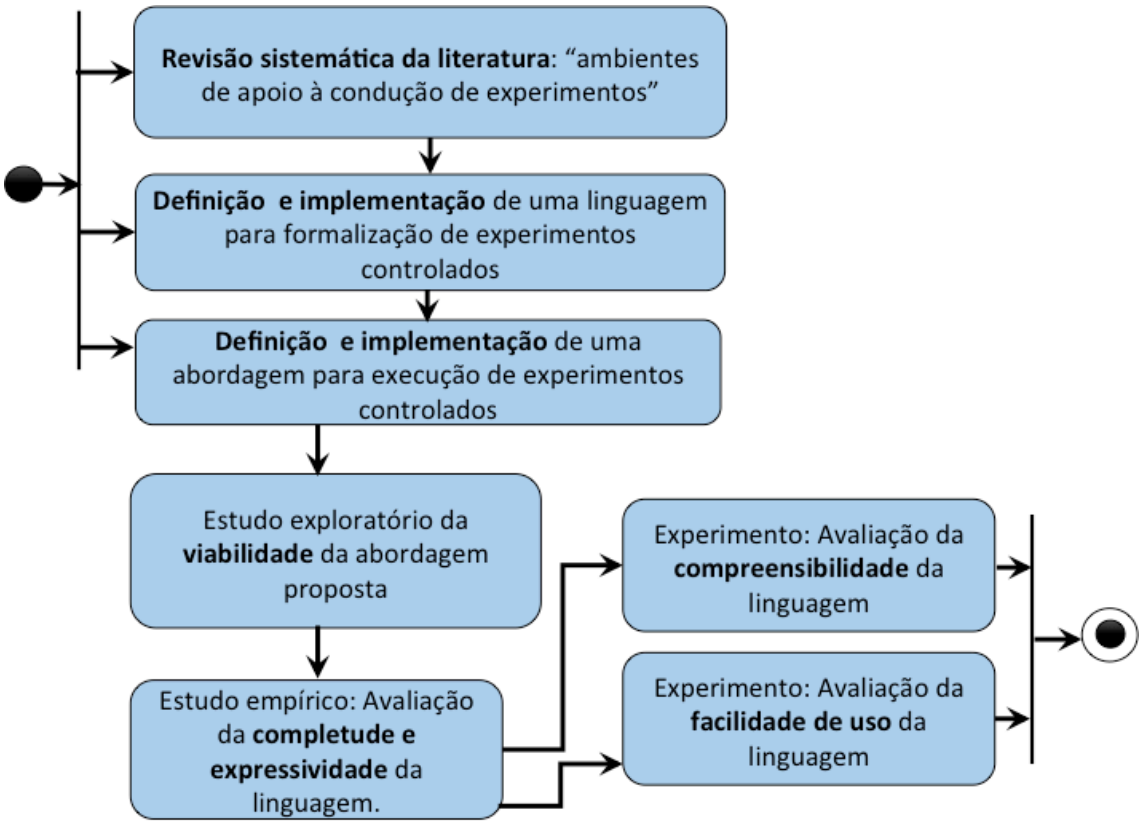


Figura 1: Metodologia da Pesquisa

1.6. Organização do Documento

O Capítulo 2 explora os assuntos que servirão de fundamentação teórica para o trabalho. O Capítulo 3 apresenta uma revisão sistemática da literatura relativa a ambientes (semi) automáticos de apoio a realização de experimentos controlados em Engenharia de Software. O Capítulo 4 descreve a linguagem específica de domínio proposta para à especificação de experimentos controlados em engenharia de software. A abordagem dirigida por modelos proposta para o apoio à execução de experimentos controlados em engenharia de software a partir da DSL definida é apresentada no Capítulo 5. O Capítulo 6 apresenta um estudo empírico cujo objetivo foi avaliar a expressividade e a completude da DSL proposta, através da modelagem de experimentos existentes da comunidade. Os estudos experimentais para avaliar a compreensão da linguagem e sua facilidade de uso na perspectiva dos experimentadores é apresentado no Capítulo 7. O Capítulo 8 apresenta trabalhos relacionados. Considerações finais e trabalhos futuros relativos a esta tese são apresentados no Capítulo 9.

2. Fundamentação Teórica

Este capítulo descreve de forma resumida trabalhos que representam a fundamentação teórica às proposições e discussões contidas nesse trabalho.

2.1. Estudos Empíricos na Engenharia de Software

Atualmente, a necessidade de realizar estudos empíricos é uma realidade para muitos trabalhos de pesquisa na ciência da computação, assim como em muitas outras áreas do conhecimento. Essa necessidade surgiu em resposta às aspirações da indústria, que tem exigido, cada vez mais, a comprovação da validade científica para que novas técnicas, ferramentas e métodos sejam aceitos para aplicação prática. Segundo Basili (BASILI, 2013), parte da razão para o lento progresso na realização de estudos empíricos na engenharia de software tem sido a cultura dentro dos departamentos de Ciência da Computação. Neste sentido, a comunidade de ES vem se esforçando e investindo cada vez mais na busca de comprovações científicas para seus novos métodos, ferramentas e tecnologias. *"A engenharia de software requer um mecanismo de pesquisa empírica que identifique os benefícios, limites e fronteiras das novas tecnologias"* (BASILI, 2013).

De acordo com Kitchenham et al (KITCHENHAM, DYBA e JORGENSEN, 2004), o objetivo da engenharia de software baseada em evidências é similar ao da medicina baseada em evidências, onde se deve buscar fornecer os meios pelos quais melhores evidências das pesquisas podem ser integradas com a experiência prática e os valores humanos no processo de tomada de decisão relativo ao desenvolvimento e manutenção de software.

A engenharia de software baseada em evidências tem focado seus esforços em diferentes tipos de estudos, principalmente em *surveys*, etnografias, mapeamentos sistemáticos, revisões sistemáticas da literatura, estudos de caso e experimentos controlados. Cada um dos métodos de avaliação empírica descreve diretrizes à respeito da coleta e análise de dados. Antes da utilização de um método, é necessário saber qual deles melhor se adéqua as necessidades da pesquisa em questão (EASTERBROOK, SINGER e ST, 2008).

Surveys são estudos utilizados para identificar características em uma população de indivíduos. Este tipo de pesquisa pode ser realizada através de questionários de coleta de dados, entrevistas estruturadas ou técnicas de registro de *logs* (WOHLIN, RUNESON, *et al.*, 2012). Os principais desafios deste tipo de estudo são a seleção da amostra a partir de uma população de indivíduos bem definida e da técnica de análise de dados usada para generalizar os resultados encontrados. Alguns exemplos de *surveys* em ES são (AALST, DONGEN, *et al.*, 2003) (MENS e TOURWÉ, 2004).

Etnografia é o tipo de estudo indicado para identificar ações, relações sociais e valores dos membros de uma comunidade. Para a engenharia de software, etnografia pode ajudar a entender como comunidades técnicas constroem uma cultura de práticas e estratégias de comunicação que as possibilitam desempenhar trabalho técnico colaborativamente (EASTERBROOK, SINGER e ST, 2008). A etnografia pode focar uma comunidade técnica grande (programadores Java, por exemplo) ou uma comunidade pequena (a equipe de programadores de um projeto, por exemplo). A observação participativa é uma forma especial de etnografia onde o pesquisador torna-se um membro da comunidade sendo estudada por um período de tempo. Ou seja, o pesquisador deixa de ser um estranho tentando entender a comunidade com o privilégio da visão de um membro.

O mapeamento sistemático, também conhecido como estudo de escopo, visa identificar o estado da prática ou de pesquisa em um tópico e, tipicamente, identificar tendências de pesquisas (WOHLIN, RUNESON, *et al.*, 2012). Esses estudos são guiados por questões amplas, mais exploratórias, e por muitas vezes elabora múltiplas questões de pesquisa (KITCHENHAM e CHARTERS, 2007). Este tipo de estudo vem sendo bastante explorado recentemente na engenharia de software.

Revisões sistemáticas são estudos que procuram sumarizar evidências acerca da base de conhecimento criada por estudos já realizados em uma área de interesse específica. Sua maior utilidade está na identificação de lacunas de pesquisa e na possibilidade de geração de novas hipóteses. Para ter valor científico, uma revisão sistemática deve ser (i) justa, ou seja, realizada de forma não tendenciosa; (ii) rigorosa, devendo seguir um protocolo bem definido; (iii) aberta, ou seja, o protocolo deve ser visível; e (iv) objetiva, com um protocolo reproduzível. As desvantagens de uma revisão sistemática é que elas requerem mais esforço do que revisões informais e

necessitam de pelo menos dois pesquisadores para minimizar o viés. As revisões sistemáticas podem ser relativas a estudos primários ou secundários. (KITCHENHAM e CHARTERS, 2007) (EASTERBROOK, SINGER e ST, 2008) (KITCHENHAMA, BRERETONA, *et al.*, 2009) (KITCHENHAM, PRETORIUS, *et al.*, 2010) são trabalhos que discutem revisões sistemáticas em engenharia de software e engenharia de software experimental.

Um estudo de caso é um termo bastante usado na literatura cuja definição formal destaca-o como sendo a investigação de um fenômeno realizada dentro de seu contexto real, especialmente quando é difícil distinguir claramente os limites entre o fenômeno e o seu contexto (WOHLIN, RUNESON, *et al.*, 2012). Um estudo de caso pode ser: (i) exploratório, objetivando derivar novas hipóteses e construir teorias; ou (ii) confirmatórios, usados para confirmar teorias existentes (PERRY, SIM e M. , 2004). A análise de dados é normalmente qualitativa, embora análises quantitativas possam ser realizadas. O projeto de um estudo de caso é bastante flexível e pode ser adaptado para se adequar ao ambiente do projeto. Estudos de caso são bem apropriados para responder questões do tipo “como” e “por que” onde o pesquisador não tem controle sobre as variáveis envolvidas e o foco está em um evento contemporâneo (KITCHENHAM, DYBA e JORGENSEN, 2004). Nesse tipo de estudo o pesquisador tem pouco ou nenhum controle sobre as variáveis de ambiente. É importante salientar, que o uso generalizado de estudos de caso tem mostrado, muitas vezes, o entendimento simplista que tem sido feito do termo, pois os relatos demonstram uma variação muito grande na qualidade dos estudos realizados que vai desde estudos bem organizados e estruturados até estudos completamente desorganizados, uns usando aplicações reais e outros usando aplicações não realistas.

Experimentos controlados são estudos que permitem testar uma hipótese de pesquisa para determinar os relacionamentos de causa e efeito entre variáveis de ambiente. Um experimento controlado dá maior poder ao pesquisador, que pode definir as ações a serem executadas no projeto e controlar as variáveis de ambiente, porém o contexto de experimentos são bem mais restritos que o de outros tipos de estudos.

Durante o desenvolvimento de uma pesquisa, os diferentes tipos de estudos científicos devem ser empregados, de forma a se complementarem, pois cada um tem

um objetivo importante dentro do contexto de uma pesquisa. Detalharemos, a seguir, experimentos controlados por serem o foco deste trabalho.

2.1.1 Experimentos Controlados em Engenharia de Software

Em termos práticos, o objetivo de um experimento na engenharia de software pode incluir mostrar que um particular método ou ferramenta é superior, de alguma maneira, a outro. Pode-se desejar mostrar que, para um particular elemento sob investigação, diferentes condições ambientais ou de qualidade dos recursos, podem afetar o uso ou os resultados de uma investigação.

O objetivo de um experimento deve ser traduzido numa hipótese formal. Normalmente existem duas hipóteses formais: a hipótese nula e a hipótese alternativa. A hipótese nula é assumida ser verdadeira a menos que os dados indiquem o contrário (WOHLIN, RUNESON, *et al.*, 2012) (JURISTO e MORENO, 2010). O experimento foca no desvio da hipótese nula, ou seja, testar a hipótese significa determinar se os dados são convincentes o bastante para rejeitá-la e aceitar a hipótese alternativa como verdadeira (PFLEEGER, 1995).

Um experimento é uma sequência de testes de um elemento sob investigação, e o *design* experimental descreve como estes testes serão organizados e executados. Ou seja, o design experimental é um planejamento completo da aplicação de condições experimentais diferentes aos participantes para determinar como as condições afetam o comportamento ou resultado de alguma atividade. Designs simples ajudam a tornar o experimento mais prático, minimizam o uso de tempo e recursos experimentais (PFLEEGER, 1995).

Segundo BASILI (BASILI, 1996), experimentos controlados trazem vários benefícios tais como: permitem a condução de estudos com foco bem definido e com potencial para resultados estatísticos significantes; permitem focar em varáveis e medidas específicas, e no relacionamento entre elas; ajudam na formulação da hipótese forçando a definição clara da questão sob investigação e permite maximizar o número de questões sendo avaliadas.

2.1.1.1 O Processo Experimental

Realizar um experimento controlado não é uma tarefa trivial, desde que é necessário planejar, executar e analisar cuidadosamente tais estudos como princípio essencial. O processo de realização de um experimento controlado é tipicamente composto das seguintes fases: definição, planejamento, execução, análise e empacotamento (WOHLIN, RUNESON, *et al.*, 2012), conforme ilustra a Figura 2:

(1) Definição: Nesta fase o estudo deve ser caracterizado em termos de problema, objetivos e metas. Ela determina a base para o experimento.

(2) Planejamento: É uma preparação de como o experimento deve ser conduzido. Compreende a definição da hipótese, seleção de variáveis (dependentes e independentes), seleção dos participantes e, determinação do design estatístico do experimento. Esta etapa também considera as ameaças à validade do experimento.

(3) Operação (execução): Esta fase segue a determinação do design. É composta de: (i) configuração do estudo (preparação), onde os participantes são escolhidos e os materiais necessários são preparados; e (ii) execução que coleta os dados que devem ser analisados.

(4) Análise e Interpretação: Responsável pela compilação dos dados coletados no estudo. Compreende estatística descritiva, redução do conjunto de dados e teste da hipótese.

(5) Apresentação e Empacotamento: Nesta fase a informação sobre o estudo é apresentada e o pacote, montado ao longo de todo o processo, é gerado. É uma fase essencial para permitir a replicação do estudo.

Cada uma destas fases está associada a execução de diferentes atividades que consomem e produzem artefatos relacionados ao estudo em questão. Um experimento controlado requer grande cuidado no planejamento para que ele possa fornecer resultados úteis e significativos (PFLEEGER, 1995). Além disso, planejar, conduzir e reportar as várias tarefas envolvidas em experimentos controlados em larga escala pode ser muito mais complexo e demorado sem a utilização de ferramentas de apoio apropriadas (HOCHSTEIN, NAKAMURA, *et al.*, 2008).

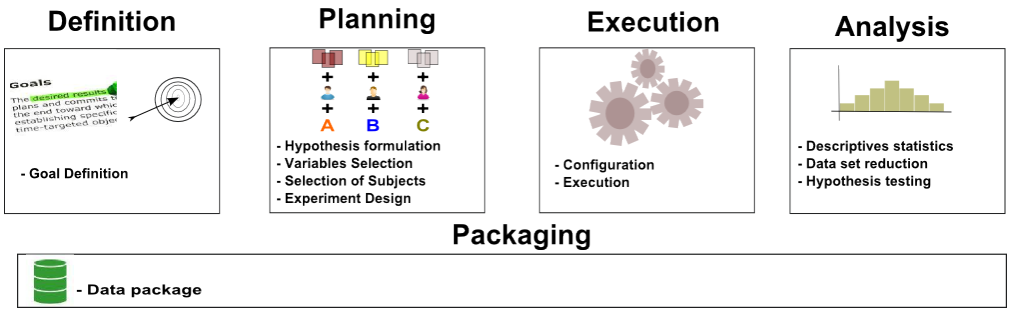


Figura 2: O Processo Experimental.

2.1.1.2 Replicação de Experimentos em ES

Experimentos controlados nos permitem testar uma hipótese de pesquisa e o relacionamento de causa e efeito entre as variáveis dependentes e independentes. Por outro lado, estudos controlados necessitam ser replicados pois um único experimento pode ser tendencioso e seus resultados limitados em termos de generalização das conclusões. Experimentos necessitam ser replicados várias vezes e sob diferentes condições antes de produzirem um pedaço de conhecimento (JURISTO, 2013). Só assim, atenderemos a necessidade de se construir uma base de conhecimento (KITCHENHAM, DYBA e JORGENSEN, 2004).

Segundo BASILI at el (1999) (BASILI, SHULL e LANUBILE, 1999), (SHULL, CARVER, *et al.*, 2008) existem vários tipos de replicação. Replicações nas quais a hipótese de pesquisa não varia: (i) rigorosas: que replica tão precisamente quanto possível o experimento original – visam aumentar a confiança na validade dos resultados de um experimento; (ii) replicações que variam a maneira como o experimento é executado – procuram aumentar a confiança nos resultados experimentais testando a mesma hipótese mas alterando detalhes do experimento de forma a atacar certas ameaças internas. Os outros tipos de replicações são as que variam a hipótese de pesquisa: (i) replicações que alteram as variáveis intrínsecas ao objeto de estudo (variáveis independentes) – para investigar quais aspectos do processo são importantes; (ii) replicações que alteram variáveis intrínsecas ao foco de avalição (variáveis dependentes) – variam a maneira como a efetividade é medida visando entender quais dimensões de uma tarefa (processo) resulta em melhores resultados; (iii) replicações que alteram as variáveis de contexto no ambiente no qual a solução é avaliada – visando

identificar fatores ambientais potencialmente importantes que afetam os resultados da solução sob investigação (MENDONÇA, MALDONADO, *et al.*, 2008).

2.2. Linguagem Específica de Domínio

Nos últimos anos, tem havido um aumento no uso de linguagens específicas de domínio (DSL) para o desenvolvimento de softwares. A adoção de DSLs eleva o nível de abstração e fornece facilidades para geração de modelos ou código fonte, especialmente usados na engenharia dirigida por modelos, trazendo, desta forma, o potencial para aumentar a produtividade do desenvolvimento de software em vários domínios (BRÄUER e LOCHMANN, 2007).

A definição dos conceitos de um domínio é realizada através da especificação de uma linguagem específica de domínio, que pode ser textual, gráfica ou híbrida. Uma DSL serve ao propósito de tornar os elementos de um domínio formalmente expressíveis e modeláveis. Toda DSL possui um metamodelo, incluindo sua semântica estática e sua sintaxe. A semântica de uma DSL deve ser bem documentada ou ser intuitivamente clara para o modelador, neste sentido, a DSL deve adotar conceitos do espaço do problema, de forma que o especialista do domínio reconheça sua "linguagem de domínio" (STAHL e VOLTER, 2005).

Com o crescente aumento do número e complexidade das DSLs surge a necessidade de uma abordagem para avaliação de DSLs. Uma DSL deve atender os objetivos dos *stakeholders*, através do aumento da sua satisfação. Apesar da necessidade, existe uma ausência de avaliações sistemáticas de linguagens específicas de domínio (GABRIEL, 2010). A fim de atender aos diversos *stakholders*, uma DSL deve ser avaliada usando diferentes critérios, que por vezes podem ser até conflitantes. Uma metodologia para guiar a avaliação de diferentes características e sub-características de uma DSL é apresentada em (KAHRAMAN e BILGEN, 2013).

2.3. Engenharia Dirigida por Modelos

A engenharia de software tem buscado cada vez mais elevar o nível de abstração na especificação, modelagem e implementação de sistemas de software. Neste cenário, uma das fortes iniciativas para alavancar o nível de abstração é engenharia dirigida por modelos (*Model Driven Engineering - MDE)* onde modelos não são apenas usados para

auxiliar no processo de desenvolvimento, mas são considerados os artefatos principais. Esta seção apresenta uma visão geral dos principais conceitos de MDE. Estes conceitos são explicados e fornecem uma base teórica para a Engenharia dirigida por modelos.

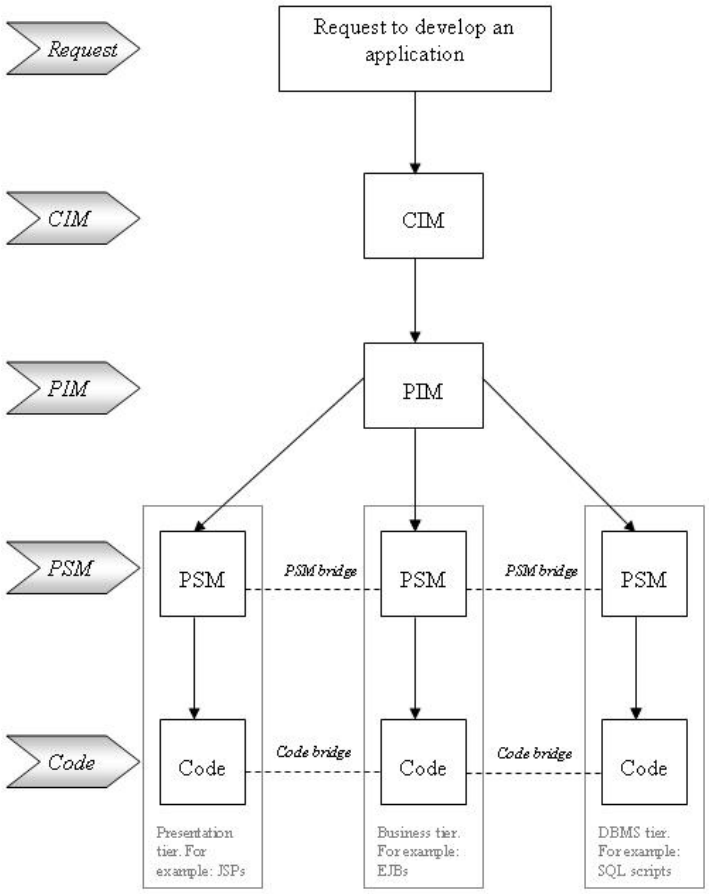


Figura 3: Arquitetura MDA

MDA (*Model-Driven Architecture*) é o termo definido pela OMG[1] (*Object Management Group*) para a abordagem MDE baseada em padrões da indústria e com a visão de independência de plataforma. De acordo com o OMG, MDA fornece uma abordagem para a especificação de sistemas e seu comportamento independentemente da sua plataforma. Em MDA temos três tipos de modelos (ver Figura 3), o modelo CIM (*Computacional Independent Model*), o modelo PIM *(Platform Independent Model*) e o modelo PSM (*Platform Specific Model*). O Modelo CIM é o modelo que descreve o domínio da aplicação sem nenhuma referência a uma tecnologia em particular ou menção de como o sistema é implementado. O modelo CIM apresenta o sistema em seu

[1] http://www.omg.org/

ambiente e ajuda a mostrar o que o sistema deverá realizar, ou seja é um modelo conceitual derivado de uma realidade com a finalidade de adquirir um entendimento melhor desta realidade. O modelo PIM é uma parte importante do MDA esse modelo descreve o sistema sem referência à plataforma e permite uma representação independente de plataforma para o sistema, este modelo é o primeiro passo para a criação do sistema, o modelo PIM pode ser transformado em um modelo PSM utilizando transformação entre modelos.

A especificação de modelos e a transformação de tais modelos em outros modelos ou artefatos de software é a essência da abordagem de desenvolvimento de software orientado a modelos (*Model-driven Development* - MDD) (STAHL e VOLTER, 2005). Os modelos associados a diferentes atividades do processo de software são construídos com base em rigores semânticos especificados em alguma linguagem de descrição de modelos. Dessa forma, a integração entre as informações oriundas de tais atividades pode ser feita através da especificação de regras de transformação que definem como um modelo pode ser derivado a partir de outro.

A transformação de modelos é uma estratégia importante quando se deseja separar os níveis de abstração do sistema, ou seja, a partir de um modelo mais abstrato ir construindo modelos mais específicos. No caso de sistemas de software, por exemplo, podemos trabalhar em um modelo independente de plataforma, um modelo PIM, e a partir de requisitos funcionais do sistema transformar este modelo em um modelo específico de plataforma, ou seja, em um modelo PSM. No contexto do DDM, é mandatório ser claro sobre a estrutura de um domínio para que seja possível formalizar esta estrutura. Esta é a base para qualquer tentativa de automação. A formalização é realizada na forma de um metamodelo. O metamodelo compreende a sintaxe abstrata e a semântica estática de uma linguagem, e é uma instância de um meta metamodelo. Um meta metamodelo descreve conceitos que podem ser usados para modelar um domínio.

2.4. Conclusões

Este capítulo apresentou as bases teóricas para a realização da presente proposta. A linguagem específica de domínio e a abordagem dirigida por modelos propostas neste trabalho está inserida no domínio da engenharia de software experimental, mais precisamente abordando a definição e execução de experimentos controlados.

3. Ambientes de apoio à Condução de Experimentos em ES: Uma Revisão Sistemática

Este capítulo descreve os resultados de uma revisão sistemática da literatura (RSL) que analisa o atual estado da arte em ferramentas, *frameworks*, ambientes e infraestruturas desenvolvidas para fornecer suporte automatizado para a condução de experimentos controlados em engenharia de software. Esta RSL encontrou sete ferramentas especificamente desenvolvidas ou adaptadas para conduzir experimentos em ES. Além disso, um conjunto de questões foi definido para avaliar a qualidade dos estudos. O objetivo do estudo é proporcionar resultados que apontem apropriadamente novos esforços de pesquisas e novas oportunidades relacionadas ao desenvolvimento do suporte automatizado ao processo da engenharia de software experimental.

Inicialmente, será caracterizado o problema central que é alvo da análise sistematizada (Seção 3.1). Após a caracterização do problema é apresentado o protocolo de revisão utilizado, onde são detalhados os principais tópicos do processo que conduziu a mesma (Seção 3.2). Os resultados da revisão são apresentados, em seguida, onde para cada questão de pesquisa são apresentados os seus resultados de acordo com os dados coletados (Seção 3.3). Por fim, uma análise qualitativa dos estudos encontrados é realizada (Seção 3.4), e em seguida as limitações do estudo e as discussões relativas aos resultados apresentados pela RSL são descritos (Seção 3.5 e Seção 3.6).

3.1. Caracterização do Problema

Recentes trabalhos de pesquisa vêm apresentando *guidelines* (diretrizes) para condução e/ou relato de estudos empíricos em engenharia de software (JEDLITSCHKA, CIOLKOWSKI e PFAHL, 2008) (JEDLITSCHKA e PFAHL, 2005: 95-104) (KO, LATOZA e BURNET, 2015), entretanto essa ajuda é meramente passiva, ou seja, sem nenhum tipo de automação ou suporte computacional.

Apesar da crescente necessidade de experimentos controlados em engenharia de software, a realização de tais experimentos é ainda muito complexa. A condução e replicação de tais experimentos em larga escala é ainda mais complexa. Uma razão para isto pode ser o pouco suporte ferramental existente para a definição, planejamento, execução, análise e replicação de tais experimentos.

3.2. Protocolo da Revisão

Esta seção descreve o protocolo da revisão usado para conduzir este estudo sistematizado, que foi definido baseado em guias específicos (KITCHENHAM e CHARTERS, 2007). O processo foi executado no início de 2012.

3.2.1 Questões de Pesquisa

Nossa pesquisa foi guiada por questões sobre ferramentas, *frameworks*, ambientes ou infraestruturas que fornecem apoio (semi) automatizado às diferentes fases de um experimento controlado. As quatro questões de pesquisa (QP) investigadas na revisão sistemática foram:

QP.1 Quais ferramentas vêm sendo propostas para dar apoio a experimentos controlados em engenharia de software?

QP.2 Quais fases de um experimento controlado os ambientes propostos estão dando suporte?

QP.3 Quais são as principais características/funcionalidades suportadas pelos ambientes propostos?

QP.4 Quem vem propondo ferramentas para suporte a experimentos controlados? Quando?

3.2.2 Critério de inclusão/exclusão

O critério de inclusão/exclusão definido para avaliar os estudos primários nesta revisão sistemática foram: (i) apenas estudos escritos em língua inglesa foram considerados; (ii) estudos que não apresentam ferramentas para a condução de experimentos controlados em ES foram excluídos; (iii) estudos que eram específicos para um único domínio da ES foram excluídos, tal como experimentos em testes; e (iv) estudos que apresentaram informações insuficientes para realizar a avaliação da ferramenta foram também excluídos.

3.2.3 Equipe

Uma equipe de seis pesquisadores realizou o estudo. Dois professores com dedicação exclusiva, um aluno de doutorado, e três alunos de mestrado. Todos os

pesquisadores são afiliados ao Departamento de Informática e Matemática Aplicada (DIMAp) da Universidade Federal do Rio Grande do Norte (UFRN), Brasil.

3.2.4 Procedimento de Decisão

Ao executar uma revisão sistemática, é importante definir como resolver possíveis situações de conflitos. Estes conflitos podem acontecer durante a seleção do estudo, a avaliação de qualidade, ou a extração dos dados. Portanto, foi definido um procedimento para apoiar as tomadas de decisão e consensos da seguinte forma: dois membros do time leem cada estudo selecionado, com atribuição realizada de forma randômica. Qualquer ocorrência de discordância entre estes pesquisadores foi resolvida por um terceiro revisor (um dos professores).

3.2.5 Origem dos dados e Estratégias de Pesquisa

A estratégia de pesquisa foi estabelecida em dois passos: (i) uma pesquisa manual nos principais veículos de publicação na área de engenharia de software experimental; e; (ii) uma análise da seção de referências de cada estudo primário em busca de novos artigos relacionados às questões da revisão. Para o primeiro passo as seguintes conferências e periódicos foram considerados: *Journal of Empirical Software Engineering, Journal of Systems and Software*, ACM/IEEE *International Symposium on Empirical Software Engineering and Measurement* (ESEM) (e os anteriores METRICS e ISESE), *Experimental Software Engineering Latin American Workshop* (ESELAW), e *International Conference on Software Engineering* (ICSE). É importante considerar que para as origens de publicação como ICSE, *Journal of Empirical Software Engineering*, e *Journal of Systems and Software*, nós limitamos o período de busca de Janeiro/2002 a Dezembro/2011, visto que estes eventos já são realizados há bastante tempo e não teríamos tempo para checar todos eles. Entretanto, no segundo passo, relativo a análise dos artigos referenciados, nós não consideramos nenhuma restrição em relação à origem e data da publicação.

3.2.6 Seleção do Estudo

O processo de seleção dos estudos foi realizado em duas etapas: (i) títulos e resumos dos artigos foram lidos/analisados durante a pesquisa manual buscando remover os artigos obviamente irrelevantes; e (ii) a leitura completa do estudo primário para avaliar se ele atendia aos critérios de inclusão/exclusão.

3.2.7 Avaliação de Qualidade

Foi realizada uma avaliação de qualidade dos estudos primários. Os critérios foram baseados em cinco questões:

QQ1. O estudo primário apresenta dados empíricos sobre o uso da solução proposta (ferramenta/ambiente/framework)?

QQ2. O estudo primário apresenta as técnicas e tecnologias usadas para o desenvolvimento da solução proposta?

QQ3. O estudo primário apresenta a metodologia utilizada nele?

QQ4. O estudo primário apresenta a arquitetura da solução proposta?

QQ5. A solução proposta é comparada com outras existentes?

Cada questão poderia ser respondida da seguinte forma:

QQ1: S (sim), um estudo empírico é explicitamente descrito no artigo; P (parcialmente), um estudo empírico está implícito no artigo; N (não), nenhum estudo empírico é mencionado nem pode ser inferido na leitura.

QQ2: S (sim), as tecnologias utilizadas para implementar a solução proposta são detalhadas; P (parcialmente), as tecnologias usadas para implementar a solução são citadas; N (não), as tecnologias utilizadas para construir a solução não são descritas e não podem ser inferidas a partir da leitura.

QQ3: S (sim), a metodologia usada pelo artigo é detalhada; P (parcialmente), a metodologia usada pelo artigo é citada; N (não), a metodologia utilizada não é mencionada e não pode ser inferida da leitura.

QQ4: S (sim), a arquitetura da solução é detalhada; P (parcialmente), a arquitetura da solução é brevemente descrita; N (não), a arquitetura da solução não é mencionada e não pode ser inferida da leitura.

QQ5: S (sim), a solução proposta é comparada com outras já existentes; P (parcialmente), outras soluções existentes são listadas e brevemente comparadas; N (não), outras soluções já existentes não são mencionadas e não podem ser inferidas da leitura.

Para cada resposta, uma pontuação foi definida como S = 1, P = 0.5, e N = 0, como realizado por Kitchenham et al (KITCHENHAM e CHARTERS, 2007). Estes critérios

foram usados na avaliação de qualidade para produzir uma pontuação de qualidade para todos os estudos primários. Esta avaliação foi realizada usando um formulário em planilha para registrar as respostas de cada questão (S/P/N) e as evidências dos estudos primários que justificam as respostas.

3.2.8 Processo de Extração dos Dados e Síntese

Como habitual, cada questão de pesquisa motivou a extração de alguns dados (Tabela 1). Além disso, informações gerais foram extraídas dos estudos tais como título, autores, editor e ano de publicação. A extração dos dados foi também auxiliada por formulários em planilha eletrônica onde fragmentos dos artigos investigados foram inseridos.

Tabela 1 Dados Extraídos

Questão de Pesquisa	**Atributo**	**Dados**
RQ.1	Ferramenta	Título do estudo, nome da ferramenta, origem (acadêmica ou industrial), ferramentas comparadas
RQ.2	Processo	Fases do experimento apoiadas
RQ.3	*Feature*	Funcionalidades
RQ.4	Mapeamento	Afiliação e país do primeiro autor

3.3. Resultados Encontrados

Esta seção apresenta os resultados encontrados de acordo com cada questão de pesquisa estabelecida na Seção 3.2.1. Esta revisão estava interessada em estudos primários que apresentem soluções (semi) automáticas de apoio à execução de experimentos em engenharia de software.

A Figura 4 ilustra o processo da revisão mostrando os estudos primários que foram encontrados e selecionados. O número de resultados inicial foi de cinquenta e cinco (55) artigos, durante a busca manual nas origens de publicação previamente especificados. Destes cinquenta e cinto (55), vinte e cinco (25) artigos foram selecionados a partir da leitura dos seus títulos e resumos. Estes vinte e cinco (25)

artigos foram completamente lidos visando descartar os estudos irrelevantes. Após esta seleção restaram nove (9) estudos primários. Em seguida, uma pesquisa foi realizada na seção de referências de cada artigo com o objetivo de selecionar outros estudos potencialmente relevantes (onde cada novo estudo selecionado tinha sua seção de referências também avaliada). Durante este processo foram encontrados sete (7) outros estudos primários. No passo final nós removemos um estudo que era específico para simulação de experimentos e, consequentemente, tinha recebido pontuação de qualidade igual a zero (0). O número final de estudos investigados foi quinze (15).

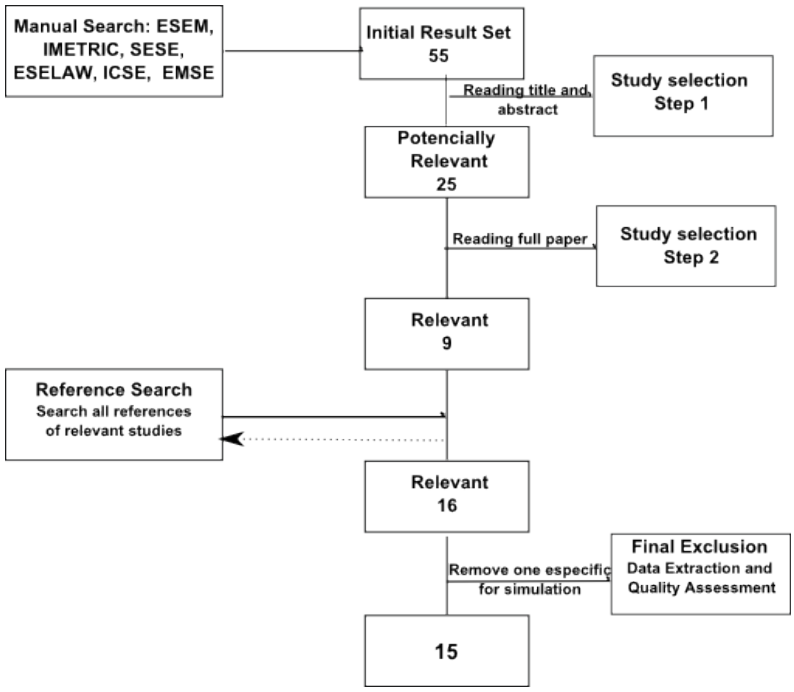


Figura 4: Processo de Pesquisa e Seleção de estudos.

Além das conferências/periódicos previamente selecionados, foram encontrados artigos de outras conferências durante a busca na seção de referência dos artigos. Estas conferências/periódicos foram: IEEE TSE (IEEE Transactions on Software Engineering), Advances in Computing Journal, NJC (Nordic Journal of Computing), NWPE (Nordic Workshop on Programming and Software Development Tools and Techniques), JIISIC (Jornadas IberoAmericanas en Ingenieria del Software e Ingenieria del Conocimiento), e ICECCS (International Conference on Electronics, Communication and Computer Science).

Os resultados encontrados para cada questão de pesquisa são discutidos nas subseções seguintes.

3.3.1 Ferramentas de suporte a experimentos controlados (QP.1)

As ferramentas de ES empírica investigadas nesta revisão foram desenvolvidas para simplificar o trabalho complexo envolvido na realização de um experimento controlado. Estas ferramentas apoiam algumas atividades do processo de engenharia de software experimental.

Os sete ambientes – ferramentas, frameworks, infraestruturas – seguintes foram encontrados: Simula Experiment Support Environment (SESE), experimental Software Engineering Environment (eSEE), value-based empirical research (VBER), Ginger2, Experiment Manager framework, Framework for Improving the Replication of Experiments (FIRE), e Mechanical Turk. A seguir é apresentado um pequeno resumo de cada uma das sete ferramentas e seus respectivos identificadores (ID) de acordo com a Tabela 2:

SESE (ARISHOLM, SJøBERG, *et al.*, 2002) (ARISHOLM, SJøBERG, *et al.*, 2002) (KARAHASANOVIC, ANDA, *et al.*, 2005) (SJøBERG, ANDA, *et al.*, 2002): É uma ferramenta baseada na web que apoia o gerenciamento de participantes, captura o tempo gasto durante o experimento, possibilita a coleta de artefatos, e o monitoramento das atividades dos participantes. Seu ponto fraco é a coleta e análise dos dados (PS1, PS2, PS3, PS4).

eSEE (DIAS NETO, BARCELOS, *et al.*, 2004) (MIAN, TRAVASSOS, *et al.*, 2004) (MIAN, TRAVASSOS e ROCHA, 2004) (MIAN, TRAVASSOS e ROCHA, 2005) (TRAVASSOS, SANTOS, *et al.*, 2008) (CHAPETTA, SANTOS e TRAVASSOS, 2005): É um ambiente para gerenciar vários tipos de estudos empíricos em engenharia de software. Ele funciona em três níveis de organização do conhecimento sobre o processo experimental: conhecimento geral para qualquer tipo de estudo experimental (*meta level*), conhecimento específica para cada tipo de estudo experimental (*configuration level*), e conhecimento para um estudo experimental específico (*instance level*). Existe um protótipo da ferramenta e um conjunto inicial de ferramentas utilizadas para compor a infraestrutura eSEE vem sendo desenvolvido (PS5, PS6, PS7, PS8, PS9, PS10).

VBER (BIFFL e WINKLER, 2007): É um framework para dar apoio a fase de planejamento de um experimento (value-based framework). Ele auxilia os *stakeholders* na análise de riscos e benefícios das potenciais variáveis de um estudo empírico (PS3).

Mechanical Turk (STOLEE e ELBAUM, 2010): É uma ferramenta *crowdsourcing* usada para apoiar a realização de estudos empíricos no contexto da engenharia de software. Ela oferece funcionalidades para acessar e gerenciar um grande número de participantes do estudo e habilita o recrutamento do tipo certo de participantes para avaliar uma técnica ou ferramenta da ES (PS11).

Ginger2 (TORII, NAKAKOJI, *et al.*, 1999)*:* É um ambiente experimental construído com base no framework CAESE. Embora o CAESE apoie o processo completo, Ginger2 está restrito as fases de execução e análise. Seu ponto forte é a variedade de dados de baixo nível que são coletados (PS15).

FIRE (MENDONÇA, MALDONADO, *et al.*, 2008): É um framework (*Framework for Improving Replication of Experiments*) que foca no compartilhamento de conhecimento visando permitir cooperação entre grupos de pesquisa. Ele é composto de sete passos que considera que os pesquisadores estão colaborando de perto usando uma variedade de mecanismos de colaboração. Seu ponto fraco é que ele dá apenas um apoio conceitual, ou seja, não tem um software de apoio (PS13).

Experiment Manager Framework (HOCHSTEIN, NAKAMURA, *et al.*, 2008): Este framework é um conjunto integrado de ferramentas de apoio à experimentos em engenharia de software. Ele foi utilizado apenas em experimentos de computação de alta performance (*high performance computing* - HPC). Ele auxilia os participantes aplicando heurísticas para inferior as atividades dos programadores. Sua ferramenta de análise é simples (PS14).

A Tabela 2 apresenta os 15 estudos incluídos na revisão, mostrando: ID, título, ano de publicação, nome do ambiente (ferramenta, *framework* ou infraestrutura) relacionado, universidade ou empresa que desenvolveu o ambiente, e a fonte da publicação.

Nós podemos observar que alguns estudos são relacionados ao mesmo ambiente experimental, onde cada estudo foca em alguma estágio e/ou funcionalidades do ambiente investigado. Este é o caso dos ambientes SESE e eSEE: PS1, PS2, PS3, e PS4 referem-se ao ambiente SESE; e PS5 até PS9 referem-se à infraestrutura eSEE. Para os estudos que são parte de um trabalho colaborativo nós preenchemos a tabela com

informações relacionadas ao primeiro autor. Outro ponto para esclarecer é que a ferramenta *Feedback-collecting*, descrita em PS2, é implementada como parte da ferramenta web *Simula Experiment Support Environment* (SESE).

Tabela 2 Estudos Investigados

ID	Título	*Ano*	Ferramenta/Ambiente	*Universidade/ Indústria*	Publicação
S1	Conducting realistic experiments in software engineering	2002	Web-based Simula Experiment Support Environment (SESE)	Simula Research Laboratory	ISESE
S2	Collecting Feedback During Software Engineering Experiments	2005	Feedback-collecting tool	Simula Research	ESEM
S3	A Web-based Support Environment for Software Engineering Experiments	2002	Simula Experiment Support Environment (SESE)	Simula Research Laboratory + KompetanseWeb AS	NJC
S4	SESE – an Experiment Support Environment for Evaluating Software Engineering Technologies	2002	Simula Experiment Support Environment (SESE)	Simula Research Laboratory + KompetanseWeb AS	NWPER
S5	Infrastructure for SE Experiments Definition and Planning	2004	experimental Software Engineering Environment (eSEE)	COPPE/UFRJ	ESELAW
S6	eSEE: a Computerized Infrastructure for Experimental Software Engineering	2004	experimental Software Engineering Environment (eSEE)	COPPE/UFRJ	ESELAW
S7	A computerized infrastructure for supporting experimentation in software engineering	2005	experimental Software Engineering Environment (eSEE)	COPPE / UFRJ	ESELAW
S8	Supporting Meta-Description Activities in Experimental Software Engineering Environments	2005	Meta-configurator from experimental Software Engineering Environment (eSEE)	COPPE/UFRJ	ESELAW
S9	An environment to support large scale experimentation in software engineering	2008	experimental Software Engineering Environment (eSEE)	COPPE/UFRJ	ICECCS
S10	Towards a Computerized Infrastructure for Managing Experimental Software Engineering Knowledge	2004	experimental Software Engineering Environment (eSEE)	COPPE/UFRJ	JIISIC
S11	Exploring the use of crowdsourcing to support empirical studies in software engineering	2010	Mechanical Turk	University of Nebraska	ESEM
S12	Value-Based Empirical Research Plan Evaluation	2007	value-based empirical research (VBER) planning framework	Vienna Univ. of Technol.	ESEM
S13	A Framework for Software Engineering Experimental Replications	2008	FIRE - Framework for Improving the Replication of Experiments	Salvador University	ICECCS
S14	An Environment for Conducting Families of Software Engineering Experiments	2008	Experiment Manager framework	University of Nebraska	Advances in Computers
S15	Ginger2: An Environment for Computer-Aided Empirical Software Engineering	1999	CAESE Framework and Ginger2	Nara Institute of Science and Technology	IEEE TSE

PS11 realizou uma adaptação na infraestrutura do *Mechanical Turk*. Ele compreende, de uma forma geral, um serviço que permite que pessoas colaborem para completar tarefas que necessitam de interação humana (envio de artefatos) ou respondam questões em *surveys*. No estudo em questão a ferramenta foi usada para apoiar o processo de execução de um experimento.

FIRE (PS13) não é especificamente uma ferramenta, mas um framework que descreve sete passos para executar e replicar um experimento.

Embora o estudo PS15 tenha sido publicado em 1999, nós o incluímos na revisão após a busca na seção de referência dos artigos pela fato do mesmo apresentar o Ginger2 que é uma ferramenta baseada no *framework* CAESE (*Computer-Aided Empirical Software Engineering*). Este framework define uma solução completa para condução de experimentos, mas, de acordo com o estudo, Ginger2 implementa apenas suporte às fases de coleta de dados e análise de um experimento. A principal funcionalidade desta ferramenta é a coleta de vários tipos de dados como cliques do mouse e eventos do teclado, movimentos tridimensionais, nível de resistência da pele, e vídeos gravados em fitas.

Não foi possível encontrar informações sobre o estágio atual de desenvolvimento destas soluções. A ferramenta eSEE é a única que relata um site na web, embora nós não tenhamos conseguido encontrar informações atualizadas nele.

3.3.2 Fases Suportados de um Experimento Controlado (RQ.2)

Esta questão da revisão visa apontar as fases de um processo experimental que são cobertas pelas ferramentas investigadas. Um experimento controlado típico é composto das seguintes fases, conforme detalhado no Capítulo 2: (1) definição, (2) planejamento, (3) execução, (4) análise e (5) empacotamento.

Tabela 3 Fases de um Experimento Apoiadas pelas Ferramentas

	(1)	**(2)**	**(3)**	**(4)**	**(5)**
SESE	■	■	■		■
eSEE	■	■	■		■
VBER		■			
Mechanical Turk	■	■	■		
FIRE	■	■	■		■
Ginger2	■	■	■	■	■
Experiment Manager			■	■	

A Tabela 3 apresenta as fases cobertas por cada ambiente experimental encontrado nesta revisão sistemática. Quase todos os ambientes empíricos selecionados no estudo dão algum tipo de suporte nas fases de definição, planejamento e execução do experimento. A fase de análise é suportada por apenas duas das ferramentas. Entretanto, quatro dos sete ambientes estudados dão suporte ao passo de empacotamento, que é muito importante para alcançar a replicação, embora nenhuma delas defina um formato explícito ou padrão para empacotamento dos experimentos. O *framework* CAESE é o único que menciona dar suporte ao processo completo, mas o Ginger2 não implementa todas estas fases, como já foi destacado na seção anterior.

Em geral, nós não encontramos mais detalhes sobre o suporte fornecido para cada fase do processo experimental nos estudos primários selecionados. Além disso, cinco estudos são *short papers*, e portanto já não era esperado encontrar muitas informações sobre os ambientes neles descritos.

3.3.3 Funcionalidades Oferecidas pelas Ferramentas Empíricas (RQ.3)

Embora muitas fases do processo experimental estejam sendo cobertas pelas ferramentas, é fundamental entender o nível de suporte fornecido por cada diferente ferramenta para cada diferente fase. Cada ferramenta tem funcionalidades relativas às fases apoiadas de um experimento controlado. A Tabela 4 mostra algumas funcionalidades descritas pelos estudos primários.

Pelo fato do foco de alguns artigos ser em partes específicas das ferramentas ou mesmo na sua abordagem e implementação, não é possível entender e detalhar todas as funcionalidades das ferramentas. Para os ambientes que têm mais que um estudo publicado, foi mais fácil conseguir detalhes sobre as funcionalidades, tais como os ambientes eSEE e SESE. Para o FIRE e VBER, as funcionalidades mostradas na tabela estão associadas com o propósito descrito para o framework.

Tabela 4 Principais Funcionalidades Encontradas

Funcionalidades	**Ferramentas**
Definir o experimento (questionários, tarefas, artefatos e papéis)	SESE, eSEE, Ginger2
Permitir ao pesquisador definir o número e tipo de participantes.	SESE, eSEE
Permitir ao participante o preenchimento de questionários e o download de tarefas e outros documentos. O participante executa a tarefa, responde a questão e faz o upload dos documentos finalizados.	SESE, eSEE, Mechanical Turk
Calcular o tempo gasto nas atividades	SESE, Mechanical Turk, Experiment Manager framework, Ginger2
Monitorar o experimento (o progresso de cada participante)	SESE, eSEE
Coletar e armazenar os dados	SESE, Ginger2, Experiment Manager framework
Publicar o experimento em um repositório de processos.	eSEE
Especificar e visualizar o plano experimental como um processo bem definido.	eSEE
Elaborar os documentos consumidos/produzidos durante o processo experimental.	eSEE
Tornar as tarefas dos experimentos disponíveis	eSEE, Ginger2,

	Mechanical Turk
Controlar o experimento	eSEE
Registrar as lições aprendidas	eSEE
Coletar feedback dos participantes	Feedback-collecting tool (SESE)
Caracterizar o estudo (usando GQM)	Ginger2, VBER
Apoiar Design estatístico do Experimento	Ginger2
Apoiar a análise de dados	Ginger2, Experiment Manager framework
Integrar dados entre diferentes ferramentas, integrar diferentes ferramentas de controle, integrar ferramentas de análise.	Ginger2
Apoiar o empacotamento	eSEE, SESE, Ginger2, FIRE
Gerenciar o pagamento (dos participantes)	Mechanical Turk
Compartilhar conhecimento intra-grupo (replicação interna) e inter-grupo (replicação externa).	FIRE
Permitir a interação entre grupos através de email ou chamadas telefônicas.	FIRE
Executar estudo piloto.	FIRE
Ajudar a identificar conflitos em potencial (indicar riscos do projeto).	VBER
Elicitar os principais stakeholders (indústria e academia) e suas expectativas.	VBER

Em um processo experimental, após estabelecer a hipótese e determinar quanto controle é necessário para as variáveis envolvidas, seja automaticamente ou manualmente, é necessário escolher o design estatístico do experimento. Esta escolha determina como organizar (participantes, material experimental, e tratamentos), para executar o experimento, e analisar os dados coletados usando o método de análise estatística específico. Entretanto, nenhuma ferramenta detalha qual suporte é dado na configuração do design do experimento durante o planejamento.

O único ambiente que explicitamente menciona a existência de ferramentas internas para suporte a análise observacional e computacional é o Ginger2. Outra questão que não é atendida pelos ambientes é como definir métricas a serem coletadas durante a execução do experimento. Ginger2 menciona uma ferramenta de métricas estatística (*Statistical Metrics Tool*) para análise de dados que calcula e retorna vários valores e métricas estatísticas que tenham sido definidas pelo experimentador mas ela não detalha como esta ferramenta funciona.

Finalmente, embora Ginger2 e SESE possibilitem a definição do experimento, elas possuem um processo predeterminado que não pode ser ajustado de acordo com as necessidades de cada novo experimento.

3.3.4 Desenvolvimento de Ferramentas para Experimentos em ES (RQ.4)

Nesta revisão, o período de varredura dos artigos foi previamente definido como sendo entre 2002 e 2010, para o primeiro passo da pesquisa, buscando artigos nas conferências e periódicos definidos. No segundo passo do estudo, foram incluídos estudos encontrados nas seções de referências dos artigos já selecionados, independente do período de publicação destes artigos, como já descrito no protocolo da revisão. O principal objetivo desta decisão foi tentar capturar um conjunto maior de trabalhos de pesquisa relacionados. Embora o número de execução de experimentos em ES venha crescendo, os resultados do estudo demonstram que pesquisas visando automatizar a condução de experimentos em ES é ainda pequeno. A Tabela 5 ilustra a concentração dos estudos por ano.

Tabela 5 Distribuição dos Estudos por Ano

Ano	Estudos Encontrados	Ano	Estudos Encontrados
1999	1	2006	0
2000	0	2007	1
2001	0	2008	3
2002	3	2009	0
2003	0	2010	1
2004	3	2011	0
2005	3	2012	0

A Tabela 6 mostra que os estudos investigados foram originados de cinco diferentes países. É importante enfatizar que alguns estudos têm pesquisadores de diferentes origens em colaboração mas nós consideramos apenas a origem do primeiro autor. Entre os estudos selecionados, 47% vem do Brasil (principalmente da COPPE/UFRJ) e os outros 27% vem do *Simula Research Laboratory*/ Noruega.

