

João Pucci Neto

Apoio à execução de experimentos controlados
usando uma ontologia para empacotamento: a fer-
ramenta *OntoExpTool*

São José do Rio Preto
2015

João Pucci Neto

Apoio à execução de experimentos controlados
usando uma ontologia para empacotamento: a fer-
ramenta *OntoExpTool*

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

Orientador: Prof. Dr. Rogério Eduardo Garcia

São José do Rio Preto
2015

Pucci, João Neto

Apoio à execução de experimentos controlados usando uma ontologia para empacotamento: a ferramenta *OntoExpTool* / João Pucci Neto. - São José do Rio Preto: [s.n], 2015.

105 f. : il. ; 30 cm.

Orientador: Prof. Dr. Rogério Eduardo Garcia

Dissertação (mestrado) - Universidade Estadual Paulista "Júlio de Mesquita Filho", Instituto de Biociências, Letras e Ciências Exatas

1. Engenharia de Software Experimental. 2. Experimento Controlado. 3. Pacote de Laboratório. 4. Ontologia. 5. *ExperOntology*. I. Garcia, Rogério Eduardo. III. Universidade Estadual Paulista "Júlio de Mesquita Filho", Instituto de Biociências, Letras e Ciências Exatas. IV. Título

João Pucci Neto

Apoio à execução de experimentos controlados
usando uma ontologia para empacotamento: a fer-
ramenta *OntoExpTool*

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

Comissão Examinadora

Prof. Dr. Rogério Eduardo Garcia
UNESP - Presidente Prudente
Orientador

Prof. Dr. Reginaldo Ré
UTFPR - Campo Mourão

Prof. Dr. Danilo Medeiros Eler
UNESP - Presidente Prudente

São José do Rio Preto
20 de Fevereiro de 2015

Este trabalho é dedicado a DEUS, à minha mãe Carmo, a meu pai Luiz (in memorian) e meus irmãos Marcelo (in memorian) e Valéria, pessoas fundamentais em minha vida.

Agradecimentos

Agradeço, em primeiro lugar, a Deus, por me abençoar com o presente de ser escolhido neste mestrado e, por me dar forças para seguir firme em todos os momentos de alegrias e dificuldades que tive no decorrer do mesmo.

Agradeço em especial à minha *Super Mãe* Carmo pela paciência, compreensão, força dada e pelas muitas orações. Te amo!

Agradeço ao meu Grande orientador Rogério, por ter me proporcionado um grande aprendizado durante todo o mestrado e me apoiado em não desistir nunca. Hoje entendo as chamadas de atenção, tudo bem que algumas poderiam ter sido mais *light* né (risos).

Agradeço à minha namorada Livia por sempre estar ao meu lado apoiando e dando forças no decorrer do mestrado, compreendendo e tolerando as minhas alterações de humor (risos); e também a toda minha família, em especial, à minha irmã Valéria.

Agradeço a meus colegas do mestrado em Ciência da Computação: Álvaro, Lilian, Fernanda, Jorge, Helton, Livia, Ingrid e aos demais do *Lapessa* da Unesp de Presidente Prudente; aos vários colegas da turma do Mestrado em Matemática, em particular ao José Vanterler e Irineu, companheiros de estudo até altas horas. Em especial à amiga e companheira de mestrado Vanessa pelo apoio e conselhos. A todos meu muito obrigado pela colaboração e amizade.

Agradeço aos professores do mestrado em Ciência da Computação: Hilda, Ivan, João Paulo e Rogéria pelos ensinamentos em suas disciplinas, e, em especial, à professora Rogéria pela atenção e conselhos dispensados.

Agradeço aos funcionários da secretaria da Pós Graduação do Ibilce, pela sempre prestação em seus atendimentos.

Agradeço aos professores Danilo Eler e Reginaldo Ré que compuseram a minha banca de qualificação e me deram muita contribuição para a continuidade deste trabalho.

Agradeço a todos meus amigos de trabalho da Etec, em especial: aos diretores Claudemir, Marcelo e Edson; a todos os professores da área de informática pelas ajudas dadas e aos demais de outras áreas também, em especial à Renata Prates e à minha amiga Luciane pela força, conselhos e apoio no decorrer

deste mestrado.

Agradeço muito aos conselhos, orações e orientações espirituais recebidas de meus amigos Padre Lindomar e Padre Sandro: vocês foram de suma importância na minha caminhada no mestrado.

Agradeço em geral a todos os meus colegas e amigos que direta ou indiretamente contribuíram para o meu mestrado, em especial, Evandro, Danilo Filito, Ivan/Raquel, Profas. Mara e Gilda.

Agradeço a Instituição Centro Paula Souza por conceder afastamento parcial de minhas aulas durante os dois primeiros anos do mestrado, como maneira de incentivo aos estudos.

*"Se os bons combates eu não combater
Minha coroa não conquistarei
Se minha carreira eu não completar
De que vale a minha fé tanto guardar"
(Padre Fábio de Melo)*

Índice

Índice	vi
Lista de Figuras	ix
Resumo	xiii
Abstract	xv
1 Introdução	1
1.1 Contexto e Justificativa	1
1.2 Motivação e Formulação do Problema	2
1.3 Objetivo e Metodologia	2
1.4 Organização do Trabalho	3
2 Experimentos em Engenharia de Software	5
2.1 Considerações Iniciais	5
2.2 Método Experimental em Engenharia de Software	5
2.2.1 Vocabulários de uma experimentação	6
2.2.2 Experimentos Controlados	8
2.2.3 Processo de Experimentação	9
2.2.4 Atividade 1 - Definição	10
2.2.5 Atividade 2 - Planejamento	12
2.2.6 Atividade 3 - Operação	19
2.2.7 Atividade 4 - Análise e Interpretação	20
2.2.8 Atividade 5 - Apresentação e Empacotamento de um Ex- perimento	23
2.3 Considerações Finais	25
3 Empacotamento de Experimentos Controlados	27
3.1 Considerações Iniciais	27
3.2 A transferência de conhecimento e as replicações de experimentos	28
3.3 Ontologia e seu uso no empacotamento	30
3.3.1 <i>ExperOntology</i> - uma ontologia para experimentos controlados	31

3.4	Ambiente de Apoio à Experimentação Aplicada à Engenharia de Software	35
3.5	Considerações Finais	36
4	OntoExpTool - Ferramenta de Apoio à Experimentação	37
4.1	Considerações Iniciais	37
4.2	Estrutura da <i>OntoExpTool</i>	37
4.3	Arquitetura da <i>OntoExpTool</i>	38
4.4	Diagramas de Sequência – <i>OntoExpTool</i>	40
4.4.1	Caso de Uso – Cria Experimento	40
4.4.2	Caso de Uso – Definição	42
4.4.3	Caso de Uso – Planejamento	42
4.4.4	Caso de Uso – Operação	43
4.4.5	Caso de Uso – Análise	43
4.4.6	Caso de Uso – Empacotamento	43
4.4.7	Diagrama de Classes - <i>OntoExpTool</i>	45
4.5	Implementação da ferramenta	45
4.5.1	Módulo de controle do tempo de execução de tarefas	47
4.5.2	Módulo de segurança	47
4.5.3	Armazenamento dos dados	47
4.5.4	Pacote <i>OWL</i> gerado	47
4.6	Considerações Finais	49
5	Empacotamento com uso da <i>OntoExpTool</i>	51
5.1	Considerações Iniciais	51
5.2	Experimento utilizado como demonstração: Avaliação da <i>SofVis_{OA}H</i>	52
5.3	Perspectiva do Experimentador	52
5.3.1	Definição do Experimento	52
5.3.2	Planejamento do Experimento	53
5.3.3	Operação do Experimento	59
5.3.4	Análise e Interpretação dos dados do Experimento	60
5.3.5	Empacotamento e Apresentação dos dados do Experimento	60
5.4	Considerações Finais	63
6	Conclusões	65
A	Diagramas da Ferramenta	69
A.1	Diagramas de Sequência	69
A.2	Diagramas de Classes - Particionados	70
A.3	Estruturas da Base de Dados da ferramenta	70