Tabela 6 Distribuição de Estudos por Afiliação e País de Origem

País	Afiliação	Distribuição dos Estudos	Ambiente
Japan	Nara Institute of Science and Technology	1	Ginger2
USA	University of Nebraska	2	Experiment Manager Mechanical

			Turk
Norway	Simula Research Laboratory	4	SESE
Austria	Vienna University of Technology	1	VBER
Brasil	COPPE / UFRJ	6	eSEE
Brasil	Salvador University	1	FIRE

3.4. Avaliação Qualitativa dos Estudos Primários

A avaliação de qualidade nos dá alguns detalhes dos estudos selecionados, que nos ajudam a entender melhor os resultados da revisão. A Tabela 7 apresenta os resultados da avaliação. As colunas da tabela são: identificador do estudo (ID), questão de qualidade (QQ1 até QQ5), pontuação final (P), e um indicador se o estudo é ou não um *short paper* (SP). Cada estudo foi avaliado através das questões de qualidade de acordo com as possíveis respostas (como detalhado na Seção 3.2): sim (S), parcialmente (P) ou não (N). Os estudos estão ordenados pela pontuação final para facilitar a visualização de como o conjunto total de estudos se saiu na avaliação. Uma pontuação baixa na tabela não significa que o estudo é irrelevante ou não está bem escrito. Todos os estudos são importantes e eles trazem contribuição interessante para a área da engenharia de software experimental.

Apenas dois estudos receberam pontuação 4 ou superior. Uma avaliação mais profunda mostra que ambos estudos com essa pontuação realizaram os requisitos relacionados com a maioria das questões de qualidade, mesmo que eles não tenham apresentado muitos detalhes sobre a arquitetura da solução proposta, e tenham descrito brevemente uma comparação da solução proposta com outras já existentes. É importante salientar que estas comparações foram limitadas à ferramentas desenvolvidas para dar apoio a realização de *surveys*, onde a maioria foi projetada por psicólogos, e não desenvolvidas para dar apoio a realização de estudos experimentais.

Apenas um dos estudos selecionados (PS14) tem uma comparação detalhada com as ferramentas/ambientes existentes (QQ5). Isto pode estar relacionado ao fato de não existir, atualmente, um grande número de ferramentas propostas para o apoio a engenharia de software experimental, o que pode ser observado com o resultado geral desta revisão sistemática.

A avaliação de qualidade também mostra que 80% dos estudos primários apresentaram a metodologia do ambiente proposto (QQ3). Além disso, a maioria dos estudos com baixa pontuação são short papers, o que é razoável considerando que estes estudos têm espaço restrito para prover informações sobre suas respectivas abordagens.

Tabela 7 Resultado da Avaliação de Qualidade dos Estudos

ID	QQ1	QQ2	QQ3	QQ4	QQ5	PF	SP
PS3	S	S	S	P	P	4	N
PS4	S	S	S	P	P	4	N
PS15	S	P	S	S	N	3,5	N
PS2	S	P	S	P	N	3	N
PS14	P	N	P	S	S	3	N
PS7	N	N	S	S	N	2	S
PS8	N	S	N	S	N	2	N
PS9	N	N	S	S	N	2	N
PS10	N	N	S	S	N	2	N
PS13	S	N	S	N	N	2	N
PS6	S	N	S	N	N	2	N
PS5	N	P	P	P	N	1,5	S
PS1	P	N	P	N	N	1	S
PS12	P	N	P	N	N	1	S

PS11	N	P	N	N	N	0,5	S

Uma descoberta importante a ser observada é que apenas 30% dos estudos primários descrevem algum tipo de avaliação empírica que foi realizada para testar hipóteses sobre as próprias soluções propostas (QQ1). Isto pode ser em virtude da maioria dessas soluções estarem ainda em desenvolvimento com apenas parte de suas funcionalidades planejadas implementadas no momento da escrita do artigo.

Por fim, a Tabela 7 também mostra que é difícil analisar detalhes sobre todas as soluções propostas, porque a maioria dos estudos não descrevem detalhes sobre tecnologia, arquitetura e/ou funcionalidades. Isto é refletido no número de estudos – cerca de 66% – que apresenta escore menor ou igual a 2, como retrata a tabela.

3.5. Limitações do Estudo

As limitações desta revisão sistemática estão associadas a estratégia de pesquisa. Nosso primeiro planejamento foi realizar pesquisas automáticas e manuais. Quando nós inicializamos a definir nossa *string* de busca percebemos que nós estamos em um escopo muito largo devido à diversidade de nossa pesquisa. Muitos trabalhos de pesquisa em engenharia de software apresentam frameworks, ferramentas, ambientes e/ou infraestruturas para outros contextos diferentes da engenharia de software empírica. Além disso, também existem muitos estudos que descrevem estudos experimentais e experimentos controlados. Por estas razões, foi extremamente difícil realizar uma busca automática sem que a mesma resultasse um imenso número de resultados (milhares) que não eram relacionados aos propósitos da revisão. Como resultado, definimos executar apenas a estratégia de busca manual. Nós sabemos que há um esforço significativo ao realizar buscas automáticas devido ao número de estudos não relevantes que são retornados, mas por outro lado nós podemos garantir a coleta de estudos relevantes ao selecionar as conferências e periódicos. Estratégias similares vêm sendo adotada por outras revisões sistemáticas (ALVES, NIU, *et al.*, 2010).

3.6. Discussões

Após executar a revisão sistemática foram identificadas algumas fragilidades e oportunidades de melhoria futura em relação as ferramentas/ambientes/infraestruturas de apoio à condução de experimentos. Nesta seção, nós apresentamos estas novas

perspectivas baseados no resultado da revisão que são fundamentais para o desenvolvimento da proposta descrita neste trabalho.

3.6.1 Customização da execução de acordo com o design do experimento

A proposta das ferramentas investigadas é facilitar o planejamento e a condução de um experimento, minimizando ameaças a validade e reduzindo o tempo gasto na preparação, execução e análise de um experimento controlado. Entretanto, eles não mencionam como configurar o design experimental ou como organizar a execução do experimento de acordo com o design estatístico escolhido, nem mesmo para os mais conhecidos, tais como o completamente aleatorizado (*completely randomized design* - CRD), completamente aleatorizado em blocos (*randomized complete block design* - RCBD), ou quadrado latino (*Latin square* - LS). Nossa experiência na condução de experimentos controlados em ES tem demonstrado que não é uma tarefa trivial organizar um experimento de acordo com o design estatístico durante a fase de planejamento. Além disso, esta atividade consome tempo e, frequentemente, é propensa a erros. O fornecimento de auxílio em como experimentos controlados devem realmente ser organizados, de acordo com o design estatístico selecionado, pode não apenas reduzir o esforço de aprendizado estatístico dos pesquisadores na engenharia de software experimental, mas também encorajar pesquisadores que não são especialistas a realizarem tais experimentos.

3.6.2 Melhoria na capacidade de análise

Em geral, as poucas abordagens encontradas nesta revisão para apoiar a condução de experimentos controlados em ES contemplam funcionalidades relacionadas às diferentes fases do processo experimental. Embora a fase de análise tenha sido contemplada por duas delas, todos os ambientes demonstram pontos fracos que precisam ser endereçados, tais como: (i) auxílio para configurar o design estatístico do experimento; (ii) geração automática de workflows dos procedimentos de execução para cada participante visando facilitar a coleta automática de seus dados; e (iii) finalmente, capacidade de análise que facilita a produção de gráficos e dados que ajudam análise do estudo de acordo com o design estatístico escolhido.

3.6.3 Guias e coleta automática de dados

Da mesma forma que o conhecimento requerido para definir um estudo experimental é considerável, o esforço para executar e gerenciar um grande volume de informações coletadas em um experimento é igualmente substancial. Neste contexto, algumas ações são necessárias visando minimizar o esforço da coleta manual de dados, e o tempo consumido na execução do experimento. Isto contribui para melhorar a precisão e qualidade dos dados. Desta forma, há uma forte necessidade de que os ambientes existentes possam fornecer suporte à execução de experimentos para garantir resultados melhores e mais precisos para o experimento.

Sendo assim, tais ambientes devem possibilitar que os participantes mantenham-se atualizados em relação as suas atividades através dos guias e da coleta automática dos dados. Isto pode simplificar o processo, desde que os participantes não necessitam coletar dados tais como o tempo consumido por cada atividade realizada, e pode acompanhar suas atividades através de um workflow sistemático e customizado das suas tarefas. Isto representa uma deficiência dos ambientes existentes que quando atendida poderá minimizar o trabalho extra de coleta de dados e aumentará a precisão da coleta de dados.

3.6.4 Flexibilidade dos Ambientes

Pode ser observado após os resultados desta revisão sistemática que apenas alguns dos ambientes atuais de experimentação em engenharia de software são adaptáveis e extensíveis para as necessidades específicas de certos experimentos. Em particular, estes ambientes deveriam atender aos seguintes requisitos: (i) flexibilidade para integração com ferramentas externas – em muitos domínios, a execução de um experimento controlado envolve uma variedade de ferramentas externas que devem ser integradas e monitoradas para apoiar um experimento completo, tais como ferramentas de gerência de processo, ambientes de desenvolvimento integrado (*integrated development environments* - IDE), ferramentas de testes, e ferramentas estatísticas. Portanto, esta capacidade é essencial para melhorar o desempenho e a precisão dos experimentos; e (ii) flexibilidade para estender o ambiente – visando atender uma variedade de requisitos dos experimentos em diferentes domínios, é também fundamental promover a extensibilidade do ambiente para apoiar diferentes designs experimentais, métricas, estratégias de coleta de informações pelos participantes, entre outros.

3.7. Conclusões

Este capítulo reportou os resultados de uma revisão sistemática sobre suporte (semi) automatizado à condução de experimento em engenharia de software. Os resultados indicam um número restrito de ambientes/infraestruturas/frameworks/ferramentas (sete no total). Entretanto, existem poucos estudos empíricos realizados para validar o uso de tais ambientes. Foi identificado que um melhoramento em potencial para as ferramentas existentes são o suporte à customização de forma a atender necessidades específicas dos experimentos, dando mais flexibilidade para estender as suas funcionalidades básicas, e permitir a integração com ferramentas externas.

Inspirados nos resultados desta revisão foi idealizado o ambiente dirigido por modelo para suporte à condução de experimentos controlados em ES proposto neste documento. Nossa experiência anterior no desenvolvimento de abordagens dirigidas por modelo para a especificação de linhas de processo e monitoramento de processos de software (ALEIXO e FREIRE, 2012) (ALEIXO, FREIRE, *et al.*, 2011) (FREIRE, ALEIXO, *et al.*, 2011) (FREIRE, ALENCAR, *et al.*, 2012) foi benéfica para definir um ambiente automatizado que pode ser adaptado e flexível para diferentes necessidades de um experimento controlado.

4. ExpDSL: Uma Linguagem para Formalização de Experimentos Controlados em Engenharia de Software

Após uma análise criteriosa dos trabalhos relacionados foi possível a estruturação desta linguagem específica de domínio para a formalização de experimentos controlados no contexto da engenharia de software. Este capítulo apresenta ExpDSL – uma linguagem para experimentação em ES.

4.1. Visão Geral da Linguagem ExpDSL

ExpDSL é organizada em três visões, cada uma responsável por especificar uma perspectiva diferente de um experimento controlado:

- ***Visão de Plano Experimental***: configura, entre outros aspectos, os fatores a serem controlados e o projeto estatístico do experimento a ser executado;
- ***Visão de Processo***: define tarefas, artefatos e papéis envolvidos no processo de execução do experimento considerando os tratamentos investigados, e coletando dados relevantes para os experimentos;
- ***Visão de Questionário***: define questionários usados para coletar informações dos participantes do experimento.

Em um modelo ExpDSL, podemos formalizar um experimento controlado (`ExperimentElement` elemento), como mostra o metamodelo da linguagem (Figura 5). Cada experimento tem um identificador e um título para o experimento (opcional

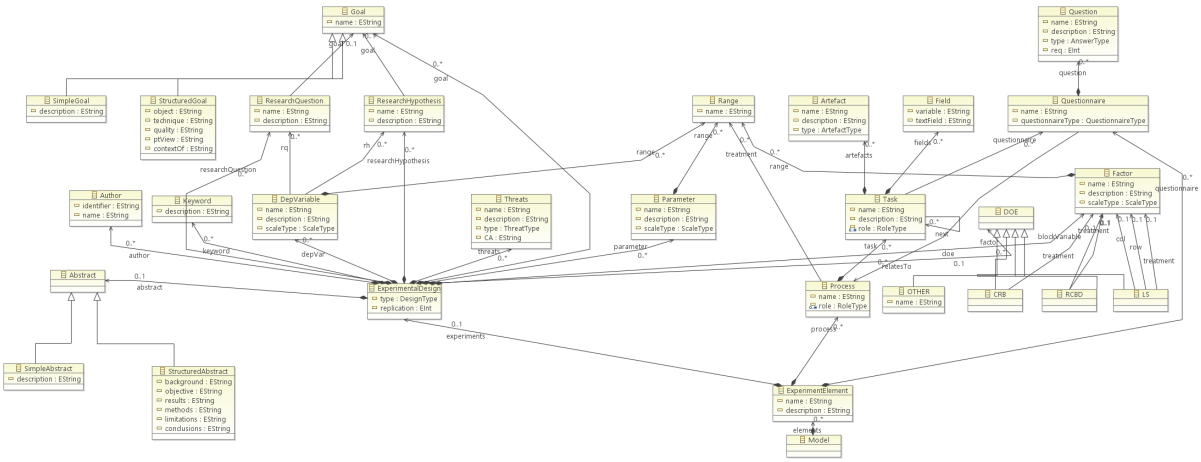


Figura 5: Metamodelo ExpDSL

description). Ele é composto por um plano experimental (`ExperimentalDesign`) e um conjunto de processos (`Process`) e questionários (`Questionnaire`). Estes três elementos representam as diferentes visões do experimento. A gramática da linguagem é apresentada no Apêndice I.

As seções seguintes descrevem cada uma destas visões em detalhes, seguindo a ordem sugerida em (JEDLITSCHKA, CIOLKOWSKI e PFAHL, 2008). Na medida em que formos apresentando a linguagem, exemplificaremos através da especificação de fragmentos de um experimento real, chamado de experimento de trabalho neste capítulo.

4.2. Experimento de Trabalho

Esta seção apresenta um experimento real na área de engenharia de software, que será usado ao longo do capítulo para ilustrar os mecanismos de formalização fornecidos pela linguagem específica de domínio ExpDSL. O experimento (CIRILO, NUNES, *et al.*, 2011) foi originalmente conduzido pelo Laboratório de Engenharia de Software (LES) da PUC-Rio (Brasil), e foi também replicado pelo nosso grupo de pesquisa na UFRN.

O objetivo deste experimento é investigar a compreensão do conhecimento de configuração usando três diferentes ferramentas de derivação de produto (CIDE, GenArch+, pure::variants) no contexto de linha de produto de software (LPS). Uma LPS (CLEMENTS e NORTHROP, 2011) representa uma família de software de um segmento específico de mercado que compartilha funcionalidades em comum, mas que também define variabilidades específicas para membros da família de software (produtos) (CZARNECKI e HELSEN, 2006). *Features* são usadas para capturar as similaridades e discriminar as variabilidades em LPS. Uma *feature* pode ser uma propriedade ou funcionalidade do sistema importante para um interessado (*stakeholder*). Derivação de produto (DEELSTRA, SINNEMA e BOSCH, 2005) refere-se ao processo de construir um produto final a partir de um conjunto de ativos de código (*code assets*) que implementam as *features* da LPS. A seleção, composição e customização destes ativos de código baseado em um conjunto selecionado de *features* constitui a configuração da LPS, e a maneira de realizar esta configuração muda de acordo com a ferramenta de automação adotada. O experimento especificado neste capítulo investiga o uso de diferentes ferramentas de derivação com 3 abordagens com o objetivo de

analisar se as diferentes técnicas influenciam na compreensão do conhecimento de configuração.

O projeto estatístico adotado pelo experimento foi um quadrado latino de três dimensões. Ele define três fatores – ferramenta, LPS e participante – com três níveis cada um. O fator ferramenta representa as três ferramentas de derivação de produto que estão sob avaliação. Elas também constituem os tratamentos do experimento. O fator LPS é modelado como três LPS existentes: *OLIS*, *Buyer Agent* e *eShop*. Cada uma destas LPSs foi desenvolvida usando diferentes *frameworks* e tecnologias. Finalmente, o fator participante representa os participantes do experimento. Todos os participantes necessitam analisar uma implementação diferente de LPS para cada um dos três tratamentos. A ordem e combinação do par Ferramenta/LPS sob avaliação pelos participantes são selecionadas de forma aleatória de acordo design estatístico (DoE) do quadrado latino. Várias LPSs são usadas para evitar o fator efeito de aprendizagem e a subsequente influência nos resultados. Há três processos (conjunto/roteiro de tarefas) relacionados ao experimento, um para cada tratamento. Cada processo define dez tarefas linearmente ligadas. Durante a execução do experimento, os participantes necessitam executar as tarefas respondendo questões sobre a compreensão do conhecimento de configuração. Duas variáveis dependentes foram analisadas no experimento: (i) corretude (*correctness*); e (ii) tempo (*time*). Isto é, foi considerado não apenas se os participantes foram capazes de entender o conhecimento de configuração mas também o tempo gasto por cada um deles para responder as questões usando diferentes ferramentas de derivação de produto. As questões de pesquisa do experimento de trabalho foram:

QP1: A disponibilidade de modelos específicos de domínio aumenta a correta compreensão do conhecimento de configuração?

QP2: A disponibilidade de modelos específicos de domínio reduz o tempo necessário para a correta compreensão do conhecimento de configuração?

QP3: As diferenças individuais de especialização dos engenheiros de LPS impacta na correta compreensão do conhecimento de configuração?

QP4: Que tipo de tarefas de configuração se beneficiaram mais das técnicas de modelos específicos de domínio e das técnicas orientadas a código?

A Tabela 8 apresenta uma visão geral resumida dos elementos do experimento de trabalho. Além disso, cada participante respondeu um questionário de experiência após terminar de executar o processo (roteiro de perguntas) de cada tratamento.

Tabela 8: Visão geral dos elementos do experimento de trabalho

Elemento	**Valor**
Projeto experimental	Quadrado latino de três dimensões
Variávies Dependentes	Corretude
	Tempo
Variávies independentes	Cada linha de produto (LPS)
Número de replicações internas	2 (duas)
Fatores	Ferramenta de LPS (*GenArchPlus*, *pureVariants*, *CIDE*)
	LPSs (*EShop*, *BuyerAgent*, *OLIS*)
	Participantes

4.2.1 Visão do Plano do Experimento

Esta visão é usada para formalizar as principais informações e configurações de um projeto experimental. Ela permite que o pesquisador defina os fatores a serem controlados (tratamentos, variáveis de blocos, etc), assim como o *design* estatístico do experimento (DoE) usado para organizar a alocação de tratamentos, participantes e material experimental. Esta visão é representada na gramática ExpDSL pelo elemento `ExperimentalDesign`. A Figura 6 mostra o fragmento de ExpDSL que descreve o plano do experimento de trabalho.

ExperimentalDesign – Cada definição de experimento tem um elemento que representa o projeto experimental. Ele é composto por um conjunto opcional de informações relacionadas ao experimento em si, tais como, autores (elemento `Authorship`), resumo (elemento `Abstract`) e palavras chaves (elemento `keyword`). Além disso, é composto também por um conjunto de objetivos (elemento `Goal`), um conjunto de questões de pesquisa (elemento `ResearchQuestion`), um conjunto de hipóteses de pesquisa (elemento `ResearchHypothesis`), um conjunto de variáveis dependentes (elemento `DepedentVariable`), fatores e seus níveis (elementos `Factor` e `Level`), um design estatístico do experimento (elemento `DoE`), um contexto que é representado por um conjunto de parâmetros (elemento `Parameter`), e o número de replicações internas (elemento `internalReplication`).

Author – A informação sobre os autores do experimento pode ser definida usando este elemento. Para nosso experimento de trabalho, por exemplo, o identificador do autor (*`identifier`*) é `CiriloElder` e o nome do autor (*`name`*) é `"Elder Cirilo"`. Esta informação é importante para indexar estudos por autores e incrementar a geração de relatórios mais completos.

Abstract – O resumo do estudo pode ser expresso de uma forma plana ou estruturada. Na especificação do experimento de trabalho temos a forma estruturada, onde deve ser definida as seguintes informações para o experimento: *Background*, objetivos, método, resultados, limitações e conclusões do estudo. Esta informação pode ajudar a pesquisa por experimentos controlados, e é importante para a geração de relatórios.

Keyword – O elemento `Keyword` é informativo e usado para apresentar as palavras chave do estudo. As palavras chave do experimento de trabalho estão definidas entre aspas duplas na Figura 6. Assim com o elemento `Abstract`, esta informação opcional pode ajudar na categorização de estudos e complementar a geração de relatórios.

```
Experiment ConfExp "Configuration Knowledge Experiment"
Experimental Design
Authorship { CiriloElder “Elder Cirilo” …}
Abstract {
        Background “The configuration knowledge is ….”
        Objective “The goals of this study is ...”
        Results “Our results suggest that: ...”
        Limitations “Confounding questions, and insufficient training session…”
        Conclusions “GenArch+ tends to better support participants in the task of localizing
        and correctly answering CK questions”
}
Keywords { “Software Product Line” “Controlled Experiment” “Configuration knowledge” }
Goals {
        G1_Comprehesion "Investigate whether the different techniques influence the correct
        comprehension of the CK"
        G2_Time Analyze “different SPL techniques” for the purpose of “Investigating” with
        respect to their “time of configuration knowledge comprehension” the point of view
        of the “SPL Designer” in the context of “SPL”
        G3_Expertise }
Research Questions {
        RQ1 “Does the configuration knowledge comprehension depend on the specification
        technique?” relates to G1_Comprehesion
... }
Research Hypotheses {
        RH1 "The correct comprehension of the configuration knowledge does not depend on the
        different specification techniques" relates to G1_Comprehesion
        RH2 "The time to correct comprehension of the configuration knowledge depends on the
        different specification techniques" relates to G2_Time
... }
Dependent Variables {
 Correctness "Measure of correct answers" Scale Absolute relates to RH H1
 Time "Time used on the correct answers" Scale Absolute relates to RH H2
}

Factors {
 Technique "Approach" Scale Nominal Range {CIDE PureVariants GenArch}
 SPL "Product Line" Scale Nominal Range {EShop OLIS Buyer}
 Participants "Experiment Participats" Scale Nominal Range {P1 P2 P3}
}
DoE = LS - Latin Square( column = SPL , row = Participants , treatment = Technique
Context {
        ParticipantLevel "schoolar level of the participants" Scale Nominal Range {MSc PhD}
 ParticipantKnowledge "schoolar level of the participants" Scale Nominal Range {JAVA XML SPL}
        trainningTime "participants trainning time" Scale Nominal Range {OneHour}}
Internal Replication  2
Threats to Validity {
   TV1 description "The engagement of the subjects, due to the length (time) of the
   questionnaires" type Conclusion control action(s) "However, there was a rotation of the
   approaches order given that we adopted the Latin square."
   ...}
```

Figura 6: Fragmento da especificação do experimento de trabalho para a visão de plano experimental.

Goal – Os objetivos de pesquisa a serem alcançados pelo experimento devem ser definidos no elemento Goal. Esta informação é importante para o entendimento do plano experimental. Há três maneiras de definir este elemento, e elas são exemplificadas no experimento de trabalho. A primeira tem um identificador G1_Comprehension e sua descrição correspondente. A segunda tem um identificador e uma forma estruturada de formalizar a descrição que é baseada no método GQM (BASILI, CALDIERA e ROMBACH, 1994): *Analyze* "different SPL techniques" *for the purpose of* "Investigating" *with respect to their* "time of configuration knowledge comprehension" *the point of view of the* "SPL Designer" *in the context of* "SPL". Nesta forma, a informação entre aspas duplas são preenchidas pelo pesquisador ao definir o experimento. A última maneira permite apenas a definição do identificador do objetivo, como o objetivo G3_Expertise do nosso experimento de trabalho. Todas as maneiras de definição de objetivos habilitam a rastreabilidade das questões e hipóteses de pesquisa.

ResearchQuestion – ExpDSL também permite define questões de pesquisa para o experimento através do elemento ResearchQuestion. Este elemento representa o que o experimento deve responder. Cada questão de pesquisa necessita ser relacionada a um elemento Goal a fim de permitir a rastreabilidade das informações. A Figura 6 apresenta um fragmento da especificação das questões de pesquisa para nosso experimento de trabalho. A questão de pesquisa que é exibida tem o identificador RQ1, a descrição "Does the configuration knowledge comprehension depend on the specification technique?" e está relacionada ao objetivo G1_Comprehesion definido previamente.

ResearchHypothesis – Este elemento permite definir o que é esperado que o resultado do experimento apoie. Cada hipótese de pesquisa necessita estar relacionada a um objetivo. O fragmento do nosso experimento de trabalho mostra duas questões de pesquisa definidas e associadas a objetivos previamente definidos. A hipótese RH1, por exemplo, tem uma descrição "The correct comprehension of the configuration knowledge does not depend on the different specification techniques" e está relacionada ao objetivo G1_Comprehesion. Cada hipótese deve ser definida através de um identificador e uma descrição textual (não mandatória), além da associação com o objetivo de pesquisa.

Embora as questões e hipóteses de pesquisa sejam elementos opcionais, é importante definir pelo menos um deles para nortear o experimento. Neste caso, há uma regra semântica na linguagem que adverte o pesquisador sobre a importância desta informação caso nenhuma delas seja definida. Portanto, o editor da linguagem é responsável por apresentar uma mensagem de alerta com tal informação.

Antes de explicar os próximos elementos, vamos sumarizar como devem ser definidas as variáveis do experimento. Todas elas, variáveis dependentes, independentes e variáveis de contexto, devem ser definidas através da mesma estrutura. Qualquer variável terá um identificador, uma descrição, uma escala (Absolute, Nominal, Ordinal, Interval ou Ratio) (KITCHENHAM, HUGHES e LINKMAN, 2001), e uma série de possíveis valores (não mandatória).

DepVariable – Este elemento é usado para descrever variáveis que são medidas para verificar se as variáveis independentes têm efeito nos resultados. Há duas variáveis dependentes em nosso experimento de trabalho: `correctness` e `time`, como mostra a Figura 6. Cada variável dependente deve ser relacionada a uma ou mais questão de pesquisa ou hipótese de pesquisa. A variável `correctness`, por exemplo, tem a descrição `"Number of correct answers"`, é do tipo `Absolute (ScaleType) (KITCHENHAM, HUGHES e LINKMAN, 2001) e` está relacionada a hipótese de pesquisa `RH1.` Esta associação é importante para permitir a rastreabilidade das informações. A formalização deste conceito é bastante importante para a realização de análises, pois ele é responsável por mensurar as variáveis de resposta do experimento.

Factors – Este elemento é usado para descrever as variáveis que são manipuladas no experimento e podem influenciar as variáveis dependentes. Cada fator, e seus correspondentes níveis, devem ser especificados na estrutura de uma variável, como aconteceu com as variáveis dependentes. Este elemento deve ser sempre definido como sendo da escala `Nominal`, para permitir a especificação dos níveis. Os níveis dos elementos devem ser definidos através do elemento `range`. O fator `SPLTechnique`, por exemplo, tem os níveis `GenArchPlus, pureVariants` e `CIDE.`

DoE – Este elemento é responsável pela definição do design estatístico do experimento (DoE) (ANTONY, 2003). Atualmente a linguagem suporta tipos de elementos que permitem escolher um entre três *designs* estatísticos para o experimento. Os tipos

suportados são: (i) completamente aleatorizado (CRD - *Completely Randomized Design*), (ii) aleatorizado em blocos completos (CRBD - *Randomized Complete Block Design*) ou (iii) quadrado latino (*LS – Latin Square*). Existem vários outros tipos de *designs* estatísticos para experimentação na literatura, entretanto, apenas esse subconjunto é atualmente atendido pela nossa linguagem. Quando o pesquisador necessitar definir qualquer outro `DoE`, ele deverá selecionar o tipo "`Other`" de forma a conseguir modelar seu experimento e adicionar a informação de qual é esse design num campo texto. Nosso experimento de trabalho define o tipo `Latin Square` para o experimento. Neste `DoE`, a configuração tem que ser definida como um valor ternário composto de coluna, linha e tratamento, selecionados através de uma referência cruzada com os fatores definidos. A Figura 6 mostra o valor deste elemento (**`column`** `= SPL,` **`row`** `= Participants,` **`treatment`** `= Technique)` para o experimento de trabalho.

É importante destacar, que no caso da opção pelo DoE "Other" pode restringir o processamento de várias informações sobre o experimento, e foi definido apenas como uma alternativa para permitir a formalização de experimentos que não usam os DoE suportados pela linguagem.

Parameter – Esta informação é usada para caracterizar o contexto do experimento, o que significa dizer que os resultados da experimentação serão particulares para as condições definidas por estes parâmetros (JURISTO e MORENO, 2010). Um experimento pode ter um conjunto de parâmetros, onde cada parâmetro deve definir valores fixos para representá-los. A Figura 6 mostra, por exemplo, três parâmetros definidos para o experimento de trabalho. Um deles é o `ParticipantLevel` cuja descrição é `"schoolar level of the participants"` definido como sendo da escala `Nominal` com apenas dois tipos de valores que caracterizaram esse parâmetro para o experimento especificados através do `range {MSc PhD}.`

Internal Replication – Este elemento é usado para definir o número de réplicas internas para os tratamentos. Para cada DoE, ele pode ter um significado diferente. Para o RCD, ele significa quantas vezes cada tratamento é aplicado. Para o RCBD ele significa o número de vezes que cada tratamento é aplicado dentro do bloco, e para o LS, significa o número de quadrado. No experimento de trabalho, existem duas réplicas de quadrados, como mostra a Figura 6.

Threats to Validity – Este elemento é usado para especificar as ameaças a validade de um experimento. Cada ameaça (elemento TheatToValidity) é definida através de um identificador, uma descrição, o tipo de ameaça (`Conclusion, Internal, Construct, External`) (WOHLIN, RUNESON, *et al.*, 2012), e, opcionalmente, uma ação de controle (atributo control action). A Figura 6 mostra as ameaças à validade para o experimento exemplo. A ameaça TV1 com descrição "`"The engagement of the subjects, due to the length (time) of the questionnaires`", que representa uma ameaça de conclusão (`type Conclusion`). A ação de controle adotada na definição foi `"However, there was a rotation of the approaches order given that we adopted the Latin square."`

4.2.2 Visão de Processo

Esta visão permite a especificação do procedimento de coleta de dados (processo) a ser seguido pelos participantes. Ela é similar a uma linguagem de definição de processos de software (FREIRE, ACCIOLY, *et al.*, 2013) (FREIRE, 2013), onde podemos definir tarefas, artefatos e papéis envolvidos nos processos.

```
Process Eshop to SPL.EShop {
  Role Participant

  Task T1 description "Which Spring Bean implements the Email
feature?" to T2 {
 artefacts SPLCode description "SPL - Eshop" type input
 field Answer "Answer"
 }
  Task T2 description "Which Bundles are related to the Browsing
feature?" to T3 {
 artefacts SPLCode description "SPL - Eshop" type input
 field Answer "Answer"
 }


...
```

Figura 7: Fragmento da definição do processo denominado Eshop para o experimento de trabalho

Process – Um experimento pode ter um conjunto de procedimentos. Cada procedimento é definido como um processo. A Figura 7 mostra um fragmento do processo, para o experimento de trabalho, cujo nome é EShop. Ele está associado ao fator `SPL.Eshop` e é definido para o papel `Participant`. Portanto, cada participante ao usar a SPL EShop seguirá este processo.

Cada processo tem, portanto, um identificador (`name`) e pode ter um papel associado (`Role`). Um processo é composto com um conjunto de tarefas (elemento Task).

Task – A(s) tarefa(s) de cada participante durante execução de um experimento é descrito através deste elemento. Ele guia o participante durante a execução do experimento. Figura 7 mostra duas tarefas do processo Eshop. A tarefa `T1` tem a descrição `“1 - Which Spring Bean implements the “Email” feature?"` e aponta para a próxima tarefa na sequência: **`“to T2”.`** Esta tarefa tem um campo associado onde deverá ser coletada uma informação do usuário. O campo é `Answer` e seu rótulo é `"Answer".`

Generalizando, uma tarefa possui um identificador (`name`), uma descrição (`description`), e, opcionalmente, um papel (`role`). Ela pode possuir uma lista de artefatos de entrada e/ou saída relacionados (`input` ou `output`), um campo texto (Field), ou uma referência cruzada para um elemento de questionário (definido pela visão de questionário) significando que ele deve ser coletado no momento de execução da tarefa. Sendo assim, durante a execução do experimento, o participante será guiado para executar as tarefas, que podem ter artefatos para consulta, artefatos para envio pelo participante, além disso pode ser necessário enviar alguma informação através dos campos associados ou, até mesmo, responder questionários.

Artefact – Este elemento tem um nome (name), uma descrição e um tipo (input, output ou input/output). Ele é usado para representar qualquer instrumento necessário à execução da tarefa. Ele é essencial na coleta de dados.

Field - no fragmento processo da Figura 7, este elemento é responsável por representar a necessidade de resposta por parte do usuário. Ele possui um nome `Answer` e um rótulo **`“Answer”.`**

4.2.3 Visão de Questionário

A visão de questionário, permite a definição de questionários que são utilizados para coletar dados qualitativos e quantitativos dos participantes do experimento. Estes questionários podem ser aplicados antes ou depois do experimento, assim como durante a execução de alguma tarefa, de acordo com a especificação.

Questionnaire – A mostra Figura 8 um fragmento do questionário chamado `BuyerKnowledgeTecnologies` e relacionado ao processo Buyer. O tipo `Post` indica que este questionário deve ser coletado após a aplicação do tratamento relacionado ao

```
Questionnaires
Questionnaire BuuyerKnowledgeTechnologies relates Buyer type Post {
      Q1BuyerJadex {
 description "What is your experience using Jadex framework?"
 type SingleChoice required 1
 { "Start" "Professional" "Expert”}
}
…
```

Figura 8: Fragmento da especificação de questionários no experimento de trabalho

processo Byer. Ele é um questionário de *feedback*.

Cada questionário é composto de um conjunto de elementos Question. Quando um questionário não especifica um tipo (Pre ou Post), significa que ele será referenciado por alguma tarefa e deverá ser coletado durante a execução de tal tarefa.

Question – A questão nomeada `Q1BuyerJadex na` Figura 8 tem a descrição `"What is your experience using Jadex framework?"`, ela é definida como do tipo **`SingleChoice`**, é uma questão mandatória (**`required`** `1`) e tem um conjunto de alternativas.

De uma forma geral, cada elemento `Question` é com posto de um nome (name), uma descrição, um tipo (`SingleChoice, MultipleChoice, Text, ParagraphText`), um atributo (required `1` ou `0`) e um conjunto de alternativas, quando os tipos forem SingleChoice ou MultipleChoice.

Alternative – As alternativas para a questão do experimento de trabalho são `"Start", "Professional", "Expert"`. Estas opções representam os valores que serão exibidos a fim de que o participante selecione um deles (`SingleChoice`).

4.2.4 O ambiente de edição ExpDSL

A utilização do xText na implementação da ExpDSL permitiu a incorporação de facilidades do *Eclipse Modeling Project*. Uma dessas facilidades é a geração automática do editor para a linguagem definida. O editor gerado integra *features* que facilitam o processo de edição pelo usuário, minimizando as possibilidades de erro. Algumas dessas *features* são: (i) o destaque (*highlight)* das palavras reservadas da linguagem; (ii) a geração automática da função de auto completar no editor, de acordo com a gramática da linguagem; (iii) a geração automática de janelas *pop-ups* com sugestões de nomes a serem referenciados, funcionando como um guia de edição; e (iv) a possiblidade de implementação de validadores relativos a restrições extras a serem executados durante a edição.

Durante a fase de especificação da DSL, sempre que uma das visões de *ExpDSL* necessita referenciar elementos de outra visão, o *xText* possibilita o gerenciamento automático dessas referências. Para a visão de Processo, por exemplo, o editor permite que, através de janelas pop-ups, seja selecionado qualquer um dos níveis dos fatores já definidos, como ilustrado na Figura 9.

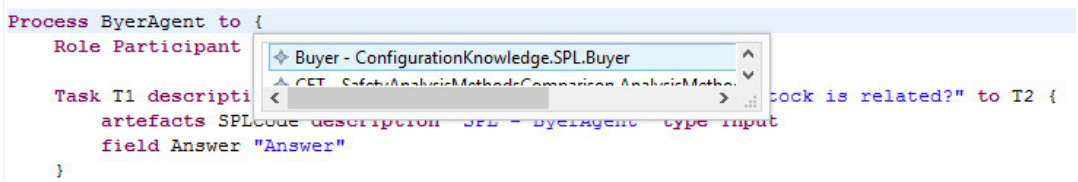


Figura 9: Gerenciamento automático dessas referências.

Outro ponto está relacionado a possibilidade que o ambiente *xText* incorpora de implementação de restrições adicionais para a linguagem sendo definida. Essas restrições demandam a criação de rotinas extras de validação, implementadas na linguagem Java. Nas gramáticas baseadas em *Ecore*, como é o caso de ExpDSL, o gerador de modelos cria também classes Java equivalentes a cada um dos elementos da DSL além de classes *helper* com funções específicas, tais como formatação, validação,

e assim por diante. Essas *features* xText foram utilizadas para codificar restrições adicionais nos casos onde elas foram necessárias em *ExpDSL*.

Alguns exemplos das validações básicas já implementadas na visão de plano experimental são as checagens semânticas. Ao selecionar uma variável (dependente, fator ou parâmetro) como sendo da escala Ordinal ou Nominal, por exemplo, o validador cobra que o usuário informe a série de valores possíveis para a mesma (range). Outro exemplo diz respeito às restrições impostas pela escolha do plano estatístico (*design of experiment*): (1) ao selecionar o *design* estatístico do tipo "*LS – Latin Square*" para o experimento é checado se a tripla: coluna, linha e tratamento foram definidas pelo pesquisador. Na violação da regra, o editor irá apresentar uma mensagem de "*error*" apropriada, para o usuário, conforme ilustrado na Figura 10; (2) ao selecionar o design estatístico do tipo "*RCBD - Randomized Complete Block Design*" para o experimento é checado se existe um fator selecionado como sendo o tratamento variação e um fator de bloco; (3) ao selecionar o design estatístico do tipo "*CRD - Completely Randomized Design*" para o experimento é checado se um fator foi selecionado como sendo o tratamento. Da mesma forma, as violações as checagens (2) e (3) também geram mensagens de erro para o usuário.

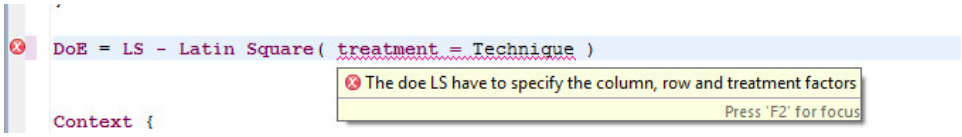


Figura 10: Validação Quadrado Latino com configuração diferente da correta.

Além disso, validadores foram implementados para permitir que ao relacionar um questionário a uma tarefa em um processo, o editor valide se o questionário pertence a experimento sendo modelado. Caso o usuário relacione um questionário não pertencente ao experimento, o editor irá apresentar uma mensagem de erro referencial.

4.3. Discussões

Esta seção discute pontos relevantes relativos a linguagem proposta.

Replicação de Experimentos. A replicação de um experimento pode objetivar repetir o experimento sob condições que são, tanto quanto possível, similares à primeira execução, ou, alternativamente, podem ser executadas variando um ou mais parâmetros

em relação ao experimento original. Neste caso, e dependendo da variação, deve ser analisado se o segundo experimento deve ser considerado uma replicação do primeiro ou um novo experimento. No caso da replicação sob condições similares, o objetivo pode ser confirmar as hipóteses do primeiro experimento, e já no segundo caso de replicação, na qual uma variável é alterada, o objetivo pode ser checar se uma determinada variável pode ser generalizada para certos valores dos resultados. Muitas vezes, os resultados da alteração de um parâmetro do experimento original na replicação irão ajudar a determinar exatamente em que circunstâncias uma tecnologia é melhor utilizada (JURISTO e MORENO, 2010). Apesar da linguagem proposta não ter como foco a replicação de experimento, a utilização desta linguagem facilita a replicação de um experimento visto que o time de pesquisadores necessitará apenas criar uma nova instância do experimento, sem a necessidade de uma nova especificação. Alternativamente, parâmetros podem ser alterados para que seja possível análises mais detalhadas em relação a tecnologia (métodos/metodologias) sendo avaliadas. Além disso, as informações históricas obtidas durante a execução individual de cada rodada das réplicas de um experimento podem ajudar a compartilhar o conhecimento entre os pesquisadores envolvidos, permitindo a não recorrência de erros já relatados por algum pesquisador em seu histórico de monitoramento.

Flexibilidade do Ambiente. A abordagem permite fácil adaptação para a integração de novos tipos de *design* estatístico. Isso se dá em virtude de termos o processo de geração do arquivo de configuração da execução (organização e aleatorização do experimento) independente da geração do workflow. Neste caso, qualquer necessidade de alteração para estender os tipos de *design* estatístico irá afetar apenas a transformação modelo para texto responsável pela criação do arquivo de configuração, essa alteração ocorre apenas pela inserção do código referente a geração do novo tipo de organização (distribuição de participantes e tratamento, além da ordem de execução, se for o caso).

Geração Automática de Workflows Executáveis. A formalização do experimento através de uma linguagem pode permitir a fácil geração, via desenvolvimento dirigido por modelos, de workflows executáveis para cada participante do experimento. Estes workflows (que representam a execução de um tratamento) seriam distribuídos de forma automática aos participantes de um experimento de acordo com o design estatístico definido (DoE). A ideia é que estes workflows permitam a criação de um ambiente customizado que permita ao pesquisador monitorar a execução do experimento.

4.4. Conclusões

Este capítulo apresentou ExpDSL, uma linguagem específica de domínio para a formalização de experimentos controlados em engenharia de software. A utilização de uma linguagem para formalização de um experimento possibilita a troca de informações entre stakeholders (projetista do experimento, estatísticos, especialistas do domínio) e entre a comunidade de pesquisadores, permitindo acesso para replicação destes estudos e realização de meta-análise. A linguagem permite a definição e o planejamento de um experimento e foi detalhada através da modelagem de um experimento real. Por fim, foram discutidos pontos relativos a linguagem proposta. O próximo capítulo apresenta uma abordagem dirigida por modelos para a geração automática de workflows executáveis a partir da formalização de um experimento.

5. Abordagem Dirigida por Modelos para Execução e Monitoramento de Experimentos Controlados em ES

A formalização de experimentos controlados através de uma linguagem específica de domínios possibilitou o desenvolvimento de uma abordagem dirigida por modelos para geração de um ambiente de execução e monitoramento de experimentos controlados que objetiva ser customizável e flexível.

5.1. Visão Geral da Abordagem

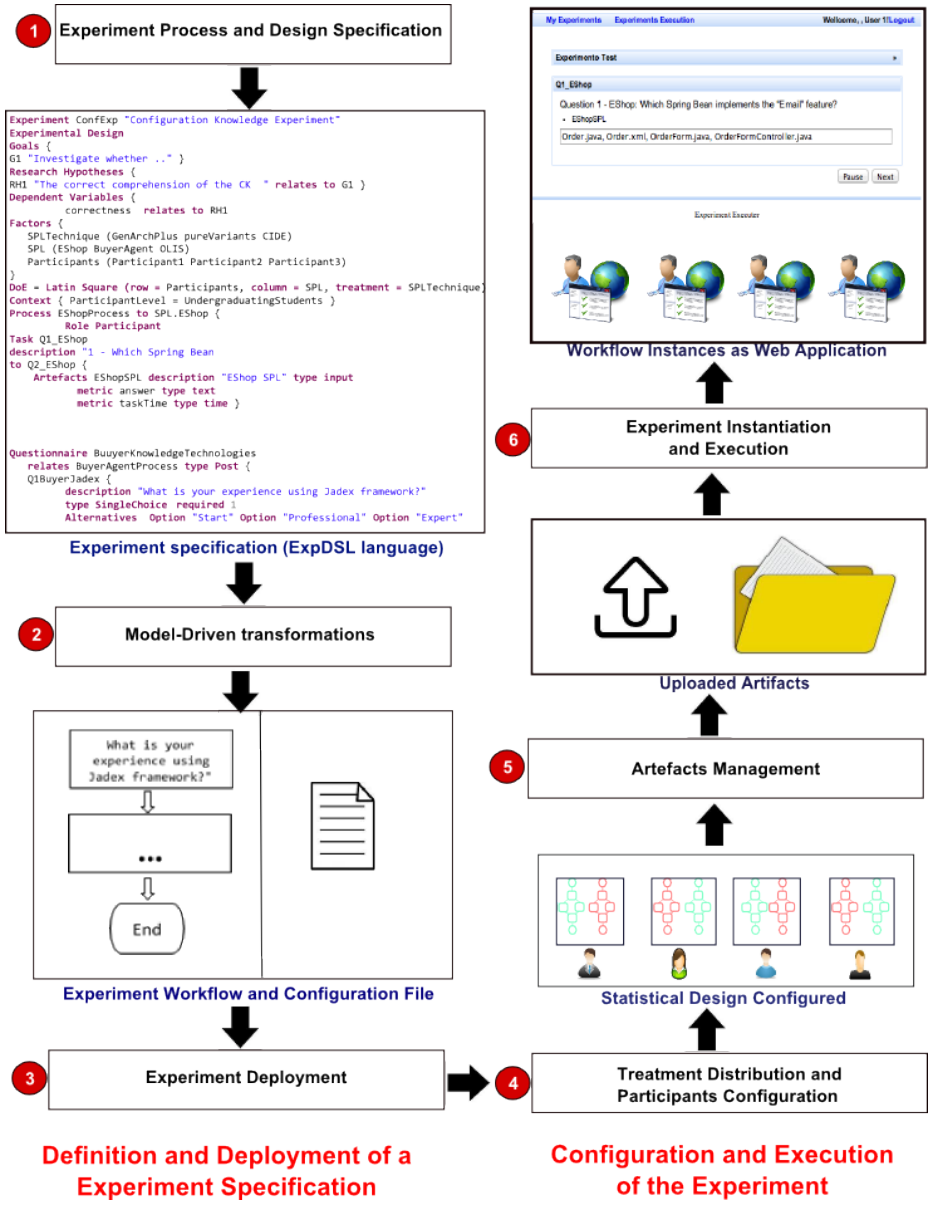


Figura 11: Visão geral da abordagem de geração do ambiente de execução

Esta abordagem é parte de um projeto maior que pretende fornecer uma infraestrutura completa de apoio a execução de experimentos controlados em engenharia de software. Neste contexto, o ambiente é fundamentado pela abordagem dirigida por modelos (FREIRE, ACCIOLY, *et al.*, 2013) (FREIRE, 2013). A Figura 11 ilustra os dois principais estágios da abordagem: (i) a definição e instalação de uma especificação de experimento; e (ii) a configuração e execução do experimento.

5.1.1 Estágio 1: Definição e Instalação de uma Especificação de Experimento

O primeiro estágio da abordagem é responsável pelo suporte à definição do experimento usando a linguagem específica de domínio (ExpDSL), que é então usada para produzir modelos de workflows. Estes workflows são instalados em um motor de execução de worflows para guiar os participantes durante a execução do experimento.

No primeiro passo (Figura 11 – (1)) deste estágio, nós necessitamos definir o projeto e o(s) processo(s) do experimento, ou seja, é o passo responsável pela completa especificação do experimento. Esta especificação é apoiada pela linguagem ExpDSL. Como apresentado no Capítulo 4, esta DSL fornece uma abstração para o projetista do experimento de tal forma que ele pode especificar todos os principais aspectos de um plano experimental. Cada especificação contém informações sobre aspectos como: objetivos, questões de pesquisa, hipóteses de pesquisa, variáveis dependentes e independentes, projeto estatístico do experimento (DoE), contexto do experimento, processo de aplicação dos tratamentos a serem seguidos pelos participantes, e os questionários extras necessários à avaliação qualitativa do experimento.

Após especificar o experimento, o segundo passo (Figura 11 – (2)) são as transformações dirigidas por modelo (modelo para modelo e modelo para texto). Estas transformações processam as especificações dos processos em ExpDSL juntamente com as informações sobre o experimento para gerar um modelo de workflow para cada participante do experimento bem como um arquivo de configuração. Os workflows são usados para guiar os participantes durante a execução do experimento, e eles são compostos por um conjunto de atividades definidas na especificação do experimento. Finalmente, o arquivo de configuração é responsável pela distribuição automática dos tratamentos de acordo com o DoE especificado para o experimento. O terceiro e último

passo deste estágio da abordagem é a instalação do experimento. Ela envolve a instalação do modelo de workflow e do arquivo de configuração no motor de execução a fim de habilitar o início da execução do experimento.

5.1.2 Estágio 2: Configuração e Execução do Experimento

O Segundo estágio representa o ambiente proposto, uma aplicação web baseada em um motor de execução de workflows. Este estágio envolve toda a configuração necessária antes de iniciar a execução do experimento. Portanto, para possibilitar a correta execução do experimento, o ambiente irá distribuir automaticamente os tratamentos e configurar os participantes (Figura 11 – (4)). Os tratamentos podem ser automaticamente distribuídos de acordo com as informações disponíveis no arquivo de configuração. Os participantes necessitam ser cadastrados no ambiente. Após isso, eles podem ser aleatorizados, respeitando o DoE definido no arquivo de configuração. Neste passo, nós necessitamos fazer o envio (upload) de todos os artefatos de entrada configurados nas tarefas do experimento (Figura 11 – (5)) especificados no passo 1 do primeiro estágio da abordagem, durante a definição dos processos relacionados à aplicação dos tratamentos. Estes artefatos fornecerão aos participantes as informações necessárias a execução das tarefas durante a execução guiada..

Finalmente, estes workflows podem ser instanciados para que possam ser executados (Figura 11 – (6) por cada participante. Os workflows são executados na aplicação web responsável por guiar e monitorar os participantes durante a execução de um experimento, incluindo a coleta dos questionários de opinião dos participantes.

5.2. Geração Automática dos Workflows

O segundo passo do primeiro estágio da abordagem compreende as transformações dirigidas por modelo que recebem como entrada a definição do experimento especificado em *ExpDSL* e gera como saída workflows customizados para cada participante do experimento de acordo com *design* estatístico especificado para o experimento. Portanto, os workflows gerados são responsáveis por guiar cada participante através das atividades e tarefas necessárias para conduzir o experimento e coletar os dados relacionadas às variáveis dependentes especificadas.

A Figura 12 ilustra o processo de transformação, que é composto de duas etapas na abordagem proposta: uma transformação M2M e transformações M2T. Inicialmente

o arquivo texto que representa a definição do experimento (gerado a partir da especificação realizada com *ExpDSL*) é convertido em um arquivo *Ecore* que representa o modelo (gerado automaticamente pelo xText) em conformidade com o metamodelo *Ecore* da *ExpDSL* (o metamodelo pode ser visto no Apêndice I). Este arquivo *Ecore* que representa o experimento é o artefato de entrada de duas transformações: (i) a transformação M2M (modelo para modelo) é responsável por gerar os workflows relativos a coleta de dados dos participantes que é gerado a partir das informações dos processos e métricas definidos; e (ii) a transformação M2T que é responsável por gerar arquivos de configuração que irão determinar a organização da execução do experimento de acordo com o *design* estatístico definido, fazendo assim a distribuição dos tratamentos e a aleatorização dos participantes.

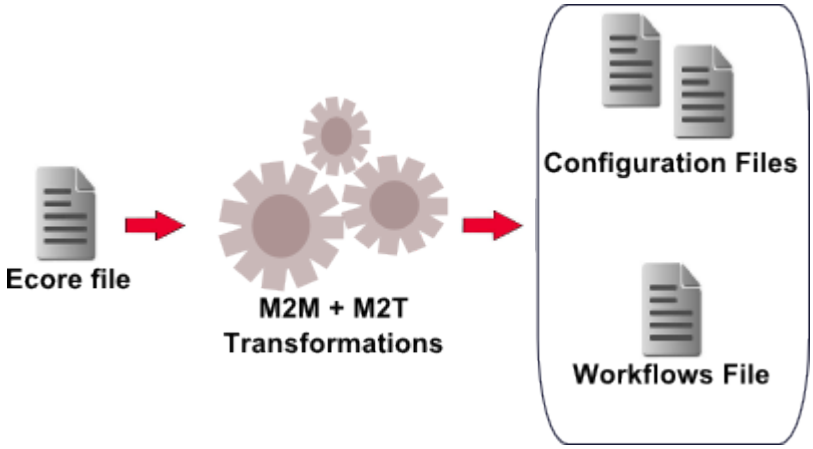


Figura 12: Geração dos Worflows Executáveis.

As transformações M2M são implementadas usando a linguagem QVTo (*Operational* QVT) e as transformações M2T usando a linguagem de *templates* Acceleo e a linguagem de programação Java. O metamodelo de workflow utilizado é baseado no esquema jBPM *Process Definition Language* (JPDL), mas ele tem mais elementos que o original, os quais são específicos para o domínio de experimentação e de processos da engenharia de software (o código da transformação M2M em QVTo e o fragmento do metamodelo JPDL adaptado pode ser visto no Apêndice I).

A Tabela 9 apresenta o mapeamento entre os elementos nas três etapas da abordagem: elementos especificados em *ExpDSL*, no modelo de workflow e no motor de execução. O elemento *Process* de ExpDSL é mapeado como um elemento *process-*

definition no modelo de workflow, finalizando como uma instância do workflow no motor de execução. O número de aplicação dos tratamentos é definido de acordo com o atributo *internalReplication*. Cada elemento *Task* na visão de processo é mapeado como um elemento *task-node* no modelo de workflow, que é mapeado para um formulário web responsável por apresentar as tarefas ao participante durante o experimento. Cada atributo "*next*" do elemento *Task* é mapeado para um elemento *transition* no modelo de workflow e um botão de transição no formulário web do motor de execução. Os elementos *Artefact in* e *Artefact out* são mapeados em elementos similares no modelo de workflow. O elemento *Artefact in* representa um link para download no formulário da atividade correspondente, e o elemento *Artefact out* representa um link para *upload* no formulário web da respectiva atividade. Os elementos *Field* das tarefas são mapeados em variáveis no modelo de workflow e serão representados como caixas de texto no formulário referente à tarefa. Cada questionário ExpDSL é transformado em um elemento equivalente no modelo de workflow e é representeado com um link para um formulário no ambiente web. Cada questão específica é mapeada em um elemento semelhante no modelo de workflow e é representada por um formulário que vai variar de acordo com o tipo da questão (*SingleChoice, ParagraphText, Text* ou *MultipleChoice*). Cada alternativa das questões será mapeada numa alternativa equivalente em JPDLPlus e será representada pelo tipo de componente no formulário da questão de acordo com o mapeamento encontrado na tabela.

Tabela 9: Mapeamento entre elementos

ExpDSL	**Workflow Model**	**Web Application – engine**
Process	process-definition	workflow instance*
Task	task-node	web-form
next (Task)	transition	botão de transição
Artefact in	artefact (type in)	link para download
Artefact out	artefact (type out)	formulário de upload

artefact inout	artefact (type inout)	link para download + formulário de upload
Field	Variable	Caixa de Texto
Participant	Role	users*
Questionnaire	Questionnaire	Link para um form de questões
Question	Question	Formulário de questão de acordo com o tipo da questão
SingleChoice Alternative	SingleChoice Alternative	Opção comboBox
ParagraphText Alternative	ParagraphText Alternative	Campo de Texto (256 caracteres)
Text Alternative	Text Alternative	Campos de Texto (50 caracteres)
MultipleChoice Alternative	MultipleChoice Alternative	Opção checkBox

5.3. Ambiente de Execução

Como mencionado anteriormente, a formalização da especificação do experimento através do uso de uma DSL possibilitou a geração de workflows customizados para os participantes, de acordo com o projeto estatístico.

Os workflows gerados devem ser executados a fim de permitir a realização prática do experimento controlado. Como vimos anteriormente, os *workflows* são executados em um motor de execução. A Figura 13 ilustra os passos necessários à execução de um

experimento utilizando o ambiente proposto. Para iniciar a execução do experimento, o pesquisador necessita fazer o *upload* dos arquivos gerados na etapa anterior (especificação dos workflow e arquivos de configuração), que representam o experimento completo, no ambiente web do motor de execução. Na sequência, o pesquisador deve executar algumas configurações simples, tais como, registrar participantes e carregar os artefatos de entrada especificados para as atividades dos processos envolvidos. Por fim, ele pode habilitar a execução do experimento. Deste ponto em diante, cada participante do experimento pode dar início a execução das suas atividades seguindo o fluxo descrito no seu workflow. Durante a execução, as métricas serão coletadas de acordo com a especificação e os dados coletados estarão disponíveis para a fase de análise.

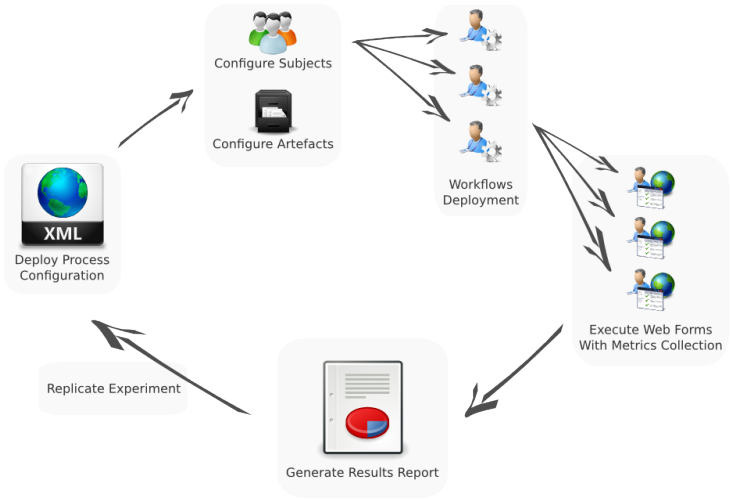


Figura 13: Processo de Execução deo Experimento.

O ambiente de execução proposto foi projetado para a web e aplica o padrão arquitetural MVC (*Model-View-Controller*). As tecnologias abordadas no desenvolvimento possuem em sua base a linguagem Java. Para o desenvolvimento do projeto foi escolhido o *framework JBoss Seam*[2], abordando o *JavaServer Faces* (JSF)[3] para desenvolvimento das páginas web, Enterprise Java Beans (EJB 3.0)[4] para construção dos componentes de negócio e a *Java Persistence* API (JPA)[5] para o mapeamento objeto-relacional. O banco de dados utilizado é o PostgreSQL, atuando

[2] http://www.seamframework.org/
[3] http://www.oracle.com/technetwork/java/javaee/documentation/index-137726.html
[4] www.jboss.org/ejb3
[5] http://www.oracle.com/technetwork/java/javaee/tech/persistence-jsp-140049.html

como repositório de dados para armazenamento da execução e configuração dos workflows. Algumas telas do ambiente podem ser visualizadas no Apêndice III.

5.4. Avaliação Exploratória da Abordagem

Esta seção apresenta um estudo exploratório que analisa a viabilidade da abordagem e a da DSL na modelagem de um experimento não-trivial em engenharia de software que já havia sido executado anteriormente (ACCIOLY, BORBA e BONIFÁCIO, 2012). O estudo modelado foi um experimento controlado que compara duas técnicas de testes usadas em linhas de produto de software (LPS): (i) uma Técnica Genérica (GT – *Generic Technique*) que usa especificações gerais sem representação de variabilidades; e (ii) uma Técnica Específica (ST – *Specific Technique*) que usa suítes de testes personalizadas por produtos. O experimento analisou os benefícios e inconvenientes destas duas técnicas do ponto de vista do gerente de testes medindo o esforço de execução dos casos de testes resultantes. É importante destacar que esta avaliação se deu com uma versão anterior da linguagem ExpDSL (diferente da apresentada no Capítulo 4) desde que a mesma sofreu melhorias após o estudo que será apresentado no Capítulo 6.

5.4.1 Critérios de Avaliação da Abordagem

Esta seção descreve cinco critérios levados em consideração no apoio à especificação e condução de experimentos controlados. Estes critérios foram identificados a partir da experiência relatada por outros pesquisadores quando conduzindo experimentos em engenharia de software. Eles são importantes para minimizar custos e esforços, e para melhorar a precisão dos dados quando conduzindo experimentos controlados:

C1. *Suporte à Definição Experimental em Múltiplas Visões*: Experimentos controlados são planejados e documentados de uma forma que abrange diferentes visões. Neste contexto, definir um experimento, por si só, compreende diferentes tipos de perspectivas (pesquisadores e participantes, por exemplo) e visão de estrutura (diferentes procedimentos de execução envolvendo vários artefatos, diferentes métricas, tipos de fatores e variáveis de blocos, coleta de *feedback*). Cada uma destas visões identifica múltiplos pontos de falha, requerendo, desta forma, um grande esforço para organizar e entender o experimento completo. Modelar o experimento usando múltiplas

visões pode melhorar o entendimento da equipe envolvida com suas responsabilidades específicas, e pode também especializar o conhecimento envolvido minimizando as chances de ocorrerem problemas de entendimento. Além disso, essa capacidade de múltiplas visões é particularmente importante para a replicação do experimento, uma vez que pode abranger a execução remota do mesmo e envolver um grande número de participantes. Como replicação de estudos é necessária na construção do conhecimento empírico com confiança, muita atenção deve ser tomada para garantir que os artefatos e os protocolos de coleta de dados dos experimentos foram bem entendidos e estão consistentes.

C2. *Monitoramento Online*: Controlar a execução de um experimento controlado é uma tarefa não-trivial. No contexto de experimentos em larga escala, esta tarefa é ainda mais complexa. O cuidado com o qual o dado é coletado e reportado afeta profundamente a precisão do processo. Consequentemente, a coleta de auto relato de dados pobres pode distorcer os resultados (SJOEBERG, HANNAY, *et al.*, 2005) e invalidar o experimento. Fornecer ferramentas e mecanismos para auxiliar a detecção antecipada de problemas durante a execução pode contribuir para garantir a validade do experimento. Além disso, se o estudo é feito em várias localidades em colaboração com outros pesquisadores, o uso do monitoramento online habilita o pesquisador remoto apoiar o monitoramento do experimento. Isso também possibilita que os pesquisadores registrem anotações relacionadas à execução contribuindo, desta forma, para manter os dados históricos do experimento e para compartilhar esse conhecimento com a comunidade de pesquisa. Além disso, diferentes técnicas de controle de processo podem ser usadas para monitorar a execução do experimento.

C3. *Prover geração e execução do processo de acordo com o design do experimento*: Durante a fase de planejamento de um experimento, o pesquisador tem que saber como lidar com conhecimentos estatísticos necessários para organizar um experimento controlado e, consequentemente, analisá-lo. Isso está intrinsicamente ligado ao design estatístico escolhido para o experimento, pois o tipo do *design* determina a análise que pode ser realizada e, portanto, as conclusões que podem ser tomadas (PFLEEGER, 1995). Embora existam designs estatísticos simples, não é fácil para um pesquisador engenheiro de software que não é familiarizado com designs estatísticos de experimentos saber como configurá-lo. Isto evidencia o problema da falta de abstração e a necessidade de um ambiente de suporte que não exija conhecimentos estatísticos

adicionais. No contexto de experimento em larga escala, esta tarefa é muito mais complexa e difícil;

C4. *Fornecer orientação e coleta de dados para cada participante*: Se o conhecimento necessário para definir um experimento é crítico, o esforço para rodar o experimento é também substancial. Neste contexto, atenção e interesses são exigidos a fim de minimizar o esforço de usar papel e caneta (coleta de dados manual), minimizar o tempo para executar o experimento, e, consequentemente, melhorar a precisão e qualidade dos dados. Fornecer um ambiente de apoio automatizado é um requisito chave neste caso e pode contribuir positivamente para alcançar estes objetivos.

C5. *Prover capacidade de integração de ferramentas externas*: Ambientes de engenharia de software experimental têm que lidar com vários estudos empíricos que pertencem a diferentes domínios de aplicação e tecnologias. Alguns dos dados necessários para realizar tais estudos experimentais podem ter origem em ferramentas externas como *IDEs*, ferramentas de teste, ferramentas de análise estática de código, ferramentas de modelagem, entre outras. Portanto, é importante que estes ambientes forneçam pontos de extensão visando dar suporte à integração. A integração com outras ferramentas externas também contribui para fornecer um ambiente único e completo para os participantes executarem o experimento.

5.4.2 Aplicação da Abordagem

Nesta seção apresentamos o estudo exploratório realizado com o objetivo de avaliar a expressividade de nossa abordagem e seus elementos. Os principais objetivos foram: (i) analisar a expressividade da DSL proposta e as transformações dirigidas por modelo da abordagem; e (ii) identificar pontos de melhoria no nosso motor de workflow. Para alcançar tais objetivos, foi modelado um experimento controlado que já havia sido executado previamente (ACCIOLY, 2012), e que utilizou fortes princípios de design estatístico. O fácil acesso à equipe responsável pela pesquisa e os resultados do experimento contribuíram também para a seleção deste experimento. Nós também simulamos o experimento com os pesquisadores que o executaram com participantes reais, objetivando coletar o feedback e impressão deles sobre o estudo.

5.4.2.1 O experimento modelado: investigando técnicas de testes caixa-preta em LPS

Em Linhas de Produto de Software (LPS), escrever testes caixa-preta baseado em cenários de casos de uso para produtos derivados pode ser um desafio devido ao potencialmente grande número de produtos e ao espalhamento dos pontos de variação através de diferentes *features* da LPS. Para lidar com este problema, várias técnicas vêm sendo propostas. Entretanto, esta área de pesquisa ainda carece de evidências empíricas sobre a real contribuição destas abordagens para melhorar a qualidade e a produtividade do ciclo de testes de uma LPS. Neste contexto, um experimento controlado foi projetado para analisar o impacto de duas técnicas de teste caixa-preta (denominados aqui de tratamentos) para produtos derivados de uma LPS (ver Figura 14) : (i) a Técnica Genérica (GT – *Generic Technique*), que usa uma especificação de testes geral sem especificação de variabilidades, isto é, uma única suite de testes para todos os produtos derivados; e (ii) a Técnica Específica (ST - *Specific Technique*), que usa suítes de testes customizadas para os produtos, isto é, uma suíte de teste específica para cada produto derivado. O objetivo é analisar estas técnicas avaliando o tempo gasto para executar cada suíte de testes e o número de Solicitação de Mudanças (*Change Requests* - CRs) inválidas. Uma CR inválida é uma indicação inválida de defeito de software, uma consequência da imprecisão em um caso de teste e não no software sendo testado.

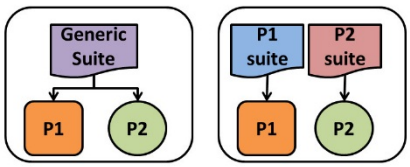


Figura 14: Tratamentos investigados pelo estudo realizado.

O experimento tem dois fatores de controle: (i) os participantes envolvidos, já que diferentes níveis de conhecimento e habilidades em testes geralmente impactam o esforço de execução dos testes; e (ii) o número de pontos de variação existentes nos casos de teste, desde que o número de pontos de variação existente em casos de testes, pois testes com mais pontos de variação podem conduzir a casos de testes mais inconsistentes. Estes fatores são controlados pelo design estatístico experimental chamado Quadrado Latino (Latin Square) (JURISTO e MORENO, 2010), que bloca o efeito de dois fatores que não são de interesses do pesquisador. Na realidade, um

quadrado latino 2x2 foi usado (dois tratamentos a serem investigados) usando participantes (com suas experiências individuais) e *features* (com seus pontos de variação) como variáveis blocantes nas linhas e colunas do quadrado latino.

5.4.2.2 Definição do Experimento

Durante a definição do experimento, foram modeladas as quatro visões do estudo experimental, as quais são detalhadas abaixo.

Visão de Processo. Nesta visão, nós definimos quatro processos, um para cada técnica (ST e GT) e combinação de *feature* (*features* denominadas como 1 e 2). Cada processo é composto de 12 atividades, onde cada atividade representa a execução de um caso de teste diferente. Nós selecionamos duas suítes de testes (cada uma com seis casos de testes) para serem usadas no experimento. Cada atividade do processo é composta de duas tarefas: uma relacionada a execução do caso de teste e a outra relacionada ao relatado de uma solicitação de mudança (*change request* – CR), quando falhas são observadas. A tarefa relativa ao caso de teste tem o próprio caso de teste como artefato de entrada e a tarefa de relato da CR tem o relatório da CR como um artefato de saída.

A Figura 15 mostra um fragmento parcial da especificação da DSL para a visão de processo que descreve duas atividades do processo de execução da técnica específica (ST) combinada com a *feature* 1 (processo "ST_F1"). A atividade chamada "SP1_F1_1" tem a descrição "Inserir um novo membro - Produto 1" cuja responsabilidade é do papel "Participante". Ela também aponta a atividade "SP1_F1_2" como sendo a próxima atividade a ser executada. A atividade "SP1_F1_1" tem duas tarefas: a primeira é chamada "Execute CT SP1_F1_1", com a descrição "Execute the test case SP1_F1_1", com o artefato de entrada chamado "Test case SP1_F1_1". As demais tarefas, atividades e processos são similarmente especificadas.

```
Process "ST_F1"{
    Activity "SP1_F1_1" "Inserir um novo membro - Produto 1" "SP1_F1_2" Role "Participante"; {
        Task "Execute CT SP1_F1_1" description "Execute the test case SP1_F1_1"
            artefacts
            artefact  "Teste case SP1_F1_1" description "Test case details SP1_F1_1" type input;;
        Task "Report CR SP1_F1_1" description "Describe TC SP1_F1_1 results"
            artefacts artefact "CR SP1_F1_1" description "Reporting the CR for Test Case SP1_F1_1" type output;;
    }
    Activity "SP1_F1_2" "Inserir um novo Artigo de Conferência  - Produto 1" "SP1_F1_3" Role "Participante";{
        Task "Executar CT SP1_F1_2" description "Executar o Procedimento do Teste SP1_F1_2"
            artefacts artefact  "Caso de Teste SP1_F1_2" description "Detalhes do Caso de teste SP1_F1_2"  type input;;
        Task "Reportar CR SP1_F1_2" description "Descrever o resultado do CT SP1_F1_2"
            artefacts artefact "CR SP1_F1_2" description "Resultado da Execução do Caso de Teste SP1_F1_2" type output;;
    }
```

Figura 15: Fragmento da Escpecifção de Processo em ExpDSL.

Visão de Plano Experimental. O fragmento da visão experimental da DSL define o plano experimental. Ela foi usada para definir os fatores e tratamento, os parâmetros do experimento, e o design estatístico escolhido para o experimento controlado. A Figura 16 mostra um fragmento da DSL para esta especificação. O elemento *Parameter* é usado para especificar qualquer característica (qualitativa ou quantitativa) do contexto do experimento que deve ser invariável através da experimentação. Neste caso, nós especificamos os parâmetros tal como "Os participantes devem pausar o tempo ao tirar dúvidas". O elemento *Factor* com valor "TestTechnique" representa a técnica investigada, portanto o atributo "isDesiredVariation" tem valor "True". Existem dois tratamentos (níveis ou alternativas) especificadas para este fator, que são: (i) o *Level* "Generic" (Generic Technique); e (ii) o *Level* "Specific" (Specific Technique). O segundo elemento *Factor* tem valor "Feature" e tem dois níveis/tratamentos que são chamados "F1" e "F2", representando cada *feature* a ser usada no experimento. O último fator é especificado como "Participante" com dois níveis "Subject1" e "Subject2" (em nosso quadrado latino, dois participantes são necessários por réplica do quadrado). Finalmente, o design experimental selecionado é "LS – Latin Square".

Durante a modelagem do experimento, nós identificamos a necessidade de dois novos elementos para melhorar os resultados da nossa transformação de modelos. O primeiro foi o elemento "*Internal Replication*" que é usado para indicar o número de aplicações de cada tratamento do experimento. Este parâmetro tem um significado diferente de acordo com o design estatístico do experimento selecionado. Em um quadrado latino, ele representa o número de réplicas do quadrado. Para nosso caso, nós indicamos quatro réplicas, que significa que nós temos oito participantes no experimento. Cada participante é responsável por executar duas instâncias do workflow, de acordo com a alocação definida pela técnica do quadrado latino.

O segundo novo elemento identificado no nosso estudo de avaliação da modelagem veio da necessidade de conectar os procedimentos de coleta de dados aos níveis de fatores que eles estão associados. Esta informação é necessária para organizar a ordem de execução do experimento, principalmente quando aplicando o design experimental do quadrado latino. Sendo assim, no nosso estudo, o elemento *Link* é usado para associar o elemento *Process* aos elementos *Level* correspondentes. Neste experimento, o processo chamado "ST_F1" (que é uma abreviação para *Specific Technique* e *Feature 1*). Assim, ele é conectado aos níveis "*TestTechnique.Specific*" e

"Feature.F1", significando que este procedimento de coleta é responsável por conduzir os participantes que estão usando a técnica específica para testar a *feature* F1. De forma similar, o processo "ST_F2" está conectado aos níveis "*TestTech-nique.Specific*" e "Feature.F2". Cada processo definido tem que ser conectado aos seus níveis associados a origem de variação).

```
Experimental Plan Design "TestDesign"
    type LS - Latin Square  {
        Parameter "Participants have to pause time for asking questions"
        Parameter "The LPS complexity is low"
        Parameter "Artefacts are written in the participant source language"
        Parameter "Participants are undergraduating students"

        Factor "TestTechnique" isDesiredVariation True
            Level "Generic";
            Level "Specific";
            ;
        Factor "Feature" isDesiredVariation False
            Level "F1";
            Level "F2";
            ;
         Factor "Subject" isDesiredVariation False
            Level "Subject1";
            Level "Subject2";
            ;

         Internal Replication 4

         Link "ST_F1" to "TestTechnique.Specific" "Feature.F1"
         Link "ST_F2" to "TestTechnique.Specific" "Feature.F2"
         Link "GT_F1" to "TestTechnique.Generic" "Feature.F1"
         Link "GT-F2" to "TestTechnique.Generic" "Feature.F2"
}
```

Figura 16: Fragmento da Especificação de Plano Experimental em ExpDSL.

Visão de Métricas. Nesta visão, nós especificamos as métricas que tem que ser coletadas durante a execução do experimento. Estas métricas são associadas as variáveis de resposta do estudo. Nós modelamos aqui duas diferentes métricas para cada processo. Uma métrica é responsável pela quantificação do tempo gasto para executar um caso de teste, e a outra está relacionada ao tempo gasto para relatar um defeito (CR).

A Figura 17 mostra um fragmento da especificação da DSL para a perspectiva das métricas. Ele descreve as duas métricas modeladas para o processo "*ST_F2*". A primeira é chamada "*TimeExecutingTest_ST-F2-Produto1*" cuja descrição é "*Tempo de execução dos casos de teste da F2 no P1*". A forma de coleta desta métrica é definida como "*continuous*" significando que o tempo será coletado para cada tarefa e para o

total das tarefas. A unidade definida é minutos. Todas as tarefas que devem ser interceptadas pela métrica têm seus nomes listados no elemento *tasks*, as quais são: “Executar TC SP1_F2_1”, “Executar TC SP1_F2_2”, “Executar TC SP1_F2_3”, “Executar TC SP1_F2_4”, “Executar TC SP1_F2_5”, e “Executar TC SP1_F2_6”. A segunda métrica é “TimeReportingTest_ST-F2-Produto1”, que é modelada de maneira similar à métrica anterior.

```
//Metrica para tempo de execução do CT F2 Produto 1
Metric "TimeExecutingTest_ST-F2-Produto1" relates "ST_F2" {
                description "Tempo de execução dos casos de teste da F2 no P1"
                form continuous
                unit minutes
                tasks "ST_F2.SP1_F2_1.Executar CT SP1_F2_1" "ST_F2.SP1_F2_2.Executar CT SP1_F2_2"
                 "ST_F2.SP1_F2_3.Executar CT SP1_F2_3" "ST_F2.SP1_F2_4.Executar CT SP1_F2_4"
                  "ST_F2.SP1_F2_5.Executar CT SP1_F2_5"  "ST_F2.SP1_F2_6.Executar CT SP1_F2_6" }

//Metrica para tempo de relato do CT F2 Produto 1
Metric "TimeReportingTest_ST-F2-Produto1" relates "ST_F2" {
                description "Tempo de execução dos casos de teste da F2 no P1"
                form continuous
                unit minutes
                tasks "ST_F2.SP1_F2_1.Reportar CR SP1_F2_1" "ST_F2.SP1_F2_2.Reportar CR SP1_F2_2"
                 "ST_F2.SP1_F2_3.Reportar CR SP1_F2_3" "ST_F2.SP1_F2_4.Reportar CR SP1_F2_4"
                  "ST_F2.SP1_F2_5.Reportar CR SP1_F2_5" "ST_F2.SP1_F2_6.Reportar CR SP1_F2_6" }
```

Figura 17: Fragmento da Especificação de Métricas em ExpDSL.

Visão de Questionário. Coletar feedback dos participantes de um experimento é muito importante para dar suporte à análise qualitativa do mesmo. Nós modelamos dois questionários para o experimento. O primeiro é aplicado antes do início da execução do experimento a fim de coletar o nível de experiência dos participantes. O segundo é aplicado após a execução do experimento para coletar o feedback dos participantes. Um questionário é um conjunto de questões onde cada questão possui um tipo a ser selecionado de acordo com o tipo da pergunta (Figura 18).

```
Questionnaire  "Nivelamento" type Pre
"Nome" {
    description "Qual seu nome:"
    type Text
    }
"Grau de Escolaridade" {
    description "Qual o grau de escolaridade:"
    type ComboBox
    "Graduação Em Andamento"
    "Graduação Completa"
    "Mestrado Incompleto"
    "Mestrado Completo" }
"Q1.ExperienciaTestes" {
    description "Já trabalhou ou teve experiência com testes de software?"
    type ComboBox
    required 1
    "Sim"    "Não"}
"Q2.DetalhesExperienciaTestes" {
    description "Se sim, relate brevemente sua experiência?"
    type ComboBox
    "Sim"    "Não" }
"Q3.DisciplinaLPS" {
    description "Já pagou a disciplina de Linhas de Produto de Software com Paulo Borba?"
    type ComboBox
    "Sim"    "Não" }
"Q4.FamiliaridadeRGMS" {
    description "Se sim, é familiarizado com a linha RGMS e suas funcionalidades?"
    type Paragraph Text }
"Q5.ComentariosExtra" {
    description "Participou do experimento passado?"
    type ComboBox "Sim" "Não" }
```

Figura 18: Fragmento da Especificação de Questionário em ExpDSL.

5.4.2.3 Geração dos Workflow

A especificação do experimento usando a DSL é o artefato de entrada para o segundo passo da nossa abordagem, que envolve a geração da especificação dos workflows. Esta geração é conduzida por uma transformação de modelo para modelo (M2M) onde as abstrações da DSL são mapeadas para elementos específicos do modelo de workflow. Este mapeamento é atualmente implementado através da linguagem de transformação de modelos QVTo.

O design estatístico definido para o experimento foi o quadrado latino. Portanto, os quatro processos especificados para o experimento na DSL foram associados as combinações dos tratamentos, e organizados como um quadrado onde as colunas representam as *features* da LPS e as linhas representam os participantes do experimento. Cada célula do quadrado representa o fator sob investigação (origem de variação desejada), que são as técnicas de teste caixa preta. A Figura 19 mostra um fragmento do arquivo de workflow gerado pela transformação. O metamodelo de workflows é baseado no JPDL *schema*.

```
<?xml version="1.0" encoding="UTF-8"?>
<expl:Model xmlns:expl="urn:jbpm.org:jpdl-3.1">
  <elements>
  <expl:process-definition name="ST_F1" quantity="4">
      <expl:start-state name="Starting">
        <expl:transition name="startTransitionSP1_F1_1" to="SP1_F1_1"/>
      </expl:start-state>
      <expl:end-state name="End"/>
      <expl:swimlane>
        <expl:assignment name="Subject" actor-id="Subject"/>
      </expl:swimlane>
      <expl:task-node name="SP1_F1_1" description="Insert a new member - Product 1">
        <expl:transition name="SP1_F1_1_Transition" to="SP1_F1_2"/>
        <expl:event type="task-end">
          <expl:action class="ST-F1-Product1ActionHandler" name="TimeTesting-ST-F1-Product1" />
        </expl:event>
        <expl:task description="Execute the test case SP1_F1_1" name="Execute TC SP1_F1_1">
          <artefacts name="Teste case SP1_F1_1" type="input" description="Test case details SP1_F1_1"/>
          <expl:event type="task-end">
            <expl:action class="SP1_F1_1ActionHandler" name="Execute TC SP1_F1_1" />
          </expl:event>
        </expl:task>
        <expl:task description="Describe TC SP1_F1_1 results" name="Report CR SP1_F1_1">
          <artefacts name="CR SP1_F1_1" type="output" description="Reporting the CR for Test Case SP1_F1_1"/>
          <expl:event type="task-end">
            <expl:action class="SP1_F1_1ActionHandler" name="Report CR SP1_F1_1"/>
          </expl:event>
        </expl:task>
      </expl:task-node>
      <expl:task-node name="SP1_F1_2" description="Insert a new conference paper  - Product 1">
        <expl:transition name="SP1_F1_2_Transition" to="SP1_F1_3"/>
        <expl:task description="Execute the test case SP1_F1_2" name="Execute TC SP1_F1_2">
          <artefacts name="Test case SP1_F1_2" type="input" description="Test case details SP1_F1_2"/>
          <expl:event type="task-end">
            <expl:action class="SP1_F1_2ActionHandler" name="Execute TC SP1_F1_2" />
          </expl:event>
        </expl:task>
```

Figura 19: Fragmento do Workflow Gerado.

5.4.2.4 Ambiente de Execução do Experimento

Durante este estágio da abordagem, nós fizemos o *deploy* dos workflows no nosso motor de execução: o arquivo XMI gerado que contém a definição dos workflows do experimento. Cada workflow é responsável pela coleta de dados de cada participante do experimento. Após a geração do arquivo de especificação dos workflows, o pesquisador deve fazer o upload de todos os arquivos que representam os artefatos de entrada (os casos de teste neste nosso estudo) usando o formulário de configuração do

ambiente. O pesquisador que está conduzindo o experimento é responsável também por realizar o cadastro dos participantes reais na aplicação.

Uma vez estando com a especificação dos workflows instalada e com os participantes cadastrados no ambiente web do nosso motor de execução, os pesquisadores podem começar a execução do experimento habilitando os participantes a acompanhar seus respectivos workflows. Deste ponto em diante, cada participante é capaz de iniciar a execução de suas atividades de acordo com o planejamento do experimento, ou seja, a execução e relato dos casos de testes da LPS. Eles serão guiados pela especificação do workflow. O ambiente de execução será responsável pela coleta das métricas antes e após a execução de cada atividade ou tarefa do workflow, seguindo a especificação realizada durante a definição do experimento.

Durante a execução do experimento, os pesquisadores podem monitorar todas as atividades em progresso assim como visualizar os detalhes sobre as atividades executadas por cada participante, tais como artefatos enviados, tempo gasto nas diferentes tarefas e atividades, além dos comentários por eles adicionados. Desta forma, o pesquisador pode conseguir a antecipação de informações sobre a precisão dos dados e pode detectar falhas de entendimento por parte dos participantes. Além disso, cada formulário web do workflow apresenta um botão "*pause*" que deve ser usado pelo participante se ele necessitar interromper momentaneamente a atividade atual do experimento para realizar alguma ação que possa afetar o tempo quantificado para esta atividade. Esta interrupção pode ser para atender uma ligação, ir ao banheiro ou executar alguma atividade extra que possa afetar o tempo resultante da atividade, de acordo com as regras do experimento.

5.4.3 Análise dos Critérios

Nesta seção são discutidas várias questões e lições aprendidas após a aplicação da abordagem para gerar workflows customizados para o a execução de um experimento controlado de testes para LPS. Nestas discussões, os critérios de análise definidos na Seção 5.1 são retomados.

C1. *Suporte à Definição Experimental com Múltiplas Visões*. Cada bloco de conhecimento na *ExpDSL* especifica a definição relacionada a apenas uma perspectiva do conhecimento envolvido em um experimento controlado. *ExpDSL* foi inicialmente

projetada para especificar as informações necessárias para habilitar a geração de suporte automático à condução de experimentos, tais como: (i) a visão de processo trata do procedimento do experimento – atividades e coleta de dados – para os participantes, e instruções para preparar o ambiente para o pesquisador; (ii) a visão de métricas que permite definir métricas relacionadas às variáveis de resposta; (iii) a visão de plano experimental que auxilia a configuração após a escolha de um design estatístico do experimento; e (iv) a visão de questionário que contribui para modelar a coleta de feedback. Essas informações são usadas para gerar o ambiente de suporte ao experimento, e elas contribuem para melhorar o entendimento das diferentes visões do experimento (processos, métricas, plano experimental e questionários). Além disso, a especificação da DSL também contribui para explicitamente documentar experimentos existentes, e consequentemente manter a consistência durante a replicação da execução. O atual ambiente de execução apenas suporta a perspectiva do participante, mas ele também possibilita definir perspectivas para diferentes processos e papéis desempenhados pelos diferentes pesquisadores envolvidos no estudo (configuração do ambiente, recrutamento de participantes, treinamento, execução piloto).

C2. *Monitoramento Online*: a abordagem apresentada habilita os pesquisadores a monitorar as atividades dos participantes e os dados produzidos durante a duração da execução do experimento. Ela contribui para o controle do experimento porque os pesquisadores podem conseguir a antecipação de informações sobre a precisão dos dados e detectar falhas no entendimento pelos participantes tão cedo quanto possível. Durante uma execução manual previamente conduzida do experimento de testes para LPS relatado em (ACCIOLY, 2012), por exemplo, um participante reportou uma CR em seis segundos, o que é muito pouco provável. Contudo, os pesquisadores, que conduziram o experimento, não puderam observar esta falha durante a execução do experimento ou antes do relatório dos dados na fase de análise do experimento. Em virtude deste acontecimento, foi necessário invalidar os dados deste participante, e, consequentemente, aumentar o viés do experimento. Este monitoramento online também permite ao pesquisador fazer anotações durante a execução do experimento e produzir informações históricas que podem ser úteis durante a análise qualitativa.

C3. *Prover geração e execução do processo de acordo com o design do experimento*: embora existam diferentes tipos de designs de experimentos, como explicado na Seção 5.1, nossa abordagem suporta atualmente três tipos básico de designs: (i) CRD, (ii)

RCBD, e (iii) LS. Estes designs podem abranger uma larga quantidade de tipos de experimentos em engenharia de software. Entretanto, é importante considerar que a utilização de designs simples ajuda a tornar o experimento mais prático pois pode minimizar os gastos e o uso de recursos (tempo, dinheiro, pessoas e *features* experimentais). Por último, mas não menos importante, designs simples são também mais fáceis de analisar (PFLEEGER, 1995). Baseado na escolha do design e na especificação das varáveis blocantes ou fatores, a abordagem proposta gera procedimentos customizados para serem executados em um workflow por cada participante de acordo com o design experimental selecionado. Desta forma, a abordagem contribui para ajudar os pesquisadores na aplicação de técnicas estatísticas por liberá-los de ter que lidar com a organização manual das muitas combinações de alternativas dos experimentos unitários.

C4. *Fornecer orientação e coleta automática de dados para cada participante*: uma vez especificado o processo relacionado a cada combinação de tratamento (procedimento de coleta dos dados) assim como tendo especificado o plano experimental, a abordagem é responsável pela geração do workflow para cada participante. Os workflows gerados fornecem um procedimento customizado passo a passo para cada diferente participante, ajudando o registro e armazenamento de suas atividades. Cada participante não necessita explicitamente registar e medir o tempo gasto em suas atividades. Isto pode contribuir diretamente para minimizar os gastos envolvidos na coleta de tais dados e aumenta a precisão dos dados coletados.

C5. *Prover capacidade de integração de ferramentas externas*: atualmente, o motor de execução é implementado e executado como uma aplicação web, que não suporta capacidade de integração com ferramentas externas. O ambiente de suporte está sendo refinado para oferecer pontos de extensão na execução do workflow para integrar ferramentas externas que são usadas ou necessárias durante a execução do experimento. Também pretende-se fornecer expressividade para modelar esta integração na *ExpDSL* como forma de manter o alto nível de abstração para o pesquisador que está modelando o experimento controlado.

5.5. Discussões

Esta seção discute pontos relevantes relativos a abordagem proposta.

Flexibilidade do Ambiente. A abordagem permite fácil adaptação para a integração de novos tipos de *design* estatístico. Isso se dá em virtude de termos o processo de geração do arquivo de configuração da execução (organização e aleatorização do experimento) independente da geração do workflow. Neste caso, qualquer necessidade de alteração para estender os tipos de *designs* estatístico irá afetar apenas a transformação modelo para texto responsável pela criação do arquivo de configuração, essa alteração ocorre apenas pela inserção do código referente a geração do novo tipo de organização (distribuição de participantes e tratamento, além da ordem de execução, se for o caso).

Análise do Experimento. Embora não tenha sido diretamente explorado pela abordagem, nós realizamos uma prototipação da análise no ambiente através da integração, via plataforma Java, com a ferramenta estatística *R*[6]. Esta ferramenta possui uma API para comunicação com Java (*API Java/R Interface* - JRI)[7] que habilita a chamada de funções *R* dentro do código Java e, neste caso, pode ser utilizada para realizar a análise estatísticas dos experimentos individuais e suas réplicas. Além disso, é possível apresentar os gráficos resultantes da análise no ambiente web de execução habilitando a interpretação destes gráficos em Java através da biblioteca *R* chamada *Java Graphics Device* (JavaGD)[8]. Esta biblioteca fornece objetos Java *canvas* equivalentes aos gráficos produzidos pela ferramenta *R*. Uma vez obtidos estes objetos, eles podem ser tratados e transferidos para um *framework* de visão tal como o *JavaServer Faces* (JSF) usado pelo ambiente proposto.

5.6. Conclusões

Este capítulo apresentou uma abordagem dirigida por modelo para a geração de workflows customizados para os participantes do experimento que são automaticamente distribuídos e executados de acordo com o projeto do experimento. A abordagem foi implementada utilizando tecnologias dirigidas por modelo e foi realizado um estudo exploratório através da modelagem, geração de workflows e instanciação dos mesmos no ambiente de execução. O estudo exploratório mostrou uma visão positiva do

[6] http://www.r-project.org/
[7] http://www.rforge.net/JRI/
[8] http://rosuda.org/R/JavaGD/

pesquisador responsável pelo experimento alvo em relação ao modelo criado, aos workflows e suas instâncias de execução.

6. Avaliação e Evolução de uma Linguagem Específica de Domínio para Formalizar Experimentos em Engenharia de Software: Um Estudo Empírico.

O estudo apresentado neste capítulo foca na avaliação da linguagem específica de domínio proposta neste trabalho (Capítulo 4) para a formalização de experimentos controlados em engenharia de software (FREIRE, ACCIOLY, *et al.*, 2013) (FREIRE, 2013).

6.1. Introdução

No capítulo anterior, um experimento relacionado ao teste de linha de produto (ACCIOLY, BORBA e BONIFÁCIO, 2012) (ACCIOLY, BORBA e BONIFACIO, 2014) foi modelado usando a versão avaliada da nossa DSL (ExpDSLv1). Este experimento modelado compara duas diferentes técnicas caixa-preta: uma técnica genérica e uma técnica específica por produto, conforme apresentado na Seção 5.4.2.1. O estudo avalia as técnicas do ponto de vista do processo de execução de testes. Na especificação, é possível encontrar (i) uma visão de Plano Experimental (Figura 20 - A,B,C,D,E,F): objetivo, hipóteses nula e alternativa, o design estatístico do experimento definido como quadrado latino com duas variáveis de bloco – Feature e Subjects – e o fator sob investigação (TestTechnique), além dos parâmetros – LPSComplexity e Participants; (ii) *um fragmento da Visão de Processo* (Figura 20 – G) que mostra uma atividade do processo responsável por guiar os participantes durante a execução. Nela podemos ver o caso de teste a ser executado com duas tarefas relacionadas ("execute the test" e "relate the change request"); (iii) um fragmento da visão de Métrica (Fig. 1 - H) que mostra duas métricas: uma para coletar o tempo gasto nos testes e outra para coletar e reportar solicitações de mudanças (*reported change request*); e (iv) um fragmento da visão de questionário (Figura 20 – I) onde nós podemos ver o questionário chamado ExperimentFeedback, definido para ser coletado após a execução do experimento (atributo Post). Apenas uma questão com suas alternativas é apresentada. A Seção 4 discute estes fragmentos no contexto do nosso estudo.

6.2. Configuração do Estudo

Esta seção apresenta o estudo em termos de objetivo principal e questões de pesquisa (Seção 6.1.1), os experimentos selecionados para serem modelados (Seção 6.1.2), os critérios de avaliação (Seção 6.1.3) e a metodologia do estudo (Seção 6.1.4).

6.2.1 Objetivo do Estudo e Questões de Pesquisa

O principal objetivo deste estudo é avaliar a linguagem específica de domínio (ExpDSL) com respeito a modelagem de experimentos em engenharia de software sob a perspectiva dos experimentadores. Para alcançar este objetivo, quatro questões de pesquisa (QP) foram definidas:

- **QP1**: Os elementos/construtores de ExpDSL são adequados para a modelagem de diferentes conceitos do domínio de engenharia de software experimental?
- **QP2**: Os elementos/construtores de ExpDSL são adequados para expressar exatamente um conceito distinto no domínio?
- **QP3**: Há uma e apenas uma maneira de expressar cada conceito de interesse em ExpDSL?
- **QP4**: Os elementos/construtores de ExpDSL correspondem a conceitos relevantes do domínio?

Com o objetivo de responder estas questões, nosso estudo investigou como os diferentes aspectos de um experimento controlado poderiam ser atendidos por ExpDSL. Critérios diferentes e complementares foram adotados para analisar a completude e expressividade da DSL durante a modelagem de um conjunto de experimentos existentes. A próxima seção detalha os experimentos modelados no estudo e como eles foram selecionados.

```
Experiment "SPL Testing Strategies"
Experimental Plan
Goals
{
G1 "Analyze test execution effort, for the purpose of evaluating the effect of two differen test case
design techniques for SPL (GT vs. ST)"
}
Hypotheses{
  H0 "The time to execute the test suites is the same for the two techniques" null from G1
  H1 "The time to execute the test suites using ST technique is less than using GT technique"
alternative from G1
  H20 "The number of terminated CRs using ST is the same compared with the GT" null from G1
  H21 "The number of terminated CRs using ST is less than using GT technique" alternative from G1
}

DesignType LS - Latin Square  {

 Parameter LPS_complexity = "low"
 Parameter Participants = "undergraduating students"

  Factor TestTechnique isDesiredVariation True
    Level Generic Level Specific

  Factor Feature isDesiredVariation False
    Level F1 Level F2

  Factor Subject isDesiredVariation False
    Level Subject1 Level Subject2

  Internal Replication 4

  Link SpecificTecFeature1_Tests to TestTechnique.Specific Feature.F1
  Link SpecificTecFeature2_Tests to TestTechnique.Specific Feature.F2
                …
}
...

Process SpecificTecFeat1_CT{
  Activity SP1_F1_1 description "Insert a new member - Product 1" to SP1_F1_2 Role "Subject" {
    Task ExecuteTCSP1_F1_1 description "Execute the test case SP1_F1_1"
    artefacts artefact  TesteCaseSP1_F1_1 description "Test case details SP1_F1_1" type input


    Task ReportCRSP1_F1_1 description "Describe TC SP1_F1_1 results"
    artefacts artefact CR_SP1_F1_1 description "Reporting the CR for Test Case SP1_F1_1" type output
  }
…
...
//Metrics
Metric TimeTesting_ST_F1_Product1 relates SpecificTecFeature1_Tests {
description "Test Case Execution time for F1 in P1"
unit minutes
activityBegin SpecificTecFeat1_Tests.SP1_F1_1 activityEnd  SpecificTecFeat1_Tests.SP1_F1_6
}
Metric ReportedCR1 relates SpecificTecFeature1_Tests {
description "Valid Terminated CR"
artefacts CRs
SpecificTecFeat1_CT.SP1_F1_1.ReportCRSP1_F1_1.CRSP1_F1_1
SpecificTecFeat1_CT.SP1_F1_2.ReportCRSP1_F1_2.CRSP1_F1_2
SpecificTecFeat1_CT.SP1_F1_3.ReportCRSP1_F1_3.CRSP1_F1_3
SpecificTecFeat1_CT.SP1_F1_4.ReportCRSP1_F1_4.CRSP1_F1_4
SpecificTecFeat1_CT.SP1_F1_5.ReportCRSP1_F1_5.CRSP1_F1_5
SpecificTecFeat1_CT.SP1_F1_6.ReportCRSP1_F1_6.CRSP1_F1_6
}
...
Questionnaires
Questionnaire  ExperimentFeedback type Pos {
         Q1_Scholling {
         escription "What is your scholling:"
         type ComboBox
         Alternatives
         Option "Undergraduate student" Option "Graduate" Option "Graduate student" Option "Master"
               }
```

Figura 20: Especificação do Experimento Testing em ExpDSLV1

6.2.2 Experimentos Alvo Modelados

O estudo selecionou um total de 16 experimentos – 3 quasi-experimentos e 13 experimentos controlados – com diferentes designs estatísticos, executados e documentados pela comunidade de engenharia de software. Oito deles foram experimentos publicados na literatura, e executados por nosso grupo de pesquisa ou grupos de pesquisa parceiros de diferentes institutos de pesquisa ou universidades. Foram considerados experimentos que avaliaram tecnologias de diferentes áreas da engenharia de software, tais como requisitos, desenvolvimento orientado a modelos, linhas de produto de software, testes e fatores humanos. Finalmente, o conjunto de experimentos foi também incrementado através da seleção de oito experimentos que foram publicados no *Journal of Empirical Software Engineering* (ESE) e no *International Journal of Software Engineering and Knowledge Engineering* (IJSEKE) nos últimos 5 anos. É importante mencionar que nossa seleção também considerou a disponibilidade de documentação sobre o planejamento e a condução do experimento. A Tabela 10 lista os experimentos usados em nosso estudo.

Tabela 10: Experimentos modelados neste estudo

Experiment	**Institution-Country**	**Experiment**	**Institution-Country**
Testing (ACCIOLY, BORBA e BONIFÁCIO, 2012)	UFPE-Brazil	Abstract type (BUDGEN, J. BURN e KITCHENHAM, 2011)	Durham University – United Kingdom
Human Factors (ACUÑA, GÓMEZ e JURISTO, 2009)	UAM-Spain	Event-driven (GÜLESIR, VAN DEN BERG, *et al.*, 2009)	UT - The Netherlands
Requirements (ARANDA, DIESTE e JURISTO, 2012)	UPM-Spain	UML statechart (CRUZ-LEMUS, GENERO, *et al.*, 2009)	UCLM- Spain
SPrL (ALEIXO, KULESZA e JUNIOR, 2013)	UFRN-Brazil	Early Usability (PANACH, CONDORI-FERNÁNDEZ, *et al.*, 2011)	UPV-Spain
MDD	UPV-Spain	Program Comprehension (KOSAR, MERNIK e CARVER, 2012)	UM - Slovenia
SPL (CIRILO, NUNES, *et al.*, 2011)	PUC-Rio-Brazil	LPS background color (FEIGENSPAN, KÄSTNER, *et al.*, 2013)	OVGU Magdeburg - Germany
CFT (JUNG, JEDLITSCHKA, *et al.*, 2013)	Fraunhofer-IESE-Germany	TDD (FUCCI e TURHAN, 2014)	University of Oulu - Finland
PBR (BASILI e SELBY, 1987)	University of Maryland-USA	Feature Maintenance (RIBEIRO, BORBA e KÄSTNER, 2014)	UFAL - Brazil

6.2.3 Critérios de Avaliação

O estudo adotou os seguintes critérios existentes para avaliação de DSLs (KAHRAMAN e BILGEN, 2013):

Completude: Este critério analisa se todos os conceitos do domínio podem ser expressos na DSL. Para analisar a completude, nós investigamos se diferentes aspectos dos experimentos selecionados podem ser especificados adequadamente usando a versão original da nossa DSL, daqui em diante chamada de ExpDSLv1. Os seguintes aspectos de experimentos em engenharia de software (JEDLITSCHKA, CIOLKOWSKI e PFAHL, 2008) foram avaliados durante nosso processo de especificação: Título, Resumo, Objetivos, Questões de Pesquisa, Hipóteses, Design do Experimento, Variáveis Independentes, Variáveis Dependentes, Métricas, Instrumentos de Medida, Caracterização (Contextualização), Procedimento de Coleta de Dados, Ameaças a Validade, e Questionários.

Expressividade. Este critério relata o grau no qual uma estratégia de resolução de problemas pode ser mapeada em um programa de forma natural. Neste estudo, nós avaliamos três sub-características da Expressividade: (i) a ortogonalidade, que diz que cada construtor da DSL é usado para representar exatamente um conceito distinto no domínio; (ii) a unicidade, que analisa se a DSL fornece uma e apenas uma única boa maneira de expressar todo conceito de interesse; e, finalmente, (iii) nós discutimos a correspondência a importantes conceitos do domínio (de acordo com autores de referência na ESE), que avalia se os construtores da linguagem correspondem a importantes conceitos do domínio.

A análise da ortogonalidade envolveu verificar se cada construtor ExpDSLv1 foi usado para especificar exatamente um conceito durante a especificação dos experimentos selecionados. Desta forma, cada vez que nós encontramos construtores similares sendo usados para especificar diferentes conceitos do experimento, isto indicou uma falta de expressividade da DSL.

Para analisar o critério da unicidade nós mapeamos os elementos de ExpDSL nos conceitos do domínio a fim de avaliar se a DSL fornece uma e apenas uma maneira de expressar cada conceito de interesse. Desta forma, se houver dois ou mais formas de

expressar o mesmo conceito de interesse, isto indica problema de expressividade da DSL.

Finalmente, a correspondência a importantes conceitos do domino foi explorada pela avaliação se os construtores da linguagem correspondem a conceitos do domínio em um plano experimental de acordo com a literatura da comunidade de engenharia de software experimental (JEDLITSCHKA, CIOLKOWSKI e PFAHL, 2008) (JURISTO e MORENO, 2010) (WOHLIN, RUNESON, *et al.*, 2012).

6.2.4 Metodologia do Estudo

Nosso estudo foi organizado em quarto estágios, que consistiram em: (i) seleção e especificação de diferentes experimentos usando a versão original de ExpDSL (chamada ExpDSLv1) ; (ii) a avaliação de cada experimento modelado através dos critérios do estudo (Seção 6.1.3); (iii) análise, discussão e proposta de melhorias para a linguagem (e abordagem como um todo) considerando os resultados do estudo; e (iv) modelagem e análise dos mesmos experimentos na nova versão de ExpDSL, daqui em diante chamada de ExpDSLv2.

Diferentes aspectos dos experimentos selecionados (Seção 6.1.2) foram modelados usando ExpDSLv1 de acordo com a documentação disponível sobre cada experimento. Para alguns experimentos, nós também interagimos com os pesquisadores que os conduziram para obter informações adicionais. Após a modelagem dos diferentes experimentos, nós analisamos os critérios de completude e expressividade (através de suas sub-características) das especificações produzidas em ExpDSLv1. Durantes esta análise, nós investigamos, para cada critério: (i) razões para a (não) adequada especificação de cada aspecto do experimento; e (ii) como adaptar a DSL para alcançar uma melhor modelagem dos aspectos identificados como não adequados. Após isso, nós incorporamos e implementamos as novas melhorias na linguagem, criando uma nova versão de ExpDSL (ExpDSLv2). Finalmente, no último estágio do estudo, os mesmos experimentos foram remodelados usando a versão melhorada da DSL para que fosse possível destacar os resultados alcançados. Figura 21 mostra fragmentos dos novos (ou modificados) elementos da linguagem.

6.3. Resultados do Estudo

Esta seção apresenta e discute os resultados do nosso estudo. A Seção 6.3.1 dá uma visão geral do resultado. Após isso, os resultados específicos para cada critério são apresentados e discutidos: a Seção 6.3.2 examina a análise de completude, e a Seção 6.3.3 explora o critério de expressividade, com diferentes sub-características: ortogonalidade (Seção 6.3.3.1), análise de unicidade da linguagem (Seção 6.2.3.2) e, finalmente, a correspondência aos conceitos do domínio (Seção 6.3.3.3).

6.3.1 Análise Geral

A modelagem de experimentos em nosso estudo revelou que a DSL investigada atende satisfatoriamente a maioria dos critérios avaliados. Por outro lado, oportunidades de melhorias foram expostas para elementos e aspectos específicos da DSL.