Lista de Figuras

2.1	Princípios básicos de um experimento, adaptado de Wohlin et al. (2012)	7
2.2	Ilustração de um experimento com as variáveis, adaptado de Wohlin et al. (2012).	8
2.3	Visão da atividade de definição, adaptado de Wohlin et al. (2012)	10
2.4	Visão da atividade de definição, adaptado de Wohlin et al. (2012)	11
2.5	Visão da atividade de Planejamento, adaptado de Wohlin et al. (2012)	13
2.6	Visão das três etapas da atividade de Operação, adaptado de Wohlin et al. (2012)	19
2.7	Visão das três etapas de uma análise e interpretação quantitativa, adaptado de Wohlin et al. (2012)	21
2.8	Gráfico de dispersão - detecção de um <i>outlier</i> , adaptado de Wohlin et al. (2012)	22
2.9	Fase de empacotamento do processo de experimentação, adaptado de Amaral e Travassos (2003)	24
3.1	Ciclos do FIRE, adaptado de Mendonça et al. (2008)	29
3.2	Ontologia para um Experimento Controlado, adaptado de Garcia et al. (2008)	33
3.3	Ontologia para Pacotes de Laboratório, adaptado de Garcia et al. (2008)	34
4.1	<i>Workflow</i> de sequência de atividades de condução e empacotamento de experimentos controlados	38
4.2	Diagrama de Caso de Uso - Visão geral	39
4.3	Arquitetura – <i>OntoExpTool</i>	39
4.4	Diagrama de Sequência – Cria Experimento	41
4.5	Diagrama de Sequência – Associar Experimentador	41
4.6	Diagrama de Sequência – Incluir Definição	42

4.7	Diagrama de Sequência – Seleção de Contexto	42
4.8	Diagrama de Sequência – Processa Dados Operacao – Participante	43
4.9	Diagrama de Sequência – Realiza Analise e Interpretacao	44
4.10	Diagrama de Sequência – Realiza Apresentação e Empacotamento	44
4.11	Diagrama de Classes – Simplificado	45
4.12	Esquema da Base de Dados I	48
4.13	Pacote de Laboratório gerado em OWL	49
5.1	Conceitos organizados na <i>ExperOntology</i> da Atividade <i>Definição</i> , adaptado de Garcia et al. (2008)	54
5.2	Questões e Métricas elaboradas na Atividade <i>Planejamento</i> (d’Arce, 2012)	54
5.3	Hipóteses elaboradas na Atividade <i>Planejamento</i> (d’Arce, 2012) .	55
5.4	Variáveis Independentes elaboradas na Atividade <i>Planejamento</i> (d’Arce, 2012)	55
5.5	Variáveis Dependentes elaboradas na Atividade <i>Planejamento</i> (d’Arce, 2012)	55
5.6	Instrumentação – Ferramentas Informatizadas – <i>Planejamento</i> (d’Arce, 2012)	56
5.7	Instrumentação – Materiais de Realizacao de Testes – <i>Planeja-</i> <i>mento</i> (d’Arce, 2012)	56
5.8	Instrumentação – Materiais de Treinamento – <i>Planejamento</i> (d’Arce, 2012)	57
5.9	Instrumentação – Questionários – <i>Planejamento</i> (d’Arce, 2012) . .	57
5.10	Instrumentação – Formulários – <i>Planejamento</i> (d’Arce, 2012) . . .	57
5.11	Conceitos organizados na <i>ExperOntology</i> da Atividade <i>Planejamento</i> , adaptado de Garcia et al. (2008)	58
5.12	Conceitos organizados na <i>ExperOntology</i> da Atividade <i>Operação</i> , adaptado de Garcia et al. (2008)	59
5.13	Conceitos organizados na <i>ExperOntology</i> da Atividade <i>Análise</i> , adap- tado de Garcia et al. (2008)	61
5.14	Instantâneo da Interface para a Atividade de <i>Empacotamento</i> . . .	62
5.15	Arquivo XML - Pacote de Laboratório gerado	62
5.16	Arquivo OWL - Pacote de Laboratório gerado	63
6.1	Detalhes de um processo experimental com a criação de um Pa- cote de Laboratório baseado na <i>ExperOntology</i>	66
A.1	Diagrama de Sequência – Formulação de Hipoteses – Nula e Al- ternativa	70
A.2	Diagrama de Sequência – Seleção de Variáveis – Dependentes e Independentes	70

A.3 Diagrama de Sequência – Selecionar Participantes	71
A.4 Diagrama de Sequência – Caracterização de Participantes	71
A.5 Diagrama de Sequência – Associar Técnica–Artefato	72
A.6 Diagrama de Sequência – Definir Cronograma	72
A.7 Diagrama de Sequência – Configurar Estudo	73
A.8 Diagrama de Classes I	74
A.9 Diagrama de Classes II	75
A.10Diagrama de Classes III	76
A.11Diagrama de Classes IV	77
A.12Esquema da Base de Dados II	78
A.13Esquema da Base de Dados III	79
A.14Esquema da Base de Dados IV	80
A.15Esquema da Base de Dados V	81
A.16Esquema da Base de Dados VI	82

Resumo

A condução de experimentos controlados e suas replicações gera dados que são armazenados em um pacote de laboratório, o que contribui para a criação de um corpo de conhecimento sobre Engenharia de Software. A replicação de um estudo depende do pacote de laboratório, em especial, replicações inter grupos. A transferência do conhecimento armazenado nos pacotes de laboratório é dificultada pela falta de padrão na organização das informações e dificuldades na compreensão. O uso de ontologia na instanciação de pacotes de laboratórios pode facilitar a transferência de conhecimento entre grupos de pesquisa de Engenharia de Software. Este trabalho apresenta uma ferramenta computacional que executa um *workflow* definido para as atividades da experimentação (*Definição, Planejamento, Execução, Análises e Interpretações, e Empacotamento*). Ela permite acesso diferenciado para experimentador e participante. O experimentador tem acesso a todas atividades do processo experimental, em especial, controle sobre a atividade *Operação*. O participante do experimento tem acesso restrito à tarefa a ele associada, bem como os artefatos necessários. O pacote de laboratório é instanciado de acordo com a *ExperOntology* (ontologia de domínio para experimentos controlados) gerando um arquivo OWL ou XML. Para avaliar o uso da ferramenta foi registrado e empacotado um experimento controlado.

Palavras-chave: Engenharia de Software Experimental, Experimento Controlado, Pacote de Laboratório, Ontologia, *ExperOntology*.

Abstract

Conducting controlled experiments and their replication generates data to be stored in a lab package, which contributes to create a body of knowledge on Software Engineering. Replication of a study depends on the lab package, in particular inter groups replication. The transfer of knowledge stored into lab packages is hampered by the lack of standard to organize information and its understanding. The use of ontology on lab packages instantiation might facilitate transferring of knowledge among Software Engineering research groups. This work presents a computational tool to execute the workflow defined for experimentation (Definition, Planning, Operation, Analysis and Packaging activities). The lab package is instantiated according to *ExperOntology* (domain ontology for controlled experiments) generating OWL or XML file. The tool allows different access to experimenter and participant roles. The experimenter has access to all activities of the experimental process, particularly to control the operation activity. The experiment participant has restricted access to the tasks associated to himher, as well as the required artifacts. To evaluate the use of the tool was recorded and packaged a controlled experiment.

Key-words: Experimental Software Engineering, Controlled Experiment, Lab Package, Ontology, *ExperOntology*

Introdução

1.1 Contexto e Justificativa

De acordo com [Wohlin et al. \(2012\)](#), desenvolver *software* não é uma tarefa simples e que, com a evolução de projetos de *softwares*, alguns problemas têm surgido, como a falta de funcionalidade, exagero nos custos, não cumprimento de prazos e baixo nível de qualidade. A Engenharia de Software Experimental tenta avaliar e medir a performance de modelos e técnicas em contextos práticos, a fim de estabelecer uma base de conhecimento para apoiar a tomada de decisão ([Garcia et al. \(2008\)](#)). É possível realizar experimentos e contrastar com experimentos anteriores, a fim de construir uma fundamentação científica para novas ideias.

Segundo [Basili et al. \(1999\)](#), a experimentação ajuda a determinar a eficácia de métodos e de teorias propostas. Somente experimentos verificam as teorias, podem explorar os fatores críticos, e dar luz a fenômeno novo para que as teorias possam ser formuladas e corrigidas. O processo experimental proporciona de modo sistemático, disciplinado e controlado a avaliação de processos e de atividades humanas. Novos métodos, técnicas, linguagens e ferramentas devem passar por experimentação e validação antes de serem sugeridos, publicados ou oferecidos para venda ([Travassos et al. \(2002\)](#)).

[Carver \(2010\)](#) menciona que os resultados de estudos experimentais só podem ser considerados se forem consolidados em um corpo de conhecimento significativo para a comunidade que atua na área, o que indica um aspecto importante em experimentação: a necessidade de replicar estudos experimentais para gerar dados e compor o corpo de conhecimento.

Para a replicação de um experimento replicadores necessitam revisar as informações dele para compreender seu projeto, condução e análise e para isso

é necessário o acesso a informações sobre a condução do estudo (Kitchenham, 2008). As informações, o conhecimento gerado, os procedimentos utilizados, os resultados obtidos e as conclusões são armazenados no que se denomina de pacote de laboratório (Shull et al., 2002). Porém, pesquisadores encontram dificuldades na compreensão de pacotes de laboratório (Shull et al., 2002). Para lidar com problemas, foi proposto o *FIRE* descrito na Seção 3.2.

Ontologia é uma especificação explícita de uma conceitualização (Gruber, 1995). Ontologias têm sido utilizadas em Engenharia de Software com o intuito de modelar conceitos de um domínio (Kitchenham et al., 1999, de A. Bionchini et al., 2007) e são um meio para prover o entendimento compartilhado da estrutura da informação que modelam (Noy e McGuinness, 2001). Uma ontologia para experimentos controlados foi proposta por Garcia et al. (2008), denominada *ExperOntology*. Sua principal finalidade é padronizar a organização do pacote de laboratório. A *ExperOntology* pode ser utilizada como um padrão semântico das informações de experimentos colaborando para a integração dos experimentos e para a transferência de conhecimento entre grupos de pesquisa em Engenharia de Software (Garcia et al. (2008)).

1.2 Motivação e Formulação do Problema

A execução de experimentos na área de Engenharia de Software Experimental consome tempo e produz grande volume de informação, tornando o gerenciamento um tanto complicado (Shull et al., 2002). Para isso é importante que se tenha uma estrutura computacional capaz de apoiar a execução de experimentos e que possa utilizar uma ontologia que visa a descrever formalmente os conceitos que compõem um pacote de laboratório, para facilitar a interpretação e entendimento do mesmo.

Ao realizar um experimento o experimentador registra informações que são armazenadas em um pacote de laboratório. Para interpretar um experimento, um grupo de pesquisa (eventualmente externo) deve revisar a descrição de um experimento contida em um pacote de laboratório do experimento (Scatlon et al., 2011a, Mendonça et al., 2008). Shull et al. (2002) afirmam que pesquisadores encontram dificuldades ao tentar analisar e interpretar pacotes de laboratório, como a compreensão dos conceitos subjacentes às técnicas de estudo, e domínio dos conhecimentos envolvidos na execução do experimento.

1.3 Objetivo e Metodologia

O presente trabalho tem como foco principal o apoio à execução de experimentos controlados com o uso de uma ontologia, a *ExperOntology* (apresentada na Seção 3.3.1) para gerar pacotes de laboratório padronizados.

O objetivo é prover uma infraestrutura computacional capaz de auxiliar o

experimentador no registro das informações geradas em todas as fases do processo de experimentação apresentado na Seção 2.2, seguindo uma sequência de atividades do *workflow* elaborado e apresentado na Seção 4.2.

As informações de cada fase de um experimento controlado são registradas pelo experimentador na ferramenta desenvolvida, e são armazenadas em uma base de dados durante a execução do experimento. Após a conclusão do experimento as informações registradas durante as fases são instanciadas utilizando como base os conceitos contidos na estrutura da ontologia *ExperOntology*, para montar o pacote de laboratório de maneira organizada, no formato *OWL*.

Com o pacote de laboratório criado no formato *OWL* e organizado baseado nos conceitos da *ExperOntology*, outro pesquisador pode utilizar ferramentas computacionais, como os *reasoners*, para apoiá-lo no entendimento dos conceitos registrados no Pacote de Laboratório, e com isso facilitar a transferência de conhecimento.

A metodologia empregada no desenvolvimento deste trabalho foi executada de acordo com as seguintes atividades:

- levantamento bibliográfico de artigos publicados de famílias de experimentos (Basili et al., 1999);
- implementação de uma ferramenta que auxilia o pesquisador na condução da execução de um experimento controlado, seguindo o processo proposto por Wohlin et al. (2012). Na mesma ferramenta poderá ser instanciado por meio de ontologia o experimento e gerar um pacote de laboratório usando a ontologia *ExperOntology*.
- para validação da ferramenta desenvolvida com intuito de obter indícios de suas vantagens na condução e empacotamento de experimentos controlados para futuras replicações, dois estudos experimentais controlados foram registrados e empacotados gerando arquivos *XML* e *OWL* disponíveis no servidor do grupo de pesquisa do Laboratório de Pesquisa em Engenharia de Software Aplicada (*Lapesa*) da Faculdade de Ciências e Tecnologia da Universidade Estadual Paulista "Júlio de Mesquita Filho- UNESP de Presidente Prudente.

1.4 Organização do Trabalho

Para contextualizar e apresentar a ferramenta *OntoExpTool*, esta dissertação está organizada como segue:

- No Capítulo 2 é apresentada uma revisão bibliográfica a respeito da metodologia de pesquisa adotada – Engenharia de Software Experimental – e o processo de experimentação detalhado.

- No Capítulo 3 é apresentada uma revisão bibliográfica a respeito da transferência de conhecimentos e as replicações de experimentos; é demonstrada a importância do uso de ontologias em experimentação e também é apresentada a estrutura de conceitos e relações da *EXPEROntology*, ontologia que pode ser aplicada como um padrão para lidar com os problemas apontados com relação a integração e ao compartilhamento das informações de pacotes de laboratório; é abordada a existência de um ambiente de apoio à Experimentação Aplicada à Engenharia de Software desenvolvido por outro grupo de pesquisa.
- No capítulo 4 são apresentados detalhes da arquitetura da ferramenta *OntoExpTool*, diagramas de sequência e de classes sobre a mesma, e uma parte da estrutura da base de dados, assim como as tecnologias utilizadas na sua implementação e desenvolvimento.
- No Capítulo 5 é apresentada a utilização da ferramenta contemplando passo a passo as atividades do processo experimental demonstrado no capítulo 2, com abordagem de uso na perspectiva do experimentador e do participante de um experimento.
- No Capítulo 6 são apresentadas as considerações finais deste trabalho, bem como propostas de trabalhos futuros.
- No Apêndice A são apresentados os demais diagramas de sequência, de classes e também as demais estruturas da base de dados, em complemento as que foram demonstrados no capítulo 4.

Experimentos em Engenharia de Software

2.1 Considerações Iniciais

O principal objetivo da Engenharia de Software Experimental é a realização de estudos que possam avaliar ferramentas, técnicas e métodos que fazem parte do desenvolvimento de software e confrontar com fatos as suposições, as especulações e as dúvidas existentes no *contexto* de desenvolvimento de software. Após os resultados de estudos, pode-se construir um corpo de conhecimento em Engenharia de Software para contribuir em tomadas de decisão com a sua respectiva aplicação no mercado e também prover um alicerce científico para a disciplina ([Wohlin et al., 2012](#)).

Na Engenharia de Software Experimental emprega-se a estratégia de experimentos para verificar e validar teorias, hipóteses e relacionamentos, com o objetivo de caracterizar, avaliar, prever, controlar e melhorar o Processo de Software ([Wohlin et al., 2012](#)).

Neste capítulo são apresentados conceitos básicos de experimentação controlada e seu processo. É apresentada uma visão geral do Método Experimental em Engenharia de Software, bem como as replicações dos experimentos.

2.2 Método Experimental em Engenharia de Software

Segundo [Basili et al. \(1999\)](#), a Engenharia de Software Experimental (ESE) é um segmento da Engenharia de Software que tem os experimentos como método de verificação e validação de teorias, hipóteses e relacionamentos, que servem para a correção e aprimoramento das técnicas empregadas pelo grupo de desenvolvedores de software. A caracterização, a avaliação, a previsão,

o controle e a melhoria de produtos, processos, recursos, teorias, modelos, entre outros, são alguns objetivos de execução de experimentos. De acordo com [Travassos et al. \(2002\)](#), “somente experimentos verificam teorias, somente experimentos podem explorar os fatores críticos e dar luz ao fenômeno novo para que as teorias possam ser formuladas e corrigidas”.