Para o critério de completude, nosso estudo considerou três níveis de suporte para cada aspecto considerado na modelagem dos experimentos: (i) suportado – foi possível especificar o aspecto considerado; (ii) parcialmente suportado – ExpDSL apenas fornece suporte parcial para a modelagem dos aspectos experimentais; e (iii) não suportado – a DSL não suporta a especificação do aspecto considerado. A Tabela 11 resume os resultados para o critério de completude: os aspectos experimentais que foram suportados, parcialmente suportados e não suportados para cada elemento analisado. Os resultados ilustram que ExpDSLv1 fornece suporte para aproximadamente 50% dos aspectos dos experimentos modelados, 17% deles foram parcialmente suportados, e 33% foram não suportados. Cinco conceitos não foram suportados para todos os experimentos modelados, eles foram: resumo (*abstract*), questões de pesquisa (*research question*), variáveis dependentes (*dependent variables*), instrumentos de medida (*measurement instruments*), e ameaças à validade (*threats to validity*). Entre outras mudanças, estes resultados nos motivaram a propor extensões à ExpDSLv1, tais como: (i) um elemento que permite a descrição do estudo (elemento *Abstract*); (ii) um elemento para definir as variáveis dependentes e que relaciona este conceito com os objetivos do experimento, permitindo a rastreabilidade entre eles e facilitando a condução de meta-análises (elemento *DependentVariable*); (iii) um elemento que permite expressar as questões de pesquisa como uma alternativa para a descrição do experimento, e define a conexão com os objetivos do experimento (elemento *ResearchQuestion*); e (iv) um elemento que permite documentar as ameaças a

validade que pode ser usadas para meta-análises e pode motivar futura replicação dos experimentos (elemento *ThreatToValidity*).

Tabela 11: Resultados para a completude de ExpDSLv1

Aspecto Experimental	**Resultado**	**Aspecto Experimental**	**Resultado**
Title	Suportado	*Metrics*	Parcialmente Suportado
Abstract	Não Suportado	*Measurement Instruments*	Não Suportado
Goals	Suportado	*Characterization/ Contextualization*	Suportado
Research Questions	Não Suportado	*Data Collection Procedure*	Suportado
Hypotheses	Suportado	*Experimental roles*	Parcialmente Suportado
Design of experiment (DoE)	31% Suportado 69% Não Suportado	*Threats to validity*	Não Suportado
Independent variables	Suportado	*Questionnaire*	Suportado
Dependent Variables	Não Suportado		

Para nosso outro critério de análise, o estudo concluiu os seguintes resultados, que estão resumidos na Tabela 12: (i) o critério de unicidade revelou três maneiras de expressar o conceito de domínio métrica em ExpDSLv1 usando diferentes elementos (ActivityMetric, TaskMetric e ArtefactMetric); (ii) o critério de ortogonalidade nos mostrou que dois construtores da DSL (Process e Metric) eram usados para especificar diferentes

conceitos dos experimentos especificados. Por exemplo, o elemento Metric estava sendo originalmente usado para modelar métricas e variáveis dependentes; e, finalmente, (iii) as análises sobre a sub-característica correspondência a importantes conceitos do domínio mostrou três elementos de ExpDSLv1 que não correspondem a conceitos do domínio de acordo com a literatura base (ameaças a validade, questões de pesquisa e variável dependente). Nós mapeamos cada elemento ExpDSLv1 no conceito correspondente como definido por Jedlitschka et al. (JEDLITSCHKA, CIOLKOWSKI e PFAHL, 2008), Juristo et al. (JURISTO e MORENO, 2010), e Wohlin et al. (WOHLIN, RUNESON, *et al.*, 2012).

Tabela 12: Resultados para a Expressividade de ExpDSLv1

Critério	**Conceito do Domínio**	**Elemento em ExpDSLv1**
Unicidade	Metric	ActivityMetric TaskMetric ArtefactMetric
Ortogonalidade	Data Collection Procedure Measurement Instruments	Process
	Metric Dependent Variable	Metric

Os resultados do estudo demandaram uma investigação de como nós poderíamos melhorar ExpDSLv1 a fim de possibilitar uma adequada especificação de experimentos. As próximas seções detalham os resultados do estudo em relação à avaliação de cada diferente critério apresentando limitações da linguagem, e propondo e apresentando novas melhorias.

6.3.2 Análise do Critério de Completude

Esta seção apresenta os principais resultados do critério de completude considerando os diferentes aspectos experimentais que foram modelados usando ExpDSLv1. Nós mostramos e discutimos cada resultado observado quando modelando os experimentos. Além disso, nós também descrevemos como estes resultados foram usados para propor melhorias e novas extensões para ExpDSL (ExpDSLv2).

I. Hipóteses Estatística e de Pesquisa

ExpDSLv1 permite a definição de hipóteses estatísticas para o experimento através do elemento Hypothesis. Neste estudo, foi possível modelar todas as hipóteses dos experimentos usando tal elemento da linguagem. A Fig. 1 – (B) mostra a especificação da hipótese para o experimento "Testing".

Apesar deste resultado, nós decidimos mudar o elemento para especificar a hipótese de pesquisa ao invés da hipótese estatística. Essa mudança simplifica a especificação e permita manter a rastreabilidade com o objetivo. A Fig. 3 (D) mostra as mudanças aplicadas.

II. Questão de Pesquisa

Um dos conceitos de domínio que não era possível especificar em ExpDSLv1 foi o das questões de pesquisa. ExpDSLv1 não fornece construtor para especificar este conceito. Durante as modelagens dos experimentos, nós estávamos transformando as hipóteses de pesquisa em questões de pesquisa a fim de especificá-los. Este novo elemento foi então proposto para ser incorporado à ExpDSLv2 onde é possível representar questões de pesquisa e associá-las a seu correspondente objetivo. A Fig. 3 (C) mostra, por exemplo, duas questões de pesquisa da modelagem do experimento "Testing" (RQ1 e RQ2) e associados aos seus relacionados objetivos.

III. Projeto Experimental

ExpDSLv1 suporta a definição de três projetos experimentais (Design of Experiment - DoE): (i) completamente aleatorizado (complete randomized Design - CRD); (ii) completamente aleatorizado em blocos (randomized complete block design - RCBD); e (iii) quadrado latino (Latin square - LS). A Figura 20 (C) mostra o projeto do quadrado latino especificado para o experimento "Testing". Neste estudo, foi possível modelar 10 dos 16 projetos experimentais definidos nos experimentos selecionados. Estes 6 experimentos usaram, por exemplo, outros projetos experimentais como *between subjects, within-subject* e design com medidas repetidas (*repeated measurement design*) (WOHLIN, RUNESON, *et al.*, 2012) (JURISTO e MORENO, 2010) (ACUÑA, GÓMEZ e JURISTO, 2009) (ARANDA, DIESTE e JURISTO, 2012). Este fato ocorreu, por exemplo, para o experimento PBR (BASILI e SELBY, 1987), que adota o design fatorial.

```
Experiment Testing "SPL Testing Strategies"

Experimental Design

Abstract {
"This study is an experiment that compares two different black-box manual test design
techniques: a generic technique that we have observed in an industrial test execution
environment and a product specific technique whose functional test cases could be derived using
any SPL technique that considers variations in functional tests"
}

Goals{
  G1 Analyze "test execution effort" for the purpose of "evaluating the effect of two different
          test case design techniques" with respect to their "efficiency regarding time to
          execute the test suites as well as the number of terminated CRs reported during the
          test execution process" the point of view of the "test engineers and software
          engineering researchers" in the context of  "controlled experiments done with graduate
          and undergraduate students"
}
Research Questions {
  RQ1 "Does the ST reduce the test execution effort compared with the GT?" relates to G1
  RQ2 "Does the ST reduce the number of terminated CRs compared with the GT?" relates to G1
}

Research Hypotheses {
  RH1 "The time to execute the test suites using ST technique is less than using GT technique"
relates to G1
  RH2 "The number of terminated CRs using ST is better than compared with the GT" relates to G1
}

Dependent Variables {
  TestExecutionTime "Test execution productivity" Scale Numerical relates to RH RH1
  NumberCRs "Number of terminated CRs (that report real defects)" Scale Numerical relates to RH
RH1
}

Factors {
  TestTechnique "The SPL test technique used" Scale Nominal Range {Generic Specific}
  Feature "The SPL feature tested. The features chosen to design the test suites were
Publications and ResearchLines" Scale Nominal Range {F1 F2}
  Subject "the subjects selected to execute the experiment." Scale Nominal Range {Subject1
Subject2}
}

     DoE = LS - Latin Square (column = Feature,  row = Subject, treatment = TestTechnique)

Context {
  LPSComplexity "The complexity of SPL tested" Scale Nominal Range {low}
  SubjectsScholling "The participants schooling" Scale Nominal Range {undergraduateStudents}
 }
  Internal Replication 4
...
Process SpecificTecFeat1_Tests to TestTechnique.Specific Feature.F1{
  Role Participant
  Task SP1_F1_1 description "Insert a new member - Product 1" to SP1_F1_2 {
        artefacts TesteCaseSP1_F1_1 description "Test case SP1_F1_1" type input
 CR_SP1_F1_1 description "CR Report the CR for SP1_F1_1" type output
  }

  Task SP1_F1_2 description "Insert a new conference paper  - Product 1" to SP1_F1_3 {
 artefacts TestCaseSP1_F1_2 description "Test case details SP1_F1_2"  type input
 CRSP1_F1_2 description "CR Reported for SP1_F1_2" type output
  }
...
Questionnaires
Questionnaire  ExperimentFeedback type Post {

 Q1_Scholling {
   description "What is your schooling:"
 type SingleChoice {"Undergraduate student" "Graduate" "Graduate student" "Master"}
}
...
Threats to Validity {
  TV1 description "Heterogeneous environment were used during the experiment operation" type
Internal validation control action(s) "the irreproducible defects will not be considered in the
CR analysis"
  TV2 description "The subjects involved in this experiment were not all testers" type External
validation
  TV3 description "The SPL used in tests was a non-real SPL, the RGMS" type Conclusion
}
```

Figura 21: Especifcação do Experimento Testing em ExpDSLv2

Para permitir a modelagem de experimentos que não seguem os projetos experimentais (DoE) suportados em ExpDSL, nós estendemos a linguagem para incluir a opção de projeto chamada “Other”. Esta mudança tem um impacto direto na configuração de um experimento. Sem conhecimento sobre o projeto experimental, a abordagem de suporte não poderá realizar as transformações de forma a automaticamente aleatorizar os participantes e alocar os tratamentos aos mesmos durante a segunda fase da abordagem (quando o projeto estiver configurado como “Other” em ExpDSLv2). Portanto, o experimentador tem que configurar esta informação manualmente usando o ambiente de execução. Apenas após essa configuração manual, o ambiente irá instanciar os workflows corretamente para iniciar a execução do experimento.

A escolha do DoE “Other” também implica em dificuldades para a validação automática do editor ExpDSLv2 para as regras e suposições estatísticas. O editor da DSL pode checar, por exemplo, se os requisitos do projeto estatístico selecionado (DoE) são satisfeitos: o quadrado latino, por exemplo, requer duas variáveis de bloco. Portanto, a opção pelo DoE “Other” aumenta a flexibilidade da linguagem mas, por outro lado, restringe validações na especificação. Quando usando a opção “Other” para especificar o projeto do experimento, o pesquisador registra, como um elemento texto, a DoE usado no experimento.

IV. Variável Dependente

Em ExpDSLv1, nós podemos definir métricas, mas sem fornecer mecanismos explícitos para especificá-las como variáveis dependentes (também conhecidas como variáveis de resposta). Por isto, nós estendemos ExpDSLv1 inclui a modelagem de variáveis dependentes, que podem ser associadas a questões de pesquisa ou hipóteses de pesquisa que são respondidas ou apoiadas pela variável independente.

Durante a definição do processo de coleta de dados (elemento process), o pesquisador tem diferentes formas de coletar dados relacionados as variáveis dependentes usando ExpDSLv2, tais como: (i) coletar artefatos criados ou alterados durante o experimento e que deve ser analisado pelo pesquisador; (ii) solicitar que o participante informe um dado específico através de um formulário (por exemplo: informar o número de defeitos encontrados) durante a execução das tarefas do

experimento; ou (iii) responder um questionário associado a alguma questão (ou questões) (através de uma referência cruzada a um questionário definido no experimento).

A Figura 21 – (E) mostra as variáveis dependentes especificadas para o experimento "Testing": (i) o número de solicitações de mudanças válidas "Change Request" (NumberCRs); e (ii) o tempo gasto para executar os testes (TestExecutionTime). A variável TestExecutionTime, por exemplo, tem uma descrição, e pode receber um valor numérico. Estas variáveis dependentes foram originalmente definidas na visão de Métrica em ExpDSLv1 – Figura 20 (H).

V. Procedimento de Coleta de Dados

Em ExpDSLv1, os procedimentos de coleta de dados são especificados como um processo, na visão de Processos, incluindo atividades, tarefas, papéis, e artefatos. A versão atual de ExpDSL não fornece suporte a loops e caminhos condicionais, mas apesar disso, todos os experimentos puderam ser modelados como procedimentos sequenciais. Em ExpDSLv1, um processo era composto de uma sequência de atividades e uma atividade poderia agrupar um conjunto de tarefas. Nos experimentos modelados neste estudo, não foi necessário agrupar tarefas em atividades. Desta forma, a fim de simplificar a especificação de um experimento, nós removemos o elemento atividade (elemento Activity) de ExpDSLv1. Portanto, em ExpDSLv2, o processo é diretamente composto de uma sequência de tarefas.

Além disso, a informação semântica definida através do elemento Link – que determina a ligação semântica entre um processo e o tratamento correspondente – em ExpDSLv1 (Figura 20 – F) foi movido para a definição do processo (Figura 21 – I). Por esta razão, o elemento Link foi removido da DSL.

VI. Ameaças à Validade

Durante a avaliação de completude, nós observamos que não era possível especificar ameaças à validade em ExpDSLv1. Este conceito de domínio não foi abordado na DSL. Portanto, ExpDSLv2 incluiu este conceito de interesse. A Figura 21 (K) mostra as ameaças à validade para o experimento "Testing" usando ExpDSLv2. A ameaça TV1, por exemplo, está descrita como "Heterogeneous environment was used

during the experiment operation", que representa uma ameaça interna (internal validation). A ação de controle definida é “the irreproducible defects will not be considered in the change request analysis”.

Análise do Critério de Completude para ExpDSLv2. Nós reavaliamos a completude da versão melhorada da DSL através da remodelagem de todos os experimentos analisados em ExpDSLv1. Os novos resultados mostraram de 13 de 15 (87%) dos aspectos experimentais dos experimentos modelados foram suportados, 1 de 15 (6,5%) foi parcialmente suportado, e apenas 1 de 15 (6,5%) foi não suportado. A Tabela 13 apresenta esses novos resultados. A eliminação do elemento métrica (elemento Metric) não afetou a capacidade de especificar experimentos porque os dados medidos continuam a ser coletados durante a execução do processo através das tarefas (elemento Task) como artefatos, questionários ou campos de texto. Finalmente, nós também observamos uma melhoria de completude relacionada a inclusão da opção “Other” para especificar um tipo de projeto estatístico (DoE), além do quadrado latino, completamente aleatoriazado em blocos, e completamente aleatorizado.

6.3.3 Análise do Critério de Expressividade

Nesta seção, nós apresentamos e discutimos os resultados para o critério de expressividade aplicado em ExpDSLv1. A Seção 6.2.3.1 descreve os resultados para a sub-característica unicidade. A Seção 6.2.3.2 apresenta os resultados para ortogonalidade. Finalmente, a Seção 6.2.3.3 discute o critério de correspondência a importantes conceitos do domínio ExpDSLv1.

Tabela 13: Análise de Completude para ExpDSLv2

Aspecto Experimental	**Resultado**	**Aspecto Experimental**	**Resultado**
Title	Suportado	*Metrics*	Não Suportado
Abstract	Suportado	*Measurement Instruments*	Parcialmente Suportado
Goals	Suportado	*Characterization/Contextualization*	Suportado

Research Questions	Suportado	*Data Collection Procedure*	Suportado
Hypotheses	Suportado	*Experimental roles*	Suportado
Design of experiment (DoE)	Suportado	*Threats to validity*	Suportado
Independent variables	Suportado	*Questionnaire*	Suportado
Dependent Variables	Suportado		

6.3.3.1 Unicidade

Para avaliar este critério, nós analisamos a maneira de expressar cada elemento do domínio em ExpDSLv1. A fim de proceder com esta análise, nós relacionamos os aspectos do domínio com o elemento correspondente da DSL que é usado para expressar este conceito. Os mesmos aspectos do domínio que nós usamos para avaliar a completude foram mapeados para os elementos que os expressam em ExpDSLv1. Os aspectos foram: Título (Title), Resumo (Abstract), Objetivos (Goals), Questão de Pesquisa (Research Question), Hipóteses (Hypotheses), Projeto Experimental (Design of experiment - DoE), Variáveis independentes (Independent variables), Variáveis Dependentes (Dependent Variables), Métricas (Metrics), Instrumentos de Medida (Measurement Instruments), Caracterização do Experimento (Characterization/Contextualization), Procedimento de Coleta de Dados (Data Collection Procedure), Papéis Experimentais (Experimental roles), Ameaças à Validade (Threats to Validity), e Questionários (Questionnaires). Após mapear todos os conceitos experimentais na DSL, nós avaliamos quais deles apresentam mais de uma maneira de ser expresso em ExpDSLv1.

O estudo constatou que ExpDSLv1 continha apenas um conceito de domínio – Métrica – que pode ser expresso de mais de uma maneira. As métricas de um experimento podiam ser expressas através do elemento ActivityMetric quando ele é relacionado à uma atividade do experimento, através do elemento TaskMetric quando está relacionado a uma tarefa do experimento, ou mesmo com um elemento ArtefactMetric quando se referia aos artefatos de tarefas ou atividades. A fim de evitar tais variedades de construtores para especificar as métricas do experimento, o elemento

```
SimpleAbstract : description=STRING;
StructuredAbstract: {StructuredAbstract}
    ('Background' background=STRING)?
    ('Objective' objective=STRING)?
    ('Methods' methods=STRING)?
    ('Results' results=STRING)?
    ('Limitations' limitations=STRING)?
    ('Conclusions' conclusions=STRING)?;
```

```
SimpleGoal: name=ID (description=STRING)?;

StructuredGoal: name=ID 'Analyze' object=STRING  'for the purpose of'
technique=STRING 'with respect to their' quality=STRING 'the point of
view of the' ptView=STRING 'in the context of' contextOf=STRING;
```

Figura 22: Possibilidades de especificação do aspecto "Abstract"

Metric foi removido para simplificar a especificação em ExpDSL. A semântica da especificação de como medir uma variável dependente foi então transferida para a especificação de processo, como mostrado na Seção 6.2.3.IV.

Por outro lado, ExpDSLv2 contém dois conceitos de domínio que podem ser expressos de duas diferentes maneiras cada um. Os conceitos Resumo (Abstract) e Objetivos (Goal) apresentam duas maneiras de ser expressos: simples ou estruturada. A Figura 22 ilustra estas duas diferentes maneiras para ambos os conceitos. A versão foi estendida para prover estas duas maneiras a fim de facilitar a especificação e melhorar a meta-análise dos experimentos documentados. Estes resultados diminuíram o nível de suporte da sub-característica unicidade, mas por outro lado deu duas maneiras alternativas de descrever estes conceitos dependendo da necessidade dos usuários de ExpDSL.

6.3.3.2 Ortogonalidade

Na análise da ortogonalidade de ExpDSLv1, foi observado dois elementos que são, cada um, usado para descrever diferentes conceitos. Estes elementos foram: (i) Elemento Process – usado para descrever o procedimento de coleta de dados e o os instrumentos de medida. A Figura 20 (G) mostra, por exemplo, um processo que mantém uma atividade com dois artefatos que são instrumentos de medida: test cases e reported CRs; e (ii) Elemento Metric – usado para descrever as variáveis dependentes e métricas juntas. A Figura 20 (H) mostra duas métricas, uma responsável pela variável dependente tempo e outra pela variável dependente corretude. Consequentemente, ExpDSLv1 não suporta completamente a característica da ortogonalidade, porque

existem dois elementos – Process e Metric – que são usados para descrever diferentes conceitos.

Em ExpDSLv2, apenas o elemento Process foi preservado para expressar dois conceitos do domínio: os procedimentos de coleta de dados e os instrumentos de medida. Esta decisão de não separar foi tomada porque os instrumentos de medida são usados durante o processo do experimento em muitas tarefas. Portanto, é adequado representá-lo durante a definição de um processo. Por outro lado, o elemento Metric foi removido. Como uma conclusão, ExpDSLv2 não fornece suporte a ortogonalidade apenas para o elemento Process. Todos os outros elementos da linguagem são usados para expressar apenas um único conceito.

6.3.3.3 Correspondência a Importantes Conceitos do Domínio

Este critério pretende avaliar se os elementos da linguagem correspondem a importantes conceitos do domínio. A fim de checar esta correspondência, nós mapeamos cada elemento ExpDSLv1 em seu correspondente conceito de domínio considerando os conceitos de domínio definidos em três relevantes referências para a engenharia de software experimental (WOHLIN, RUNESON, *et al.*, 2012) (JURISTO e MORENO, 2010) (JEDLITSCHKA, CIOLKOWSKI e PFAHL, 2008). A Tabela 14 detalha este mapeamento.

Esta análise de correspondência concluiu que apenas dois elementos de ExpDSLv1 não tiveram correspondência com importantes conceitos do domínio: elementos Link e Sub-hypotheses. O elemento The Link (Figura 20 – I) é responsável para semântica de ligação entre cada processo e seu correspondente tratamento, e o elemento Sub-hypotheses é usado para decompor as hipóteses. A fim de simplificar a linguagem para apenas conter importantes conceitos do domínio foi optado por mover a semântica do elemento Link para o elemento Process em ExpDSLv2, assim, quando especificando um processo nós temos que associar o correspondente tratamento, como mostra a Figura 21 (I). O elemento Subhypotheses foi eliminado para simplificar as especificações ExpDSL, porque ele não representa um conceito concreto do domínio reportando na literatura.

Tabela 14: ExpDSLv1 – Correspondência a importante conceitos do domínio

Elemento ExpDSLv1	Conceitos de Domínio		
	Jedlitschka et al.	Wohlin et al.	Juristo et al.
Goal	Goal	Goal	Goal
Hypotheses	Hypotheses (null and alternative)	Hypotheses (null and alternative)	Hypotheses (null and alternative)
Subhypotheses	N/A	N/A	N/A
Parameter	Parameter	Context	Parameter
Factor	Variable (independent variable)	Factor	Factor
Level	Level	Level	Alternative or Level
Link	N/A	N/A	N/A
Questionnaire	Experimental materials	Measurement instruments	Experimental Object
Process	Procedure	Process	Procedure
Metric	Metric	Metric	Metrics
DesignType	Design	Experiment design	Experimental design
Não Suportado	Threats to validity	Threats to validity	Validity threats
Não Suportado	Research question	Research question	N/A
Não Suportado	Dependent variable	Dependent variable	Response variable

As análises também mostraram três conceitos de domínio que não correspondem a qualquer elemento/construtor da linguagem: ameaças à validade, questão de pesquisa, e variável dependente. Estes conceitos de domínio foram incorporados em ExpDSLv2, com nós já discutimos na análise de completude.

6.4. Discussões e Lições Aprendidas

Nesta seção, nós apresentamos e discutimos lições aprendidas relacionadas aos resultados deste estudo empírico.

Critério de Adequação (Appropriateness). O critério de adequação *funcional* permite avaliar o grau ao qual a DSL suporta o desenvolvimento de soluções para alcançar necessidades do domínio da aplicação (KAHRAMAN e BILGEN, 2013). Ele é composto das sub-características completude e adequação. A completude foi avaliada e

discutida neste estudo. Por outro lado, a adequação reflete a análise de se a DSL é apropriada para modelagem de aplicações específicas do domínio. Neste estudo, nós tentamos atender este critério especificando 16 diferentes experimentos. Além disso, nós também modelamos mais que uma replicação de 3 destes experimentos. Desta variabilidade de experimentos, nós modelamos experimentos: (i) de diferentes domínios da engenharia de software; (ii) com diferentes projetos experimentais; e (iii) definidos e executados por diferentes grupos de pesquisa. Esta variedade de experimentos escolhidos apoiou a conclusão de adequação fornecida por ExpDSL no contexto do conjunto de experimentos modelados. Entretanto, nós reconhecemos que é importante modelar uma larga quantidade de experimentos adicionais a fim de aumentar a variabilidade de experimentos que podem ser expressos em ExpDSL.

Replicação de Experimentos. O fato de não haver uma maneira padrão de formalizar experimentos prejudica sua replicação. Em uma pesquisa experimental colaborativa, os resultados de estudos anteriores são necessários a fim de transferir conhecimento entre os pesquisadores envolvidos e usando uma terminologia comum. Neste contexto, prover uma linguagem específica de domínio (DSL) para formalizar experimentos em engenharia de software e a replicação dos mesmos é um dos principais objetivos do trabalho proposto. A linguagem funciona como um vocabulário controlado e pode facilitar a comunicação e troca de informação entre os pesquisadores, contribuindo para preencher o abismo relacionado a completa definição de um experimento. Neste estudo, nós modelamos réplicas da especificação de 3 experimentos. Nós observamos que quando modelando uma réplica, nós pudemos reusar a maioria da especificação porque as mudanças são localizadas e mais relacionadas as seguintes ações: (i) adicionar uma nova variável dependente; (ii) mudar o contexto, e/ou (iii) trocar alguns instrumentos de medida. Nós pretendemos avaliar de forma sistemática e explorar os benefícios de ExpDSL no que diz respeito a replicação em trabalhos futuros.

6.5. Ameaças à Validade

Nós analisamos as ameaças à validade de acordo com a classificação apresentada em (WOHLIN, RUNESON, *et al.*, 2012). A primeira ameaça a validade identificada neste estudo é uma validade interna chamada instrumentação. Este é o efeito causado pelos artefatos usados pelo estudo. Aqui ele está relacionado a escolha dos experimentos modelados assim como ao tamanho da amostra. Esta escolha define para

quais tipos de experimentos as conclusões são válidas, restringindo a extensão para as quais os resultados podem ser generalizados. Nós controlamos esta ameaça selecionando experimentos reais de diferentes origens (grupos de pesquisa), diferentes domínios da engenharia de software, e com diferentes projetos experimentais.

Outra ameaça à validade do estudo é chamada confiança (Reliability), que diz respeito com qual extensão os dados e as análises são dependentes de específicos pesquisadores envolvidos na modelagem do experimento. Nós controlamos esta ameaça modelando e validando especificações de experimentos usando ExpDSL com pesquisadores do Fraunhofer Institute for Experimental Software Engineering (IESE) e da Universidad Politecnica de Madrid (UPM), que não foram originalmente envolvidos no desenvolvimento de ExpDSL.

6.6. Conclusão

Este capítulo apresentou um estudo empírico de avaliação e melhoramento da linguagem ExpDSL que apoia a formalização de experimentos controlados. Um total de 14 experimentos, reportados pela comunidade de engenharia de software, foram especificados usando a DSL. Estas especificações foram avaliadas de acordo com alguns critérios a fim de responder as questões de pesquisa. O resultado do estudo demonstrou o valor da linguagem, mas, por outro lado, expôs uma série de oportunidades de melhoria. Estas melhorias foram discutidas e atendidas na nova versão da DSL. Como resultado final do estudo, nós concluímos que: (i) com relação ao critério de completude, a DSL revelou ser adequada a modelar aproximadamente 13 dos 15 (87%) diferentes conceitos do domínio dos experimentos da ES avaliados (QP1); (ii) o critério de ortogonalidade mostrou que ExpDSL fornece apoio a 14 dos 15 (93%) conceitos de domínio usando apenas um construtor da linguagem (QP2); (iii) o critério de unicidade resultou que 13 dos 15 (87%) elementos ExpDSL foram adequados para expressar exatamente um conceito distinto no domínio (QP3); e, finalmente, (iv) o critério de correspondência a importantes conceitos do domínio ajudou-nos a verificar que os construtores ExpDSL endereçam importantes conceitos relatados pela comunidade de engenharia de software experimental (QP4).

7. Avaliação Experimental da Abordagem Proposta

Com o objetivo de avaliar o uso na prática da linguagem ExpDSL, foram planejados e executados dois experimentos controlados. O objetivo do primeiro experimento controlado foi o de avaliar a compreensão da referida linguagem. Já o segundo experimento teve como objetivo avaliar a facilidade de uso da linguagem por parte dos experimentadores. Este capítulo está organizado em duas grandes seções, uma para cada um dos experimentos realizados. Essas seções estão organizadas segundo o modelo proposto por (WOHLIN, 2012) para reportar experimentos controlados.

7.1. Experimento 1: Compreensão da Linguagem

7.1.1 Definição do Experimento

O objetivo do primeiro experimento foi analisar ExpDSL com o propósito de avaliar a compreensibilidade de um plano experimental do ponto de vista do projetista de experimentos no contexto de estudantes de mestrado e doutorado. A compreensibilidade de uma DSL pode ser definida como o grau no qual os elementos da linguagem podem ser entendidos (os elementos da linguagem podem ser entendidos lendo suas descrições) (KAHRAMAN e BILGEN, 2013).

O experimento foi executado em laboratório com alunos de mestrado e doutorado que cursavam a disciplina de Engenharia de Software Experimental do Programa de Pós-graduação em Sistemas e Computação. Dois experimentos reais usando diferentes planos experimentais foram selecionados a partir dos experimentos especificados no Capítulo 6. Cada um deles foi utilizado no nosso experimento de duas formas: a partir de uma descrição na linguagem ExpDSL e outra usando sua descrição em linguagem natural – que foi representada pelo próprio artigo que descreve o experimento, e é comumente usado para descrevê-lo e reportá-lo.

As seguintes questões de pesquisa foram definidas visando alcançar o objetivo estabelecido.

QP1: A compreensão de um plano experimental especificado em ExpDSL é diferente daquela descrita em um artigo científico?

QP2: O tempo para correta compreensão do planejamento de um experimento depende do tipo de especificação utilizada para sua descrição?

QP3: Qual a percepção do leitor relacionada a compreensão de um plano experimental descrito em ExpDSL?

Visando responder as duas primeiras questões de pesquisa relativas à compreensão da ExpDSL, foram definidas duas métricas: (i) corretude da compreensão do experimento; e (ii) tempo para compreender um experimento. Já para responder a terceira questão de pesquisa foi idealizado um questionário para ser respondido pelos participantes do experimento, visando coletar as suas impressões no uso de cada uma das especificações. A partir das respostas desse questionário foi realizada uma avaliação qualitativa das impressões gerais e compreensibilidade das especificações investigadas.

7.1.2 Planejamento do Experimento

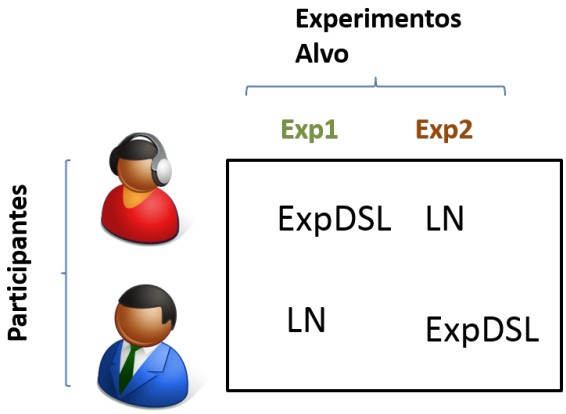


Figura 23: Organização do quadrado latino

O experimento foi organizado segundo o modelo de quadrado latino (RYAN, 2011), visando controlar a influência de dois fatores no resultado do experimento: (i) experimento alvo a ser compreendido e (ii) experiência do participante. A escolha pelo formato de quadrado latino se deu pelo fato dessa forma de organização possibilitar um bom controle da variância dos fatores envolvidos no experimento (LEROY, 2011). Cada quadrado latino foi organizado como uma matriz dois por dois (duas linhas e duas colunas), como ilustra a Figura 23, contendo: (i) dois experimentos alvo (Exp1 e Exp2); (ii) dois participantes; e (iii) cada participante devendo analisar cada um dos tipos de especificação (ExpDSL e LN: Linguagem Natural) investigado com um experimento alvo diferente, e de forma alternada. Foi planejada a utilização de 6 (seis) réplicas desta

configuração de quadrado latino, fazendo uso de dois experimentos alvo e um total de 12 (doze) participantes.

O restante desta seção apresenta o planejamento do experimento, desde a seleção das variáveis e definição das hipóteses a serem testadas, até a apresentação das ameaças à validade identificadas para o estudo, além das iniciativas visando minimizar tais ameaças.

7.1.2.1 Seleção das Variáveis

A variável sendo investigada é o tipo de especificação do plano experimental: ExpDSL e LN, que representam, respectivamente, a especificação do plano na linguagem específica de domínio e a especificação do plano na forma de um artigo científico. As variáveis dependentes são o percentual de respostas corretas (corretude) e o tempo gasto para a compreensão (eficiência). A corretude da compreensão é representada pelo número de respostas corretas em relação ao número total de respostas. O tempo para compreensão representa o tempo gasto com a leitura para a compreensão de um aspecto do experimento. Isto significa que devemos garantir que os participantes podem responder as questões de forma clara para que o pesquisador possa comparar as respostas com o conjunto de respostas esperadas. Além disso, é necessário garantir que o tempo de resposta de cada participante possa ser medido. Esse tempo deve ser medido para cada questão relativa a compreensão. Por fim, outras duas variáveis são controladas pelo quadrado latino: a experiência do participante e a complexidade do experimento alvo.

7.1.2.2 Definição das Hipóteses Estatísticas

Na definição do nosso objetivo, expressamos que gostaríamos de comparar a corretude da compreensão de aspectos de um plano experimental e o tempo para compreender esse plano quando usando diferentes tipos de especificações

Para formular a hipótese formal (estatística), tomamos N como sendo o número médio de respostas corretas, e NT o tempo médio levado para responder as questões corretas.

Considere:

• P(ExpDSL) e P(LN) como sendo a proporção de respostas corretas em ExpDSL e LN, respectivamente, e;

• μT(ExpDSL) e μT(NL) como sendo o tempo médio para compreender o experimento escritos em ExpDSL e LN, respectivamente.

Então, as hipóteses formuladas são as seguintes:

Compreensão Correta:

H0: P(ExpDSL) = P(LN)

H1: P(ExpDSL) ≠ P(LN)

Tempo para compreensão:

H0: μT(ExpDSL) = μT(LN)

H1: μT(ExpDSL) <> μT(LN)

Estas hipóteses indicam que desejamos observar, com significância estatística, que o uso de uma especificação de experimento em ExpDSL tem uma proporção de respostas corretas (compreensão) diferente da obtida com linguagem natural (artigo), e que a corretude da compreensão em ExpDSL consome uma quantidade de tempo diferente da consumida na especificação em linguagem natural. Ou seja, espera-se rejeitar as hipóteses nulas do experimento.

Essas duas hipóteses são testadas por meio da realização da análise estatísticas da diferença de proporções entre respostas corretas e da análise de variância dos tempos obtidos pelos participantes do experimento na realização das tarefas de compreensão de um experimento – ANOVA (HAIR, TATHAM, *et al.*, 2007). Para a interpretação do resultado da ANOVA será utilizado o nível de significância de 5%.

7.1.2.3 Os Experimentos Alvo

Foram utilizadas três especificações de experimentos alvo na execução do experimento. Todos os experimentos já haviam sido especificados na linguagem ExpDSL no estudo de análise da DSL (Capítulo 6). O primeiro experimento foi utilizado na atividade de preparação (aquecimento) para o experimento – para este

experimento foi solicitado ao participante a modelagem a partir da especificação em artigo (ACCIOLY, BORBA e BONIFÁCIO, 2012). Os outros dois experimentos alvo, denominados aqui de CFT (JUNG, JEDLITSCHKA, *et al.*, 2013) e LPS (CIRILO, NUNES, *et al.*, 2011), foram utilizados na execução do estudo experimental e selecionados visando permitir que os participantes pudessem compreender as questões propostas em um intervalo de tempo não superior a duas horas. Estes experimentos foram selecionados a partir do estudo anterior (Capítulo 6) por se tratarem de experimentos já validados pelo nosso grupo de pesquisa conforme critérios apresentados. A duração máxima de 2 (duas) horas teve por objetivo garantir que o experimento não se tornasse cansativo, o que prejudicaria os resultados do mesmo.

7.1.2.4 Seleção dos Participantes

Foram selecionados para participar do experimento 12 (doze) alunos da disciplina de Engenharia de Software Experimental do Programa de Pós-graduação em Sistemas e Computação da UFRN (PPgSC/UFRN). De forma geral, o público da disciplina possui conhecimento em engenharia de software experimental, porém com pouca experiência na realização de experimentos científicos. O conhecimento de tais alunos é, em geral, desenvolvido como parte da própria disciplina, onde eles aprendem os fundamentos teóricos da área e os aplicam através da definição e execução de experimentos de seus respectivos projetos de pesquisa. A distribuição dos tratamentos aos participantes em cada uma das réplicas do quadrado latino foi aleatorizada. É importante dizer que os participantes tiveram a liberdade de negar a participação, sem prejuízo individual na disciplina.

7.1.2.5 Instrumentação

O objetivo da instrumentação é prover meios para a realização do experimento e o seu monitoramento. Os instrumentos de um experimento podem ser classificados em (WOHLIN, 2012): objetos, orientações e instrumentos de medida.

A realização do experimento ocorreu em um dos laboratórios do Instituto Metrópole Digital (IMD) da Universidade Federal do Rio Grande do Norte (UFRN). Cada participante utilizou um computador (**objeto**), nos quais foram copiados os materiais necessários a realização do experimento. Como **orientação** para a realização do experimento cada participante utilizou: (i) uma descrição em questionário (roteiro)

descrevendo os aspectos experimentais a serem compreendidos; (ii) os slides utilizados nos treinamentos da linguagem; e (iii) as especificações dos experimentos a serem interpretados. Como **instrumento de medida** foi aplicado um questionário a cada participante com o objetivo de coletar a resposta de cada questão de interpretação e que mede automaticamente o tempo que o usuário leva em cada questão em milissegundos (Qualtrics[9]). Cada participante teve um período de 30 minutos para ler a respectiva especificação antes de iniciar a responder as atividades de compreensão do experimento.

As atividades dizem respeito a questões abertas relativas à compreensão de 9 aspectos do plano de um experimento controlado: objetivo, hipóteses de pesquisa (ou questões de pesquisa), fator investigado, variáveis dependentes, variáveis independentes/controladas, *design* estatístico do experimento (DoE), contexto, processo e momento da coleta de dados, e questões qualitativas levantadas. As questões estão descritas na Tabela 15.

Tabela 15 Questões relativas a compreensão dos experimentos alvo

Questão	Atividade de Interpretação
Q1	Qual(is) o(s) objetivo(s) do experimento?
Q2	Qual(is) a(s) questão(ões) de pesquisa (ou hipóstese(s)) do experimento?
Q3	Qual(is) a(s) variável(eis) dependente(s) e a qual(is) questão ou hipótese de pesquisa ela(s) está(ão) relacionada(s)?
Q4	Qual fator está sendo investigado e quais os seus níveis (tratamentos)?
Q5	Quais outras variáveis independentes estão sendo controladas no experimento e quais seus níveis (alternativas)?
Q6	Quais métricas são coletadas e em que momento do experimento?
Q7	Detalhe e explique o design estatístico do experimento (plano experimental).
Q8	Qual contexto (caracterização) do experimento?
Q9	Que informações qualitativas foram levantadas no experimento?

Ao final do experimento, foi solicitado o preenchimento de um questionário de opinião onde os participantes foram solicitados a responder questões qualitativas

[9] http://www.qualtrics.com/

relativas à linguagem ExpDSL (ver Tabela 16). Nosso questionário fez uso de três critérios de avaliação de DSL:

- **Compreensibilidade** – os elementos da linguagem são entendíveis pelo usuário (isto é, os elementos da linguagem podem ser entendidos após a leitura de suas descrições) (KAHRAMAN e BILGEN, 2013);
- **Facilidade de Uso** – impressão do usuário da linguagem do cenário trabalhado, que reflete a facilidade de uso da DSL (GABRIEL, 2010);
- **Expressividade** – quão compacta e restritiva é a DSL para expressar as intenções do usuário (GABRIEL, 2010). O grau no qual a solução de um problema pode ser mapeado para a linguagem naturalmente (KAHRAMAN e BILGEN, 2013).

Tabela 16: Questões relativas à percepção da compreensão da linguagem

Nº	Critérios	Questão	Tipo da Questão
1	Compreensibilidade	Como você classifica os textos (palavras-chave) que representam os conceitos do domínio na linguagem ExpDSL?*	Escala de Likert (5 pts)
2		Quais termos da linguagem ExpDSL você julga inadequados (caso exista algum)?	Aberta
3		É fácil compreender um experimento especificado na linguagem ExpDSL. *	Escala de Likert (5 pts)
4		Com que frequência você ficou confuso devido a semelhança entre os elementos (palavras-chave) da linguagem ExpDSL? *	Escala de Likert (5 pts)
5		Com que frequência você ficou confuso devido à falta de clareza do vocabulário da linguagem ExpDSL? *	Escala de Likert (5 pts)

6	Facilidade de Uso	Como você classifica o cenário fornecido (experimento) para compreensão da linguagem ExpDSL na avaliação realizada hoje (no que diz respeito à exigência mental)? *	Escala de Likert
7		O que você acredita ter sido mais difícil de compreender no experimento especificado em ExpDSL?	Aberta
8		Com que frequência você se encontrou "travado" ou confuso durante a interpretação do cenário do experimento escrito em ExpDLS? *	Escala de Likert
9		Foi mais fácil compreender os conceitos dos experimentos avaliados lendo o paper do que lendo a especificação em ExpDSL. *	Escala de Likert (5 pts)
10	Expressividade	Quais elementos do experimento você não pôde identificar no cenário expresso em ExpDSL?	Aberta
		Comentários Gerais	Aberta

* Questões obrigatórias

Nesse mesmo questionário, os participantes foram solicitados a registrar os comentários gerais sobre a linguagem que acabaram de analisar. As perguntas definidas para o questionário visaram caracterizar aspectos relacionados a percepção de compreensão da linguagem ExpDSL em termos de: (i) qual a percepção do usuário em relação à compreensão da linguagem (1ª a 5ª questão); (ii) se foi fácil para os participantes compreenderem o plano do experimento usando a linguagem (6ª a 9ª questão); e (iii) percepção da expressividade da linguagem por parte dos participantes (10ª questão). Além de uma questão geral para comentários do participante.

7.1.2.6 Ameaças à Validade

Esta seção apresenta e discute as ameaças à validade do estudo que foram identificadas, e como as mesmas foram tratadas. As ameaças à validade do estudo foram classificadas de acordo com as seguintes nomenclaturas: (i) validade interna, (ii) validade externa, (iii) validade de construção e (iv) validade da conclusão (WOHLIN, 2012).

Validade de Conclusão: esta validade diz respeito a questões que afetam a habilidade de realizar uma correta conclusão sobre as relações entre os tratamentos e os resultados. Foram tomadas ações para lidar com a seguinte ameaça: *Participantes de heterogeneidade randômica*. Esta ameaça aparece quando os participantes são selecionados aleatoriamente e seu background é muito heterogêneo. Nosso estudo buscou reduzir tal ameaça ao escolher o quadrado latino como design estatístico do experimento, com o objetivo de considerar o participante como uma variável controlada. Além disso, todos os participantes foram selecionados a partir da disciplina de engenharia de software experimental, onde todos, a princípio, estão com o mesmo nível de conhecimento, conhecimento suficiente para participar da pesquisa.

Validade Interna: ameaças a validade interna são influências que podem afetar os fatores com respeito a casualidade, sem o conhecimento do pesquisador. Nosso estudo buscou reduzir a ameaça de *Objeto de aprendizagem*, que indica que os participantes podem adquirir conhecimento quando eles analisam o primeiro objeto alvo (experimento a ser analisado) com um tratamento e aplicam este conhecimento para o outro experimento modelado com o outro tratamento. Nosso estudo utilizou dois diferentes experimentos alvo em nosso design (um por tratamento) para reduzir esta ameaça. *Maturação,* isto significa que os participantes reagem diferentemente à medida que o tempo passa. Isto pode acontecer no segundo tratamento, quando o participante pode ter aprendido do primeiro tratamento. De forma geral, essa ameaça foi atenuada através da aplicação do quadrado latino onde temos o intercâmbio entre os pares de participantes. *Seleção*, isto significa que o resultado pode ser afetado da forma como os participantes são selecionados. Uma participação forçada pode resultar em uma pobre motivação. Foram convidados estudantes de mestrado e doutorado, e os mesmos tiveram a opção de recusar a participação. Portanto, houve a participação apenas de voluntários no nosso experimento.

Validade de construção: esta validade diz respeito a generalização do resultado do experimento para o conceito ou teoria por trás do experimento. *Mono-tendência do método* (m*ono-method bias)*, significa que os experimentos com um único tipo de medida podem resultar em um viés de aferição. Em nosso experimento, medidas para corretude, esforço e satisfação não podem ser cruzadas umas contra as outras. Mesmo com essa ameaça, é impossível evitar o uso de uma única métrica para cada variável, e

nós tentamos minimizar este efeito. Neste sentido, nosso estudo buscou mecanizar a medição o tanto quanto possível por meio de tarefas (corretude) e tempo (esforço).

Validade Externa: esta validade diz respeito a generalização dos resultados para a prática industrial. *Interação da seleção e tratamento*, diz respeito a ter uma população de participantes que não é representativa da população que queremos generalizar. Em nosso experimento, a ameaça foi atenuada porque todos os participantes têm um background similar. Isto significa que, neste momento, podemos apenas dizer que os resultados de nosso estudo podem ser válidos para estudantes de pós-graduação com conhecimento em engenharia de software experimental e com pouca experiência na condução de experimentos controlados. *Dependência do Objeto*, significa que os resultados podem depender dos objetos usados no experimento e eles não podem ser generalizados. Essa ameaça foi minimizada em nosso estudo usando dois objetos para cada tratamento. Foram também selecionados objetos reais (experimentos alvo), fruto de publicação em eventos da área de engenharia de software experimental, e com diferentes designs experimentais. Apesar disso, não é possível generalizar os resultados para qualquer experimento ou domínio de experimentação. Podemos reduzir esta ameaça usando uma variedade de objetos com grande diferença entre eles. Entretanto, esta situação requer mais tempo e recursos para rodar o experimento.

7.1.3 Realização do Experimento

Esta seção apresenta as etapas que marcaram a execução do experimento. São descritas as etapas de (i) preparação para a execução do experimento (Seção 7.1.3.1), (ii) execução propriamente dita (Seção 7.1.3.2) e (iii) validação dos dados obtidos para o procedimento de análise dos mesmos (Seção 7.1.3.3).

7.1.3.1 Preparação

O planejamento, que antecedeu a preparação do experimento, definiu: (i) que o experimento seria configurado na forma de quadrado latino; (ii) que seriam utilizados 12 (doze) participantes, distribuídos em 6 (seis) réplicas de quadrado latino – com duas delas compostas por participantes mais experientes e quatro por participantes menos experientes; (iii) que o experimento aconteceria em 4 (quatro) dias, não ultrapassando duas horas em cada dia; e (iv) que as questões relativas à intepretação da especificação do experimento alvo deveriam ser concluídas em menos de duas horas.

Como tarefas de preparação para a realização do experimento, foram realizadas as seguintes tarefas: (i) definição do cronograma (dia-a-dia) para o experimento; (ii) definição dos experimentos alvo a serem utilizados na realização do estudo experimental; (iii) definição dos roteiros de questões para o aquecimento e para a execução do estudo experimental, com as duas especificações de experimento alvo; (iv) definição do questionário de feedback; e (v) aleatorização das réplicas de quadrado latino a serem utilizadas.

Foi definida a seguinte cronologia para o experimento: (i) 1º dia (1 hora e meia) – treinamento introdutório, revisando todo o conhecimento em experimentação e fundamentação da linguagem específica de domínio ExpDSL; (ii) 2º dia (1 hora e meia) – realização de aquecimento através da especificação de um plano de experimento, a partir da descrição em artigo, na linguagem ExpDSL; (iii) 3º dia (2 horas) – leitura e compreensão do experimento alvo (30 minutos) e realização do roteiro de atividades com o tipo de especificação e experimento alvo sorteado para o dia (1 hora e meia); e, por fim, (iv) o 4º dia (2 horas) – leitura e compreensão do experimento alvo (30 minutos) e realização do roteiro de atividades com a especificação e experimento alvo diferente da já utilizada (1 hora e meia). Além disso os participantes responderam um questionário de *feedback*.

Para a aleatorização das réplicas de quadrado latino foram realizados os seguintes sorteios: (i) posição dos participantes nas 6 (seis) réplicas correspondentes; (ii) o primeiro experimento alvo a ser compreendido na primeira coluna do quadrado; e (iii) qual o tipo de especificação a ser utilizado pelo primeiro participante de cada réplica e assim por diante. Foi garantido que as alternativas possíveis de configurações do quadrado tivessem o mesmo número de sorteios. A Tabela 17 apresenta o resultado da aleatorização dos experimentos alvo e dos tipos de especificações de plano experimental para as réplicas de quadrado latino contendo os participantes.

Tabela 17: Resultado da Aleatorização dos Quadrados Latinos

		CFT	LPS
Réplica 1	1	ExpDSL	LN
		LN	ExpDSL

	2		
Réplica 2	1	LN	ExpDSL
	2	ExpDSL	LN
Réplica 3	1	ExpDSL	LN
	2	LN	ExpDSL
Réplica 4	1	LN	ExpDSL
	2	ExpDSL	LN
Réplica 5	1	LN	ExpDSL
	2	ExpDSL	LN
Réplica 6	1	ExpDSL	LN
	2	LN	ExpDSL

O sorteio dos participantes para ocupar as 12 (doze) posições possíveis nas 6 (seis) réplicas de quadrado latino obedeceu os seguintes passos: (i) os nomes dos participantes foram listados em ordem alfabética em uma planilha; (ii) foi criada uma coluna denominada de “aleatório”, contendo o resultado da expressão randômica – “random()” (gerando um valor real aleatório entre zero e um); (iii) em seguida, os participantes foram ordenados pela coluna “aleatório”; e (iv) cada quadrado foi composto por dois participantes conforme a ordem obtida, com os primeiros dois compondo a primeira réplica e assim por diante.

O sorteio das posições dos experimentos alvo também seguiu um procedimento semelhante, onde: (i) os nomes dos experimentos alvo foram listados em uma planilha; (ii) foi criada uma coluna denominada de “aleatório”, contendo uma

fórmula – o resultado da função "*random*()"; (iii) em seguida, os experimentos alvo foram ordenados pela coluna "aleatório"; e, por fim, (iv) os experimentos alvo foram usadas nessa ordem na primeira réplica de quadrado latino e na ordem inversa na segunda réplica, e assim por diante. Aleatorização semelhante foi realizada para os tipos sob investigação.

Além disso, os roteiros de execução (questões sobre compreensão – Tabela 15) foi configurado na ferramenta Qualtrics de forma a facilitar a coleta das respostas e, também, a contabilização do tempo utilizado em cada questão. A utilização do aplicativo também permitiu a configuração do comportamento desejado de que uma vez finalizada uma questão do roteiro, o participante não poderia retornar mais a mesma questão. Houve também a configuração do questionário de feedback (Tabela 16) através de formulários no Google (Google Forms[10]**).**

7.1.3.2 Execução

Houve apenas uma exceção em relação às atividades planejadas para o experimento. A exceção foi a realização antes da execução da segunda rodada do experimento, onde um dos participantes informou que precisaria viajar, e acabou sendo marcada uma data posterior onde o participante realizou a segunda etapa do experimento. É importante destacar que a execução do experimento seguiu a ordem estabelecida: leitura e compreensão do experimento alvo (30 minutos) e realização do roteiro de atividades com o tipo de especificação e experimento alvo sorteado para o dia (1 hora e meia).

7.1.3.3 Validação dos Dados

Para a validação dos dados obtidos, foram realizadas duas verificações: (i) se o experimento ocorreu conforme foi planejado e instruído, e (ii) se a coleta de dados ocorreu conforme foi instruída. A execução do experimento foi acompanhada por um pesquisador responsável. Dado que tudo ocorreu conforme planejado e não ocorreu nenhuma anormalidade da execução do mesmo, os dados obtidos foram considerados válidos.