Para Conradi (2001) *apud* [Travassos et al. \(2002\)](#), para compreender a natureza dos processos da informação, os pesquisadores devem observar o fenômeno, encontrar explicação, formular a teoria e verificá-la. A experimentação pode ajudar a construir uma base de conhecimento confiável e reduzir assim incerteza sobre quais teorias, ferramentas e metodologias são adequadas às situações analisadas. A observação e experimentação podem levar a novos e úteis meios de introspecção e abrir novas áreas de investigação. Os experimentos são propostos quando desejamos manipular um comportamento de maneira direta, precisa e sistemática. Eles envolvem mais do que um tratamento para comparar os resultados.

A principal razão para se utilizar a experimentação na Engenharia de Software é aprimorar a compreensão e a identificação de relações entre diferentes variáveis envolvidas no processo de software. Para melhorar a compreensão do processo de software e a maneira de desenvolver trabalhos, a experimentação pode ser importante para responder algumas questões do tipo se as inspeções são efetivas e se a orientação do objeto melhora o reuso ([Wohlin et al., 2012](#)).

2.2.1 Vocabulários de uma experimentação

Os princípios básicos de um experimento são apresentados na Figura 2.1. Na parte superior da Figura 2.1 (acima da linha tracejada) está a área da teoria e na parte inferior a observação. Em um experimento controlado, no caso de atividades de desenvolvimento de software é gerado e avaliado um modelo que representa profissionais aplicando métodos, técnicas e ferramentas (considerados objetos de estudo). Os participantes escolhidos para o experimento aplicam tais objetos de estudo sob condições controladas, gerando resultados que possibilitam a análise e obtenção de conclusões a respeito dessa aplicação. Para avaliação de um objeto de estudo em um experimento, é necessário isolar os fatores que o influenciam e também captar os efeitos obtidos com a sua aplicação. Dessa maneira, o ponto de partida fica sendo a ideia de uma relação de causa e efeito em teoria ([Wohlin et al., 2012](#)). Tal relação é formalizada em apenas uma *hipótese* ou em um conjunto de *hipóteses*, que é (são) testada(s) durante o processo. A relação causal que é expressa na hipótese é avaliada por meio da execução do experimento.

As variáveis denominadas *independentes* são os elementos que influenciam a aplicação do objeto de estudo. Em um experimento estuda-se o efeito de uma ou mais variáveis *independentes*, denominado fatores ([Wohlin et al., 2012](#)).

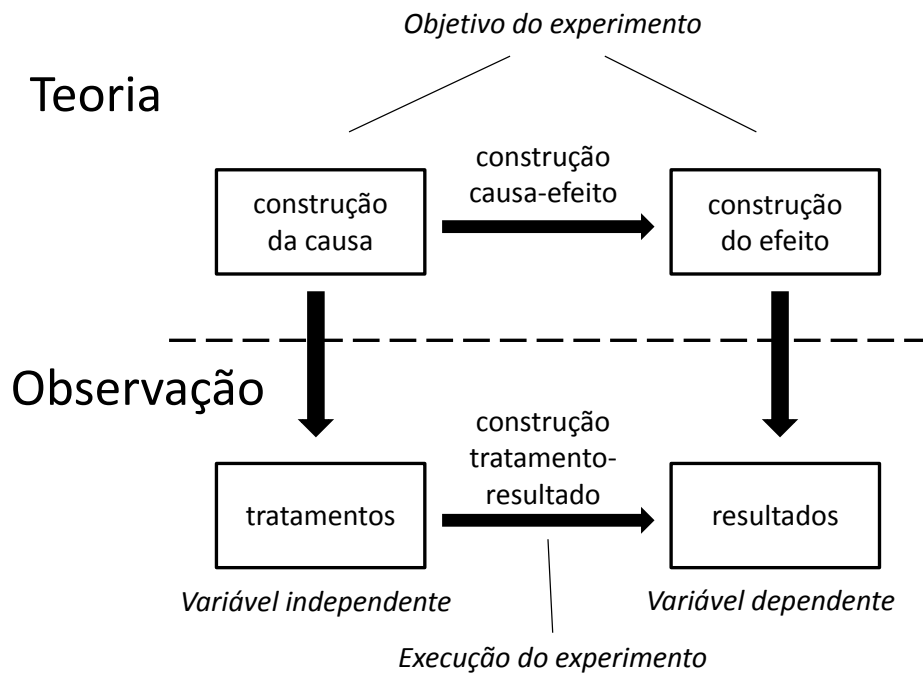


Figura 2.1: Princípios básicos de um experimento, adaptado de Wohlin et al. (2012)

Cada *fator* pode assumir valores denominados *tratamentos*. Como exemplo, cita-se um experimento em que o objetivo é a análise de técnicas de Verificação e Validação (V&V) de software, conforme Basili e Selby (1987), de maneira que alguns fatores podem ser isolados, como o tipo de técnica de teste aplicada e a experiência dos participantes, na qual os tratamentos do fator tipo de técnica podem ser: leitura de código, teste estrutural e teste funcional.

O efeito obtido pela execução varia de acordo com a atribuição de tratamentos. Tal efeito é representado por meio da *variável dependente*, na qual os valores correspondem aos resultados da execução de um experimento. Seguindo o mesmo exemplo, a *variável dependente* pode ser o número de defeitos localizados por um participante com o auxílio de uma determinada técnica de teste. Como ilustrado na Figura 2.2, as variáveis *independentes* são as entradas da execução do experimento e conforme os valores que elas assumem, o efeito das mudanças refletem no valor da *variável dependente*, considerada variável de saída (Lopes e Travassos, 2009a). As pessoas que aplicam o tratamento são chamadas de participantes. Em um experimento, as características dos objetos e também dos participantes podem ser consideradas variáveis *independentes* (Wohlin et al., 2012).

Os tratamentos são aplicados a uma combinação de objetos (exemplo: os programas em que devem ser aplicadas as técnicas de V&V) e participantes (exemplo: são as pessoas que aplicam os tratamentos). Ainda, seguindo o mesmo exemplo, a execução do experimento consiste no processamento de

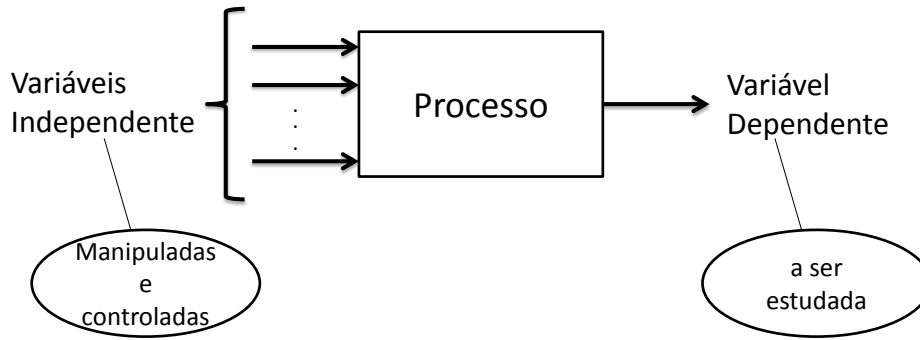


Figura 2.2: Ilustração de um experimento com as variáveis, adaptado de [Wohlin et al. \(2012\)](#).

um conjunto de testes (*trials*), no qual cada teste é uma combinação de tratamento, participante e objeto ([Wohlin et al., 2012](#)), ou seja, um teste pode ser um participante N, usar um método M (tratamento) para inspecionar o programa P (objeto). Durante a execução dos testes, os resultados do experimento são coletados. Deve-se observar que este tipo de teste não deve ser confundido com o uso de testes estatísticos. O número de testes afeta o erro experimental e fornece uma oportunidade para estimar o significado do efeito de qualquer fator experimental. O erro experimental auxilia diretamente a conhecer a quantidade de confiança que podemos colocar nos resultados do experimento.