7.1.4 Apresentação dos Resultados

[10] http://www.google.com/forms/about/

Os resultados do experimento são apresentados e discutidos levando em conta a corretude e o tempo gasto em cada uma das questões de compreensão do plano do experimento; bem como as totalizações referentes ao tempo total gasto por cada participante e a corretude total por participante. Ao final são discutidas as percepções da dos participantes em relação à compreensão das especificações.

7.1.4.1 Resultados Obtidos pelos Participantes do Experimento 1

As Tabelas 18 e 19 apresentam os resultados obtidos pelos 12 (doze) participantes do experimento, nas duas rodadas de execução do experimento. Por questão de impessoalidade foram omitidos os nomes dos mesmos. A Tabela 18 registra os resultados da correção de cada questão do roteiro. A primeira coluna lista a réplica do quadrado (variando de 1 a 6), a segunda lista o participante (variando de 1 a 12), a terceira o experimento alvo, a quarta lista o tratamento. As demais listam o resultado para cada uma das questões (de Q1 a Q9). As questões foram avaliadas como: (i) CERTA – C, onde o participante respondeu o esperado em relação à questão de compreensão do plano; (ii) ERRADA – E, quando o participante respondeu de forma incorreta à questão; e (iii) PARCIAL – P, quando participante respondeu a questão de forma incompleta.

Tabela 18. Resultados obtidos pelos participantes do experimento

Replica	Part.	EXP	Trat.	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
R1	P1	CFT	ExpDSL	C	C	C	C	P	C	C	C	C
R5	P10	CFT	ExpDSL	C	C	C	C	P	P	E	E	C
R6	P11	CFT	ExpDSL	C	C	C	C	C	C	C	C	E
R6	P12	LPS	ExpDSL	C	C	C	C	E	E	C	C	E
R1	P2	LPS	ExpDSL	C	C	C	P	C	C	C	C	C
R2	P3	LPS	ExpDSL	C	C	C	C	C	C	C	C	C
R2	P4	CFT	ExpDSL	C	C	C	C	P	C	C	C	C
R3	P5	CFT	ExpDSL	C	C	C	C	C	C	C	C	C
R3	P6	LPS	ExpDSL	C	C	C	P	C	E	E	C	E
R4	P7	LPS	ExpDSL	C	C	C	P	E	P	C	E	P
R4	P8	CFT	ExpDSL	C	C	C	C	P	C	C	C	P
R5	P9	LPS	ExpDSL	C	C	C	P	E	P	C	E	E
R1	P1	LPS	LN	C	C	C	E	E	P	C	E	E
R5	P10	LPS	LN	C	C	E	P	E	P	E	E	P
R6	P11	LPS	LN	C	C	C	C	C	C	C	E	P

R6	P12	CFT	LN	C	C	C	P	E	P	E	P	E
R1	P2	CFT	LN	C	C	C	C	P	E	C	C	P
R2	P3	CFT	LN	C	C	C	C	E	C	E	P	C
R2	P4	LPS	LN	C	C	P	P	E	P	C	P	E
R3	P5	LPS	LN	C	C	C	P	E	C	C	C	C
R3	P6	CFT	LN	C	C	C	E	E	E	E	E	E
R4	P7	CFT	LN	C	C	P	C	P	E	E	P	E
R4	P8	LPS	LN	C	C	C	E	C	C	P	P	E
R5	P9	CFT	LN	C	C	C	P	E	C	C	C	C

A Tabela 19 segue as mesmas colunas da tabela anterior mas registra os tempos que os participantes levaram para a conclusão de cada uma das questões de interpretação solicitadas no roteiro do experimento A coluna tempo total representa o tempo gasto para conclusão do roteiro de questões.

Tabela 19. Resultados obtidos pelos participantes do experimento

Replica	**Part.**	**EXP**	**Trat.**	**Tempo Total**	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**	**Q9**
R1	P1	CFT	ExpDSL	37	3,6	3,5	1,7	0,9	3,4	10,3	10,3	2	1,5
R5	P10	CFT	ExpDSL	28	0,6	5,2	4,2	0,9	2,9	5,6	2	3,9	3,1
R6	P11	CFT	ExpDSL	24	0,8	4,1	1,8	1,9	2,9	5	3,9	1,3	2,9
R6	P12	LPS	ExpDSL	94	4,8	6,6	4,1	14,8	7,1	17,1	5,5	10,7	22
R1	P2	LPS	ExpDSL	24	1,1	0,5	0,5	0,2	0,9	13	0,3	0,3	3,8
R2	P3	LPS	ExpDSL	18	0,2	1,8	2,3	1,4	1	9,4	0,8	0,9	0,6
R2	P4	CFT	ExpDSL	41	2,6	2,8	12,2	1,8	9,8	3,6	2,1	3,7	1,9
R3	P5	CFT	ExpDSL	33	0,5	4,9	2,1	0	4,9	9,8	4	1	1,7
R3	P6	LPS	ExpDSL	18	0,5	0,5	0,8	0,5	5,5	3	1,9	0,4	4,6
R4	P7	LPS	ExpDSL	37	3,1	4,2	3,2	2,1	4,5	5,7	5,5	1,4	3,9
R4	P8	CFT	ExpDSL	26	0,9	1,2	4,5	2,4	9,3	4,3	1,1	1,2	1,3
R5	P9	LPS	ExpDSL	36	0,9	6,9	3,2	1,8	3,7	3,2	4,4	3,2	9
R1	P1	LPS	LN	60	11,1	5,7	2,6	10	10	3,3	1,2	5,4	3
R5	P10	LPS	LN	62	8,5	8,6	10	10,5	8,5	3	6,6	3,9	2,5
R6	P11	LPS	LN	59	5,7	4,3	12	2,7	5,9	10,5	5,9	7,5	0,9
R6	P12	CFT	LN	76	6,4	10,5	10,5	21,9	10,2	5	6,7	4,2	0,3
R1	P2	CFT	LN	73	3	2,5	13,7	30,7	3,3	5,1	9,7	3,3	1,7
R2	P3	CFT	LN	68	12,5	10	5,7	14	14,9	4,4	1,9	2,2	1,9
R2	P4	LPS	LN	74	9,4	9,9	15,2	9,1	17,9	2,6	7,1	0,6	2,8
R3	P5	LPS	LN	63	3,8	5,7	3,8	3,3	28,9	6,7	0,9	2,2	6

R3	P6	CFT	LN	55	4,6	6,6	21,5	4,6	3,3	2,1	0,4	0,4	3,1
R4	P7	CFT	LN	68	3,6	6,4	8,8	6,9	18,8	5,8	10	1,9	2,3
R4	P8	LPS	LN	55	6,2	3,3	7,3	9,6	11,2	7,2	5,9	1,5	2,7
R5	P9	CFT	LN	45	1,5	6,2	4,2	6,7	3,8	8,2	7	6,5	1,5

Conforme já foi citado, as tarefas solicitaram a identificação de vários aspectos de um plano experimental: objetivo, hipóteses ou questões de pesquisa, variáveis dependentes e sua relação com as hipóteses e/ou questões de pesquisa, fator sob investigação e seus níveis (tratamentos), variáveis controladas e seus níveis, métricas e momento da coleta das mesmas, design estatístico do experimento (plano experimental), contexto (caracterização) do experimento, que informações adicionais (*feedback*) foram levantadas pelo experimento. Os tempos estão apresentados, na referida tabela, apenas em minutos, embora tenham sido originalmente registrados na forma de milisegundos. Os tempos foram convertidos em minutos, para facilitar a manipulação e análise estatística dos mesmos.

As análises dos resultados foram realizadas tomando por base: (i) o percentual de corretude de cada questão; (ii) a corretude por participante; (iii) os tempos de cada questão individualmente; e (iv) o tempo total gasto por cada participante para responder todas as questões. Foram realizadas análises estatísticas variadas dos resultados obtidos, validando ou não, as hipóteses definidas.

7.1.4.2 Avaliação Relativa à Corretude da Compreensão

A primeira avaliação compreendeu a análise da corretude das questões. Para isso, cada questão foi corrigida e rotulada como: CERTA, ERRADA ou PARCIAL. O rótulo PARCIAL foi usado para denominar questões respondidas de forma incompleta.

A Figura 24 apresenta o gráfico de proporções de respostas de cada tipo para o conjunto de todas as questões do roteiro. ExpDSL apresentou 75% do total de respostas corretas contra, aproximadamente, 52% de LN. Em relação ao número total de erros, ExpDSL apresentou 13% contra 28% de LN. Já as questões incompletas, rotuladas como PARCIAL, o número total em ExpDSL representou 12% do total, e em LN esse número representou 20% do total.

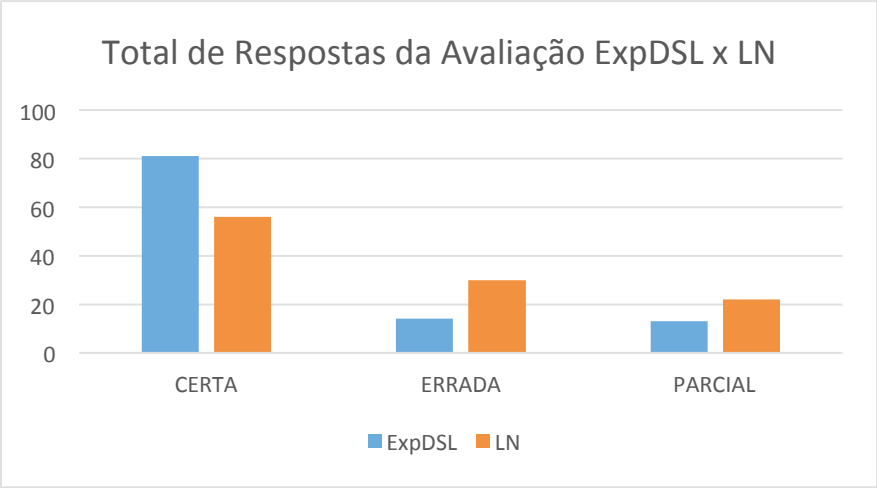


Figura 24: Gráfico do resultado total em relação a avaliação das questões do roteiro

Com base nessas proporções, realizamos uma análise para avaliar se há, estatisticamente, diferença entre as proporções de cada tipo de especificação. Usamos, para isso, o teste de análise de proporções Fisher´s Test. Ele analisa se há significância estatística entre a diferença de duas populações (FIELD, MILES e FIELD, 2012). A hipótese nula para o teste diz que há diferença estatística entre as duas proporções. A hipótese alternativa diz que não há diferença significativa entra as proporções.

Tabela 20: Teste Estatístico para duas proporções

Respostas	Teste e Intervalo de Confiança para duas proporções	Interpretação
CERTA	Estimate for difference: 0,231481 95% CI for difference: (0,106785; 0,356178) Test for difference = 0 (vs ≠ 0): Z = 3,64 P-Value = 0,000 Fisher's exact test: P-Value = 0,001	As proporções das respostas certas são diferentes, sendo MAIOR com o uso da ExpDSL.
ERRADA	Estimate for difference: -0,148148 95% CI for difference: (-0,253736; -0,0425599) Test for difference = 0 (vs ≠ 0): Z = -2,75 P-Value = 0,006 Fisher's exact test: P-Value = 0,011	As proporções das respostas erradas são diferentes, sendo MENOR com o uso da ExpDSL.
PARCIAL	Estimate for difference: -0,0833333 95% CI for difference: (-0,180984; 0,0143176) Test for difference = 0 (vs ≠ 0): Z = -1,67 P-Value = 0,094 Fisher's exact test: P-Value = 0,139	Não há evidência estatisticamente significativa de que as proporções das respostas parciais sejam diferentes.

O resultado da análise aponta que a proporção de respostas corretas em ExpDSL é estatisticamente diferente da proporção de respostas corretas em relação à compreensão de planos de experimentos especificados em LN. A partir do gráfico (Figura 24) observamos que o número de respostas corretas é maior em ExpDSL. O número de erros encontrados também é significativamente diferente nas análises de compreensão de planos descritos nos dois métodos avaliados (ExpDSL e LN), e pelo gráfico, podemos perceber que o número de erros em ExpDSL foi menor que o número de erros em LN. Já para o número de respostas classificadas como incompletas, não houve diferença entre as duas populações (ExpDSL e LN), ou seja, não houve diferença estatisticamente significativa em relação à proporção de respostas incompletas nas especificações em ExpDSL e em LN.

A análise anterior foi realizada para o resultado da avaliação da compreensão do plano do experimento como um todo. A seguir, realizamos uma avaliação por questão, para verificar a diferença de proporções nos aspectos avaliados do plano experimental. A Figura 25 apresenta o gráfico de proporção da avaliação para cada questão do roteiro do experimento: CERTA, ERRADA, e PARCIAL.

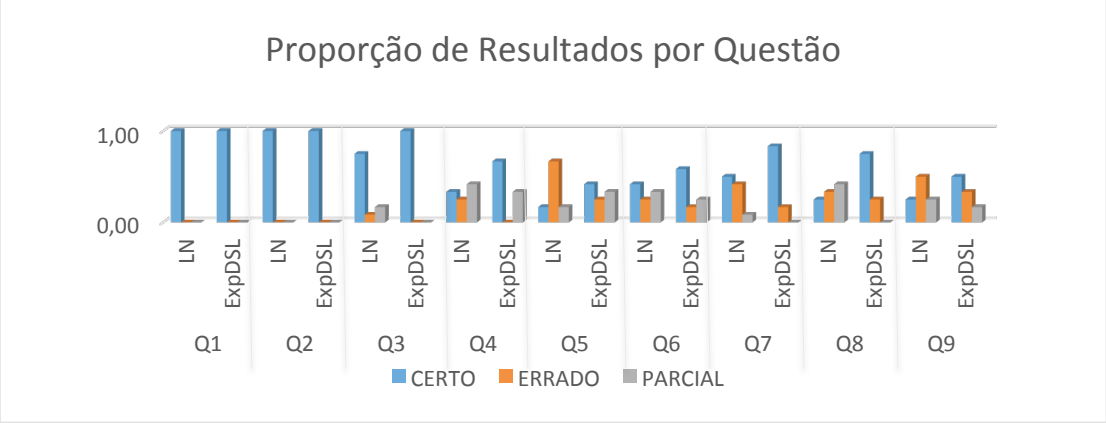


Figura 25: Gráfico de proporção da correção de cada questão do roteiro para o experimento 1

De uma forma geral, podemos observar que nas questões Q1 e Q2 tivemos 100% de acerto em ambos os tratamentos (ExpDSL e LN). Essa equidade na avaliação tanto dos objetivos como das questões de pesquisa (ou hipóteses de pesquisa) dos experimentos alvo foi equilibrada por esta ser uma informação presente e destacada nos planos de experimentos descritos em linguagem natural (na forma de artigos), assim como é na linguagem específica de domínio (ExpDSL). Para a terceira questão (Q3),

não houve respostas incompletas ou erradas na especificação em ExpDSL, e apenas uma resposta errada e duas incompletas para a especificação tradicional (LN). Ambas as respostas incompletas em LN estavam relacionadas à não associação, pelo participante, entre a variável dependente e o objetivo a que a mesma estava relacionada, talvez pelo fato de não haver essa associação explícita durante uma descrição em artigo, o que não é o caso de ExpDSL, onde existe uma amarração (referência cruzada) entre as variáveis dependentes e o(s) objetivo(s) ao qual a mesma está associada. Para a questão 4 (Q4) não houve respostas erradas em ExpDSL, 66,67% de respostas foram certas e 33,33% incompletas e em LN houve 33,3% de respostas corretas, 25% incorretas e 42% incompletas. A questão 5 apresentou o maior número de erros das questões, foram aproximadamente 67% de respostas erradas para a especificação em LN e esse número cai para 25% na compreensão da especificação em ExpDSL. A sexta questão apresenta um resultado balanceado entre as duas especificações, com 58,33% de acertos em ExpDSL e 41,67% em LN. A questão 7 apresenta 83,33 de acerto em ExpDSL contra 50% em LN, essa diferença aumenta na questão 8, onde 75% das respostas foram corretas para ExpDSL contra 25% em LN. Por fim, para a questão 9 tivemos 50% de acertos para ExpDSL contra 25% para LN, além disso o número de respostas erradas em LN foi 50% contra 33% em ExpDSL. De forma geral, em nenhuma das questões houve um número maior de acertos em LN que em ExpDSL.

Através do Fisher's Test, nosso estudo avaliou com nível de confiança de 95%, que a diferença estatística entre as proporções de respostas corretas é significante entre os dois grupos (ExpDSL e LN). A hipótese nula é que a proporção é a mesma. A hipótese alternativa é que a proporção de respostas é estatisticamente diferente nas duas especificações. Para a análise das questões, em particular, não utilizamos o teste da aproximação normal porque o mesmo é válido apenas quando, para ambas as amostras, o número de eventos é maior que quatro e a diferença entre o número de acontecimentos e eventos é maior que quatro.

Tabela 21: Resultado do Fisher's Test por questão

	Aspecto	CERTA	ERRADA	PARCIAL
Q1	Objetivo	P-Value = 1,000		
Q2	Hipótese (Questão) de	P-Value = 1,000		

	Pesquisa			
Q3	Variável Dependente	P-Value = 0,217	P-Value = 1,000	P-Value = 0,478
Q4	Fator sob investigação	P-Value = 0,220	P-Value = 0,217	P-Value = 1,000
Q5	Variáveis Independentes	P-Value = 0,371	P-Value = 0,100	P-Value = 0,640
Q6	Métricas e Momento da Coleta	P-Value = 0,684	P-Value = 1,000	P-Value = 1,000
Q7	Design do Experimento	P-Value = 0,193	P-Value = 0,371	P-Value = 1,000
Q8	Contexto do Experimento	P-Value = 0,039*	P-Value = 1,000	P-Value = 0,037
Q9	Informações adicionais (qualitativas)	P-Value = 0,400	P-Value = 0,680	P-Value = 1,000

A Tabela 21 mostra, baseado no teste de proporção, que apenas para a questão 8, que trata da compreensão do contexto do experimento, há uma diferença significativa entre as duas especificações. Neste caso, calculamos também o intervalo de confiança tanto para o caso das respostas corretas quanto das respostas parciais. No caso das respostas corretas, a diferença estimada foi de 0,5 e o intervalo de confiança foi de: (0,153524; 0,846476), significando que a média da população deve se encontrar entre 15 e 84%. Já para as respostas incompletas, a diferença estimada foi de -0,416667 e o intervalo de confiança foi de (-0,695606; -0,137727), significando que a diferença pode variar entre -69 e -13%.

Nós avaliamos que a diferença se deu pelo fato de o contexto (características) do experimento ser uma informação descrita de forma não explícita e não agrupada ou rotulada nas descrições de planos de experimentos em artigos. Sendo assim, muitos participantes compreenderam que o contexto se referia apenas as informações sobre os participantes e/ou informações sobre as variáveis sendo controladas. Já na especificação

em ExpDSL essa informação está agrupada em um bloco coeso, cuja palavra chave é "*Context*", facilitando, assim, a identificação do mesmo.

Como uma tentativa de avaliar se houve diferença entre o número de respostas de cada tipo (CERTA, ERRADA e PARCIAL) em LN devido ao fator "Experimento Alvo", nós avaliamos a questão 8 também para verificar se há diferenças significativas em relação ao artigo alvo sendo avaliado. A realização do teste estatístico de proporção (ver Tabela 22) mostrou que não há diferença entre o número de respostas de cada tipo para os dois artigos alvo.

Tabela 22: Avaliação de proporção do fator experimento alvo para a questão 8

LN		ExpDSL	
CERTA	P-Value* = 1,000	CERTA	P-Value* = 1,000
ERRADA	P-Value* = 0,545	ERRADA	P-Value* = 1,000
PARCIAL	P-Value* = 1,000	PARCIAL	P-Value* = 1,000

* P-Value calculado pelo Fisher's Test

Ao final desta análise, podemos concluir que para o resultado geral do nosso experimento o número de respostas corretas aumentou e a taxa de erros de compreensão diminuiu, quando usando ExpDSL, com significância estatística. Apesar do resultado da análise por questões não ter apresentado significância estatística para todas as questões, o que pode ter acontecido devido ao baixo número de dados, os resultados individuais para ExpDSL foram sempre maiores ou iguais do que para LN. De toda forma, podemos concluir, respondendo nossa questão de pesquisa QP1, que para o plano experimental como um todo, há evidências de que a compreensão de um plano especificado em ExpDSL é diferente que quando descrita em um artigo científico (linguagem natural).

7.1.4.3 Avaliação Relativa ao Tempo

Em relação a avaliação do tempo, a primeira avaliação que fizemos foi em relação ao tempo total de avaliação do plano experimental. A Figura 26 exibe o *boxplot* para a representação do tempo total gasto pelos participantes em ExpDSL e LN. É possível perceber que o tempo de compreensão de ExpDSL foi menor do que o tempo de compreensão da especificação em LN.

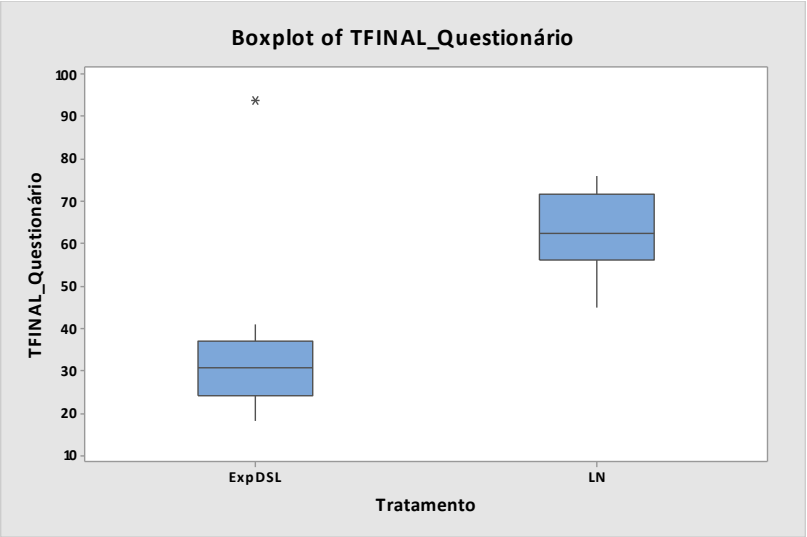


Figura 26: Boxplot do tempo total gasto por todos os participantes

O teste de normalidade (P=0,295) mostrou que não há evidências para rejeitar a hipótese nula de que os dados seguem a distribuição normal. Partindo deste ponto, usamos a análise de variância (ANOVA) para verificar se o tempo é estatisticamente diferente nas duas abordagens (Figura 27).

Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Replica	5	1182,8	236,57	0,59	0,709
Tratamento	1	4873,5	4873,50	27,28	0,000
EXP	1	28,2	28,17	0,16	0,700
Participante(Replica)	6	2399,0	399,83	2,24	0,125
Error	10	1786,3	178,63		
Total	23	10269,8			

Figura 27: Resultado da ANOVA para o tempo total de compreensão do plano experimental

A Tabela 25 exibe o resultado da ANOVA para o tempo total de compreensão. O valor-p calculado foi bem próximo a 0% (0,000), o qual nos permite rejeitar a hipótese nula. Portanto, os resultados apontam que o tempo para compreender o plano experimental é diferente para os dois tipos de especificações (ExpDSL e LN). Além disso, a equação de regressão (ver Apêndice II) mostrou que o fator tratamento (ExpDSL) tem um impacto estimado de 14,25 minutos para menos no tempo. A análise dos resíduos para este ANOVA encontra-se no Apêndice II.

Para complementar a avaliação, resolvemos analisar o tempo gasto para cada questão do experimento. A Figura 28 apresenta os gráficos *boxplot* dos tempos gastos

para compreender cada um dos aspectos do plano experimental de acordo com o roteiro do experimento (questões 1 a 9). De forma geral, o tempo médio de resposta para a compreensão da maioria dos aspectos nas especificações de experimento alvo em ExpDSL foram menores do que em LN. É possível perceber que a abordagem ExpDSL obteve resultados melhores do que a abordagem LN para a maioria das questões. Para podermos generalizar os resultados é necessária a realização de testes estatísticos adicionais.

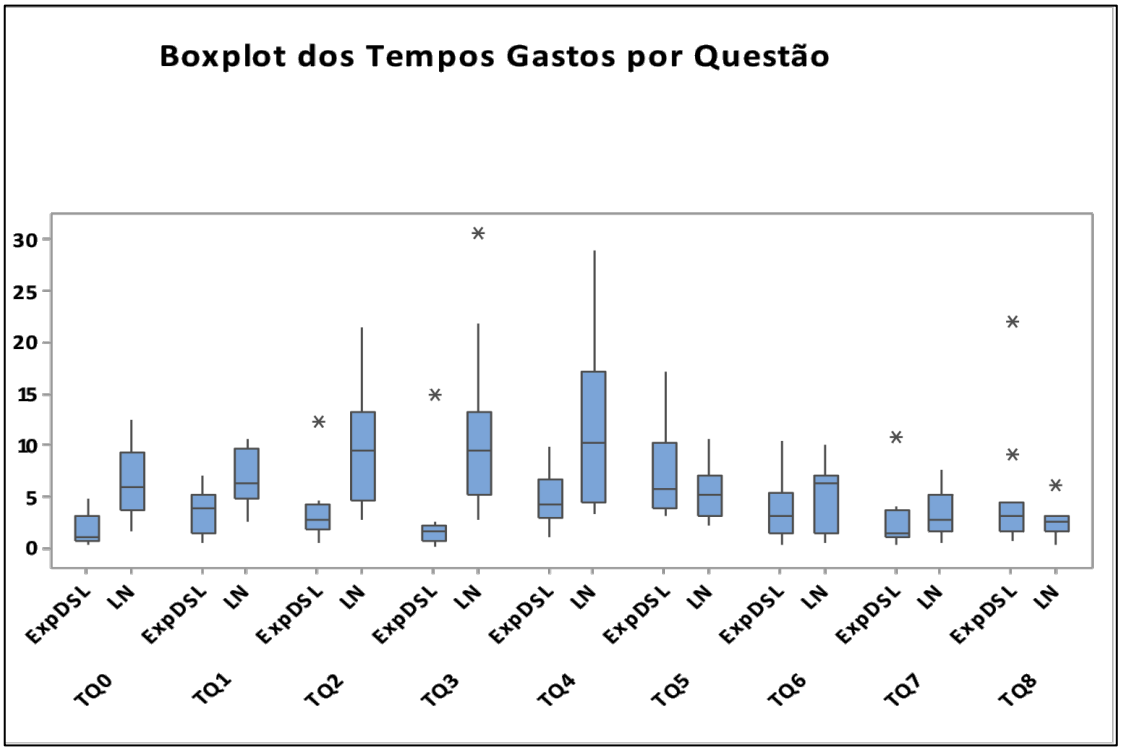


Figura 28: Gráficos Boxplot para os tempos gastos por questão

Tabela 23: Teste de Normalidade para o tempo de cada questão

Questão	Teste de Normalidade	Distribuição Normal	Teste Estatístico
Questão 1	p-value = 0,018	Não	Mann-Whitney
Questão 2	p-value = 0,776	Sim	ANOVA
Questão 3	p-value = 0,006	Não	Mann-Whitney
Questão 4	p-value <0,005	Não	Mann-Whitney
Questão 5	p-value <0,005	Não	Mann-Whitney
Questão 6	P-Value = 0,021	Não	Mann-Whitney
Questão 7	P-Value = 0,089	Sim	ANOVA
Questão 8	P-Value = 0,006	Não	Mann-Whitney

Questão 9	p-value <0,005	Não	Mann-Whitney

O primeiro teste realizado foi o teste de normalidade. A análise das questões 1, 3, 4, 5, 6, 8 e 9 foram realizadas através de testes não-paramétricos, já que os dados não seguem uma distribuição normal. Foi então utilizado o teste Mann-Whitney (two-sample Wilcoxon) que avalia a igualdade da mediana de duas populações e calcula o intervalo de confiança. A hipótese nula: H0: $\mu 1 = \mu 2$ e a hipótese alternativa é H1: $\mu 1 \neq \mu 2$, onde μ é a mediana da população.

Tabela 24: Teste Mann-Whitney para as questões: Q1, Q3, Q4, Q5, Q6, Q8 e Q9

Questão	Resultado do Teste	Interpretação
Q1	N Median TQ0_1_ExpDSL 12 0,900 TQ0_1_LN 12 5,950 Point estimate for η1 - η2 is -4,250 95,4 Percent CI for η1 - η2 is (-7,500;-2,500) W = 88,5 Test of η1 = η2 vs η1 ≠ η2 is significant at 0,0004 The test is significant at 0,0004 (adjusted for ties)	P-Value de 0,0004 . Há evidências de que há diferença entre a mediana, sendo MENOR a do ExpDSL.
Q3	N Median TQ3_ExpDSL 12 2,750 TQ3_LN 12 9,400 Point estimate for η1 - η2 is -5,950 95,4 Percent CI for η1 - η2 is (-9,700;-2,100) W = 96,5 Test of η1 = η2 vs η1 ≠ η2 is significant at 0,0022 The test is significant at 0,0022 (adjusted for ties)	P-Value de 0,0022. Há evidências de que há diferença entre a mediana, sendo MENOR a do ExpDSL.
Q4	N Median TQ4_ExpDSL 12 1,60 TQ4_LN 12 9,35 Point estimate for η1 - η2 is -7,40 95,4 Percent CI for η1 - η2 is (-10,00;-3,30) W = 88,0 Test of η1 = η2 vs η1 ≠ η2 is significant at 0,0004 The test is significant at 0,0004 (adjusted for ties)	P-Value de 0,0004. . Há evidências de que há diferença entre a mediana, sendo MENOR a do ExpDSL.
Q5	N Median TQ5_ExpDSL 12 4,100 TQ5_LN 12 10,100 Point estimate for η1 - η2 is -5,600	P-Value de 0,0102. Não há evidências de que há diferença entre a mediana das populações.

	95,4 Percent CI for η1 - η2 is (-10,802;-0,899) W = 105,0 Test of η1 = η2 vs η1 ≠ η2 is significant at 0,0102 The test is significant at 0,0102 (adjusted for ties)	
Q6	N Median TQ6_ExpDSL 12 5,650 TQ6_LN 12 5,050 Point estimate for η1 - η2 is 1,300 95,4 Percent CI for η1 - η2 is (-1,200;5,201) W = 168,0 Test of η1 = η2 vs η1 ≠ η2 is significant at 0,3123 The test is significant at 0,3121 (adjusted for ties)	P-Value de 0, 3123. Não há evidências para definir que há diferença entre a mediana das populações.
Q8	N Median TQ8_ExpDSL 12 1,350 TQ8_LN 12 2,750 Point estimate for η1 - η2 is -1,000 95,4 Percent CI for η1 - η2 is (-2,898;0,601) W = 125,0 Test of η1 = η2 vs η1 ≠ η2 is significant at 0,1572 The test is significant at 0,1569 (adjusted for ties)	P-Value de 0,1572. Não há evidências para definir que há diferença entre a mediana das populações.
Q9	N Median TQ9_ExpDSL 12 3,000 TQ9_LN 12 2,400 Point estimate for η1 - η2 is 0,800 95,4 Percent CI for η1 - η2 is (-0,800;2,302) W = 168,0 Test of η1 = η2 vs η1 ≠ η2 is significant at 0,3123 The test is significant at 0,3119 (adjusted for ties)	P-Value de 0,3119. Não há evidências para definir que há diferença entre a mediana das populações.

Para analisar o tempo das questões 2 e 7, além da normalidade da distribuição dos erros amostrais, a aplicação dos testes paramétricos exige que as variâncias sejam homogêneas e que os efeitos dos fatores de variação sejam aditivos; ou, em outras palavras, que sejam passíveis de serem somados uns aos outros. Os efeitos de dois ou mais fatores de variação são ditos não-aditivos quando, na associação de um ou mais desses fatores, em vez de se somarem, esses efeitos se multiplicam, de tal forma que o efeito resultante pode ser ampliado (quando o fator multiplicativo é maior que 1), ou reduzido (quando esse fator é menor que 1). É o que comumente ocorre nas chamadas interações entre dois ou mais fatores de variação (RYAN, 2011). Após a verificação da

não interação entre os fatores e da homogeneidade das variâncias, realizamos o teste da ANOVA para as questões. A Tabela 25 exibe o resultado da análise de variância. Para a questão 2, que avalia a compreensão das questões de pesquisa o P-Value foi de 0,003, significando que podemos rejeitar a hipótese nula de que não há diferença entre os tempos para os dois tratamentos (especificação em ExpDSL e LN). Para a questão 7, que avalia a compreensão do design estatístico do experimento, o P-Value foi de 0,228, significando que não há evidência suficiente para rejeitar a hipótese nula, ou seja, o valor do tempo é igual para qualquer um dos tratamentos (ExpDSL e LN). Para garantir a adequação da ANOVA, analisamos os resíduos tanto para a questão 2, quanto para a questão 7 (ver Apêndice II).

Tabela 25: Resultado da ANOVA para as questões 2 e 7

Q2

Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Replica	5	46,912	9,382	1,37	0,354
Tratamento	1	58,594	58,594	14,50	0,003
EXP	1	1,450	1,450	0,36	0,562
Participante(Replica)	6	41,223	6,870	1,70	0,218
Error	10	40,401	4,040		
Total	23	188,580			

Q7

Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Replica	5	51,237	10,247	1,79	0,248
Tratamento	1	19,260	19,260	1,65	0,228
EXP	1	7,150	7,150	0,61	0,452
Participante(Replica)	6	34,278	5,713	0,49	0,802
Error	10	116,534	11,653		
Total	23	228,460			

Ao analisarmos o resultado da avaliação do tempo, observamos que as questões de Q1 a Q5 apresentaram evidência suficiente para concluir que os tempos gastos para a compreensão em ExpDSL foram estatisticamente diferentes (menores). Nas demais questões, os tempos foram equivalentes, embora a taxa de erro tenha sido menor para ExpDSL, embora só generalizável para a questão 8.

7.1.5 Influência dos Participantes e dos Experimentos Alvo

Para avaliar a influência das variáveis de bloco no resultado do experimento, nós observamos que na análise da ANOVA tanto os experimentos alvo quanto os participantes aparecem com valores pouco significantes para o campo quadrado médio ajustado (Adj MS) em relação ao valor observado para o erro

experimental (Error), o que representa um indício de que esses fatores não tiveram influência e que, talvez, não precisassem terem sido controlados.

7.1.6 Análises Qualitativas – Experimento #1

Após a realização do experimento com cada uma das especificações, os participantes responderam a um questionário de avaliação qualitativa sobre a DSL. O questionário possuía 12 (doze) questões ao todo (ver Tabela 16). Em oito das questões, o participante respondia de acordo com a escala Likert (LIKERT, 1932) com 5 níveis (representada por um número de 1 a 5). Esta seção apresenta uma consolidação das respostas dos participantes, bem como uma análise e discussão sobre as mesmas.

A primeira questão da avaliação qualitativa perguntou "Como você classifica os textos (palavras-chave) que representam os conceitos do domínio na linguagem ExpDSL?". A Figura 29 apresenta uma consolidação das respostas dos participantes para essa questão. Nessa consolidação podemos perceber que a maioria dos resultados foi "Muito bom" e o restante "Bom", significando que, de uma forma geral, os participantes estavam satisfeitos com as palavras-chaves da linguagem para representar os conceitos de experimentação.

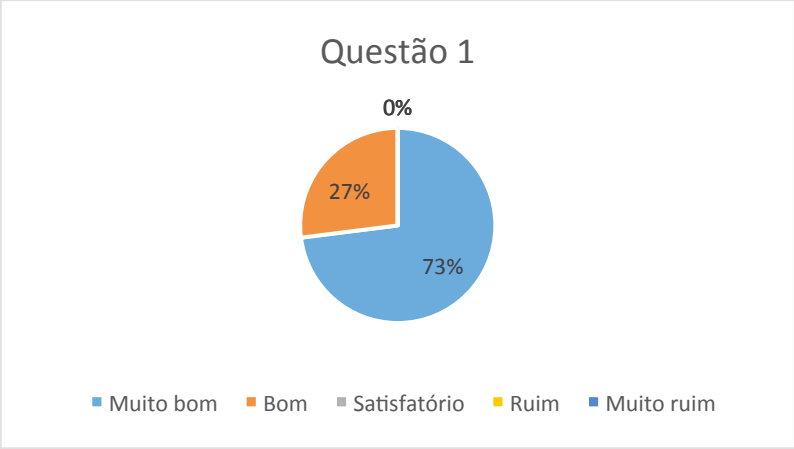


Figura 29: Análise da questão qualitativa 1.

Como um complemento da primeira questão, a segunda questão da avaliação qualitativa perguntou "Quais termos da linguagem ExpDSL você julga inadequados (caso exista algum)?". Essa questão foi opcional e apenas 5 participantes responderam. Destes 5, 3 apenas reafirmaram que não encontraram termos inadequados. Um dos participantes sugeriu que a palavra chave DoE, que designa o design estatístico do experimento, não fosse abreviado, podendo ser representado por

DesignOfExperiment. Um outro participante julgou inadequado definir os fatores (sobe investigação e bloqueantes) em um único bloco. Ele sugeriu que houvesse uma forma de determinar qual o fator sob investigação chamando-o de tratamento já no momento da definição.

A terceira questão do questionário de avaliação qualitativa afirmou "É fácil compreender um experimento especificado na linguagem ExpDSL?". A Figura 30 apresenta uma consolidação das respostas dos participantes para essa questão. O resultado mostra que todos os participantes acharam fácil a compreensão de um experimento especificado em ExpDSL.

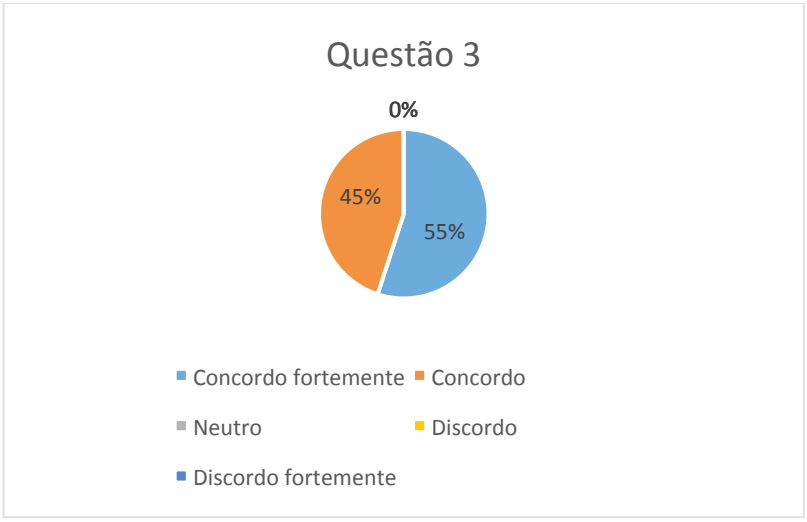


Figura 30: Anáise da questão qualitativa 3

A quarta questão da avaliação qualitativa perguntou "Com que frequência você ficou confuso devido a semelhança entre os elementos (palavras-chave) da linguagem ExpDSL?". A Figura 31 apresenta uma consolidação das respostas dos participantes para essa questão. Nenhum participante relatou se sentir confuso com frequência ou alta frequência. Na verdade, a grande maioria (64%) relatou sentir-se confuso raramente. Diante deste resultado e relembrando o resultado da questão 3, concluímos que, embora os participantes possam ter achado a linguagem (ExpDSL) fácil, em alguns momentos eles se sentiram confusos.

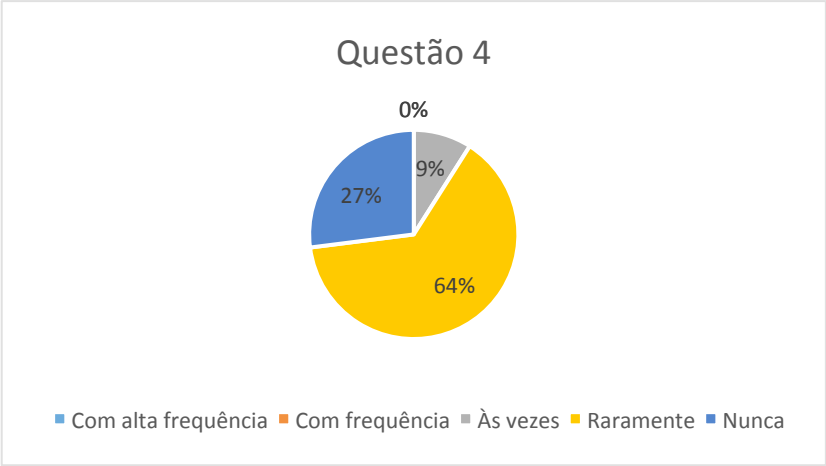


Figura 31: Análise qualitativa da questão 4

A quinta questão do questionário de avaliação qualitativa perguntou "Com que frequência você ficou confuso devido à falta de clareza no vocabulário da linguagem ExpDSL?". A Figura 32 apresenta uma consolidação das respostas para tal questão. Analisando as respostas dos participantes para essa questão, temos que a grande maioria (82%), nunca (18%) ou raramente (64%) ficaram confusos com o vocabulário da linguagem. A questão aberta "Questão 7" foi inserida para tentar capturar que elementos da linguagem foram identificados como confusos, onde o participante teve dificuldade de compreensão (ver Tabela 26).

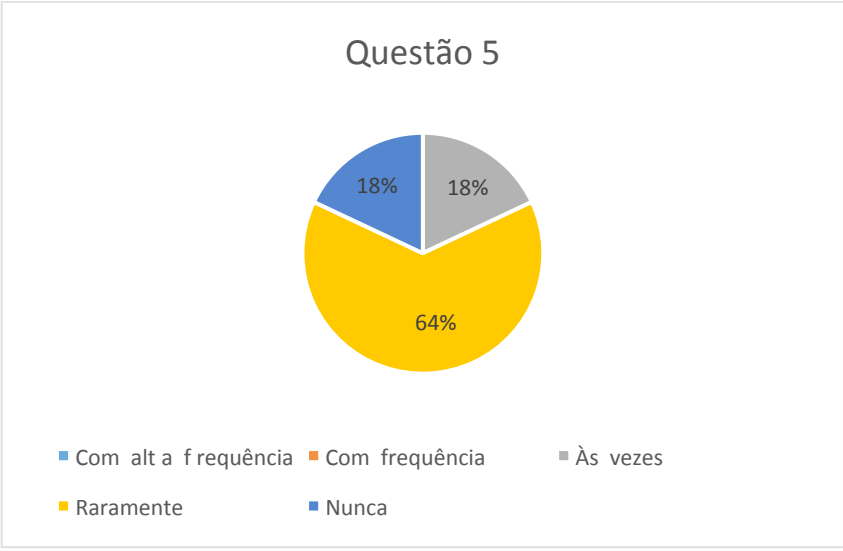


Figura 32: Resultado da questão 5.

A questão 6 avalia a percepção da exigência mental relativa ao experimento alvo: "Como você classifica o cenário fornecido (experimento) para compreensão da linguagem ExpDSL na avaliação realizada hoje (no que diz respeito à exigência

mental)?". A Figura 33 apresenta uma consolidação das respostas para tal questão. Analisando as respostas dos participantes para essa questão, temos que a grande maioria, 73%, achou o cenário (experimento analisado) simples, o que nos leva a crer que não houve dificuldades quanto ao entendimento do cenário em si. De toda forma, para 27% o cenário apresentado foi regular.

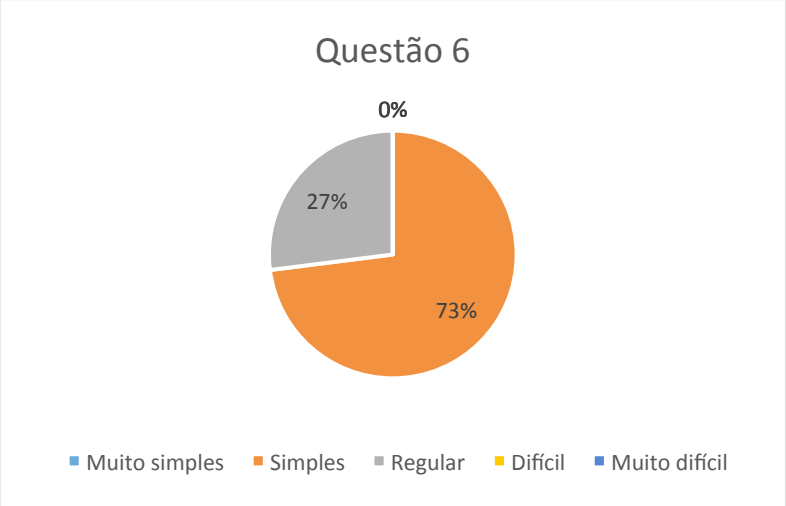


Figura 33: Análise qualitativa da questão 6

A questão 7 visa complementar a avaliação da duas questões anteriores e pergunta "O que você acredita ter sido mais difícil de compreender no experimento especificado em ExpDSL?" Essa questão foi aberta e opcional, sendo respondida por 6 participantes. A Tabela 26 sumariza os pontos difíceis identificados pelos participantes relativos ao cenário do experimento. Podemos verificar que não houve convergência em relação a dúvida dos participantes.

Tabela 26: Pontos difíceis identificados pelos participantes

Pontos mais difíceis de compreender no experimento especificado em ExpDSL
O design do experimento (DoE).
As variáveis independentes e as métricas.
Identificar as variáveis dependentes.
Métricas qualitativas e quantitativas
O próprio contexto do experimento.
A parte da descrição dos processos.

A questão 8 visa identificar se o cenário (experimento alvo) deixou o participante confuso. A pergunta é: “Com que frequência você se encontrou "travado" ou confuso durante a interpretação do cenário do experimento escrito em ExpDLS?”. A Figura 34 consolida o resultado para a questão. Podemos concluir que, de alguma forma, os participantes ficaram confuso apenas “as vezes” (55%) ou “raramente” (45%) em relação ao cenário, corroborando com a questão anterior, onde a maioria achou o cenário (experimento alvo) especificado em ExpDSL simples.

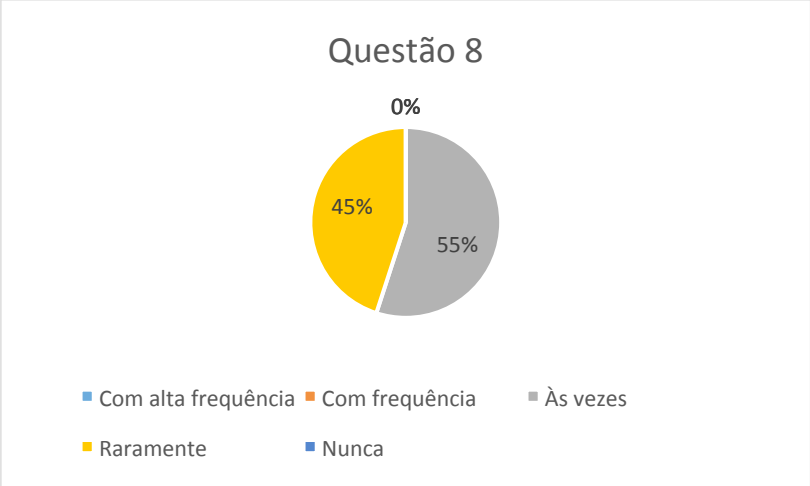


Figura 34: Resultado da questão 8

A questão 9 busca comparar a compreensão da especificação em ExpDSL com a especificação do experimento através do *paper*. A questão é uma afirmação “Foi mais fácil compreender os conceitos dos experimentos avaliados lendo o paper do que lendo a especificação em ExpDSL”. A maioria dos participantes (Figura 35) discordaram da afirmação (36% Discordaram Fortemente e 27% Discordaram), porém, 27% concordaram fortemente. A próxima questão (Questão 10) visa capturar qual a informação complementar para identificarmos as razões para discordância.

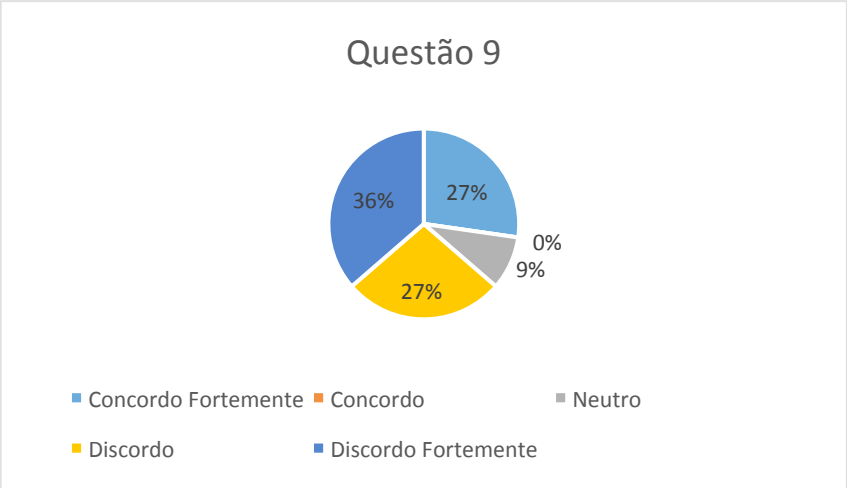


Figura 35: Resultado da Questão 9

A questão 10 busca identificar o que o usuário não foi capaz de identificar em ExpDSL. A pergunta foi "Quais elementos do experimento você não pôde identificar no cenário expresso em ExpDSL?".

Tabela 27: Elementos não identificados pelo participante no cenário em ExpDSL

Elementos não identificados pelo participante no cenário em ExpDSL
Nenhum
Métricas do Experimento.
Variáveis independentes e análise qualitativa
As variáveis independentes não estão muito claras.
Identificar as variáveis dependentes.
Os dados qualitativos
Alguns detalhes sobre o DoE, já que o design do experimento não é um dos designs básicos definidos para a linguagem.
Métricas

Na análise desta questão, pudemos identificar que dos 3 participantes (27%) que concordaram com a afirmação da questão anterior (Questão 9), um destacou a dificuldade em determinar as “Métricas do Experimento” e outro em “Identificar as variáveis dependentes”. O terceiro participante não respondeu à questão aberta. Nós analisamos que a dificuldade em identificar as métricas pode vir do fato de não haver um elemento denominado “Métrica” na linguagem, muito embora as métricas estejam especificadas dentro do elemento Processo (junto das tarefas onde as mesmas devem ser

coletadas), como explicado aos participantes durante o treinamento. Para a segunda observação, nós não conseguimos julgar qual pode ter sido a dificuldade do participante, pois há um bloco de elementos identificados como “Dependent Variable” na especificação em ExpDSL. É ainda interessante observar que o participante (P3) que citou a “Métrica do Experimento” respondeu de forma correta a questão do roteiro sobre as métricas coletadas. O mesmo aconteceu com o participante (P4), que identificou as variáveis dependentes como não sendo claras, mas respondeu corretamente a questão do roteiro sobre as variáveis dependentes.

Para finalizar o questionário, foi deixada uma questão aberta e opcional denominada “Comentários Gerais” e foi instruído aos participantes para deixarem qualquer comentário que julgassem relevante em relação à linguagem e aos cenários avaliados. Apenas 4 participantes responderam à questão. Houve um comentário de percepção do usuário, dois relatos de problemas e uma sugestão em relação à linguagem.

Tabela 28: Comentários gerais dos participantes ao final do experimento 1.

Comentários Gerais
Um ótimo recurso para construção e planejamento de experimentos.
Muito bom. Só um detalhe: o Eclipse não estava reconhecendo o "Scale Absolute" em CorrectAnswers "Measure of correct answers" Scale Absolute relates to RH H1 Time "Time used on the correct answers" Scale Absolute relates to RH H2 Quando alterei pra Ratio funcionou.
Na questão de procedimentos de coleta ficou faltando falar sobre Process OLIS e Process Eshop.
A ExpDSL é útil para planejar e entender um experimento controlado, principalmente relacionado ao design do estudo, objetivos, hipóteses e identificação das variáveis dependentes. É fácil a compreensão da maioria dos conceitos relacionados ao experimento. Como sugestão poderia ter uma opção para que o estudante/profissional que fosse realizar o experimento, utilizado esta especificação pudesse também definir as métricas, assim como ele definir, hipóteses e objetivos, por exemplo.

7.2. Experimento 2: Facilidade de Uso da Linguagem

7.2.1 Definição do Experimento #2

Este experimento busca analisar a linguagem específica de domínio ExpDSL com o propósito de avaliá-la com respeito a sua facilidade de uso do ponto de vista dos projetistas de experimentos no contexto de estudantes de mestrado e doutorado.

Este segundo experimento também foi executado em laboratório com alunos de mestrado e doutorado que faziam parte da disciplina de Engenharia de Software Experimental. Dois experimentos reais com diferentes planos experimentais foram selecionados, também a partir dos experimentos utilizados no capítulo anterior (mas diferentes dos utilizados no Experimento #1), para serem especificados em ExpDSL pelos participantes. Ambos os experimentos foram disponibilizados em linguagem natural (através do próprio artigo que descrevia cada experimento). Cada participante foi encarregado de especificar apenas um dos experimentos em ExpDSL, de forma aleatória (realizada por sorteio).

As questões de pesquisas que guiaram este experimento a fim de alcançarmos o objetivo geral foram:

QP1: É fácil especificar experimentos controlados em ExpDSL?

QP2: Qual a opinião dos experimentadores sobre o uso de ExpDSL para especificar seus experimentos?

7.2.2 Planejamento do Experimento #2

Neste segundo experimento, cada participante realizou uma especificação de experimento utilizando a linguagem ExpDSL (o tratamento). Foram utilizados dois experimentos alvo como objetos de estudo (FUCCI e TURHAN, 2014) e (RIBEIRO, BORBA e KÄSTNER, 2014). Esses experimentos foram distribuídos de forma aleatória entre o total de 10 (dez) participantes. Além disso, foi realizada, ao final da especificação, uma pesquisa de opinião com os participantes à respeito da facilidade de uso da linguagem. O restante desta seção apresenta o planejamento do experimento, desde a definição das hipóteses a serem testadas, até a apresentação das ameaças à validade identificadas para o estudo, além das iniciativas visando a minimização de tais ameaças.

7.2.2.1 Seleção das Variáveis

A variável sendo investigada é a forma de especificar experimentos controlados (ExpDSL). As variáveis dependentes são a taxa de respostas corretas (taxa de acerto), o tempo gasto para a especificar um plano experimental (eficiência) e alguns critérios sobre a percepção do usuário quanto a facilidade de uso de ExpDSL. A taxa de acerto se refere ao número de aspectos corretamente especificados em relação ao número total de aspectos, enquanto a satisfação do usuário se refere a percepção do usuário a respeito da especificação de experimentos em ExpDSL. Isto significa que nós devemos garantir que os participantes podem especificar um experimento a partir de uma descrição textual clara para que o pesquisador possa comparar os aspectos que foram especificados em ExpDSL com a descrição textual para avaliar a corretude. Além disso, temos que garantir que o tempo de resposta de cada participante possa ser medido. Um questionário final será aplicado visando avaliar diferentes critérios sobre a percepção do usuário após usar ExpDSL.

7.2.2.2 Definição das Hipóteses #2

Na definição do nosso objetivo, expressamos que gostaríamos de avaliar a facilidade de uso da linguagem ExpDSL ao formalizar a especificação de um experimento, bem como a satisfação dos usuários em usar ExpDSL para especificar experimentos. As hipóteses de pesquisa definidas foram:

HP1: A ExpDSL é fácil de usar

HP2: A ExpDSL é considerada pelos experimentadores como boa para especificar experimentos.

Formulação da hipótese estatística: A correta especificação de um experimento será avaliada a fim de determinar a facilidade de uso e um questionário de satisfação será usado para avaliar a satisfação do usuário em usá-la.

Para formular a hipótese estatística, consideramos N como sendo o número de aspectos corretamente definidos no uso da linguagem.

Se tomarmos:

μNExpDSL como sendo a taxa média de aspectos especificados corretamente usando ExpDSL para especificar o experimento
Então, as hipóteses que formulamos são:

Taxa de Acertos:

H0: μNExpDSL <= 80%

H1: μNExpDSL > 80%

A hipótese significa que gostaríamos de mostrar, com significância estatística, que as especificações em ExpDSL resulta num número correto maior que 80%. Nosso experimento busca rejeitar a hipótese nula. Nós calcularemos também o intervalo de confiança para a taxa média estimada para cenários semelhantes ao do experimento.

Além disso, calcularemos também o intervalo de confiança para tempo médio estimado para especificar um experimento em ExpDSL.

7.2.2.3 Seleção dos Participantes

Foram selecionados para participar do experimento 10 (dez) alunos da disciplina de Engenharia de Software Experimental do Programa de Pós-graduação em Sistemas e Computação da UFRN (PPgSC/UFRN), mesmos participantes do Experimento 1, com exceção de 2 (dois) que optaram por não participar, já que a participação era voluntária. De forma geral, o público da disciplina possui conhecimento em engenharia de software experimental, porém com pouca experiência na realização de experimentos científicos. O conhecimento que possui foi, em parte, fruto da própria disciplina, onde aprenderam a teoria e tiveram que definir e executar os experimentos de seus respectivos projetos de pesquisa.

7.2.2.4 Instrumentação

O objetivo da instrumentação é prover meios para a realização do experimento e o seu monitoramento. Os instrumentos de um experimento podem ser classificados em (WOHLIN, 2012): objetos, orientações e instrumentos de medida.

A realização do experimento ocorreu em um dos laboratórios do Instituto Metrópole Digital da Universidade Federal do Rio Grand do Norte (UFRN). Cada participante utilizou um computador com o editor ExpDSL instalado (**objeto**), nos quais foram copiados os materiais necessários a realização do experimento. Como **orientação**

para a realização do experimento cada participante utilizou: (i) uma descrição em artigo de um experimento a ser especificado; e (ii) os slides utilizados nos treinamentos da linguagem. Como **instrumento de medida** foi realizado, por cada participante, a especificação, em ExpDSL, de um experimento científico descrito em formato de artigo científico. O tempo de duração da especificação, para cada participante, foi registrado pelos dois pesquisadores que aplicaram o experimento, com um limite máximo de 2:00hs para a especificação, porém este tempo limite não foi necessário.

O objetivo do experimento também é avaliar o grau de satisfação do usuário da linguagem. Segundo (GABRIEL, 2010), a satisfação do usuário quando desempenhando tarefas específicas, é influenciada pela intuitividade da linguagem – facilidade de uso, facilidade de aprendizado, método de interação, e proximidade entre a representação mental do usuário da DSL e a capacidade da DSL de satisfazer suas intenções. Para avaliar o grau de satisfação dos usuários, alguns critérios de avaliação de DSL (KAHRAMAN e BILGEN, 2013) foram utilizados:

Capacidade de aprendizado: facilidade de assimilação dos conceitos do domínio e das funcionalidades da ferramenta de especificação (editor) (GABRIEL, 2010). Os conceitos e símbolos da linguagem são fáceis de aprender e lembrar (KAHRAMAN e BILGEN, 2013).

Facilidade de Uso – impressão do usuário a respeito do cenário executado, que por sua vez reflete a facilidade de uso da DSL.

Eficiência – a capacidade da DSL habilitar os usuários a alcançarem seus objetivos determinados com precisão e de forma completa.

Expressividade – quão compacta e restritiva é a DSL para expressar as intenções do usuário.

Impressão Geral – impressão final do usuário da linguagem.

Intenção de Uso – o grau em que uma pessoa tem a intenção de usar a DSL (MOODY, 2003).

Após realizar a especificação cada participante respondeu um questionário de percepção de facilidade de uso e satisfação a respeito da linguagem ExpDSL. A *Tabela*

29 apresenta os itens avaliados no questionário, e foram adaptadas de (GABRIEL, 2010).

Tabela 29: Questionário de opinião sobre uso de ExpDSL

	Critério	**Questão**	**Tipo**
1	Capacidade de aprendizado	Com que frequência você realizou perguntas ao supervisor durante a especificação do experimento?	Escala de Likert
2		Com que frequência você achou necessário consultar a documentação de ExpDSL durante a especificação do experimento?	Escala de Likert
3	Facilidade de uso	As palavras-chave da linguagem ExpDSL são fáceis de aprender e lembrar.	Sim ou Não
3.1		Quais deles você julga inadequadas?	Aberta
4		Quão fisicamente exigente foi especificar o cenário (experimento) em ExpDSL?	Escala de Likert
5		Quão confiante você se sentiu durante a especificação do experimento em ExpDSL?	Escala de Likert
6	Impressões gerais	Você se sente mais produtivo especificando um experimento em ExpDSL em relação a especificar em linguagem natural?	Escala de Likert
6.1		Por que?	Aberta
7		ExpDSL te protege de cometer erros na especificação do experimento.	Escala de Likert
8		Que mudanças ou adições que você propõe para a linguagem?	Aberta
9	Eficiência	Como você julga a corretude do cenário que	Escala de

		você especificou em ExpDSL?	Likert
10		O resultado da especificação em ExpDSL realizada hoje corresponde ao que você esperava?	Escala de Likert
11	Expressividade	Com que frequência você se sentiu incapaz de expressar o que pretendia em ExpDSL?	Escala de Likert
12		Quão simples você achou o cenário (experimento) especificado?	Escala de Likert
13		Um experimento pode ser mapeado em ExpDSL facilmente.	Escala de Likert
14	Intenção de uso	Eu, definitivamente, não usaria ExpDSL para especificar / experimentos.	Escala de Likert
15		Eu pretendo usar ExpDSL para especificar experimentos controlados ao invés de especificar em linguagem natural	Escala de Likert
		Comentários	

7.2.2.5 Ameaças à Validade

Esta seção apresenta e discute as ameaças à validade do estudo que foram identificadas, e como as mesmas foram tratadas. As ameaças à validade do estudo foram classificadas, igualmente ao apresentado na Seção 7.1.2.3, com as seguintes nomenclaturas: (i) validade interna, (ii) validade externa, (iii) validade de construção e (iv) validade da conclusão (WOHLIN, 2012).

Validade de Conclusão: *Participantes de heterogeneidade randômica*. Esta ameaça aparece quando os participantes são selecionados aleatoriamente e seu background é muito heterogêneo. Todos os participantes foram selecionados a partir da disciplina de engenharia de software experimental, onde todos, a princípio, estão com o mesmo nível

de conhecimento, suficiente para participar da pesquisa. Confiabilidade da Medição: A validade de um experimento é altamente dependente de sua medida, que pode depender de diferentes fatores. Neste segundo experimento, a subjetividade na correção das especificações representa uma ameaça pois depende de julgamento humano. Para minimizar o efeito desta ameaça, nós criamos um padrão de correção e as especificações dos usuários foram corrigidas por dois pesquisadores.

Validade Interna: Ameaças à validade interna são influências que podem afetar os fatores com respeito a casualidade, sem o conhecimento do pesquisador. *Ameaça de grupo simples (Single group threats)*, esta ameaça se aplica a experimentos onde não há um grupo de controle (grupo onde não é aplicado o tratamento). A ameaça diz respeito a problemas para determinar se o tratamento, ou outro fator, causou o efeito observado. Nós tratamos as seguintes ameaças: *Seleção*, isto significa que o resultado pode ser afetado por como os participantes são selecionados. Uma participação forçada pode resultar em uma pobre motivação. Nós recrutamos estudantes de mestrado e doutorado do curso de experimentação e os mesmos tiveram a opção de recusar a participação. Portanto, nós tivemos apenas participantes voluntários, dos 12 (doze) alunos do curso, apenas 10 (dez) aceitaram participar deste segundo experimento. *Instrumentação*. Este efeito é causado pelos artefatos usados na execução do experimento. Desde que os participantes devem especificar um experimento em ExpDSL a partir da sua descrição em artigo, o conteúdo do artigo em relação ao plano do experimento pode afetar negativamente o desempenho dos participantes. Para tentar evitar esta ameaça nós aleatorizamos dois artigos entre os participantes cujos conteúdos foram previamente avaliados em relação as informações necessárias para corretamente especificar um experimento em ExpDSL, além disso ambos os artigos foram oriundos de conferências/periódicos de alto nível em experimentação.

Validade de construção: esta validade diz respeito a generalização do resultado do experimento para o conceito ou teoria por trás do experimento. *Mono-tendência do método* (m*ono-method bias)*, significa que os experimentos com um único tipo de medida podem resultar em um viés de aferição. Em nosso experimento, nós minimizamos este efeito com medidas para taxa de acerto, esforço e percepção do usuário quanto a facilidade de uso da DSL, mesmo não sendo possível cruzar umas contra as outras. *Confiabilidade da medida*, diz respeito a adequabilidade dos instrumentos de medidas (questões e roteiros). Apesar da subjetividade inerente, nós

minimizamos este efeito ao utilizar a avaliação de principais aspectos experimentais no roteiro e utilizar medidas de percepção do usuário baseadas na literatura existente.

Validade Externa: esta validade diz respeito a generalização dos resultados para a prática industrial. *Interação da seleção e tratamento*, diz respeito a ter uma população de participantes que não é representativa da população que queremos generalizar. Em nosso experimento, a ameaça foi evitada porque todos os participantes têm um background similar. Isto significa que, neste momento, nós podemos apenas dizer que os resultados de nosso estudo podem ser válidos para estudantes de pós-graduação com conhecimento em engenharia de software experimental e com pouca experiência na condução de experimentos controlados. *Dependência do Objeto*, significa que os resultados podem depender dos objetos usados no experimento e eles não podem ser generalizados. Nós minimizando esta ameaça em nosso estudo usando dois objetos para o tratamento. Nós também selecionamos objetos reais (experimentos alvo), fruto de publicação em eventos da área de engenharia de software experimental, e com diferentes designs experimentais. Apesar disso, nós não podemos generalizar os resultados para qualquer experimento ou domínio de experimentação. Nós poderíamos evitar esta ameaça usando uma variedade de objetos com grande diferença entre eles. Entretanto, esta situação requereria mais tempo e recursos para rodar o experimento.

7.2.3 Apresentação dos Resultados: Experimento #2

7.2.3.1 Resultados Obtidos pelos Participantes do Experimento 2

A *Tabela 30* apresenta os resultados obtidos pelos 10 (dez) participantes na execução do experimento. Por uma questão de impessoalidade foram omitidos os nomes dos mesmos. A tabela registra os resultados da correção do experimento (Taxa de Acertos) por participante, assim como a duração de cada especificação.

Tabela 30: Resultados do Experimento 2

Participante	**Experimento Alvo**	**Taxa de Acerto**	**Duração (em minutos)**
P1	A	0,86	73
P2	A	0,64	74
P3	A	0,79	69

P4	A	0,86	47
P5	A	0,71	108
P6	B	0,57	70
P7	B	0,86	73
P8	B	0,57	99
P9	B	0,86	73
P10	B	0,36	36

7.2.3.2 Análise dos Resultados Obtidos do Experimento 2

A Figura 36 apresenta o relatório geral para a análise do tempo gasto para especificação de um experimento em ExpDSL. O resultado mostra que o tempo médio de especificação de experimentos como o utilizado ficou entre 57 e 87 minutos. O P-Value 0,080 para o teste de normalidade confirmam que os dados seguem uma distribuição normal (suposição do intervalo de confiança estimado).

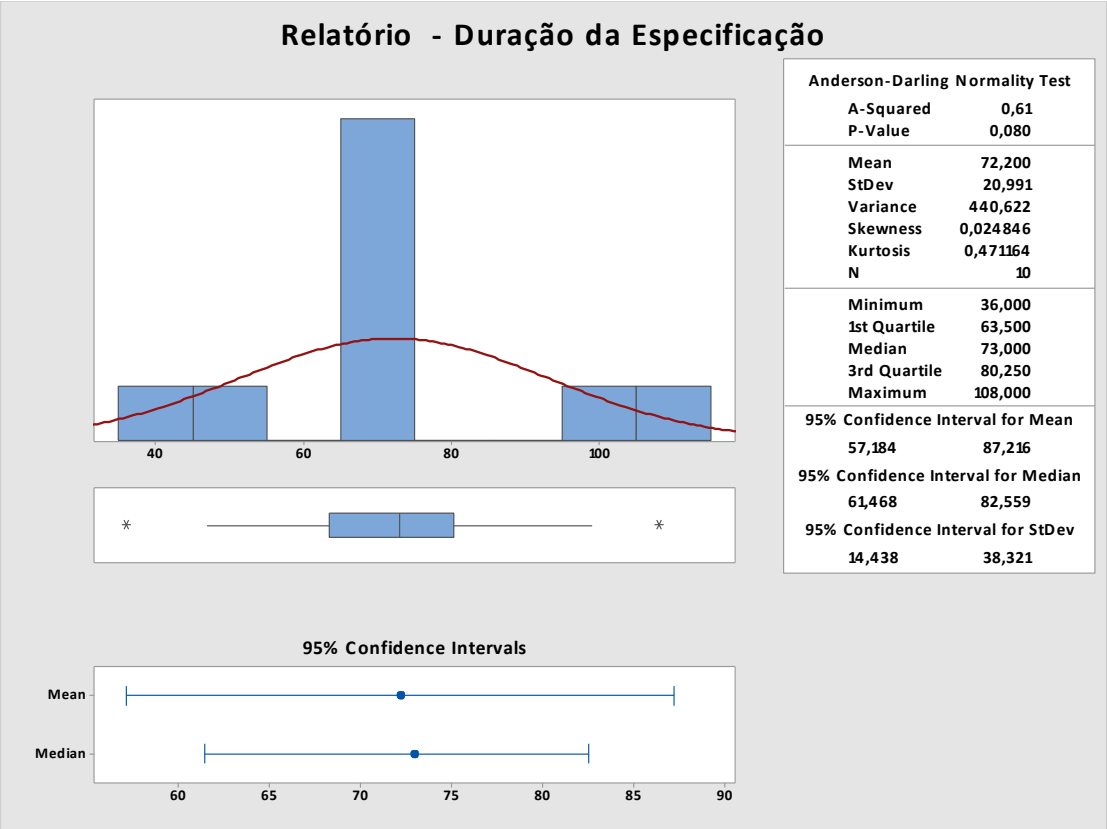


Figura 36: Relatório da Duração das Especificações pelos participantes

A **Error! Reference source not found.** apresenta o relatório estatístico para a análise da taxa de corretude para as especificações criadas em ExpDSL. O resultado mostra que a taxa de acerto média foi de aproximadamente 71%, sendo entre 59% e 83% o intervalo de confiança para a taxa média estimada para cenários semelhantes ao do experimento. Esse resultado indica que não há evidências para rejeitar a hipótese nula definida para o experimento, já que o intervalo de confiança contém valores menores que 80%.

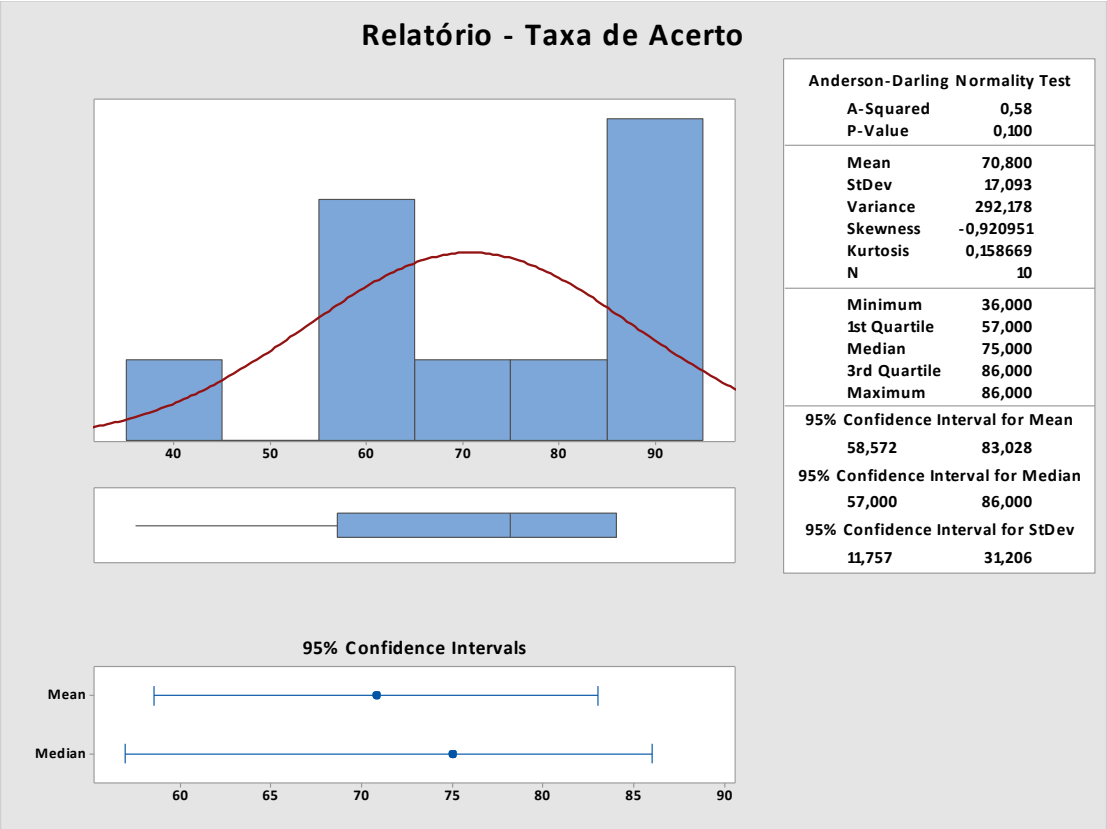


Figura 37: Relatório Estatístico da Taxa de Acerto dos Participantes

Apesar deste resultado, é importante salientar que um dos aspectos avaliados foi a especificação do procedimento de coleta de dados (representado por um processo na DSL), porém nenhum participante realizou esta especificação pois não há informações suficientes na descrição do artigo para este nível de detalhes. Esse fato impactou negativamente a taxa de acertos, mas também nos mostra que apesar de ser um meio efetivo para relatar um experimento, sua descrição em artigo não é precisa (o que dificulta uma replicação, por exemplo). Um segundo aspecto que foi respondido de forma incompleta por todos os participantes, gerando efeito negativo na taxa de acertos, foi o contexto do experimento. Como em ambos os artigos não havia uma seção de "caracterização do experimento", esta informação estava diluída no texto ao longo do plano, o que pode ter dificultado o entendimento do aspecto por parte dos participantes.

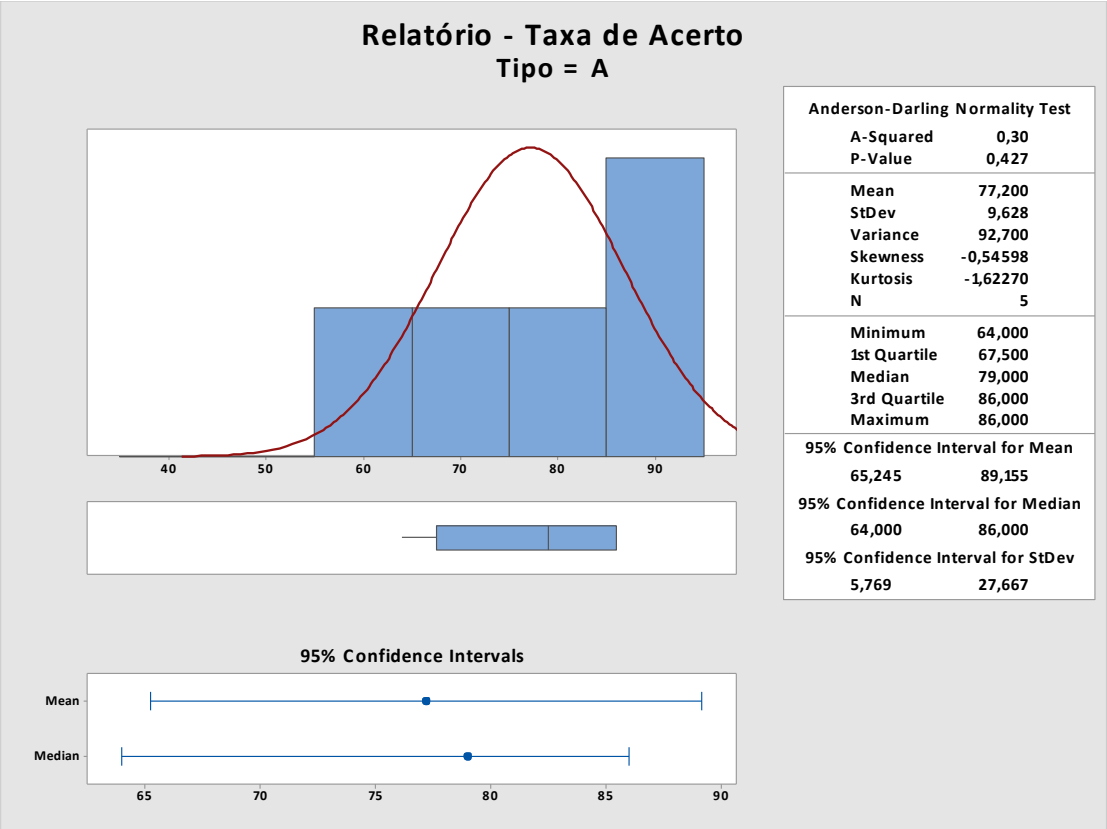


Figura 38: Taxa de acerto para Experimento Tipo A

Para tentar avaliar a influência do tipo de experimento sendo modelado no resultado obtido, geramos os gráficos da taxa de acerto para cada tipo. As Figura 38 e Figura 39 ilustram o resultado, onde podemos observar que a média de acertos para o experimento modelado A é de 77% (variando de 65% a 89%), já a taxa de acertos para o tipo B é de 64% (variando de 38% a 91%), o que mostra uma variação muito grande para o experimento do tipo B. Esse resultado sugere que o tipo do experimento pode ter influenciado neste resultado, já que os mesmos elementos de ExpDSL poderiam ser utilizados tanto para o experimento A quanto para o experimento B.

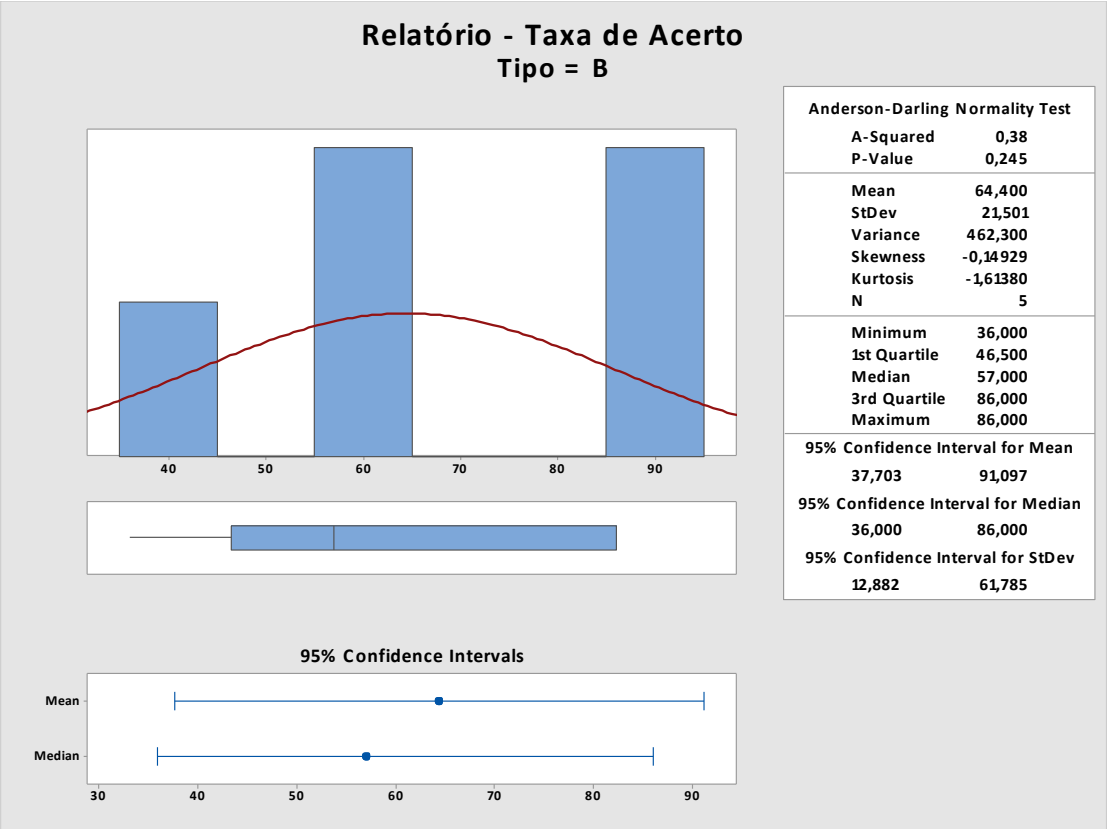


Figura 39: Taxa de acerto para Experimento Tipo B

7.2.4 Análises Qualitativas – Experimento #2

Após a realização do experimento através da realização de uma especificação, os participantes responderam a um questionário de avaliação qualitativa sobre ExpDSL. O questionário possuía 15 (quinze) questões ao todo (ver Tabela 29). Em oito das questões, o participante respondia de acordo com a escala Likert (LIKERT, 1932) com 5 níveis (representada por um número de 1 a 5). Esta seção apresenta uma consolidação das respostas dos participantes, bem como uma análise e discussão sobre as mesmas. Um dos objetivos dessa avaliação qualitativa é relacionar, sempre que possível, as suas respostas com os resultados da realização das questões de compreensão do roteiro do experimento.

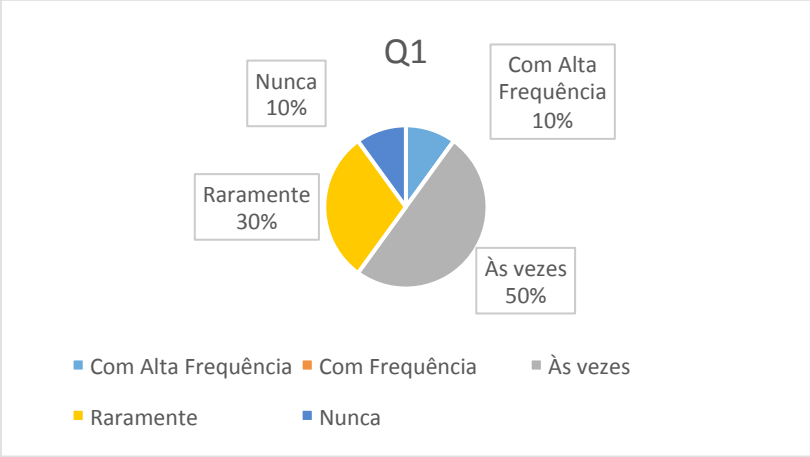


Figura 40: Resultado da Questão 1 do Questionário de Opinião

As duas primeiras questões do questionário de opinião do Experimento #2 visam avaliar a capacidade de aprendizado dos participantes. Neste sentido, a primeira questão pergunta: "*Com que frequência você realizou perguntas ao supervisor durante a especificação do experimento?*". O resultado mostra que 50% dos participantes realizaram perguntas ao supervisor somente "Às vezes", e 30% "Raramente" e 10% Nunca fizeram perguntas, porém 10% realizou perguntas "Com Alta Frequência".

A segunda questão pergunta: "*Com que frequência você achou necessário consultar a documentação de ExpDSL durante a especificação*?". A questão apresentou 50% dos participantes indicando que consultou a documentação "Com Frequência" (30%) ou "Com Alta Frequência" (20%). 40% Nunca ou Raramente utilizou a documentação, e 10% fez a consulta "Às vezes".

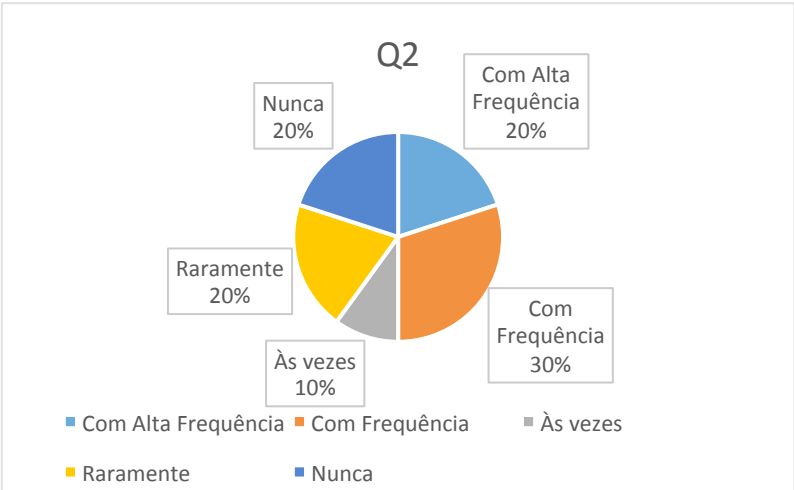


Figura 41: Resultado da Questão 2 do Questionário de Opinião

A questões 3, 4 e 5 visam avaliar a percepção do usuário quanto a facilidade de uso da DSL. Para a terceira questão existe uma afirmação: "*As palavras-chave da linguagem ExpDSL são fáceis de aprender e lembrar*". O resultado mostra que 40% dos participantes concordaram fortemente com a afirmação e 60% concordaram. Essa resposta mostra que a percepção do usuário é de que a sintaxe da linguagem é fácil de aprender e lembrar.

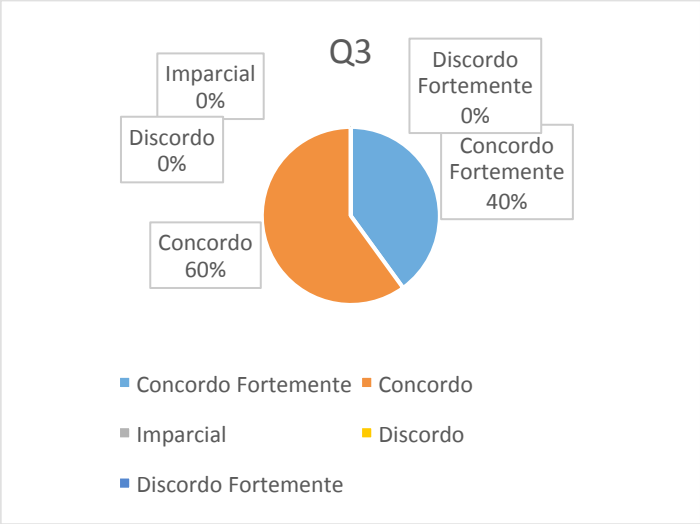


Figura 42: Resultado da Questão 3 do Questionário de Opinião

A questão 3.1 é uma questão aberta que complementa a questão 3. Ela diz "Quais delas você julga inadequadas?". Para essa questão, apenas 7 participantes responderam. Dentre eles, apenas um, o participante P4, apontou uma palavra-chave (Factor) que julgou inadequada, os demais apenas afirmaram que não julgavam nenhuma palavra inadequada.

Tabela 31: Respostas para a Questão 3.1 do questionário de opinião

Participante	**Resposta**
P1	Nenhuma
P3	Nenhuma. Estão todas claras em seu propósito.
P4	Factor
P5	N/A
P7	Todas são adequadas e de fácil aplicação.
P9	Nenhuma
P10	Nenhuma

A questão 4 pergunta: "*Quão fisicamente exigente foi especificar o cenário (experimento) em ExpDSL?*" e visa saber qual foi a percepção do usuário em relação ao cenário sendo especificado em termos de complexidade. A análise dos resultados mostra que 40% dos participantes acharam o nível de exigência do cenário como sendo Regular, outros 40% acharam o nível de exigência Baixo, e apenas 20% considerou o nível de exigência Alto. Neste sentido, fomos levados a acreditar que o nível de exigência do cenário influenciou na percepção da facilidade de uso, visto que 80% dos participantes não o consideraram nem Alto, nem Muito Alto, o que pode levar o participante a crer que a linguagem é fácil influenciado pela simplicidade do cenário que especificou.

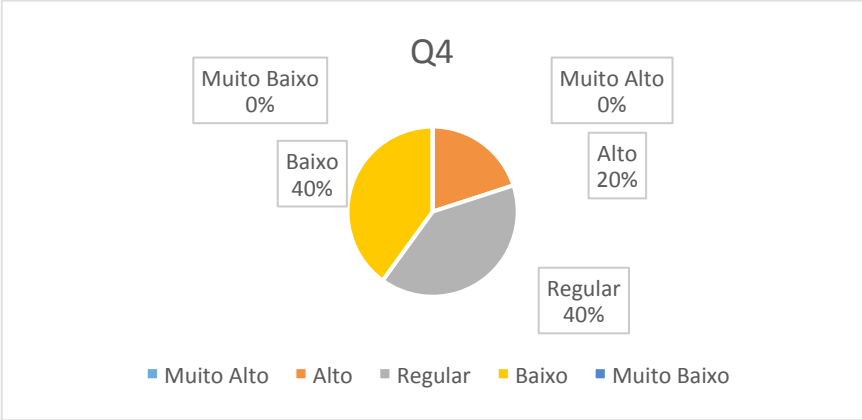


Figura 44: Resultado da Questão 4 do Questionário de Opinião

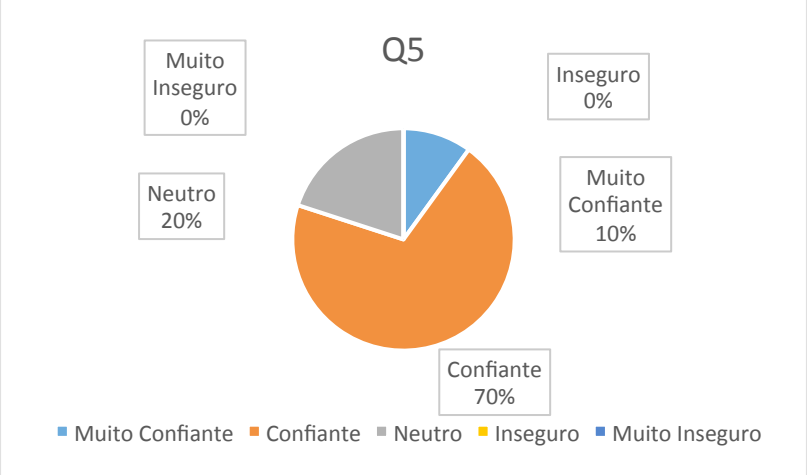


Figura 43: Resultado da Questão 5 do Questionário de Opinião

A questão 5 pergunta "Quão confiante você se sentiu durante a especificação do experimento em ExpDSL?". O gráfico da Figura 43, mostra que 80% dos participantes

se sentiram Confiante(s) (70%) ou Muito Confiante (10%), os outros demais participantes ficaram Neutro(s) em relação à confiança nos cenários que haviam especificado. De toda forma, nenhum participante relatou que se sentia inseguro ou muito inseguro em relação as suas especificações. Esse resultado mostra que a linguagem ajuda o participante a se sentir confiante quando especificando experimentos.

De uma forma geral, pelos resultados das questões 3, 4 e 5, concluímos que a percepção do usuário é que a linguagem ExpDSL é fácil de se usar.

O objetivo das questões 6, 7 e 8 é coletar impressões gerais dos participantes em relação à linguagem ExpDSL. A questão 6 perguntou: "*Você se sente mais produtivo especificando um experimento em ExpDSL em relação a especificar em linguagem natural?*" a fim de avaliar a percepção do participante em relação a sua própria produtividade. Como resultado, 90% dos participantes responderam que "Sim", se sentem mais produtivos utilizando a linguagem para especificar experimentos, e 10% (participante P5) respondeu que "Não", não se sente mais produtivo utilizando a linguagem ExpDSL.

Para complementar esta questão, a questão 6.1 levanta o porquê da resposta anterior. A resposta de cada participante é apresentada na Tabela 32. É possível observar que todas as respostas para questão 6.1 são positivas em relação à linguagem, o que nos leva a questionar a resposta do participante P5 para a questão 5 (única resposta que diz que usar ExpDSL não o deixa mais produtivo).

Tabela 32: Repostas para Questão 6.1 do questionário de Opinião

Participante	**Resposta da questão "Por que?"**
P1	O foco é bem direcionado, portanto, torna mais fácil especificar.
P2	As palavras chaves e validações auxiliam muito.
P3	Você pode seguir um passo a passo na especificação do experimento e relacionar todos os elementos sem se perder facilmente. Num documento de texto é preciso pensar em como organizar a especificação, coisa que já é feita em ExpDSL
P4	Fica mais fácil descrever e visualizar os componentes do experimento.
P5	Checa a consistência entre hipótese, goals, contexto, etc.

P6	Na ExpDSL eu sei exatamente qual é a estrutura e quais são as opções que eu tenho a cada momento (auto completar).
P7	Porque serve como guia para construção/planejamento do experimento, de forma estruturada e completa.
P8	A especificação fica mais estrutura, facilitando a escrita e leitura.
P9	Por ser especificado em formato de tópicos sendo mais fácil de lembrar e gerenciar
P10	Porque a especificação com ExpDSL é mais produtiva e direta.

A questão 7 apresenta uma afirmação "ExpDSL te protege de cometer erros na especificação do experimento.". Como resultado, 80% dos participantes Concordam (20%) ou Concordam Fortemente (60%) que ExpDSL os ajuda a não cometer erros na especificação do experimento. Nenhum participante discordou ou discordou fortemente da afirmação.

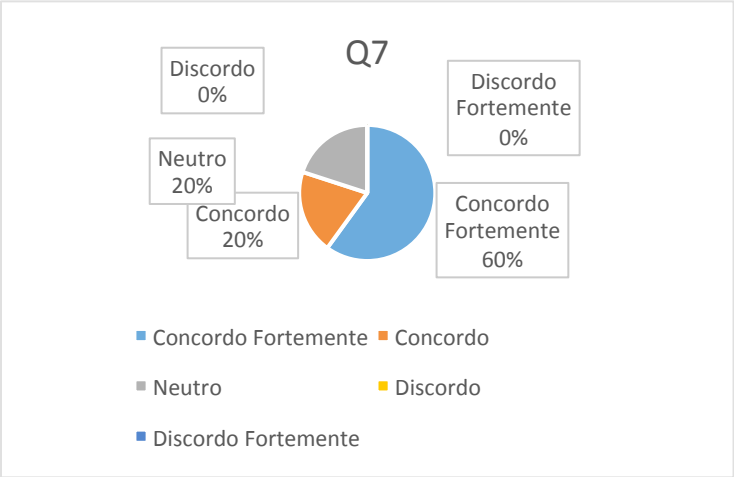


Figura 45: Resultado da Questão 7 do Questionário de Opinião

A questão 8 é aberta e opcional e questiona "*Que mudanças ou adições que você propõe para a linguagem?*". Apenas 7 participantes responderam, e destes, 4 apenas responderam que nenhuma mudança ou adição seria necessária. O resultado completo é apresentado na Tabela 33. O participante P2 e P4 sugeriram a adição de mecanismos de "Ajuda" rápida que pudessem ser exibidos durante a digitação das palavras-chave. O

participante P6 deu várias sugestões em relação aos elementos da linguagem, a organização dos elementos e também em relação a ajuda para formatação do texto.

Tabela 33: Respostas da Questão 8.

Participante	**Resposta para a questão 8**
P1	Nenhuma
P2	Adicionar documentação nos comandos, para quando colocar o mouse sobre as palavras chaves poder identificar ainda mais fácil o seu significado.
P3	Não vejo nenhuma mudança necessária
P4	Dicas nas palavras-chaves, mostrando o que os componentes da linguagem são e o que eles descrevem, talvez até exemplificando cada componente.
P5	N/A
P6	As "aspas" dos Strings as vezes atrapalham um pouco. Em alguns casos, eu acho que elas são desnecessárias. Por exemplo no "abstract" é preciso colocar "chaves" e "aspas" para delimitar o conteúdo. Em keywords poderia usar a quebra de linha (\n) como delimitador entre as strings. Em outros casos poderia delimitar strings usando apenas { } como em LaTeX. Outro ponto é a ordem dos elementos da gramática da linguagem. A estrutura poderia ser mais flexível para alterar a ordem dos elementos. Mostrar os erros de ortografia dos Strings seria bom. Por fim, poderia usar o recurso do auto formatar do Eclipse (CTRL+SHIFT+F) para arrumar o código fonte do experimento.
P7	Nada a adicionar.
P8	
P9	
P10	

As questões 9 e 10 visam avaliar a percepção do usuário em relação à sua própria eficiência quando utilizando a linguagem. A questão 9 pergunta: "*Como você julga a corretude do cenário que você especificou em ExpDSL?*". O gráfico da Figura 46 apresenta os resultados e mostra que 50% dos participantes consideram a corretude de sua especificação "Satisfatória", enquanto 40% considera "Boa" e 10% "Muito Boa". Nenhum participante considerou que a corretude de sua especificação estaria Ruim ou Muito Ruim.

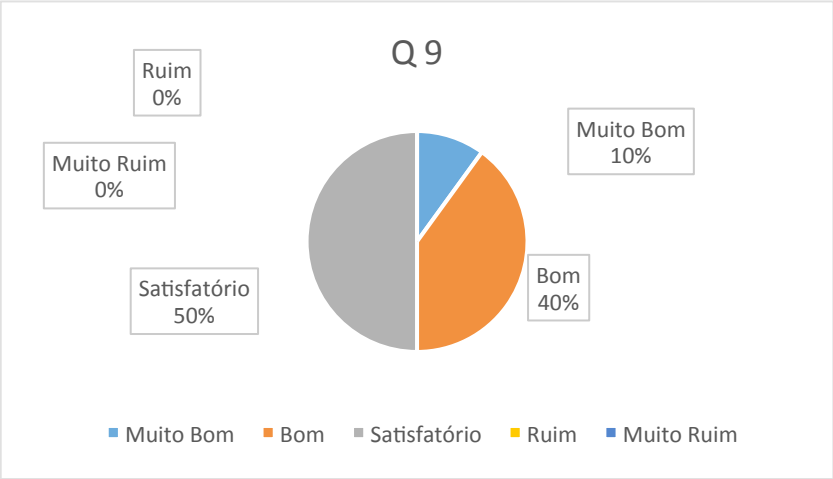


Figura 46: Resultado da Questão 9 do Questionário de Opinião

A questão 10 afirma: "*O resultado da especificação em ExpDSL realizada hoje corresponde ao que você esperava".* Como resultado, temos que 40% dos participantes concordam fortemente com a afirmação, 50% concordam, e 10% se mostrou neutro. Nenhum participante julgou que sua especificação estava abaixo do que esperavam.

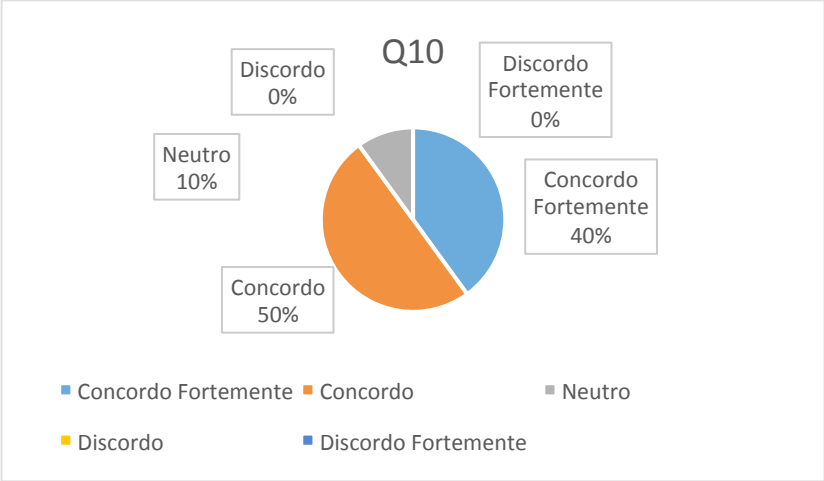


Figura 47: Resultado da Questão 10 do Questionário de Opinião

De forma geral, pelos resultados das questões 9 e 10, concluímos que o usuário se julga mais eficiente quando especificando um experimento usando ExpDSL.

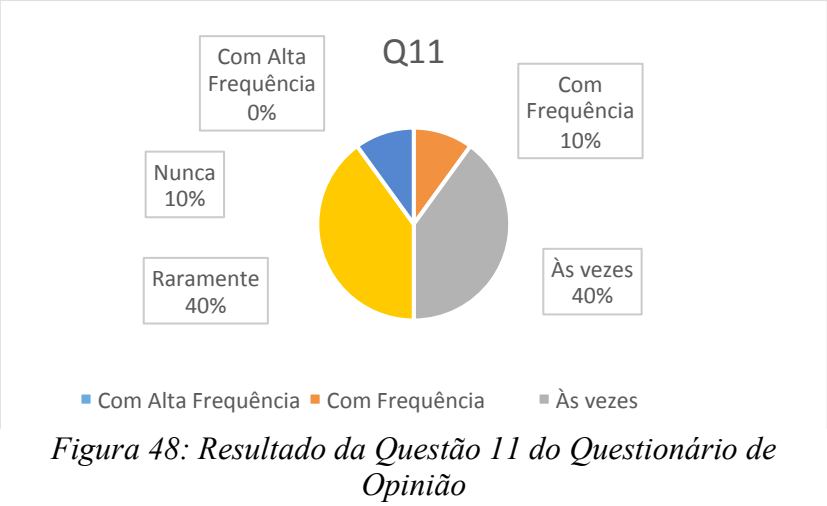


Figura 48: Resultado da Questão 11 do Questionário de Opinião

As questões 11, 12 e 13 visam avaliar a percepção do usuário em relação à expressividade da linguagem.

A questão 11 pergunta: "*Com que frequência você se sentiu incapaz de expressar o que pretendia em ExpDSL?*". O resultado da análise mostra que 50% dos participantes Nunca ou Raramente se sentiram incapazes de expressar o que pretendiam em ExpDSL. 40% dos participantes julgaram que "Às vezes" se sentiram incapazes de expressar o que pretendiam com a linguagem, e apenas 10% dos participantes julgaram-

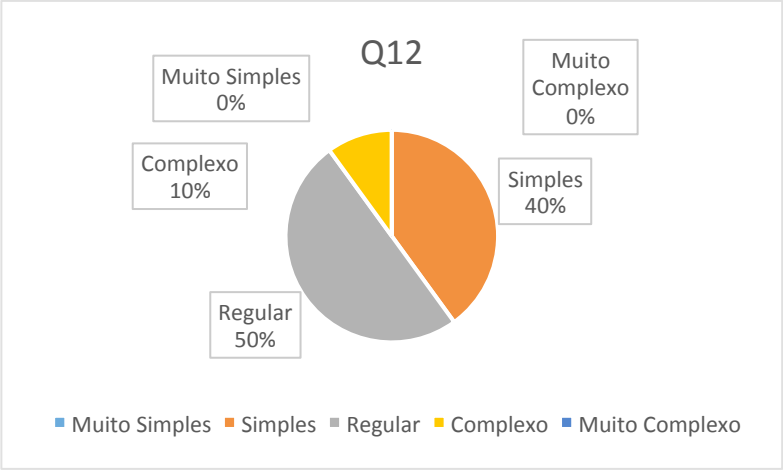


Figura 49: Resultado da Questão 12 do Questionário de Opinião

se incapazes de expressar o que desejavam "Com Frequência".

A questão 12 pergunta: "Quão simples você achou o cenário (experimento) especificado?". Como resultado, 90% dos participantes consideram o cenário especificado simples (40%) ou regular (50%), e 10% considerou o cenário complexo. Nenhum participante considerou o cenário Muito Complexo ou Muito Simples. De toda forma, o cenário apresentado para especificação ajudou o participante à expressar o que desejava na linguagem ExpDSL. Neste sentido, pretendemos analisar novos cenários em trabalhos futuros.

A questão 13 afirma: "*Um experimento pode ser mapeado em ExpDSL facilmente.*" A análise da resposta dos participantes mostrou que 90% dos participantes

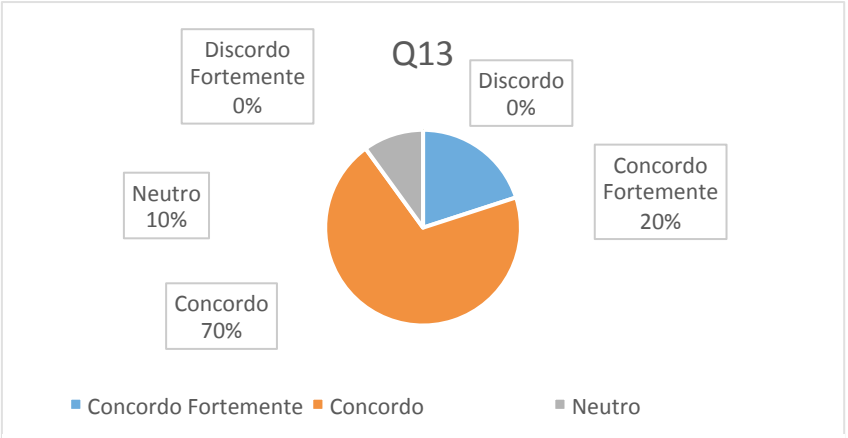


Figura 50: Resultado da Questão 13 do Questionário de Opinião

ou Concorda(m) com afirmação (70%) ou Concorda(m) Fortemente. Os demais participantes (10%) são neutros em relação à afirmação. Este resultado foi bastante satisfatório em relação à percepção de expressividade da linguagem.