2.2.2 Experimentos Controlados

As atividades básicas de um processo de experimentação têm como foco proporcionar diretrizes e suporte para executar os experimentos, além disso, deve ser observado que experimentos verdadeiros (aqueles que possuem aleatoriedade completa) são difíceis de se obter na Engenharia de Software. Em Engenharia de Software são considerados como *quasi*-experimentos (na qual não se pode selecionar de maneira aleatória os participantes do experimento). O *quasi*-experimento é mantido por uma discussão completa de ameaças aos experimentos ([Wohlin et al., 2012](#)).

Um experimento é proposto para testar uma teoria ou hipótese. No projeto do experimento existem alguns tratamentos (valores que a variável estudada pode ter) que se pode controlar. O experimento é conduzido e com isso pode ser observado o resultado. Pode-se concluir que foi testada a relação entre o tratamento e o resultado. Se o experimento for corretamente estabelecido, devem ser geradas conclusões sobre a relação entre a causa e o efeito para os quais foi estabelecida uma hipótese.

Em geral, a experimentação envolve observação, projeto experimental, coleta de dados, análise qualitativa ou quantitativa, avaliação do objeto de es-

tudo (processo ou produto). Existem vários tipos de estudos na Engenharia de Software Experimental. Um estudo é o ato de descobrir algo desconhecido ou de testar uma hipótese, podendo incluir todos os tipos de análise quantitativa e ou qualitativa ([Günther, 2006](#)).

Esses estudos devem responder a algumas questões:

1. O que estudar e por que estudar. Por exemplo: Estudos de fatores humanos, ou estudos de projetos e produtos, ou estudo de métodos e técnicas, ou estudos da organização e de seus processos.
2. Que tipo de estudo experimental realizar. Por exemplo: Estudos *In Vivo* (Envolve pessoas no seu próprio ambiente de trabalho em condições reais de trabalho), ou Estudos *In Vitro* (Realizado em condições controladas como em um laboratório ou um grupo fechado), ou Estudos *In Virtuo* (Realizado em condições controladas nas quais os participantes interagem com modelos computacionais da realidade (simuladores)), ou Estudos *In Silico* (Participantes e o mundo real são descritos por modelos computacionais (dinâmica de sistemas) ([Lopes e Travassos, 2009b](#)).
3. Qual objeto de estudo. Exemplo é um processo ou produto.
4. Qual finalidade do estudo. Caracterizar (Exemplo: o quê está acontecendo.), ou avaliar (Exemplo: se é bom.), ou prever (Exemplo: se é possível estimar o comportamento futuro.), ou controlar (Exemplo: se é possível manipular eventos e situações.), ou melhorar (Exemplo: se é possível melhorar eventos e situações.).
5. Qual o foco. Exemplo: Quais aspectos e variáveis do objeto de estudo são de meu interesse.
6. Qual a perspectiva. Exemplo: Quais são os grupos de pessoas interessadas. ([Juristo e Moreno, 2001](#)).

2.2.3 Processo de Experimentação

O processo de experimentação é formulado para que as ações apropriadas sejam adotadas para garantir o sucesso de um experimento. Infelizmente pode acontecer de algum fator ser negligenciado antes do experimento e, consequentemente, pode impedir que sejam feitas análises planejadas, impossibilitando assim, obter conclusões válidas. Segundo [Wohlin et al. \(2012\)](#), os processos se tornam importantes conforme possam ser usados como diretrizes do que fazer e de como fazer e também como listas de checagem. Para realizar um experimento é necessário um processo que envolva várias atividades a serem seguidas em uma determinada ordem.

Na Figura 2.3 são ilustradas todas as atividades do processo de formulação de um experimento. Ao identificar que um experimento é apropriado à avaliação almejada, é importante planejá-lo de maneira cuidadosa.

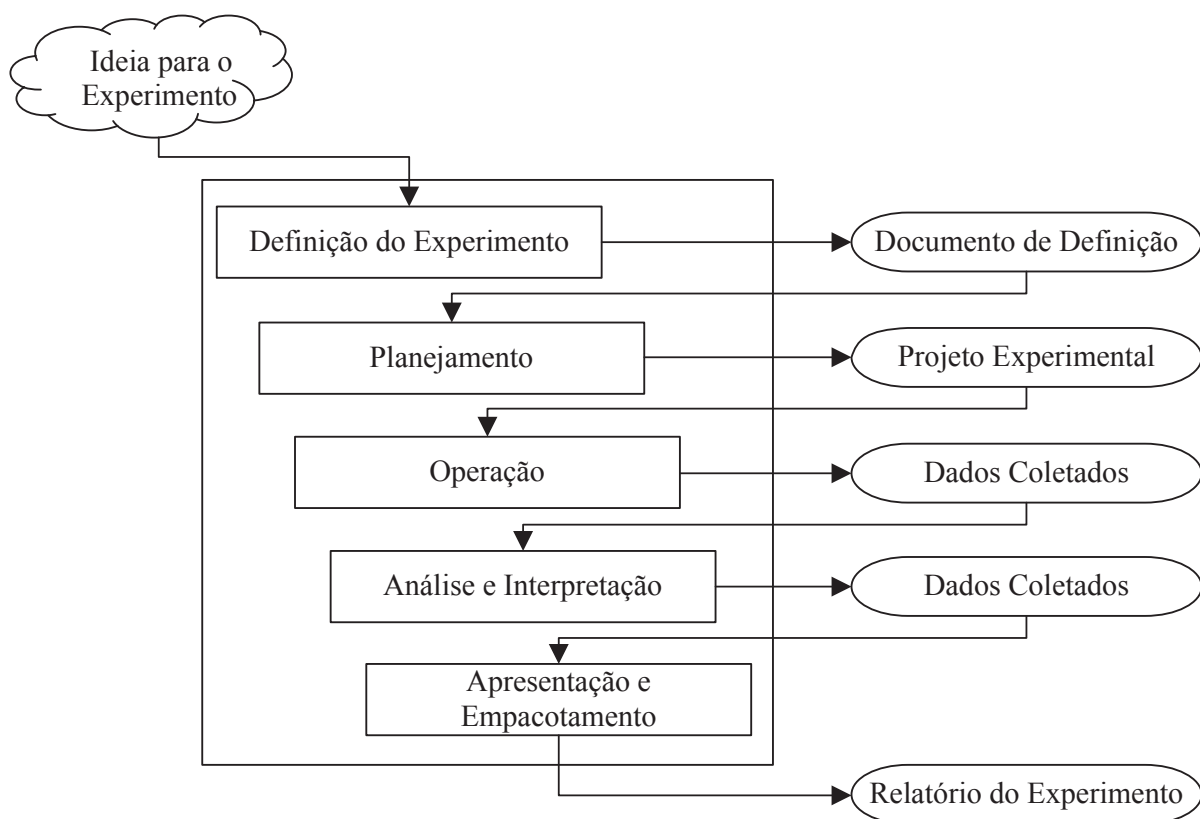


Figura 2.3: Visão da atividade de definição, adaptado de Wohlin et al. (2012)

O processo de experimentação pode ser dividido em cinco atividades: **definição**, **planejamento**, **operação**, **análise e interpretação** e **apresentação e empacotamento**. As cinco atividades são descritas em detalhes (Wohlin et al., 2012).

2.2.4 Atividade 1 - Definição

O objetivo desta atividade é conduzir um experimento, que, no entanto é tido como uma tarefa de trabalhos intensos. Na Figura 2.4 é demonstrado como a fundamentação do experimento é determinada. Se a fundamentação não for devidamente estabelecida pode ser necessário um retrabalho, ou ainda mais comprometedor, o experimento não pode ser usado para se estudar o que era pretendido. Ainda nesta atividade é necessária a definição das metas de um experimento de acordo com o *framework* definido. Para tal, foi escolhido como modelo o *G.Q.M* – abordagem de melhoria com o modelo da medição baseado em níveis – Nível Conceitual (Goal), Nível Operacional (Question) e Nível Quantitativo (Metric) (Wohlin et al., 2012).

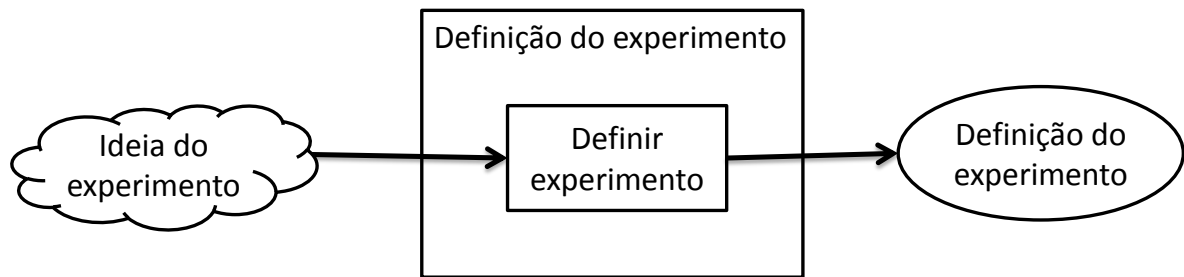


Figura 2.4: Visão da atividade de definição, adaptado de Wohlin et al. (2012)

2.2.4.1 Modelo de definição da meta

Este modelo tem como principal objetivo assegurar que aspectos fundamentais de um experimento sejam definidos antes de serem executadas as atividades de planejamento e a de execução. Após ser definida a meta do experimento (conforme esse modelo), a fundamentação é propriamente realizada e o modelo da meta é esboçado assim:

Análise <Objeto(s) de estudo>

com o objetivo do <objetivo>

com relação ao seu <foco de qualidade>

a partir do ponto de vista da <perspectiva>

no contexto do <contexto>

A seguir é detalhado cada item do esboço do modelo da meta (Wohlin et al., 2012).

O **Objeto do estudo** tem como principal objetivo identificar a entidade que é estudada no experimento. Tal item pode ser caracterizado como produtos, processos, recursos, modelos, métricas ou teorias. Como exemplos, tem-se o produto final, o desenvolvimento ou processo de inspeção ou o modelo de confiabilidade crescente.

O **Objetivo** define qual é a real intenção do experimento, podendo ser para avaliar o impacto de duas diferentes técnicas ou caracterizar a curva de aprendizagem de uma organização.

O **Foco da qualidade** pode ser eficácia, custo, confiabilidade, etc. É considerado o principal efeito sob o estudo no experimento.

A **Perspectiva** determina o ponto de vista a partir do qual os resultados do experimento são interpretados. Como exemplo, tem-se o desenvolvedor, o gerente de projeto, o cliente e o pesquisado.

O **Contexto** é o ambiente em que o experimento será executado e nele serão definidos (não detalhadamente) quais participantes estão envolvidos no experimento e quais objetos de software são usados. Os participantes podem ser caracterizados por experiência, tamanho do grupo, carga de trabalho e etc. Já os objetos são tamanho, complexidade, prioridade, domínio de aplicação e etc.

Quando se opta por estudos de objetos únicos, os mesmos são conduzidos por apenas um participante e um objeto. Os estudos de variação de multi objetos são conduzidos por apenas um participante por meio de um conjunto de objetos. Multi-testes dentro de estudos de objeto examina um objeto único por meio de conjunto de participantes. Estudos de objeto-participante examinam um conjunto de participantes e um conjunto de objetos.

Os tipos de experimentos citados anteriormente podem ser executados como um experimento ou como um *quasi*-experimento. Essa denominação *quasi*-experimento é atribuída pela falta de aleatoriedade de participantes ou de objetos. Sendo assim, o estudo de um único objeto é considerado um *quasi*-experimento se o participante único e o objeto não forem selecionados de maneira aleatória. Porém, será considerado um experimento se o participante e o objeto forem escolhidos de uma maneira aleatória.

2.2.5 Atividade 2 - Planejamento

Segundo [Wohlin et al. \(2012\)](#), na atividade de planejamento deve-se preparar como o experimento será conduzido, e para isso, seis etapas são executadas, baseadas na definição do experimento: *seleção de contexto, formulação das hipóteses e a seleção de variáveis, seleção dos participantes, projeto do experimento, instrumentação e avaliação da validade*. O processo de planejamento é repetido até uma leitura completa do projeto do experimento. Uma visão geral da atividade de planejamento é ilustrada na Figura 2.5.

2.2.5.1 Etapa de Seleção de Contexto

Para conseguir o máximo de resultados gerais em um experimento, o ideal seria utilizar projetos reais de softwares e que devem ser executados com uma equipe profissional grande. Porém, experimentar em projetos reais consiste em uma ameaça de riscos – por exemplo, um experimento que tenta provar que o novo método sendo examinado é melhor que outro, pode atrasar o desenvolvimento do projeto. Para evitar esse e outros riscos, existe a possibilidade de executar experimentos no formato *off-line*, em paralelo com os projetos reais. Outra possibilidade para minimizar custo é executá-los com a participação de estudantes, o que torna também mais fáceis para controlar e mais direcionados para um contexto do que projetos compostos por profissionais com mais experiência. O *contexto* do experimento pode ser definido

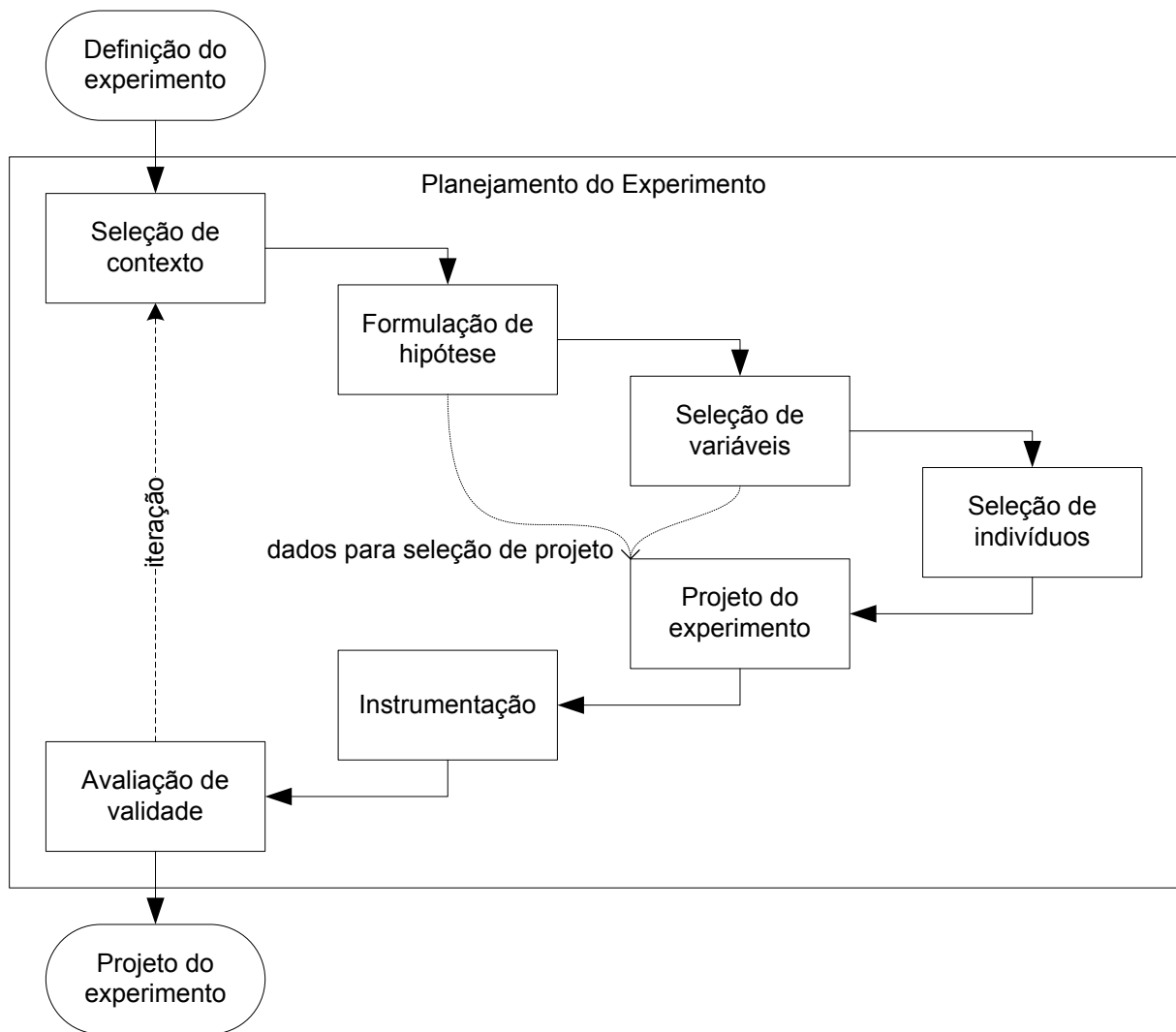


Figura 2.5: Visão da atividade de Planejamento, adaptado de Wohlin et al. (2012)

segundo quatro dimensões: ser *off-line* ou *on-line*; com estudantes ou profissionais; com problemas fictícios ou reais; com focos específicos ou gerais.