As questões 14 e 15 procuram avaliar a intenção de uso da linguagem ExpDSL pelos participantes.

A questão 14 afirma: "*Eu, definitivamente, não usaria ExpDSL para especificar experimentos*.". O resultado mostra que 90% dos participantes ou "Discorda" (30%) ou "Discorda Fortemente" (60%) desta afirmação. 10% dos participantes foram neutros em relação à afirmação.

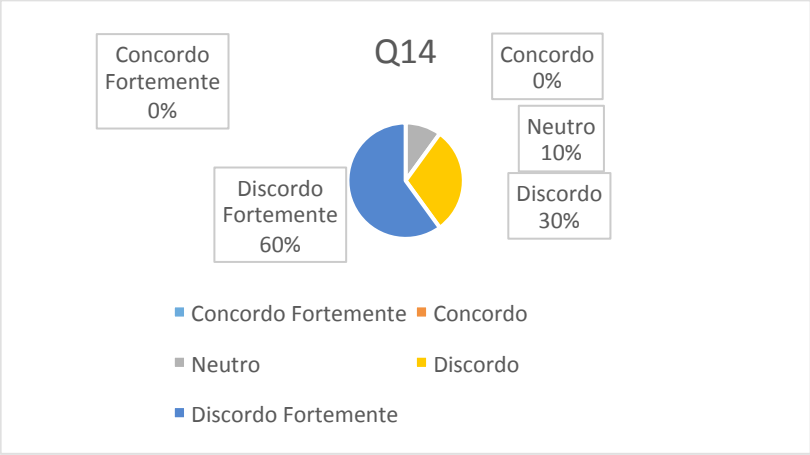


Figura 51: Resultado da Questão 14 do Questionário de Opinião

A questão 15 afirma: "*Eu pretendo usar ExpDSL para especificar experimentos controlados ao invés de especificar em linguagem natural*.". Como resultado, 100% dos participantes foram positivos e concordaram (90%) ou concordaram fortemente (10%) com a afirmação.

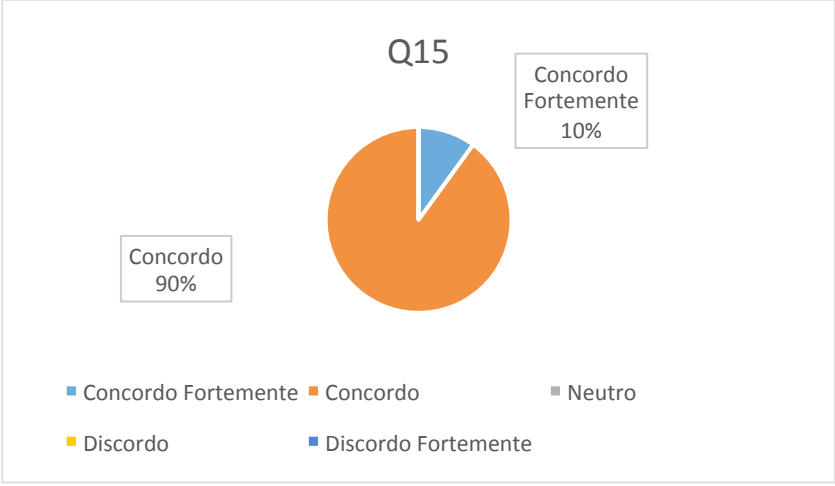


Figura 52: Resultado da Questão 15 do Questionário de Opinião

O resultado apresentado para as questões 14 e 15 nos levam a crer que os participantes estão satisfeitos com a linguagem e pretendem utiliza-la em novas especificações.

7.3. Conclusões

Este capítulo relatou a motivação, planejamento, execução e análise dos resultados de dois estudos experimentais, visando a avaliação da linguagem ExpDSL.

O primeiro experimento executado visava avaliar a compreensão da linguagem ExpDSL na especificação de um plano experimental. Como resposta a primeira questão de pesquisa do experimento 1, "A compreensão de um plano experimental especificado em ExpDSL é diferente daquela descrita em um artigo científico?', nós podemos concluir, com significância estatística, que há diferença, e que essa diferença é positiva em relação à compreensão do plano como um todo. Como resposta a segunda questão de pesquisa do experimento, **"**O tempo para compreensão do planejamento de um experimento depende do tipo de especificação utilizada para sua descrição?", nós observamos com significância estatística que sim, depende, e para a compreensão do plano como um todo esse tempo foi menor para as especificações em ExpDSL. Para a questão de pesquisa "Qual a percepção do leitor relacionada a compreensão de um plano experimental descrito em ExpDSL?", tivemos respostas positivas que nos levam a concluir que o participante teve uma boa percepção em relação à compreensão da especificação de um plano em ExpDSL.

O segundo experimento visava avaliar a facilidade de uso através do levantamento da taxa de acertos média ao se especificar um plano em ExpDSL, assim como tempo médio, além da opinião do experimentador sobre a facilidade de uso da linguagem. Respondendo as questões de pesquisa "É fácil especificar experimentos controlados em ExpDSL?", apesar de obtida uma taxa de acertos média de 70% , não podemos concluir que é fácil especificar experimentos em ExpDSL (não houve evidência para rejeitar a hipótese nula e aceitar nossa hipótese alternativa que era de uma taxa maior que 80%). Em relação ao tempo médio de especificação de experimentos como o utilizado foi estimado um intervalo de confiança entre 57 e 87 minutos. Para a questão: "Qual a opinião dos experimentadores sobre o uso de ExpDSL para especificar seus experimentos?" nós avaliamos alguns critérios através do questionário de opinião obtendo as seguintes conclusões: (i) Capacidade de Aprendizado: consideramos positiva

pois consideramos normal ter recorrido à documentação ou realizado perguntas aos pesquisadores uma vez que os participantes não são especialistas na linguagem e haviam passado apenas por um treinamento e sessão de aquecimento; (ii) Facilidade de Uso: consideramos que a percepção do usuário é que a linguagem ExpDSL é fácil de se usar; (iii) Impressões gerais: apontam que os experimentadores se sentem mais produtivos e seguros ao especificar um experimento em ExpDSL; (iv) Eficiência: o resultado aponta que os experimentadores avaliaram positivamente sua eficiência ao especificar o cenário proposto utilizando ExpDSL; (v) Expressividade: o resultado aponta que precisamos investigar melhor a percepção do usuário quanto à sua capacidade de expressão na linguagem; e (vi) Intenção de uso: a intenção dos participantes em utilizar a linguagem ExpDSL foi bastante positiva, o que indica que houve uma aceitação da linguagem para o contexto em que foi aplicada.

8. Trabalhos Relacionados

Para auxiliar pesquisadores na realização de experimentos controlados, trabalhos de pesquisa têm desenvolvido ferramentas de apoio à condução de experimentos controlados. Entretanto, a maioria destes trabalhos não descrevem como atendem à especificação detalhada dos aspectos experimentais (formalização). As subseções seguintes apresentam trabalhos relacionados com respeito ao apoio à formalização e a execução de experimentos controlados em ES.

8.1. Formalização de Experimentos Controlados

Algumas pesquisas propõem formalizações para auxiliar o registro do material, procedimentos e resultados do experimento a fim de facilitar a troca de informações entre pesquisadores e, consequentemente, a replicação dos estudos.

8.1.1 Ontologias

Garcia et al (GARCIA, HOHN, *et al.*, 2008) propôs uma ontologia, chamada experOntology, para ajudar a entender os conceitos de um experimento controlado com foco na fase de empacotamento. A ontologia abrange conceitos do estudo: definição, planejamento, execução, etc; e conceitos da definição do plano experimental que contém hipóteses iniciais, objetos de estudo, variáveis dependentes e independentes, etc. A avaliação da ontologia foi realizada pela instanciação de conceitos em um experimento de Verificação e Validação (V&V) (BASILI e SELBY, 1987). Eles concluem que a ontologia proposta pode apropriadamente instanciar um experimento controlado e também indicam que o processo de instanciação pode destacar valores indisponíveis no pacote do experimento.

De forma complementar, Scatalon et al (SCATALON, GARCIA, & CORREIA e M, 2011) propõem um *workflow* que incorpora uma abordagem para instanciação de pacotes na replicação de um estudo usando experOntology. A abordagem deles consiste em combinar conceitos fornecidos no pacote com os conceitos definidos em experOntology. Por exemplo, se um conceito de “Contexto” é combinado, ou seja, ele foi fornecido com o pacote, então é possível capturar qual foi o contexto do experimento anterior, por exemplo, “contexto acadêmico”. Entretanto, se o conceito não for propriamente combinado, então o workflow sugere a evolução de experOntology

com a adição dos valores para o conceito. Esta abordagem foi usada para instanciar o experimento publicado por De Lucia et al (DE LUCIA, GRAVINO, *et al.*, 2010).

Uma outra ontologia para apoiar experimentos controlados foi proposta por Siy and Wu (SIY e WU, 2009). Esta ontologia foca na mitigação de riscos que representem ameaças à validade do experimento através da modelagem do projeto experimental. Baseado na experiência dos especialistas, os autores formalmente especificaram uma série de restrições que podem representar ameaças à validade de um experimento. Esta abordagem é avaliada através da modelagem de uma família de experimentos que avalia técnicas de leitura para identificar falhas em documentos de especificação de requisitos. A conclusão do trabalho é que a ontologia poderia apropriadamente especificar o experimento e identificar inconsistências usando as restrições definidas.

Assim como Garcial et al (GARCIA, HOHN, *et al.*, 2008) e Siy e Wu (SIY e WU, 2009), o trabalho apresentado nesta tese de doutorado também formaliza conceitos de experimentos em ES. Entretanto, ao invés de definir uma ontologia, foi proposta uma DSL que formaliza aspectos do experimento controlado (plano do experimento, processo de coleta de dados, questionários de feedback). Adicionalmente, a DSL proposta pode ser usada para gerar especificações de workflows os quais podem ser interpretadas por um motor de execução de *workflow* permitindo a execução e monitoramento do experimento controlado. Enquanto Scatalon et al. (SCATALON, GARCIA, & CORREIA e M, 2011) focaram apenas na definição de um workflow para desenvolver o conhecimento sobre pacotes experimentais, nosso trabalho focou no apoio ao planejamento, execução e, possível análise, de um experimento controlado em ES. Além disso, ExpDSL foi avaliada de forma empírica e experimental, comprovando seus benefícios através de evidências estatísticas (Capítulos 6 e 7).

8.1.2 Linguagens Específicas de Domínio

Cartaxo et al. (CARTAXO, COSTA, *et al.*, 2012) definem uma DSL gráfica chamada ESEML (*Empirical Software Engineering Modeling Language*) para especificar o planejamento de um experimento controlado. ESEML define entidades básicas de plano experimental (variáveis dependentes, hipóteses, testes de hipóteses, e projeto do experimento, por exemplo). Após a definição do experimento, a DSL pode gerar um documento no formato PDF contendo o plano do experimento modelado. De

forma similar, nosso trabalho também define uma linguagem específica de domínio para formalizar um experimento controlado. Entretanto, a DSL ESEML não deixa claro como seus elementos poderiam ser automaticamente transformados para determinar especificações a serem usadas na execução do experimento, pois ESEML não contém elementos para especificar as atividades/tarefas dos participantes. Além disso, ESEML não foi avaliada através da modelagem de experimentos controlados existentes. Por outro lado, ExpDSL pode ser transformada e executada em um motor de execução de workflow, que dá suporte para as fases de execução e análise de um experimento controlado. Além disso, ExpDSL foi avaliada na modelagem de 16 casos de experimentos reais (Capítulo 6) e também foi avaliada experimentalmente (Capítulo 7).

8.2. Ferramentas de Apoio a Condução de Experimentos

Alguns trabalhos de pesquisa exploram o desenvolvimento de ambientes automatizados para dar apoio à condução de experimentos controlados, como apresentado no Capítulo 3 (FREIRE, ALENCAR, *et al.*, 2013) desta tese de doutorado.

Hochstein et al. (HOCHSTEIN, NAKAMURA, *et al.*, 2008) apresentam *Experiment Manager Framework*, um conjunto integrado de ferramentas que são utilizadas para realizar experimentos durante o processo de desenvolvimento de aplicações de alta performance. A ferramenta é voltada para esse subdomínio, contendo tarefas para os participantes específicas para este fim. A ideia é que a ferramenta seja utilizada durante o desenvolvimento do software, no ambiente real do desenvolvedor. A ferramenta contabiliza a duração das atividades e faz inferências para descobrir qual a próxima atividade do participante no processo de desenvolvimento. Algumas métricas relativas à computação de alta performance são coletadas e análises estatísticas específicas para o domínio são realizadas. Os autores afirmam ainda não ter realizado um experimento controlado bem elaborado mas que coletaram resultados, sem evidências estatísticas, que mostram que a ferramenta é capaz de produzir dados consistentes, é de fácil uso por pesquisadores não experientes além de exportar dados de forma sanitizada (sem identificação dos participantes).

Para dar suporte à logística de experimentos controlados em larga escala, Arisholm el al (ARISHOLM, SJøBERG, *et al.*, 2002) (ARISHOLM, SJøBERG, *et al.*, 2002) desenvolveram um ambiente de suporte experimental chamado SESE. Ele é uma ferramenta baseada na web que apoia o gerenciamento de participantes, capturando o

tempo gasto durante o experimento, habilitando a coleta de produtos de trabalho e monitorando as atividades dos participantes. Seu ponto fraco é a o apoio a fase de planejamento, já que não fornecem apoio à distribuição de tratamentos e aleatorização de participantes, além disso não há suporte à análise de dados.

Travassos et al (TRAVASSOS, SANTOS, *et al.*, 2008) apresentam um ambiente, chamado eSEE, que fornece um conjunto de facilidades para permitir engenheiros de software e pesquisadores geograficamente distribuídos realizarem e gerenciarem processos de experimentação, assim como trocarem conhecimento científico através de diferentes tipos de estudos na web. eSEE atua em três níveis de organização do conhecimento sobre processo experimental: (i) conhecimento para qualquer tipo de estudo experimental (nível meta), (ii) conhecimento para cada estudo experimental (nível de configuração), e (iii) conhecimento para um estudo experimental específico (nível de instância). Há um protótipo e um conjunto de ferramentas para popular a infraestrutura sendo desenvolvida.

ExpTool (NETO, GARCIA, *et al.*, 2014) é uma ferramenta de apoio à condução de experimentos controlados cujo foco é o empacotamento do experimento. A ferramenta se propõe a apoiar as etapas iniciais do experimento, assim como a operação do mesmo. Para a análise e interpretação, a ferramenta permite o registro de informações e arquivos obtidos externamente. Os dados experimentais são organizados em arquivos XML e os dados coletados podem ser baixados em um arquivo ZIP. Os autores relatam que pretendem, como trabalhos futuros, formalizar o conhecimento envolvido no experimento através do uso de uma ontologia.

A Tabela 35 apresenta os resultados de uma comparação entre tais ferramentas descritas nos trabalhos relacionados e o ambiente de suporte proposto neste documento. A comparação é baseada nos requisitos apresentados na Tabela 34.

Tabela 34: Requisitos desejáveis para ambientes experiemntais

RQ1	Definição formal do plano experimental
RQ2	Distribuição automática dos tratamentos
RQ3	Guia para execução das tarefas dos participantes

RQ4	Coleta dos dados de execução
RQ5	Registro automático de tempos de execução
RQ6	Monitoramento da execução
RQ7	Exportação do plano experimental (PDF, EXPDSL, HTML, ...)
RQ8	Caracterização dos participantes
RQ9	Coleta de feedback
RQ10	Preparação da execução
RQ11	Exportação dados de execução
RQ12	Suporte à análise dos dados
RQ13	Empacotamento do experimento
RQ14	Suporte à replicação de experimentos

Para cada ferramenta, os requisitos foram classificados da seguinte forma: (i) ATENDE – quando o mesmo está implementado na ferramenta; (ii) PARCIAL – quando o requisito não está completamente disponível na ferramenta; e (iii) NÃO ATENDE – quando o requisito não é atendido pela ferramenta. A Tabela 35 mostra então essa breve comparação entre as ferramentas.

Tabela 35: Comparação entre trabalhos que propõem ferramentas de auxilio à experimentação.

Requisitos	**Ambiente Proposto**	**ExpTool**	**eSEE**	**SESE**	**Experiment Manager Framework**
RQ1	ATENDE	PARCIAL	ATENDE	NÃO ATENDE	NÃO ATENDE
RQ2	ATENDE	NÃO ATENDE	NÃO ATENDE	NÃO ATENDE	NÃO ATENDE
RQ3	ATENDE	ATENDE	ATENDE	ATENDE	PARCIAL

RQ4	ATENDE	ATENDE	ATENDE	ATENDE	ATENDE
RQ5	ATENDE	ATENDE	NÃO ATENDE	NÃO ATENDE	ATENDE
RQ6	ATENDE	NÃO ATENDE	ATENDE	ATENDE	NÃO ATENDE
RQ7	ATENDE	ATENDE	ATENDE	NÃO ATENDE	NÃO ATENDE
RQ8	ATENDE	ATENDE	NÃO ATENDE	NÃO ATENDE	ATENDE
RQ9	ATENDE	NÃO ATENDE	NÃO ATENDE	NÃO ATENDE	NÃO ATENDE
RQ10	ATENDE	ATENDE	ATENDE	PARCIAL	PARCIAL
RQ11	PARCIAL	ATENDE	NÃO ATENDE	PARCIAL	PARCIAL
RQ12	PARCIAL	NÃO ATENDE	NÃO ATENDE	NÃO ATENDE	PARCIAL
RQ13	PARCIAL	ATENDE	NÃO ATENDE	NÃO ATENDE	NÃO ATENDE
RQ14	ATENDE	ATENDE	NÃO ATENDE	NÃO ATENDE	NÃO ATENDE

Apesar de fornecer a exportação da definição do experimento como um arquivo XML. A ferramenta ExpTool não formaliza os conceitos, mas (NETO, GARCIA, *et al.*, 2014) deixa esse requisito como um trabalho futuro a ser realizado através da ontologia ExperOntology (GARCIA, HOHN, *et al.*, 2008) (RQ1).

Para a ferramenta SESE, a preparação se restringe ao envio dos arquivos que serão usados pelos participantes para download durante a execução (RQ10). Além disso, os dados coletados podem ser exportados como arquivo do banco de dados Microsoft Access 2000 (RQ11).

Experimental Manager (HOCHSTEIN, NAKAMURA, *et al.*, 2008) guia as tarefas dos participantes, porém estas tarefas dizem respeito ao desenvolvimento de um sistema concorrente, onde a próxima tarefa é inferida a partir de uma heurística própria do domínio da programação concorrente (RQ3). O pesquisador pode configurar as tarefas com artefatos a serem usados durante a execução, assim como outras variáveis de execução, porém nada é mencionada a respeito da distribuição dos tratamentos entre

os participantes e da organização da execução (RQ10). Os dados do experimento podem ser exportados em vários formatos, incluindo CSV e Excel (RQ11). Além disso, possui suporte de visualização da análise, sem a realização dos testes de hipótese (RQ12).

Para a ferramenta proposta neste trabalho, os dados de execução são exportados como arquivo csv para que possam ser importados, tratados e, possivelmente, analisados em ambientes estatísticos externos (RQ11). A ferramenta apenas fornece suporte a realização da análise de variância, após a importação dos dados tratados (em formato csv) (RQ12). No estado atual, a ferramenta aqui proposta faz o empacotamento do plano experimental, em ExpDSL, e dos dados coletados através do arquivo csv (RQ13).

8.3. Conclusões

Este trabalho observou que as pesquisas que propõem auxílio computacional à condução de experimentos, em sua maioria, não explicitam ou realizam a formalização do plano do experimento, o que pode comprometer a troca de informações entre pesquisadores e, consequentemente, dificultar a replicação e meta-análise de experimentos. Por outro lado, alguns trabalhos de pesquisa focam na formalização dos experimentos, mas não apresentam ferramentas computacionais para auxiliar a condução dos mesmos. Além disso, esses trabalhos foram avaliados apenas em relação à especificação/formalização de um experimento como prova de conceito, mas não foram avaliados de maneira a produzir evidências científicas de seus benefícios.

9. Considerações Finais e Trabalhos Futuros

Este trabalho apresentou uma linguagem específica de domínio e uma abordagem dirigida por modelos para a formalização de experimentos em engenharia de software. Este capítulo final analisa as questões de pesquisa exploradas na tese, revisa suas contribuições, discute suas limitações e apresenta direções para trabalhos futuros.

9.1. Análise das Questões de Pesquisa da Tese

Para o desenvolvimento do trabalho foram definidas questões de pesquisa, as quais motivaram a elaboração e execução de estudos com o intuito de respondê-las.

Para responder a primeira questão de pesquisa – "*Quais ferramentas vem sendo propostas para apoiar à realização de experimentos controlados em engenharia de software? Quais são os benefícios e limitações destas ferramentas?*" – foi idealizada uma revisão sistemática da literatura, a qual foi executada e apresentada no Capítulo 3. Os objetivos definidos para a revisão sistemática foram o de ter um panorama geral de como o tema vem sendo tratado pela comunidade acadêmica, e confrontar tais resultados com a experiência prática do nosso grupo de pesquisa e de grupos parceiros na condução de experimentos controlados, assim como de resultados reportados por outros grupos de pesquisa. Os resultados preliminares obtidos permitiram uma identificação de carências nos ambientes de apoio a definição e execução de experimentos controlados em ES (Capítulo 3).

A resposta para a segunda questão – "*Qual a viabilidade do uso de uma DSL (domain-specific language) para geração de workflows de atividades de participantes de um experimento seguindo seu respectivo design estatístico?*" – foi explorada através da definição de uma linguagem específica de domínio (ExpDSL) que permite a modelagem de experimentos controlados em engenharia de software (Capítulo 4). Em relação a viabilidade da geração de workflows de atividades dos participantes, foi definida e desenvolvida uma abordagem dirigida por modelos para gerar os workflows dos participantes, de acordo com o projeto do experimento, permitindo sua execução através de uma aplicação web, que permite a execução e monitoramento das atividades de cada participante do experimento, obedecendo seu projeto estatístico (Capítulo 5). Tal abordagem foi definida com base em resultados de pesquisas preliminares de definição de estratégias de monitoramento de processos de software (FREIRE, ALEIXO, *et al.*, 2011) (FREIRE, ALENCAR, *et al.*, 2012).

A resposta para a terceira questão – "*Quão completa e expressava a linguagem de formalização proposta é?*" foi respondida através da realização de um estudo empírico de modelagem de experimentos controlados existentes (Capítulo 6). Nesse estudo, 16 experimentos reais foram modelados para possibilitar a avaliação da expressividade e completude de ExpDSL. Ainda durante a realização do estudo empírico, embora tenha sido atestada a proposta, algumas limitações foram identificadas e sanadas através da proposição de melhorias na linguagem. Após a implementação das melhorias, os critérios de avaliação foram novamente analisados para atestar o impacto dessas mudanças. Além dessa avaliação, dois estudos também utilizaram a DSL proposta nessa abordagem como estudo de caso (CAMPOS, BEZERRA, *et al.*, 2012) (CAMPOS, FREIRE, *et al.*, 2013).

Para responder a quarta questão de pesquisa – "*Quais são os benefícios e limitações da abordagem proposta em relação ao uso de apenas linguagem natural para especificação de experimentos?*" – foram realizados dois experimentos controlados. O primeiro experimento para avaliar a compreensão de um plano experimental em especificado em ExpDSL em comparação com um plano especificado em linguagem natural. Foi constatado, com rigor estatístico, que há diferença na taxa de corretude para a compreensão das especificações, sendo mais baixa em ExpDSL. Além disso, foi também avaliado o tempo médio para a compreensão do plano experimental, e obteve-se como resultado que tal compreensão é mais rápida quando o plano está especificado em ExpDSL. O segundo experimento realizado foi utilizado para estimar a taxa de erro e o tempo médio para especificação de um plano em ExpDSL a partir de sua descrição em artigo. Como resultado temos o intervalo de confiança estimado entre 59% e 82% para a taxa de acertos e entre 57% e 87% para o tempo médio, em situações semelhantes ao do estudo realizado. Além disso, a pesquisa de opinião mostrou que os experimentadores tiveram impressões positivas ao avaliar a linguagem após o uso em relação à capacidade de aprendizado, facilidade de uso, produtividade, confiança e eficiência ao especificar um plano experimental, como apresentado no Capítulo 7.

Os resultados obtidos neste primeiro estudo precisam ser reforçados através da realização de novos estudos empíricos. Esses novos estudos foram realizados com a participação de usuários da linguagem.

9.2. Revisão das Contribuições

Esta tese de doutorado possui as seguintes contribuições principais:

- Uma revisão sistemática sobre ambientes computacionais de apoio a condução de experimentos controlados em ES (Capítulo 3) (FREIRE, ALENCAR, *et al.*, 2013).
- Uma linguagem específica de domínio, chamada ExpDSL, para a formalização de experimentos controlados (Capítulo 4) (FREIRE, 2013).
- Uma abordagem dirigida por modelos para customização de workflows relativos ao processo de execução de experimentos controlados (Capítulo 5) (FREIRE, ACCIOLY, *et al.*, 2013).
- Um ambiente de suporte a execução de workflows experimentais (Capítulo 5) (FREIRE, SIZÍLIO, *et al.*, 2014).
- Um estudo empírico relativo para avaliar a completude e expressividade da linguagem de especificação de experimentos proposta (Capítulo 6) (FREIRE, KULESZA, *et al.*, 2014a) (FREIRE, KULESZA, *et al.*, 2014b).
- Um experimento para análise da compreensão da linguagem proposta (Capítulo 7).
- Um experimento para avaliação da facilidade de uso da linguagem proposta (Capítulo 7).

9.3. Limitações do Trabalho

O trabalho apresentado nesta tese possui várias limitações, dentre as quais podemos destacar:

- Os projetos experimentais que a linguagem e abordagem propostas atualmente oferecem suporte são limitados a três tipos: completamente aleatorizado, aleatorizado em blocos completos e quadrado latino.
- A variedade de experimentos modelados no estudo empírico (Capítulo 6) é apenas um subconjunto de experimentos existentes do domínio da ES. Embora tenhamos realizado a modelagem de 16 experimentos, entendemos que há necessidade de ampliar para abranger um maior número de subdomínios.
- O ambiente de apoio à experimentação não foi avaliado de forma empírica ou experimental, tal como nos estudos conduzidos para a DSL. O ambiente foi objeto de um estudo de viabilidade apenas e precisa ser avaliado a partir da sua utilização durante a realização de um experimento prático (Capítulo 5).

- O contexto de realização das avaliações não abrangeu especialistas da área, o contexto de avaliação foi o de pesquisadores não especialistas (estudantes de mestrado e doutorado), que embora seja um dos perfis de usuário foco do estudo, não têm experiência ampla na área, o que pode resultar em diferentes percepções quanto a avaliação da linguagem.

9.4. Trabalhos Futuros

Existem vários trabalhos futuros que podem ser desenvolvidos como consequência das pesquisas conduzidas nesta tese de doutorado. Abaixo listamos uma série de trabalho que podem ser conduzidos nessa direção:

- Implementar um ambiente web integrado que permita a definição do experimento, assim como a aplicação das transformações propostas pela abordagem dirigida por modelos, além da geração, execução e monitoramento do experimento;
- Avaliar o ambiente de execução/monitoramento experimentalmente;
- Avaliar ExpDSL no contexto dos especialistas do domínio experimental;
- Desenvolver uma base de conhecimento através da modelagem de experimentos e famílias de experimentos;
- Realizar meta-análise dos dados experimentais (base de conhecimento) a partir de especificações de experimentos definidos em ExpDSL;
- Avaliar a aplicação da linguagem e ambiente propostos em outros domínios fora da ES;
- Fornecer uma interface gráfica baseada em formulários como alternativa à descrição textual da linguagem ExpDSL.

Referências Bibliográficas

AALST, W. M. P. V. D. et al. Workflow mining: A survey of issues and approaches. **Data & Knowledge Engineering**, 2003. 237–267.

ACCIOLY, ; BORBA, P.; BONIFÁCIO, R. Comparing Two Black-box Testing Strategies for Software Product Lines. **SBCARS VI - Brazilian Symposium on Components, Architecture and Software Reuse**, 2012.

ACCIOLY,. **Comparing Different Testing Strategies for Software Product Lines (Masters Thesis)**. UFPE. Recife, Brasil. 2012.

ACCIOLY, P.; BORBA, P.; BONIFACIO, R. Controlled Experiments Comparing Black-box Testing Strategies for Software Product Lines. **J-JUCS**, v. 20, n. 5, p. 615-639, 2014.

ACUÑA, S. T.; GÓMEZ, M.; JURISTO, N. How do personality, team processes and task characteristics relate to job satisfaction and software quality? **Information and Software Technology**, v. 51, n. 3, p. 627–639, 2009.

ALEIXO, F.,Freire, M., Câmara, W., Kulesza, U.. Automating the Variability Management, Customization and Deployment of Software Processes: A Model-Driven Approach. **Lecture Notes in Business Information Processing**, v. 73, p. 372-387, 2011. ISSN 978-3-642-19802-1.

ALEIXO, F.; FREIRE, M. A. A Comparative Study of Compositional and Annotative Modelling Approaches for Software Process Lines. **Proceedings of the 26th Brazilian Symposium on Software Engineering (SBES 2012)**, Natal, Brazil, p. 51-60, September 23-28 2012.

ALEIXO, F.; KULESZA, U.; JUNIOR, E. A. D. O. **Modeling Variabilities from Software Process Lines with Compositional and Annotative Techniques:** A Quantitative Study. International Conference of Product Focused Software Development and Process Improvement (Profes). Paphos: PROFES. 2013.

ALVES, et al. Requirements engineering for software product lines: A systematic literature review. **Information & Software Technology**, v. 52, n. 8, p. 806-820, 2010.

ANTONY, J. **Design of experiments for engineers and scientists**: Elsevier, 2003.

ARANDA, A.; DIESTE, O.; JURISTO, N. **In Search of Requirements Analyst Characteristics that Influence Requirements Elicitation Effectiveness:** a Quasi-experiment. Workshop on Managing the Influence of People and Team Factors in SE. Madrid: First Workshop on Managing the Influence of People and Team Factors in Software Engineering (INTEAMSE 2012). 2012.

ARISHOLM, E. et al. A Web-based Support Environment for Software Engineering Experiments. **Nordic Journal of Computing**, n. 9(4), 2002. 231-247.

ARISHOLM, E. et al. SESE – an Experiment Support Environment for Evaluating Software Engineering Technologies. **Nordic Workshop on Programming and Software Development Tools and Techniques**, 2002. 81-98.

BASILI, V. The Past, Present, and Future of Experimental Software Engineering. **J. Braz. Comp. Soc.**, 2006. 7-12.

BASILI, V. A Personal Perspective on the Evolution of Empirical Software Engineering. In: MUNCH, J.; SCHMID, K. **Perspectives on the Future of Software Engineering**. 1st. ed.: Springer-Verlag Berlin Heidelberg, v. XVI, 2013. Cap. Part III, p. 255-273.

BASILI, V. R. **The role of experimentation in software engineering:** past, current, and future. In Proceedings of the 18th international conference on Software engineering (ICSE '96). Washington, DC, USA. 1996. p. 442-449.

BASILI, V. R.; SELBY,. Comparing the effectiveness of software testing strategies. **IEEE Transactions on Software Engineering**, v. 13, n. 12, p. 1278–1296, 1987.

BASILI, V. R.; SHULL, F.; LANUBILE, F. Building Knowledge through Families of Experiments. **IEEE Trans. Software Engineering**, 25 (4), July/August 1999. 456-473.

BASILI, V.; CALDIERA, G.; ROMBACH, H. The Goal Question Metric Approach. In: _____ **Encyclopedia of Software Engineering**. Wiley, 1994.

BIFFL, S.; WINKLER, D. Value-Based Empirical Research Plan Evaluation. **International Symposium onEmpirical Software Engineering and Measurement**, September 2007.

BRÄUER, M.; LOCHMANN, H. **Towards Semantic Integration of Multiple Domain-Specific Languages Using Ontological Foundations.** ATEM/MoDELS. 2007.

BUDGEN, ; J. BURN, A.; KITCHENHAM,. Reporting computing projects through structured abstracts: a quasi-experiment. **Empirical Software Engineering**, v. 16, n. 2, p. 244-277, 2011.

BUDGEN, D. et al. **Lessons from Conducting a Distributed Quasi-experiment**. International Symposium on Empirical Software Engineering and Measurement. 2013.

CAMPOS, E. et al. Composição de Linguagens de Modelagem Específicas de Domínio: Um Estudo Exploratório. **III Brazilian Workshop on Model-Driven Software Development, CBSoft 2012**, p. p. 41-48, 2012.

CAMPOS, E. et al. Composition of Domain Specific Modeling Languages: An Exploratory Study. **International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2013)**, 2013.

CARTAXO, B. et al. **ESEML:** empirical software engineering modeling language. Workshop on Domain-specific modeling. New York: Workshop on Domain-specific modeling. 2012.

CHAPETTA, W. A.; SANTOS, P. S. M.; TRAVASSOS, G. H. Supporting Meta- Description Activities in Experimental Software Engineering Environments. **ESELAW'05, Brazil**, 2005.

CIOLKOWSKI, M. **An Approach for Quantitative Aggregation of Evidence from Controlled Experiments in Software Engineering**. Fraunhofer Institute for Experimental Software Engineering (IESE). Kaiserslautern. 2013.

CIRILO, et al. **Configuration Knowledge of Software Product Lines:** A Comprehensibility Study. Porto de Galinhas: Workshop on Variability & composition. 2011.

CIRILO, et al. **Configuration Knowledge of Software Product Lines:** A Comprehensibility Study. International workshop on Variability & composition. Porto de Galinhas: Workshop on Variability & composition. 2011.

CLEMENTS, P.; NORTHROP, L. **Software Product Lines:** Practices and Patterns. Professional: Addison-Wesley, 2011.

CRUZ-LEMUS, J. A. et al. Assessing the understandability of UML statechart diagrams with composite states—A family of empirical studies. **Empirical Software Engineering**, v. 14, n. 6, p. 685-719, 2009.

CZARNECKI, K.; HELSEN, S. Feature-Based Survey of Model Transformation Approaches. **IBM Systems Journal: Model-driven software development**, Riverton, v. 45, n. 3, p. 621-645, Julho 2006. ISSN 0018-8670.

DE LUCIA, A. et al. An experimental comparison of ER and UML class diagrams for data modelling. **Empirical Software Engineering**, v. 15, n. 5, p. 455–492, 2010.

DEELSTRA, S.; SINNEMA, M.; BOSCH, J. Product derivation in software product families: a case study. **Journal of Systems and Software - Special issue: The new context for software engineering education and training**, New York, v. 74, n. 2, p. 173-194, 15 Janeiro 2005.

DIAS NETO, A. C. et al. Infrastructure for SE Experiments Definition and Planning. **ESELAW'04, Brasília, Brazil**, 2004.

EASTERBROOK, S.; SINGER, J.; ST, M.-A. Selecting Empirical Methods for Software Engineering Research. In: _____ **Guide to Advanced Empirical Software Engineering**. Springer Science+Business Media, 2008.

FEIGENSPAN, et al. Do background colors improve program comprehension in the #ifdef hell? **Empirical Software Engineering**, v. 18, n. 4, p. 699-745, 2013.

FIELD, A.; MILES, J.; FIELD, Z. **Discovering Statistics Using R**.: Editora SAGE, 2012.

FREIRE, M. A. **A Model-Driven Approach to Formalize and Support Controlled Experiments in Software Engineering**. Baltimore: Doctoral Symposium on Empirical Software Engineering. 2013.

FREIRE, M. et al. **Automatic Deployment and Monitoring of Software Processes:** A Model-Driven Approach. Conference on Software Engineering and Knowledge Engineering. Miami/Florida. 2011.

FREIRE, M. et al. **Software Process Monitoring using Statistical Process Control Integrated in Workflow Systems**. Conference in Software Engineering Knowledge Engineering. San Francisco/California - USA. 2012.

FREIRE, M. et al. A Model-Driven Approach to Specifying and Monitoring Controlled Experiments in Software Engineering. In: _____ **PROFES**. Paphos, Cyprus: LNCS, v. 7983, 2013. p. 65-79.

FREIRE, M. et al. **Automated Support for Controlled Experiments in Software Engineering:** A Systematic Review. SEKE. Boston/USA. 2013.

FREIRE, M. et al. **A Process-Oriented Environment for the Execution of Software Engineering Experiments.** International Conference of Product Focused Software Development and Process Improvement. Helsinki: Springer. 2014a. p. 290-293.

FREIRE, M. et al. **An Empirical Study to Evaluate a Domain Specific Language for Formalizing Software Engineering Experiments**. International Conference on Software Engineering and Knowledge Engineering. Vancouver. 2014b (to appear). p. 250-255.

FREIRE, M. et al. Assessing and Evolving a Domain-Specific Language for Formalizing Software Engineering Experiments: An Empirical Study. **International Journal of Software Engineering and Knowledge Engineering**, v. 24, n. 10, p. 1-23, 2014.

FUCCI, D.; TURHAN,. On the role of tests in test-driven development:a differentiated and partial replication. **Empirical Software Engineering**, v. 19, n. 2, p. 277-302, 2014.

GABRIEL, P. H. D. N. **Software Languages Engineering: Experimental Evolution**. Lisboa. 2010.

GARCIA, E. R. et al. **An Ontology for Controlled Experiments on Software Engineering**. International Conference on Software Engineering & Knowledge Engineering (SEKE). San Francisco: SEKE. 2008.

GÜLESIR, G. et al. Experimental evaluation of a tool for the verification and transformation of source code in event-driven systems. **Empirical Software Engineering**, v. 14, n. 6, p. 720 - 777, 2009.

HAIR, J. F. et al. **Analise Multivariada de Dados**. Bookman, 2007.

HOCHSTEIN, L. et al. An Environment for Conducting Families of Software Engineering Experiments. **Advances in Computers**, 74, 2008. 175–200.

JEDLITSCHKA, A.; CIOLKOWSKI, M.; PFAHL, D. Reporting Experiments in Software Engineering. In: **Guide to Advanced Empirical Software Engineering**. Springer Science+Business Media, 2008.

JEDLITSCHKA, A.; PFAHL, D. **Reporting guidelines for controlled experiments in software engineering**. ISESE. 2005: 95-104.

JEDLITSCHKA, et al. **Relevant Information Sources for Successful Technology Transfer:** A Survey Using Inspections as an Example. Empirical Software Engineering and Measurement. Germany: Fraunhofer Institute. 2007. p. 31-40.

JUNG, et al. **A Controlled Experiment on Component Fault Trees**. Conference on Computer Safety, Reliability and Security (SafeComp). Toulouse. 2013.

JURISTO,. **Towards Understanding Replication of Software Engineering Experiments**. ACM / IEEE International Symposium on Empirical Software Engineering and Measurement. Baltimore: IEEE. 2013. p. 4.

JURISTO, N.; MORENO, A. M. **Basics of Software Engineering Experimentation**. Madrid: Kluwer Academic Publisher, 2010.

KAHRAMAN, ; BILGEN,. A framework for qualitative assessment of domain-specific. **Software & Systems Modeling** , 23 November 2013. 1-22.

KARAHASANOVIC, A. et al. Collecting Feedback During Software Engineering Experiments. **Empirical Software Engineering**, n. Springer Science + Business Media, 2005. 113–147.

KITCHENHAM, ; HUGHES, R. T.; LINKMAN, S. G. Modeling Software Measurement. **IEEE Transactions on Software Engineering**, v. 27, n. 9, p. 788–804, 2001.

KITCHENHAM, B. A.; DYBA, T.; JORGENSEN, M. **Evidence-based software engineering**. 26th International Conference on Software Engineering. 2004. p. 273-281.

KITCHENHAM, B. et al. Systematic literature reviews in software engineering - A tertiary study. **Inf. Softw. Technol.**, August 2010. 792-805.

KITCHENHAM, B.; CHARTERS, S. **Guidelines for performing Systematic Literature Reviews in Software Engineering**. Keele University and Durham University, Tech. Rep. 2007. (EBSE 2007-001).

KITCHENHAMA, B. et al. Systematic literature reviews in software engineering – a systematic literature review. **Information and Software Technology**, 2009. 7–15.

KO, A. J.; LATOZA, T. D.; BURNET, M. M. A practical guide to controlled experiments of software engineering tools with human participants. **Empirical Software Engineering**, February 2015. 110-141.

KOSAR, ; MERNIK, M.; CARVER, J. C. Program comprehension of domain-specific and general-purpose languages: comparison using a family of experiments. **Empirical Software Engineering**, v. 17, n. 3, p. 276-304, 2012.

LEROY, G. **Designing User Studies in Informatics**. Springer, 2011.

LIKERT, R. A technique for the measurement of attitudes. In: WOODWORTH, R. S. **Archives of Psychology**. New York: Columbia University Press, v. 22, 1932. p. 5-55.

MENDONÇA, M. G. et al. A Framework for Software Engineering Experimental Replications. **IEEE International Conference on Engineering of Complex Computer Systems**, 2008.

MENS, ; TOURWÉ, T. A survey of software refactoring. **, IEEE Transactions on Software Engineerin**, 2004. 126 - 139.

MIAN, P. G. et al. Towards a Computerized Infrastructure for Managing. **Jornadas Iberoamericanas em Ingeniería del Software e Ingeniería del Conocimiento**, 2004.

MIAN, P. G.; TRAVASSOS, G. H.; ROCHA, A. R. C. A computerized infrastructure for supporting experimentation in software engineering. **Experimental Software Engineering Latin American Workshop**, 2005.

MIAN, P.; TRAVASSOS, G.; ROCHA, A. R. C. eSEE: a Computerized Infrastructure for Experimental Software Engineering. **Experimental Software Engineering Latin American Workshop (ESELAW'04)**, 2004.

MOODY, D. L. **The Method Evaluation Model:** A Theoretical. European Conference on Information Systems. Naples. 2003.

NETO, J. P. et al. **ExpTool:** a Tool to Conduct, Package and Replicate Controlled Experiments in Software Engineering. International Conference in Software Enginnering Research and Practice (SERP). Las Vegas: WORLDCOMP. 2014. p. 377-383.

PANACH, J. I. et al. Early Usability Measurement in Model-Driven Development: Definition and Empirical Evaluation. **Int. J. Soft. Eng. Knowl.**, v. 21, n. 3, p. 339, 2011.

PERRY, D. E.; SIM, S. E.; M. , S. **Case Studies for Software Engineers. IEEE Computer Society**. 26th International Conference on Software Engineering. Washington, DC, USA:. 2004. p. 736-738.

PFLEEGER, S. L. Experimental design and analysis in software engineering: Part 2: how to set up and experiment. In: _____ **SIGSOFT Softw. Eng. Notes**. v. 20, 1995. p. 22-26.

PFLEEGER, S. L. Experimental Design and Analysis in Software Engineering: Types of Experimental Design. In: _____ **SIGSOFT Softw. Eng. Notes**. v. 20, 1995. p. 14-16.

REDWINE, S.; RIDDLE, W. **Software Technology Maturation**. 8th International Conference on Software Engineering. [S.l.]: IEEE Computer Society Press. 1985. p. 189-200.

RIBEIRO, M.; BORBA, P.; KÄSTNER, C. **Feature Maintenance with Emergent Interfaces**. International Conference on Software Engineering. New York. 2014.

RYAN, T. **Estatística Moderna Para Engenharia**. Elsevier Brasil, 2011.

SCATALON, L. P.; GARCIA, & CORREIA, R. E.; M, R. C. **Packaging Controlled Experiments Using an Evolutionary Approach Based on Ontology**. International Conference on Software Engineering and Knowledge Engineering (SEKE). Miami. 2011.

SHULL, F. et al. The role of replications in Empirical Software Engineering. **Empirical Software Engineering**, v. 13, n. 2, p. 211-218, April 2008. ISSN 1573-7616.

SIY, H.; WU,. **An Ontology to Support Empirical Studies in Software Engineering**. International Conference on Computing, Engineering and Information. Fullerton: International Conference on Computing, Engineering and Information. 2009.

SJøBERG, D. I. K. et al. Conducting Realistic Experiments in Software Engineering. **International Symposium on Empirical Software Engineering** , 2002.

SJøEBERG, D. I. K. et al. A survey of controlled experiments in software engineering. **IEEE Transactions on Software Engineering**, 31 , Issue: 9, Sept. 2005. 733 - 753.

SOLARI,. **Identifying Experimental Incidents in Software Engineering Replications**. [S.l.]: IEEE International Symposium on Empirical Software Engineering and Measurement. 2013.

STAHL, T.; VOLTER, M. **Model-Driven Software Development**. Heidelberg, Germany: John Wiley & Sons, Ltd., 2005.

STOLEE, K. T.; ELBAUM, S. Exploring the use of crowdsourcing to support empirical studies in software engineering. **2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement**, 2010.

TORII, K. et al. Ginger2: An Environment for Computer-Aided Empirical Software Engineering. **IEEE Transactions on Software Engineering**, (25)4, July 1999. 474–492.

TRAVASSOS, G. H. et al. An environment to support large scale experimentation in software engineering. **13th IEEE International Conference on Engineering of Complex Computer Systems**, 2008. 193-202.

WOHLIN, C. et al. **Experimentation in Software Engineering:** An Introduction. Boston/Dordrecht/London: Kluwer Academic Publishers, 2012.

Apêndice I: Artefatos Utilizados no Desenvolvimento Da Abordagem Proposta

Esta seção apresenta os artefatos necessários para o desenvolvimento da abordagem dirigida por modelo.