2.2.5.2 Etapa de Formulação e estabelecimento de hipótese

Testar uma hipótese é o objetivo da análise estatística de um experimento. A hipótese deve ser estabelecida formalmente e, durante a execução do experimento, os dados são coletados e, posteriormente, usados para rejeitar a hipótese. Caso a hipótese seja rejeitada, então as conclusões podem ser obtidas. Wohlin et al. (2012) ressalta que é na atividade de planejamento que a definição do experimento é formalizada em hipóteses: uma *nula* e uma *alternativa*.

Na *hipótese nula* (identificada por H_0) é estabelecido que não existam tendências reais subjacentes ou padrões no experimento; as únicas razões para as diferenças observadas são coincidências. A *hipótese nula* é a que o experimentador deseja rejeitar com a mais alta relevância possível.

A *hipótese alternativa* (identificada por H_a , H_1 , etc) é aquela que tentará fazer com que a *hipótese nula* seja rejeitada. Um exemplo dessa hipótese é que programadores que usam a ferramenta *Eclipse* tem melhor produtividade que quando usam o *Netbeans*.

Para avaliar o resultado de um experimento existem vários testes estatísticos diferentes e todos eles se baseiam em que as hipóteses devem ser formuladas antes que os testes estatísticos sejam escolhidos e exibidos.

Os testes de hipóteses estão sujeitos a alguns tipos de erro identificados como: o *Erro-Tipo-I* e o *Erro-Tipo-II*. O *Erro-Tipo-I* pode ocorrer quando um teste estatístico indicar que existe um padrão ou relacionar que o mesmo não tem nenhum padrão real. Como exemplo, pode-se demonstrar que a chance de cometer o Erro-Tipo-I é a probabilidade de rejeitar a H_0 apesar de provar que usar o *Eclipse* e o *Netbeans* encontre a mesma produtividade.

$$P(\text{Erro-Tipo-I}) = P(\text{rejeita } H_0 \mid H_0 \text{ verdadeira})$$

O *Erro-Tipo-II* ocorre quando um teste estatístico não indica um padrão ou quando relaciona que o mesmo não tem nenhum padrão real. Como exemplo, pode-se demonstrar que a chance de cometer o Erro-Tipo-II é a probabilidade de não rejeitar a H_0 mesmo que prove que usar *Eclipse* é mais produtivo do que usar *Netbeans*.

$$P(\text{Erro-Tipo-II}) = P(\text{não rejeita } H_0 \mid H_0 \text{ falsa})$$

O tamanho dos erros depende de fatores diferentes. Um exemplo é a habilidade do teste estatístico em revelar um padrão verdadeiro nos dados coletados. Isso se refere à força de um teste (Wohlin et al., 2012), que é a probabilidade de o teste revelar um padrão verdadeiro se a H_0 for falsa. Um experimentador deve escolher um teste com uma força tão alta quanto possível e a força pode ser expressa da seguinte maneira:

$$\text{Força} = P(\text{rejeita } H_0 \mid H_0 \text{ falsa}) = 1 - P(\text{Erro-Tipo-II})$$

2.2.5.3 Etapa de Seleção das variáveis

A escolha das *variáveis (dependentes e independentes)* de um experimento deve ocorrer antes que qualquer projeto possa começar.

As *variáveis independentes* podem ser controladas e mudar no decorrer do experimento. Para escolha dessas variáveis é necessário o domínio de conhecimento; as mesmas devem possuir algum efeito na *variável dependente* e devem ser controladas; também inclui escolher as escalas de medida, a extensão das variáveis e os níveis específicos nas quais os testes serão feitos.

O efeito dos tratamentos é medido em uma ou nas *variáveis dependentes*, porém, normalmente só há uma *variável dependente* e que deve ser derivada diretamente da hipótese. Não é comum para este tipo de variável mensurá-la e como solução para isso pode-se medi-la de maneira indireta que deve ser cuidadosamente validada por afetar o resultado do experimento. Após a escolha das *variáveis dependentes*, a hipótese pode ser refinada e isto implica que a escala de medida e a extensão das variáveis sejam determinadas.

Tanto as *variáveis independentes* e as *dependentes* geralmente são escolhidas simultaneamente ou numa ordem contrária.

2.2.5.4 Etapa de seleção dos participantes

A seleção dos participantes está ligada diretamente à generalização dos resultados do experimento. A fim de generalizar os resultados da população desejada, a seleção deve representar essa população. Amostra da população é como é chamada a seleção dos participantes e tal amostra pode ser uma probabilidade ou uma não probabilidade. A principal diferença entre as duas é que na probabilidade a seleção de cada participante é conhecida e a não probabilidade, é desconhecida (Wohlin et al., 2012).

Existem algumas técnicas de amostras de probabilidade, a seguir descritas:

- amostra simples de aleatoriedade: os participantes são selecionados de uma lista da população ao acaso, randomicamente.
- amostra sistemática: o primeiro participante é selecionado da lista da população aleatoriamente, e cada um dos demais participantes são selecionados da lista.
- amostra aleatória estratificada: a população é dividida em um número de grupos ou estratos (classes) com uma distribuição conhecida entre os grupos. A amostra aleatória é então aplicada dentro do estrato.

Existem algumas técnicas de amostras de não probabilidade, a seguir descritas:

- amostra conveniente: são selecionadas as pessoas mais próximas e as mais convenientes.
- amostra por cota: esse tipo de amostra é usado para obter participantes a partir de vários elementos de uma população. Amostra de conveniencia é normalmente usada por cada elemento.

Wohlin et al. (2012) afirmam que quando se trata de resultados generalizados, o tamanho da amostra também causa impacto, pois, quanto maior

a amostra for, menor se torna o erro quando generalizar os resultados. O tamanho da amostra também está ligado diretamente com a força do teste estatístico. E, para escolher o tamanho da amostra, existem alguns critérios a serem analisados como o de que se houver grande variedade na população, um tamanho de amostra maior é necessário e de que a análise dos dados é necessária para tal tomada de decisão.

2.2.5.5 Etapa do projeto do experimento

Para tirar conclusões significativas a partir de um experimento, é aplicada a análise estatística nos dados coletados para interpretar os resultados. Para se ter um melhor aproveitamento do experimento, ele deve ser cuidadosamente planejado e projetado. Quais análises estatísticas podem ser aplicadas e as escalas de medição utilizadas dependem diretamente do projeto escolhido.

Deve-se levantar qual problema será investigado no experimento e com isso analisar todas as ideias sobre os objetivos do experimento. Para alcançar uma compreensão melhor e ajudar a solução final necessita-se de uma definição clara.

Um experimento é composto por um conjunto de testes dos tratamentos que devem ser muito bem planejados e projetados. É no projeto do experimento que os testes são descritos, organizados e administrados. A escolha do projeto afeta a análise estatística e o contrário também ocorre. Para tal escolha, um dos cuidados é analisar a hipótese para ver qual análise estatística terá que ser escolhida para demonstrar ou rejeitar a *hipótese nula*. Ainda como parte do projeto do experimento é preciso determinar quantos testes poderão ser feitos para se obter a certeza de que o efeito do tratamento é visível. Um projeto adequado também forma a base para permitir a replicação.

2.2.5.6 Princípios gerais de projeto

Em um projeto de um experimento vários aspectos precisam ser analisados e considerados, como a aleatoriedade, bloqueio e equilíbrio. Em geral, quase todos os projetos de experimentação utilizam uma combinação desses aspectos. Em seguida serão explicados cada um dos três aspectos de um projeto.

1. Aleatoriedade: Todos os métodos estatísticos utilizados para analisar os dados exigem que as observações sejam a partir de variáveis aleatórias independentes. Para tanto, utiliza-se a aleatoriedade, que vigorará sobre a atribuição dos objetos, participantes e em que ordem os testes são realizados. A randomização é usada para mediar o efeito de um fator que pode de um outro modo estar presente. A randomização é também utilizada para seleccionar indivíduos que são representativos da população de interesse. [Wohlin et al. \(2012\)](#) consideram um dos mais importantes.