I.A. Gramática Exp*DSL*

A Figura abaixo apresenta a representação da gramática ExpDSL utilizando o xText.

```
grammar br.ufrn.dimap.ase.dsl.Expdslv3 with
org.eclipse.xtext.common.Terminals

generate expdslv3 "http://www.ufrn.br/dimap/ase/dsl/Expdslv3"

Model:
 elements+=ExperimentElement;
ExperimentElement:
 'Experiment' name=ID description=STRING
 'Experimental Design' experiments=ExperimentalDesign
 process+=Process*
 ('Questionnaires' questionnaire+=Questionnaire*)?
 ;
QualifiedName:ID ('.' ID)*;

ExperimentalDesign:
('Authorship' '{' author+=Author* '}')?
('Abstract' '{'abstract=Abstract '}')?
('Keywords' '{' keyword+=Keyword* '}')?
('Goals' '{'(goal+=Goal)*'}')
('Research Questions' '{' researchQuestion+=ResearchQuestion*'}')?
('Research Hypotheses' '{' researchHypothesis+=ResearchHypothesis* '}')?
('Dependent Variables' '{' depVar+=DepVariable*'}')
('Factors' '{' factor+=Factor* '}')
('DoE' '=' type=DesignType '(' (doe=DOE)')')
('Context'  '{'(parameter+=Parameter)*'}')?
('Internal Replication' replication=INT)?
('Threats to Validity' '{' threats+= Threats*'}')?;

Author: identifier=ID name=STRING;

Abstract: (SimpleAbstract|StructuredAbstract);

SimpleAbstract : description=STRING;
StructuredAbstract: {StructuredAbstract}
 ('Background' background=STRING)?
 ('Objective' objective=STRING)?
 ('Methods' methods=STRING)?
 ('Results' results=STRING)?
 ('Limitations' limitations=STRING)?
 ('Conclusions' conclusions=STRING)?;
Keyword: description=STRING;

Goal: StructuredGoal|SimpleGoal;
```

```
SimpleGoal: name=ID (description=STRING)?;
StructuredGoal: name=ID 'Analyze' object=STRING  'for the purpose of'
technique=STRING 'with respect to their' quality=STRING 'the point of view of
the' ptView=STRING 'in the context of' contextOf=STRING;

ResearchQuestion: name=ID description=STRING 'relates to'
goal=[Goal|QualifiedName] ;
ResearchHypothesis:name=ID description=STRING 'relates to'
goal=[Goal|QualifiedName];
Range: name=ID;

enum ScaleType returns ScaleType:
 Absolute = 'Absolute'| Ratio = 'Ratio' |Interval = 'Interval'| Ordinal =
'Ordinal'| Nominal = 'Nominal';

DepVariable : (name=ID description=STRING ('Scale' scaleType=ScaleType)?
('Range' '{'range+=Range*'}')?) 'relates to'
 ('RQ' rq+=[ResearchQuestion|QualifiedName]*|'RH'
rh+=[ResearchHypothesis|QualifiedName]*);

Factor: name=ID description=STRING ('Scale' scaleType=ScaleType)? ('Range'
'{'range+=Range*'}')?;

DOE: LS|RCBD|CRB|OTHER;
LS: {LS} 'column =' col=([Factor|QualifiedName])',' 'row =' row=(
[Factor|QualifiedName]) ',' 'treatment =' treatment=(
[Factor|QualifiedName]);
RCBD: {RCBD}  'blockVariable =' blockVariable=( [Factor|QualifiedName]) ','
'treatment =' treatment=( [Factor|QualifiedName]);
CRB: {CRB} 'treatment =' treatment=([Factor|QualifiedName]);
OTHER:{OTHER} name=STRING;

Parameter: name=ID description=STRING ('Scale' scaleType=ScaleType)? ('Range'
'{'range+=Range*'}')?;

enum DesignType returns DesignType:
 CDR = 'CRD - Completely Randomized Design' |
 RCBD = 'RCBD - Randomized Complete Block Design' |
 LS = 'LS - Latin Square' |
 other = 'Other';

//Data procedure definitions
Process:
 'Process' name=ID ('to' treatment+=[Range|QualifiedName]*)?'{'
   ('Role' role += RoleType*)?
  (task+=Task)*
 '}';

Task:
 'Task'name=ID 'description' description=STRING
 ('to' next+=[Task|QualifiedName]*)?
 ('Role' role += RoleType*)? '{'
 ('artefacts' (artefacts+=Artefact)*)?
 ('quest' (questionnaire+=[Questionnaire|QualifiedName]*))?
 ('field'(fields+=Field)*)?
  '}';

Artefact:
 name = ID 'description' description=STRING 'type' type=ArtefactType;
```

```
enum ArtefactType returns ArtefactType:
in_ = 'input' | out_ = 'output' | inout_ = 'inoutput';

Field: variable=ID textField=STRING;

enum RoleType returns RoleType:
participant = 'Participant'| researcher = 'Researcher';


Threats: name= ID 'description' description=STRING 'type'  type=ThreatType
('control action(s)' CA= STRING)?;

enum ThreatType returns ThreatType:
iv = 'Internal' | ev = 'External' | c = 'Construction' | r = 'Reliability'|
cl='Conclusion';

//Questionnaire definitions
Questionnaire: 'Questionnaire' name=ID ( 'relates' relatesTo+=[Process]*)?
    ('type' questionnaireType=QuestionnaireType)? '{'
  (question+=Question)*'}';
enum QuestionnaireType returns QuestionnaireType:
pre = 'Pre' | post = 'Post';

Question:
name=ID '{'
    ('description' description=STRING)
 ('type' type=AnswerType)
 ('required' req=INT )?
 ( "{"alternatives+=Alternatives*"}")?
'}';
Alternatives:
description = STRING;

enum AnswerType returns AnswerType:
Text='Text' | ParagraphText = 'ParagraphText' | MultipleChoice =
'MultipleChoice'
| SingleChoice = 'SingleChoice' | Scale = 'Scale' | Grid = 'Grid';
```

I.B Transformações M2M e M2T

Esta seção apresenta fragmentos dos códigos responsáveis pelas transformações modelo para modelo e modelo para texto. A transformação M2M, implementada em QVTo é apresentada na listagem abaixo.

```
//Models
modeltype DSL  uses expdslv3('http://www.ufrn.br/dimap/ase/dsl/Expdslv3');
modeltype JPDL uses jpdl31Plus('urn:jbpm.org:jpdl-3.1Plus');

transformation ExpDSLv3ToWorkExpTese(in dsl : DSL, out outModel: JPDL);

main() {

    log('starting transformation...');
```

```
    dsl.rootObjects()[DSL::Model]->map model2model();
 log('ending transformation...');
}

mapping DSL::Model::model2model():JPDL::Model{
log('elements...');
 elements := self.elements->map model2DSLModel()->asOrderedSet();
}

mapping DSL::ExperimentElement::model2DSLModel():JPDL::DocumentRoot {

  log('process...');
  processDefinition := self.process->map
processToProcessDefinition(self.experiments.replication);
  log('questionnaires...');
  questionnaires := self.questionnaire -> map questToQuestionnaire(result);
  log('plan...');
  plan := self.experiments -> map experimentalPlanToPlan(result);

}


mapping DSL::ExperimentalDesign::experimentalPlanToPlan(in res:
JPDL::DocumentRoot):JPDL::ExperimentalPlan {
 log('goals...');
    goals := self.goal->map goalsToGoal()->asOrderedSet();
 log('research questions...');
 researchQuestions := self.researchQuestion->map RQtoRQ()->asOrderedSet();
 log('hypothesis...');
 hypothesis := self.researchHypothesis-> map RHtoRH()->asOrderedSet();
 log('dependent variables...');
 depVariables := self.depVar -> map depVarToDepVar()->asOrderedSet();
 log('factors...');
 factors := self.factor -> map factorToFactor()->asOrderedSet();
 log('context...');
 context := self.parameter -> map parameterToParameter()->asOrderedSet();
 log('design...');
 //DESIGN
 design := map doeToDesign(self.doe);
}

mapping doeToDesign(in doe: DSL::DOE): JPDL::Design{
var design: JPDL::Design;

 if(doe.oclIsTypeOf(DSL::LS))then {
 log('LS');
 result := object JPDL::LS {
   treatment := getFactor(doe.oclAsType(DSL::LS).treatment.name);
   row := getFactor(doe.oclAsType(DSL::LS).row.name);
   col := getFactor(doe.oclAsType(DSL::LS).col.name);
};}
 else {
 if(doe.oclIsTypeOf(DSL::RCBD))then {
  log('RCBD');
  result := object JPDL::RCBD {
  treatment := getFactor(doe.oclAsType(DSL::RCBD).treatment.name);
  blockVariable:= getFactor(doe.oclAsType(DSL::RCBD).blockVariable.name);
 };
 } else {
```

```
 if(doe.oclIsTypeOf(DSL::CRB))then {
  log('CRD');
  result := object JPDL::CRD {
   treatment := getFactor(doe.oclAsType(DSL::CRB).treatment.name);
 };
 }
 else {
 if(doe.oclIsTypeOf(DSL::OTHER))then {
  log('OTHER');
  result := object JPDL::Other {
   name := doe.oclAsType(OTHER).name;
 };}endif;
}endif;
} endif;
}endif;
//result := design;
}
//Map ExpDSLv3 StructuredGoal or SimpleGoal to JPDL::Goal
mapping DSL::Goal::goalsToGoal():JPDL::Goal {

 if(self.oclIsTypeOf(StructuredGoal))then {
  var g:= self.oclAsType(StructuredGoal);
  description := 'Analyze' + g._object +  ' for the purpose of ' + g.technique
+ ' with respect to their ' + g.quality + ' the point of view of the ' +
g.ptView +  ' in the context of ' + g.contextOf;
  id := g.name;
 }else {
 if(self.oclIsTypeOf(SimpleGoal)) then {

     var g:= self.oclAsType(SimpleGoal);
  id := g.name;
  description := g.description;

 }endif;
 }endif;
}


mapping DSL::ResearchQuestion::RQtoRQ():JPDL::ResearchQuestion {
 description := self.description;
 name := self.name;

 var allGoals := outModel.objectsOfType(JPDL::Goal);
   allGoals->forEach(go) {
   if(go.oclAsType(JPDL::Goal).id.equalsIgnoreCase(self.goal.name))then {
          log('goal:'+ self.goal.name);
      fromGoal := go;
 }endif;
   };
}

mapping DSL::ResearchHypothesis::RHtoRH():JPDL::ResearchHyphotesis {
 description := self.description;
 id := self.name;
 var allGoals := outModel.allInstances(JPDL::Goal);
   allGoals->forEach(go) {
   if(go.oclAsType(JPDL::Goal).id.equalsIgnoreCase(self.goal.name))then {
          log('goal:'+ self.goal.name);
      fromGoal := go;
```

```
 }endif;
  };

}
mapping DSL::DepVariable::depVarToDepVar(): JPDL::DependentVariable {
     name := self.name;
         description := self.description;
         scale := getScale(self.scaleType);
         if(self.scaleType = DSL::ScaleType::Nominal or self.scaleType =
DSL::ScaleType::Ordinal) then{
  range := self.range->map rangeToRange();
 }endif;
 if ( (self.rh) <> null ) then{
  self.rh->forEach(h){
   relatesToRH += Sequence{getRH(h)};
  };
         }endif;
       if ( (self.rq) <> null ) then{
  self.rq->forEach(q){
   relatesToRQ += Sequence{getRQ(q)};
  };
         }endif;
         }

mapping DSL::Factor::factorToFactor(): JPDL::Factor{
 name := self.name;
 description := self.description;
 if(self.scaleType = DSL::ScaleType::Nominal or self.scaleType =
DSL::ScaleType::Ordinal) then{
  range := self.range->map rangeToRange();
 }endif;
 scale := getScale(self.scaleType);
}

mapping DSL::Parameter::parameterToParameter(): JPDL::Parameter{
    name := self.name;
    description := self.description;
    scale := getScale(self.scaleType);
    if(self.scaleType = DSL::ScaleType::Nominal or self.scaleType =
DSL::ScaleType::Ordinal) then{
  range := self.range->map rangeToRange();
 }endif;
}
mapping DSL::Range::rangeToRange(): JPDL::Range{
 name := self.name;
}
mapping DSL::ResearchQuestion::relatesToRQtoRQ(): JPDL::ResearchQuestion {
  result := getRQ(self);
}

mapping DSL::ResearchHypothesis::relatesToRHtoRH(): JPDL::ResearchHyphotesis
{
  result := getRH(self);
}


mapping DSL::Questionnaire::questToQuestionnaire(in res: JPDL::DocumentRoot):
JPDL::Questionnaire {
 name := self.name;
```

```
question := self.question->map questionToQuestion();
if(type<>null) then {
 type := getQuestionnaireType(self.questionnaireType);
}endif;

self.relatesTo->forEach(p){
   res.allSubobjectsOfType(JPDL::ProcessDefinitionType)->forEach(pro){

if(pro.oclAsType(JPDL::ProcessDefinitionType).name.equalsIgnoreCase(p.name))
then{
     result.processes +=
Sequence{pro.oclAsType(JPDL::ProcessDefinitionType)};
    }endif;
   };
};
}

mapping DSL::Question::questionToQuestion():JPDL::Question{
description := self.description;
type := getQuestionType(self.type);
option := self.alternatives->alternativesToAlternatives();
if (self.req = 1) then {
    required := true.oclAsType(BooleanType);
}else {
 required := false.oclAsType(BooleanType);
}endif;
}

mapping DSL::Alternatives::alternativesToAlternatives(): JPDL::Alternative {
description := self.description;
}

mapping DSL::Process::processToProcessDefinition(in num: Real)
:JPDL::ProcessDefinitionType{

  //cria o nó inicial e final do processo
  result.startState := object JPDL::StartStateType {
  name := 'Starting';

  var act: Set(Task);
  act := self.allInstances(DSL::Task);

  act->forEach(a) {
   var comp := a.repr();
   var res := isSuccessor(comp, self.task);
//if the activity has no other refering to her as next, it is a first
activity...
   if(res.repr().equalsIgnoreCase('false')) then {
//create a transition from start to this activity...
 var start_t := object JPDL::TransitionType{
    name := 'startTransition'+a.name;
    to := a.name;
   };

           transition += Sequence{start_t}

   }endif;
  }
  };
```

```
  result.endState := object JPDL::EndStateType {
  name := 'End';
  };

  // cria swimlane com o papel determinado para a primeira atividade, caso
exista.
  result.swimlane := object JPDL::SwimlaneType {

   assignment := object JPDL::AssignmentType {
  actorId := getProcessRole(self);
 };

  };
  name := self.name;
  quantity := num.round();
  log('taskNodes...');
  taskNode := self.task->map task2taskNode(self)->asOrderedSet();

  //links
    self.allInstances(DSL::Task)-> map ligarAoNoFinal(result)-
>asOrderedSet();
}

mapping DSL::Task::task2taskNode(in pro:DSL::Process): JPDL::TaskNodeType{
 name := self.name;
 description := self.description;
 log('artefacts...');
 artefacts := self.artefacts->map artefact2artefact()->asOrderedSet();
 log('fields...');
 fields := self.fields -> map fieldToField()->asOrderedSet();
 log('questionnaires relation...');
 questionnaires := self.questionnaire -> map questToQuest()->asOrderedSet();

 // tratar última tarefa
 self.next->forEach(i){
  transition += object JPDL::TransitionType{
   name := self.name+'_Transition';
   to := i.name;
  }//end_create_transactions
 };//end forEach next
}

mapping DSL::Questionnaire::questToQuest():JPDL::Questionnaire {
 name := self.name;
}
mapping DSL::Artefact::artefact2artefact():JPDL::Artefact{
 name := self.name;
 description := self.description;

 if(self.type = DSL::ArtefactType::in_) then {
  result.type := JPDL::ArtefactType::input.repr();

 }endif;

 if (self.type = DSL::ArtefactType::out_)then {

  result.type:= JPDL::ArtefactType::output.repr();
 }endif;
```

```
 if (self.type = DSL::ArtefactType::inout_)then {

  result.type:= JPDL::ArtefactType::inoutput.repr();
 }endif;
 }

mapping DSL::Field::fieldToField():JPDL::Field{
 name := self.variable;
}
//Create links to ENDNODE
mapping DSL::Task::ligarAoNoFinal(inout process:
JPDL::ProcessDefinitionType){

 // cria transições para o fim...
 var i := getTransitionToFim(process);

    if(i>1)then{

 var join := object JPDL::JoinType {
  name := 'joinEnd';
  transition := object JPDL::TransitionType{
   name := 'to_EndTransition';
   to:= 'End';
  };
 };
 process.join += Sequence{join};
 process.allInstances(JPDL::TaskNodeType)->forEach(task) {
 if(task.transition->isEmpty())then{
  var trst := object JPDL::TransitionType {
  name := task.name + 'transition';
  to := 'joinEnd';
 };
 task.transition += trst;
 }endif;
 };

 }else {

 process.allInstances(JPDL::TaskNodeType)->forEach(task) {
  if(task.transition->isEmpty())then{
   var trst := object JPDL::TransitionType {
   name := task.name + 'transition';
   to := 'End';
  };
  task.transition += trst;
  }endif;
 };
 }endif;

}

/************
 *  QUERYS  *
 ************/

query getRQ(in rq: DSL::ResearchQuestion): JPDL::ResearchQuestion {

 var allRQs := outModel.objectsOfType(JPDL::ResearchQuestion);
```

```
  allRQs->forEach(res) {
 if(res.oclAsType(JPDL::ResearchQuestion).name.equalsIgnoreCase(rq.name))then
{
  return res;
 }endif;
 };
 return null;
}


query getRH(in rh: DSL::ResearchHypothesis): JPDL::ResearchHyphotesis {

 var allRHs := outModel.objectsOfType(JPDL::ResearchHyphotesis);
  allRHs->forEach(res) {
 if(res.oclAsType(JPDL::ResearchHyphotesis).id.equalsIgnoreCase(rh.name))then
{
  return res;
 }endif;
 };
 return null;
}


query getFactor(in name:String): JPDL::Factor {
 var allFactors := outModel.objectsOfType(JPDL::Factor);
 allFactors->forEach(f){
 if(f.oclAsType(JPDL::Factor).name.equalsIgnoreCase(name)) then {
  return f;
 }endif;
 };
 return null;
}

query getGoal(in g:DSL::Goal): JPDL::Goal {
  var allGoals := outModel.objectsOfType(JPDL::Goal);
  allGoals->forEach(go) {
  if(go.oclAsType(JPDL::Goal).id.equalsIgnoreCase(g.name))then {
     return go;
 }endif;
  };
  return null;
}


 query getTransitionToFim(in process: JPDL::ProcessDefinitionType): Integer{
    var i := 0;
    process.allInstances(JPDL::TaskNodeType)->forEach(task) {
  if(task.transition->isEmpty())then{
  i:= i +1;

  }endif;
 };
    return i;

 }

//check if activity is between begin and end
query checkBetween(in begin: String, in end_: String, in atual :String, in
pro:DSL::Process):Boolean{
```

```
 if( atual.equalsIgnoreCase(begin) or atual.equalsIgnoreCase(end_))then {
 return true;
 }else {
 var atividadeAtual : Task;

 atividadeAtual = getTask(atual,pro);
 if(atividadeAtual.next != null) then{
    atividadeAtual.next->forEach(act){
            checkBetween(begin,end_,act.name,pro);
      };
      }endif;

 }endif;
return false;
}

//QUERY QUE RETORNA A ATIVIDADE A PARTIR DO NOME
query getTask(in name: String, in pro:DSL::Process):DSL::Task {
 var activities : Set(DSL::Task);
 activities := pro.allInstances(Task);
  activities->forEach(i){

  var exp := name.equalsIgnoreCase(i.oclAsType(Task).name);
   if (exp) then {
    return i;
   }endif;
  };
}

query getProcessRole(in pro: DSL::Process): String {
//alwasy return the role from the first process activity or task ...
 var act := pro.task;
 var role:= 'unknown';
 return pro.role->at(1).repr();
}

query isSuccessor(in rep: String,in act: Set(Task)): Boolean {

 var x:=0;
 act->forEach(i){
     i.next->forEach(j){
  if(j.repr().equalsIgnoreCase(rep))then{
  x:= x +1;
 }endif;
 }
 };
 if(x>0)then{
 return true;
 } else {
 return false;
 }endif;

}

query getScale(in type: DSL::ScaleType) : JPDL::ScaleType {
 var res: JPDL::ScaleType;

    if(type.=(DSL::ScaleType::Absolute)) then {
```

```
 res := JPDL::ScaleType::Integer;
 }
 else {
 if(type.= (DSL::ScaleType::Ratio)) then {
 res := JPDL::ScaleType::Real;
 } else {
 if(type.= (DSL::ScaleType::Ordinal)) then {
 res := JPDL::ScaleType::Ordinal;
 } else {
 if(type.= (DSL::ScaleType::Nominal)) then {
 res := JPDL::ScaleType::Nominal;
 }else {
 if(type.= (DSL::ScaleType::Interval)) then {
 res := JPDL::ScaleType::Interval;
 }endif;
 }endif;
 }endif;
 }endif;
 }endif;

 return res;
}
//
query getQuestionType(in type : DSL::AnswerType) : JPDL::AnswerType {
 var res: JPDL::AnswerType;

    if(type.=(DSL::AnswerType::SingleChoice)) then {
 res := JPDL::AnswerType::comboBox;    }
 else {
 if(type.= (DSL::AnswerType::ParagraphText)) then {
 res := JPDL::AnswerType::paragraphText;
 } else {
 if(type.=(DSL::AnswerType::Text)) then {
 res := JPDL::AnswerType::text;
 } else {
 if(type.=(DSL::AnswerType::MultipleChoice)) then {
 res := JPDL::AnswerType::checkBox;
 }endif;
 }endif;
 }endif;
 }endif;

 return res;
}

query getDesign(in type : DSL::DesignType) : JPDL::DoEType {
 var res: JPDL::DoEType;
 if( type.=(DSL::DesignType::CDR))then {
 res := JPDL::DoEType::CRD;
 } else {
 if( type.=(DSL::DesignType::RCBD)) then {
 res := JPDL::DoEType::RCBD;
 } else {
 if( type.=(DSL::DesignType::LS)) then {
 res := JPDL::DoEType::LS;
 } else {
 if( type.=(DSL::DesignType::other)) then {
 res := JPDL::DoEType::other;
 }endif;
```

```
 }endif;
 }endif;
 }endif;
 return res;
}


query getQuestionnaireType(in type: DSL::QuestionnaireType):
JPDL::QuestionnaireType {

 if(type.=(QuestionnaireType::pre))then {
  return JPDL::QuestionnaireType::Pre;
 } else {
  return JPDL::QuestionnaireType::Post;

 }endif;

 return null;
}



query getBooleanValue(in type: String): JPDL::BooleanType {
 if(type.equalsIgnoreCase("true")) then {
  return true.oclAsType(BooleanType);
 } else{
  return false.oclAsType(BooleanType);
 }endif;
 return null;
}


query getDesign(in doe: DSL::DOE):JPDL::Design {
 var res: JPDL::Design;

 if(doe.oclIsTypeOf(DSL::LS))then {
  log('LS');
  res.oclAsType(JPDL::LS).treatment :=
getFactor(doe.oclAsType(DSL::LS).treatment.name);
  res.oclAsType(JPDL::LS).row := getFactor(doe.oclAsType(DSL::LS).row.name);
  res.oclAsType(JPDL::LS).col := getFactor(doe.oclAsType(DSL::LS).col.name);
 }
 else {
  if(doe.oclIsTypeOf(DSL::RCBD))then {
   log('RCBD');
   res.oclAsType(JPDL::RCBD).treatment :=
getFactor(doe.oclAsType(DSL::RCBD).treatment.name);
   res.oclAsType(JPDL::RCBD).blockVariable:=
getFactor(doe.oclAsType(DSL::RCBD).blockVariable.name);

  } else {
  if(doe.oclIsTypeOf(DSL::CRB))then {//<===ALTERAR NA DSL====
   log('CRD');
   res.oclAsType(JPDL::CRD).treatment :=
getFactor(doe.oclAsType(DSL::CRB).treatment.name);
  }
  else {
  if(doe.oclIsTypeOf(DSL::OTHER))then {
   log('OTHER');
```

```
  res.oclAsType(JPDL::Other).name := doe.oclAsType(OTHER).name;
 }endif;
}endif;
} endif;
}endif;
return res;
}
```

A transformação M2T implementada em Acceleo listada abaixo.

```
[module generate('http://www.ufrn.br/dimap/ase/dsl/Expdslv3')]


[template public generateElement(exp: ExperimentElement)]

[comment @main/]
[file (exp.name.concat('.conf').toLower().trim(),true,'UTF-8')]
<?xml version="1.0" encoding="UTF-8"?>
<experimental_plan type="[(exp.experiments.type)/]"
internalReplication="[(exp.experiments.replication)/]">


[let maxValue : Integer = exp.experiments.replication]

[if (exp.experiments.type = DesignType::CDR)]
[for (p: Process| exp.process)]
 [for (t: Range| p.treatment )]
  [for (counter : Integer | Sequence{1..maxValue})]
   [let selectedSubject : Integer = randomRange(1, maxValue*p.treatment-
>size())]
  <process name="[p.name/]" subject="[selectedSubject/]" />
   [/let]
  [/for]
 [/for]
[/for]
[elseif (exp.experiments.type=DesignType::RCBD)]
[for (p: Process| exp.process)]
 [for (it : Range| p.treatment )]
  [for (counter : Integer | Sequence{1..maxValue})]
   [let selectedSubject : Integer = randomRange(1, (maxValue*p.treatment-
>size()))]
    [let block : String =
it.eContainer(DOE).oclAsType(RCBD).blockVariable.name]
  <process name="[p.name/]" subject="[selectedSubject/]" block="[ block /]" />
    [/let]
   [/let]
  [/for]
 [/for]
[/for]
[elseif (exp.experiments.type = DesignType::LS)]

 <ls treatment="[exp.experiments.doe.oclAsType(LS).treatment.name/]"
col="[exp.experiments.doe.oclAsType(LS).col.name/]"
row="[exp.experiments.doe.oclAsType(LS).row.name/]" />


[for (f: Factor|exp.experiments.factor)]
   [for (range: Range|f.range)]
  <factor  name="[f.name/]" level="[range.name/]" />
   [/for]
 [/for]



 [for (p: Process| exp.process)]
  [for (it : Range| p.treatment )]
   <link name="[p.name/]"
```

```
  treatment="[it.eContainer(Factor).name/].[it.name/]," />
 [/for]
[/for]


[/if]
[/let]

</experimental_plan>

[/file]
[/template]

[query public randomRange(initialValue :Integer, maxValue :Integer) : Integer
= invoke('AcceleoExpDSL.main.RandomSubject','randomize(java.lang.Integer,
java.lang.Integer)', Sequence{initialValue, maxValue}) /]
```

I.C Metamodelo de Workflow

A Figura a seguir apresenta um fragmento do metamodelo Ecore JPDLPlus adaptado e gerado a partir do JPDL *Schema*. Os novos elementos estão destacados na figura. Este metamodelo é o tipo base de saída (destino) das transformações M2M na abordagem proposta.

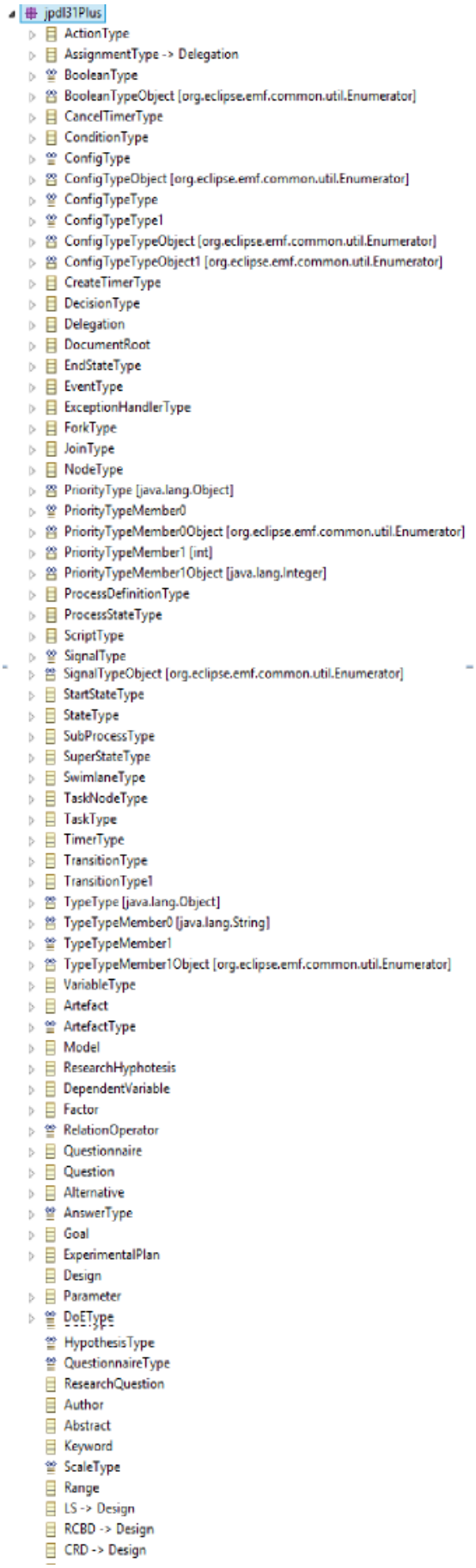


Figura 53: Metamodelo JPDL

Apêndice II: Análise de Resíduos para o experimento #1 do Capítulo 7.

Para verificar a adequação da análise de variância, foram analisados os resíduos com relação aos pressupostos de Normalidade, Independência e Homocedasticidade (HAIR, TATHAM, *et al.*, 2007). Os resíduos são obtidos por meio da diferença entre as observações individuais e a média do tratamento correspondente. A Figura 54 apresenta os gráficos utilizados para a análise dos resíduos para o tempo total de compreensão do plano experimental.

A validade do pressuposto de Normalidade pode ser verificada por meio do gráfico de probabilidade normal para os resíduos (HAIR, TATHAM, *et al.*, 2007). Nesse gráfico, os resíduos são representados em função do seu valor esperado, o qual é calculado supondo que os resíduos seguem uma distribuição normal. Considera-se válida a suposição de Normalidade se os pontos do gráfico estiverem dispostos na forma de uma linha reta. Na visualização dessa reta, devem ser enfatizados os valores centrais do gráfico e não dos extremos (HAIR, TATHAM, *et al.*, 2007). A validade da suposição de Independência foi verificada por meio do gráfico "Versus Fit"; no qual, os resíduos devem estar distribuídos em torno de uma faixa horizontal centrada em 0 (zero) (HAIR, TATHAM, *et al.*, 2007). A distribuição dos resíduos no gráfico indica a validade da suposição de Independência para o tempo total de compreensão. A validade da suposição de Homocedasticidade pode ser verificada por meio do gráfico de "Versus Order"; no qual, as abordagens envolvidas devem demonstrar variâncias semelhantes, com a média de resíduos próxima de zero (HAIR, TATHAM, *et al.*, 2007). A indicação de validade da suposição de Homocedasticidade foi verificada por meio do gráfico. Por fim, tal análise, viabiliza a conclusão, de base estatística, para os resultados apresentados pela ANOVA.

Além disso, a Figura 56 exibe a equação de regressão, onde podemos destacar o coeficiente associado ao tratamento, e temos uma diminuição de 14,25 minutos para ExpDSL.

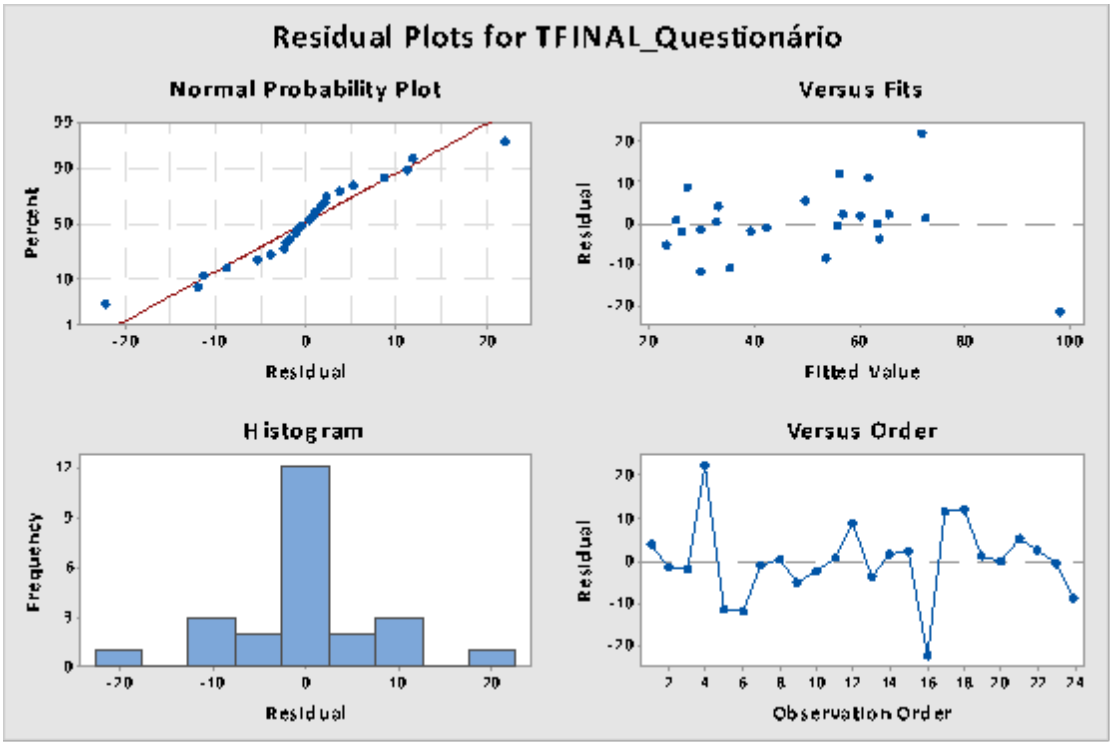


Figura 54 Gráficos de análise de resíduos para o tempo total de compreensão do plano

```
TempoTotal = 48,92 - 0,42 Replica_R1 + 1,33 Replica_R2 - 6,67 Replica_R3
- 2,42 Replica_R4 - 6,17 Replica_R5 + 14,33 Replica_R6
- 14,25 Tratamento_ExpDSL + 14,25 Tratamento_LN - 1,08 EXP_CFT
+ 1,08 EXP_LPS + 0,00 Participante(Replica)_P1(R1) -
 0,00 Participante(Replica)_P2(R1) - 7,25 Participante(Replica)_P3(R2)
+ 7,25 Participante(Replica)_P4(R2) + 5,75 Participante(Replica)_P5(R3)
- 5,75 Participante(Replica)_P6(R3) + 6,00 Participante(Replica)_P7(R4)
- 6,00 Participante(Replica)_P8(R4)+ 2,25 Participante(Replica)_P10(R5)
- 2,25 Participante(Replica)_P9(R5)- 21,75 Participante(Replica)_P11(R6)
+ 21,75 Participante(Replica)_P12(R6)
```

Figura 55: Equação de Regressão para o tempo total - Experimento #1

As Figura 56 e Figura 57 mostram os gráficos de resíduos para a ANOVA que avalia os tempos da questão 2 e questão 7. Em ambos os casos, a normalidade, independência e Homocedasticidade dos resíduos estão evidentes, donde podemos concluir que há a adequação da ANOVA.

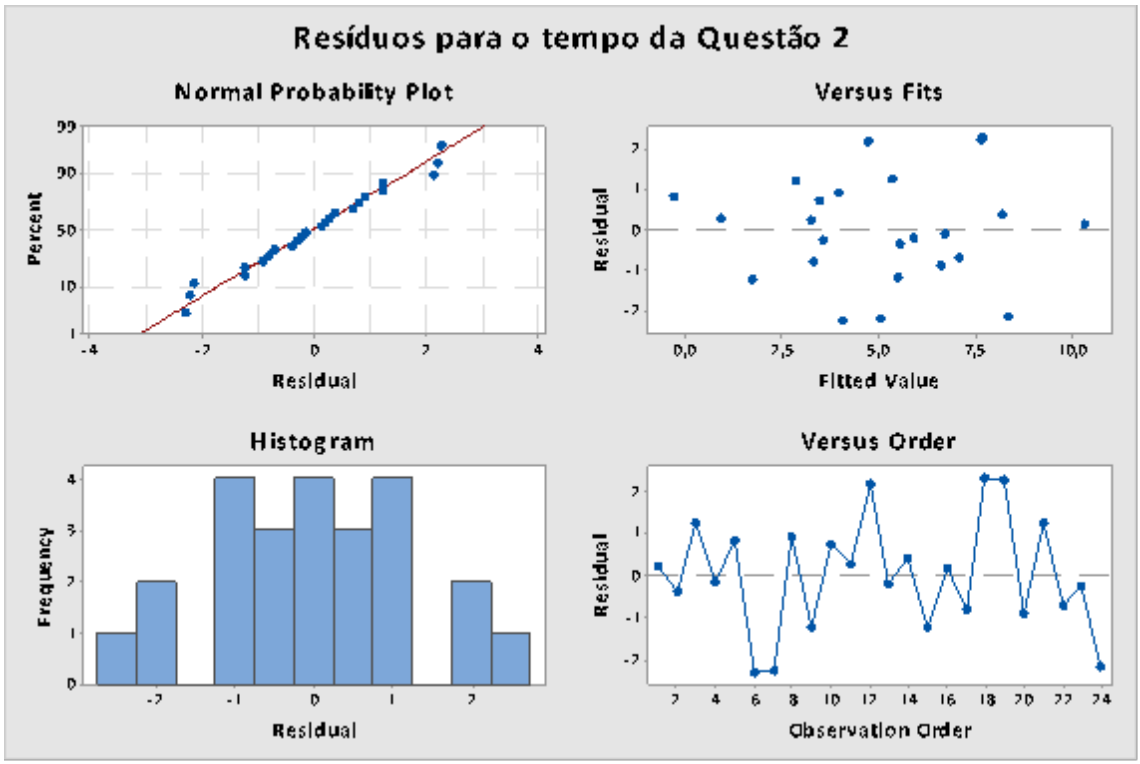


Figura 56: Gráficos de resíduos para a ANOVA da questão2

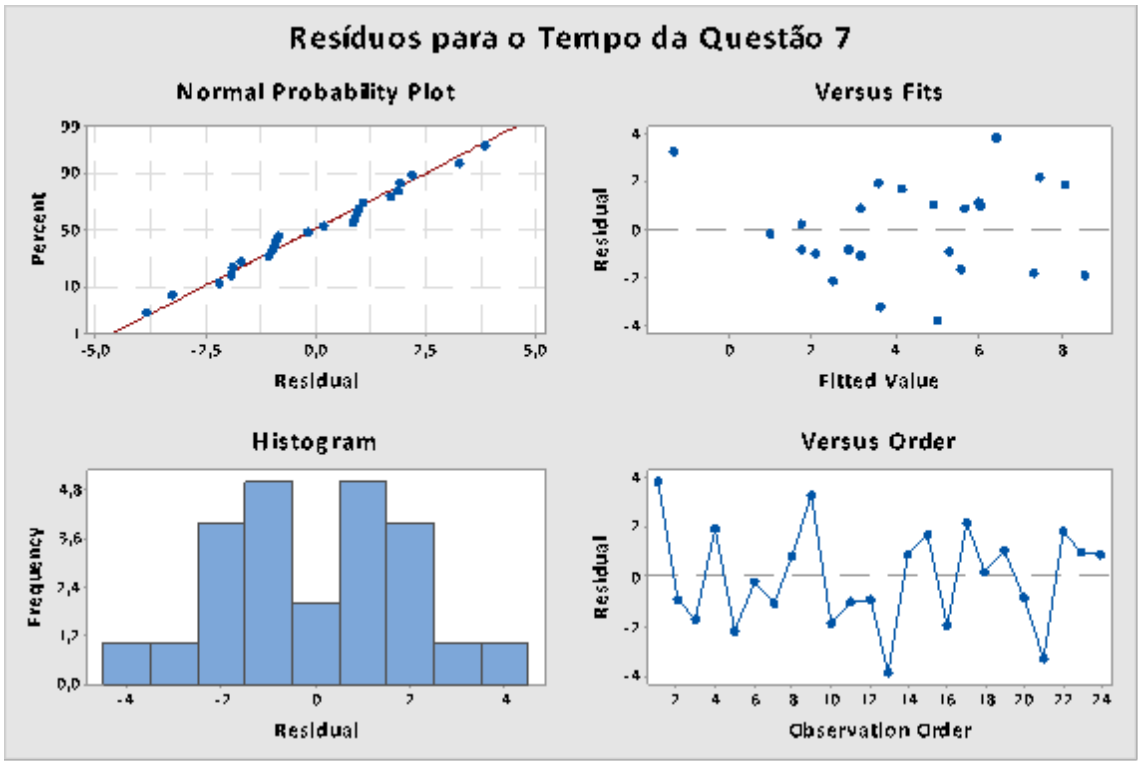


Figura 57: Gráficos de resíduos para a ANOVA da questão7

Apêndice III: Telas do Ambiente de Execução

Este apêndice ilustra algumas telas do ambiente web onde são instalados os workflows gerados pela abordagem proposta (Capítulo 5).

Iniciamos com a apresentação da tela de configuração para o experimento ilustrativo cujo título é "OO System Productivity".

A Figura 58 exibe o projeto do experimento após a distribuição automática dos tratamentos aos usuários. O projeto é um quarado latino 2x2. Podemos observar que existem dois quadrados (LS1 e LS0) e quatro usuários (User 1, User2, User3 e User 4). Além disso, "*ComparisonProcessJavaEventManagement*", por exemplo, é o nome do processo associado ao tratamento que o usuário *User3* deve executar primeiro (LS1).

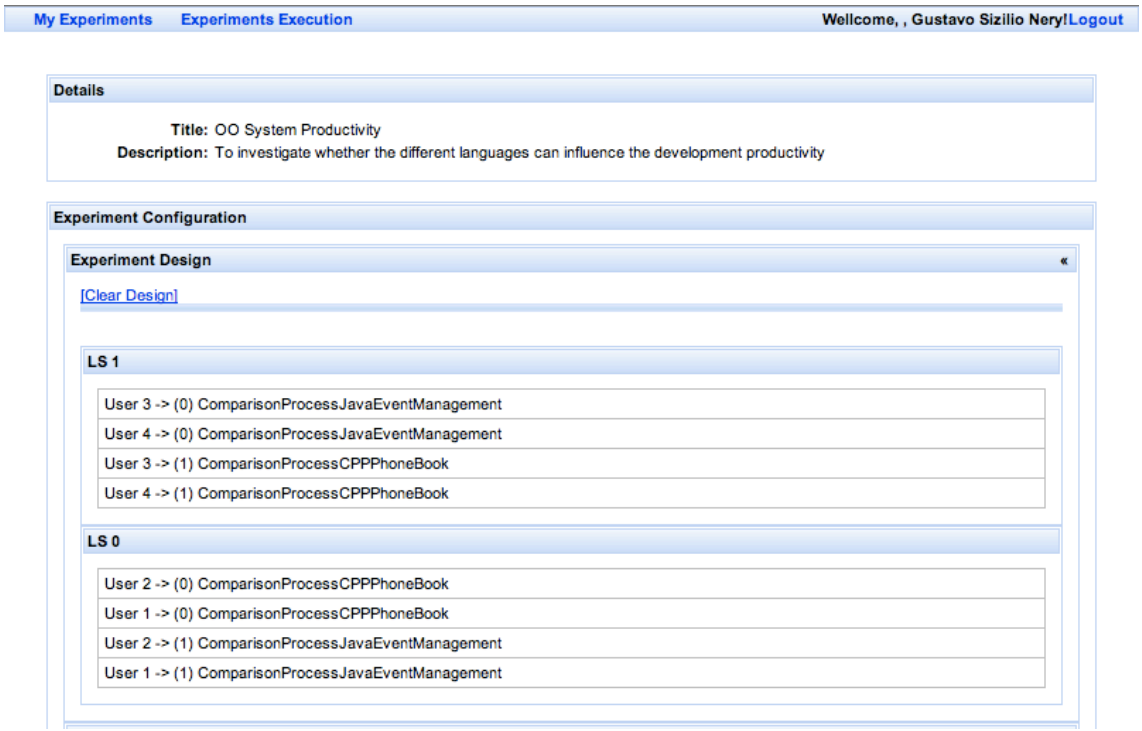


Figura 58: Tela de Apresentação da Distribuição dos tratamentos

A Figura 59 exibe a tela com a primeira tarefa apresentada ao participante do experimento. A tarefa tem a seguinte descrição "*Create de design of use cases*", no formulário, o participante deve fazer o download do artefato "*Use case specification*", após realização da tarefa ele deve fazer o upload de dois artefatos: diagrama de classe (*PhonBook_ClassDiagram*)e protótipo da interface (*PhoneBook_PrototypeInterface*). A qualquer momento o usuário pode fazer uma pausa (botão *Pause*) para realizar tarefas

fora do contexto do experimento (como atender o telefone, por exemplo), de forma a não prejudicar a contagem do tempo da tarefa. Ao finalizar a tarefa, o participante deve solicitar a próxima tarefa clicando no botão *Next*.

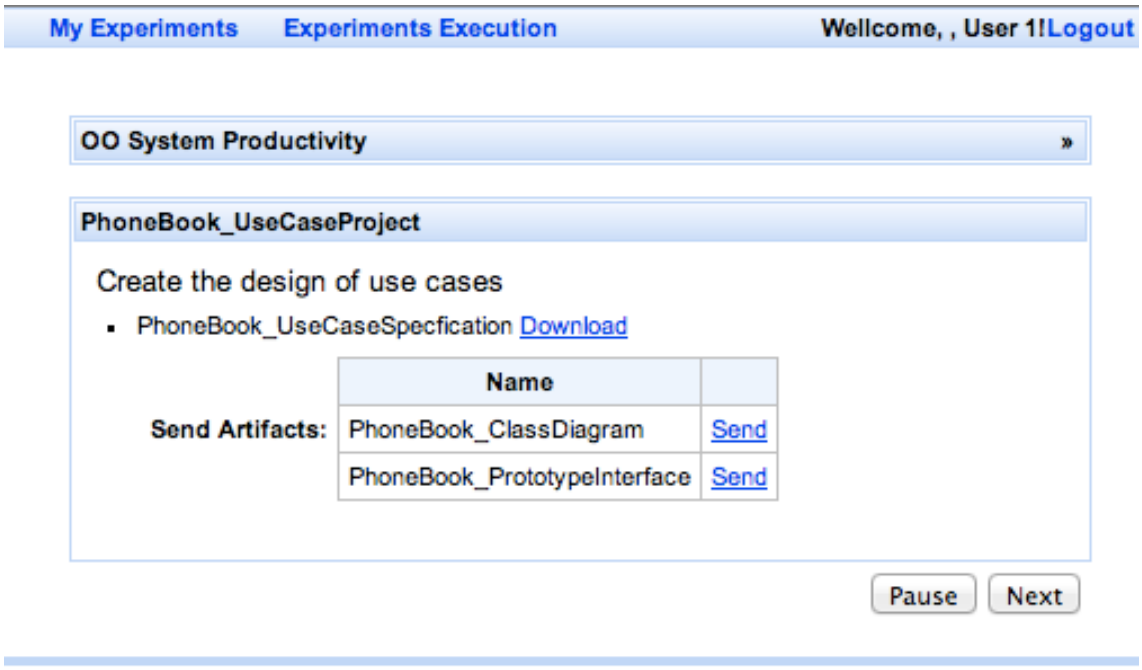


Figura 59: Formulário Web de apresentação de tarefa ao participantes.

A mostra a tela de monitoramento que é apresentada ao pesquisador responsável pelo experimento. O pesquisador pode visualizar o andamento da aplicação de todos os tratamentos, verificando quais tarefas cada usuário está desempenhando ou já desempenhou, podendo ainda verificar os artefatos gerados e visualizar também o tempo decorrido em cada uma das tarefas. O pesquisador pode ainda inserir comentários durante a execução, de forma a registrar um histórico dos acontecimentos que, por ventura, possam acontecer através do botão "Comments". O botão "Analysis" serve para o experimentador habilitar a análise após o término da execução. Ele poderá exportar os dados para que sejam tratados e, ao final, poderá importar na ferramenta para gerar, por exemplo, uma análise de variância (único tipo de análise possível até o momento). A análise é realizada através de uma integração do ambiente com a ferramenta de análise estatística R[11].

[11] http://www.r-project.org/

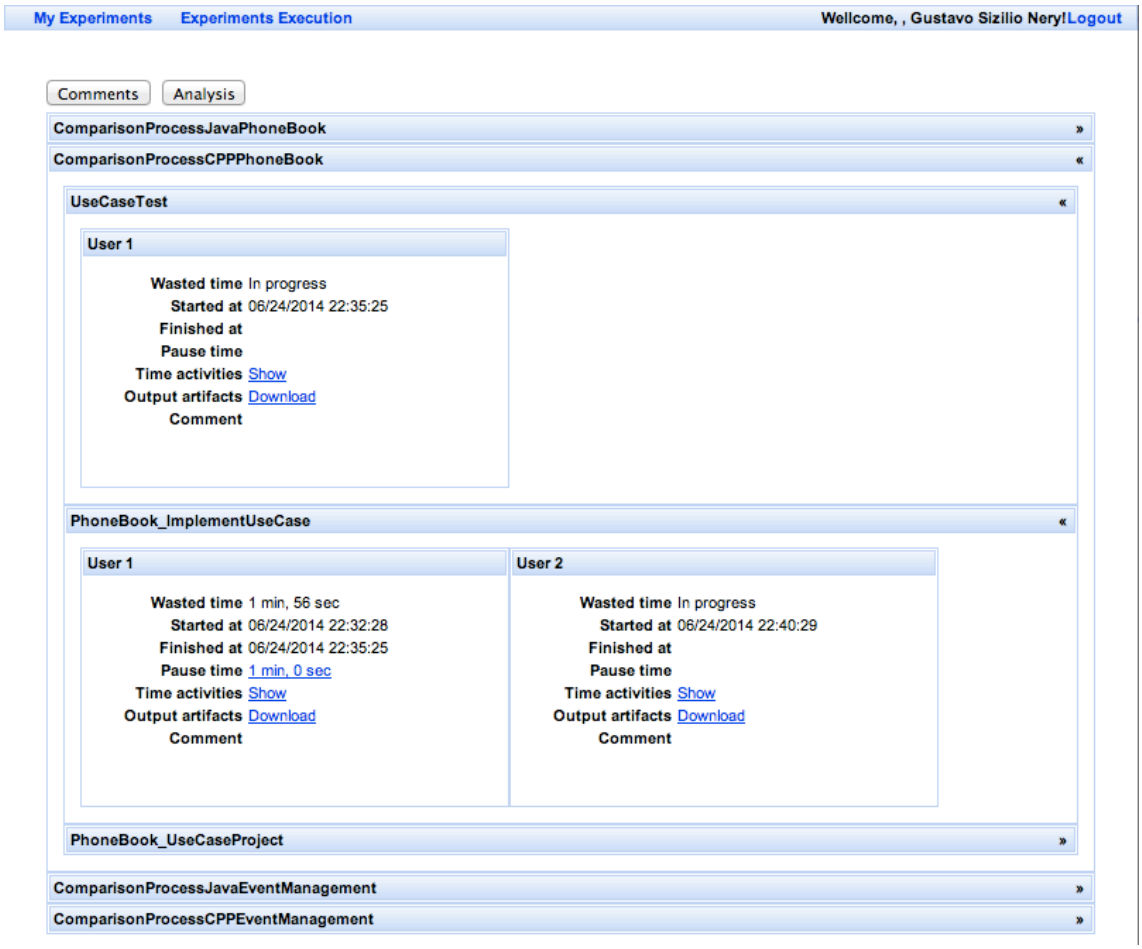
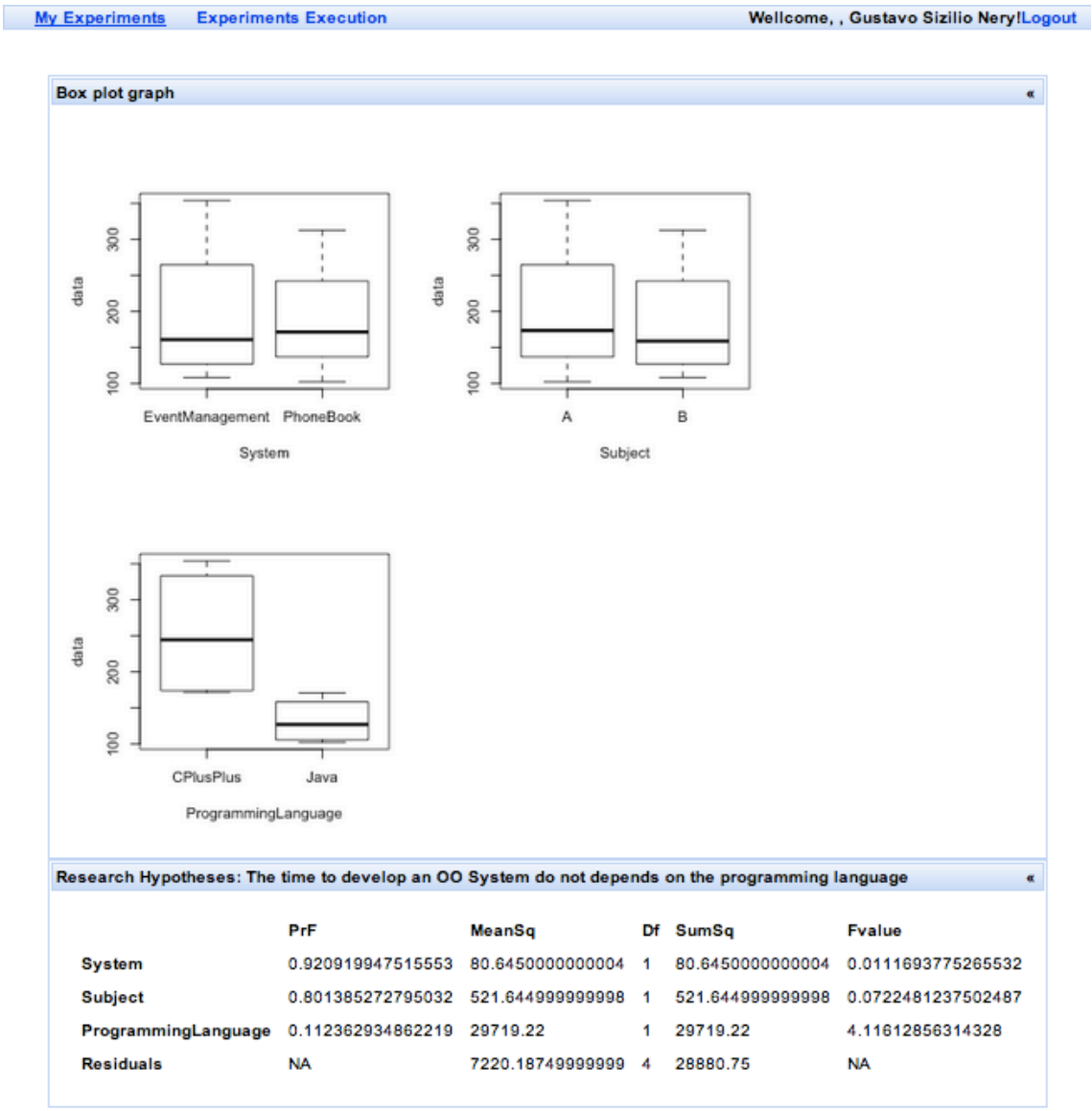


Figura 60: Tela de monitoramento dos participantes

A Figura 61 mostra o relatório gerado após a execução ANOVA para os dados simulados para o experimento ilustrado neste apêndice. O Resumo mostra os gráficos Box plot gerados para fator do experimento exemplo, além do resultado da ANOVA.



	PrF	MeanSq	Df	SumSq	Fvalue
System	0.920919947515553	80.6450000000004	1	80.6450000000004	0.0111693775265532
Subject	0.801385272795032	521.644999999998	1	521.644999999998	0.0722481237502487
ProgrammingLanguage	0.112362934862219	29719.22	1	29719.22	4.11612856314328
Residuals	NA	7220.18749999999	4	28880.75	NA

Figura 61: Resumo da execução da Análise

Por fim, ilustramos o formulário que apresenta um questionário de feedback definido para o experimento exemplo (Figura 62). O questionário apresenta apenas três questões a serem respondidas pelo participante. Ao final, o participante deve clicar no botão Save para finalizar seu questionário.

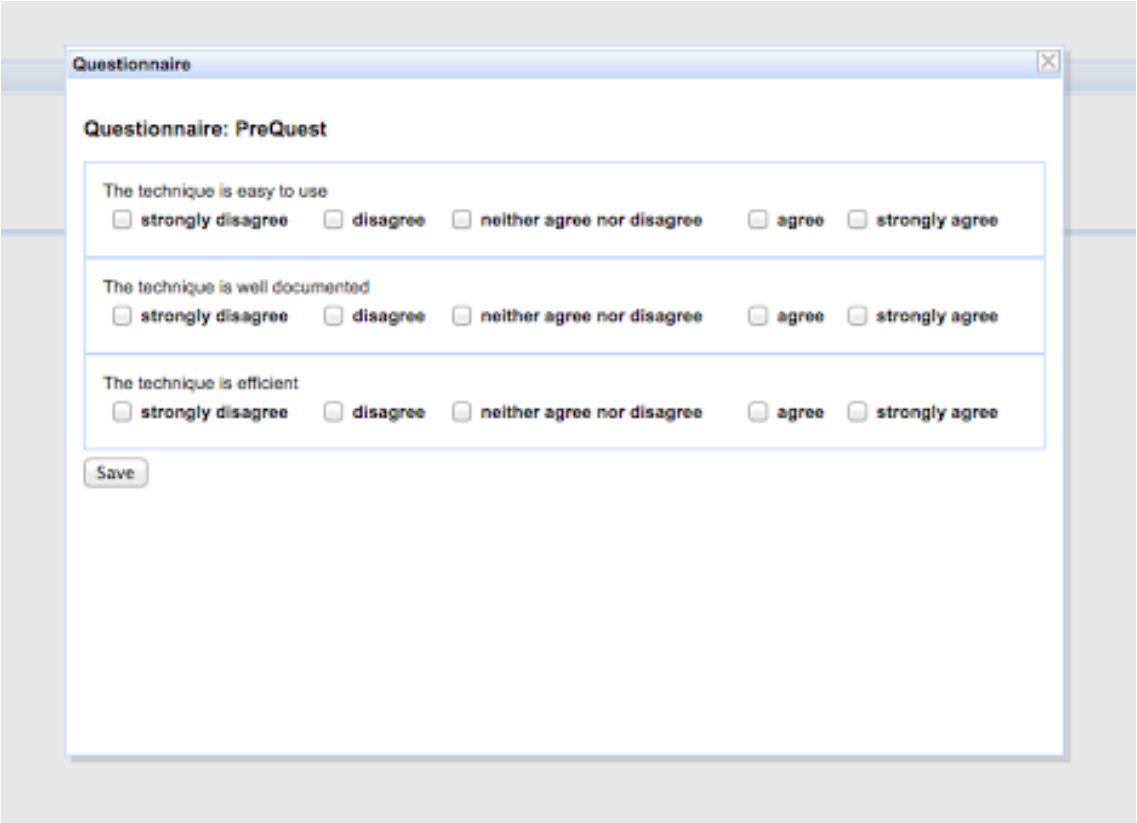


Figura 62: Formulário associado ao questionário de feedback do experimento exemplo.