Tem-se como exemplo que os desenvolvedores da empresa que farão a seleção dos participantes do experimento, por meio da seleção aleatória dos que estiverem disponíveis. A atribuição para cada tratamento (tipo de projeto) também será selecionado aleatoriamente.

2. Bloqueio: às vezes existe um fator que provavelmente tenha um efeito sobre a resposta, mas o interesse na verdade é observar o efeito. Se o efeito do fator é conhecido e controlável, é possível utilizar a técnica citada, o bloqueio, que por sua vez é utilizado de maneira sistemática para eliminar o efeito que não se quer mais, para que se possa depois efetuar a comparação entre os tratamentos. O bloqueio é usada para eliminar o efeito indesejado no estudo e, por conseguinte, os efeitos entre os blocos não são estudados. Esta técnica aumenta a precisão do experimento. (Wohlin et al., 2012).

Tem-se como exemplo que dentre os participantes escolhidos para este experimento alguns têm diferentes experiências em um determinado assunto estudado e outros participantes não têm. Para minimizar esse efeito, os participantes com experiência são agrupados e em outro grupo os que não têm experiência no assunto do experimento.

3. Equilíbrio: um experimento se torna equilibrado se ocorrer atribuição de cada tratamento para um número igual de participantes. O equilíbrio não é algo necessário em um experimento, porém, se torna desejável, pois simplifica e fortalece a análise estatística das informações.

Exemplo: o experimento utiliza um projeto equilibrado, ou seja, há o mesmo número de pessoas em cada grupo.

2.2.5.7 Etapa de Instrumentação

Dentro de um experimento existem os chamados instrumentos que podem ser de três tipos: objetos, diretrizes e instrumentos de medição. Também na atividade de planejamento de um experimento é que os instrumentos são escolhidos. A seguir, a explicação de cada um deles.

1. Objetos: podem ser considerados objetos as especificações ou documentos de código. Em um experimento de inspeção, como exemplo, deve-se usar um documento de código com um número conhecido de defeitos, pois a partir desse documento com os defeitos identificados pode-se gerar um posterior.
2. Diretrizes: incluem, por exemplo, descrições de processo e listas de checagem (*checklists*) (necessárias para guiar os participantes no experimento). Se no experimento serão comparados métodos diferentes, algumas diretrizes para os métodos precisam ser desenvolvidas, porém,

além das diretrizes, os participantes do experimento devem receber treinamento dos métodos que serão utilizados.

3. Medição: uma medição só pode ser executada em um experimento por meio de dados que foram coletados de formulários manuais ou entrevistas com os participantes do experimento. O planejamento da tarefa é elaborar formulários com perguntas sobre o assunto abordado e depois validar os formulários e as questões com algumas pessoas que tenham um conhecimento e habilidades similares como as dos participantes do experimento.

A instrumentação tem como finalidade proporcionar meios para demonstrar e monitorar o experimento sem prejudicar o controle do experimento. Por fim, é preciso deixar bem claro que se a instrumentação prejudicar o resultado do experimento, o mesmo será inválido. Portanto, independente de como o experimento é instrumentado, os resultados do experimento devem ser os mesmos.

2.2.5.8 Etapa de Avaliação da validade

A validade dos resultados de um experimento é algo fundamental. Por isso, na atividade de planejamento do projeto do experimento faz-se necessário planejar a validade adequada dos resultados do experimento, isto é, os resultados devem ser válidos em primeiro lugar para a população das quais a amostra é tirada e em segundo plano pode haver algum interesse em generalizar tais resultados de uma população maior. Porém, é importante deixar claro que validade adequada não se aplica a maioria da validade geral.

Um experimento que é conduzido dentro de uma empresa pode ser projetado para obter respostas exclusivas para essa empresa, caso validados para a mesma. Porém, se for necessário explorar conclusões mais gerais deste experimento, a validade deve cobrir um escopo mais geral também.

Há diferentes esquemas de classificação para diferentes tipos de ameaças para a validade de um experimento. Alguns autores definem até quatro tipos de ameaças para a validade dos resultados experimentais: validade de conclusão, validade interna, validade de construção e validade externa.

Cada uma das quatro categorias citadas acima explica uma questão metodológica na experimentação. A seguir são explicadas as conclusões sobre a teoria definida nas hipóteses baseadas nas observações. Destas conclusões, há um tipo de ameaça para a validade dos resultados:

1. Validade de conclusão: tem como foco o relacionamento entre o tratamento e o resultado, na qual se busca uma certeza que há uma relação estatística, ou seja, verifica, por meio do tratamento se os dados gerados pelas *variáveis independentes* são relevantes para o experimento.

2. Validade interna: se uma relação é observada entre o tratamento e o resultado, verifica a certeza de que é uma relação causal e não é o resultado da influência de outro fator que não é controlado ou mesmo não foi medido, ou seja, se o tratamento causa o resultado (o efeito) (Kitchenham et al., 2002).
3. Validade da construção: nesta validade o foco é o relacionamento entre teoria e observação, na qual verifica se o relacionamento entre a causa e o efeito é causal, é preciso fazer a validação de que o tratamento reflete bem a construção da causa e de que o resultado reflete bem a construção do efeito.
4. Validade externa: A validade externa está preocupada com a generalização. Se existe uma relação causal entre a construção da causa, e o efeito, pode o resultado do estudo ser generalizado fora do âmbito do experimento? Existe uma relação entre o tratamento e os resultados?

A validade de conclusão tem como foco questões do tipo escolha de testes estatísticos, escolha de tamanhos de amostras, cuidado na implementação e medição de um experimento, que afetam diretamente na habilidade para projetar a conclusão correta sobre relações entre os tratamentos e os resultados de um experimento.

2.2.6 Atividade 3 - Operação

Nesta atividade, o experimento é colocado em prática, no qual os tratamentos são aplicados aos participantes. A seguir, na Figura 2.6, são ilustradas as fases dentro da atividade de operação.

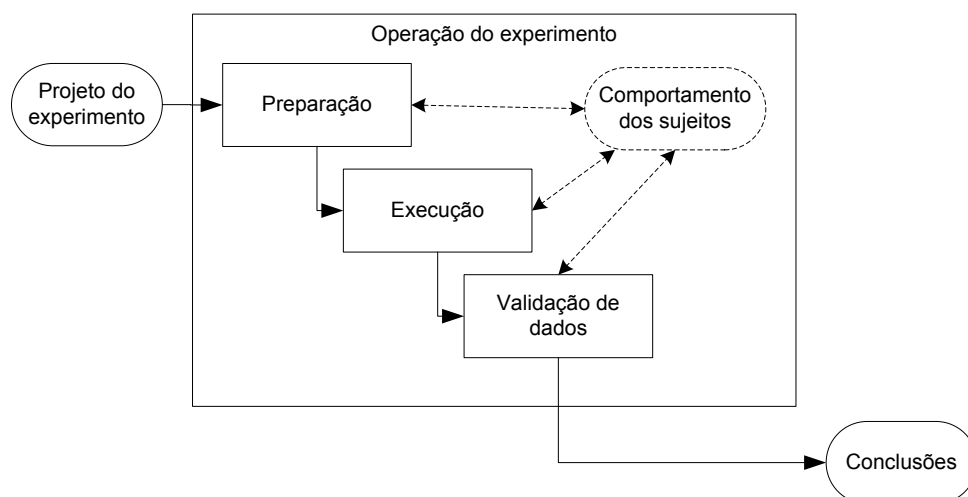


Figura 2.6: Visão das três etapas da atividade de Operação, adaptado de Wohlin et al. (2012)

Existem três etapas na atividade de operação do experimento: preparação (sujeitos e objetos são preparados); execução (os sujeitos realizam suas tarefas

de acordo com diferentes tratamentos e dados coletados); e validação de dados (os dados coletados são validados). Essas três etapas sofrem influências diretas do comportamento dos participantes do experimento, sendo assim, é preciso conscientizá-los a participar do mesmo com grande seriedade e a seguir a ordem de execução.

A primeira etapa é a de **preparação** na qual os participantes são escolhidos e informados sobre o experimento. Também ocorre a preparação do material necessário. O experimentador do experimento tem que motivar os participantes de maneira a alocá-los, se possível, em tarefas nas quais ele já está habituado, por exemplo, escrever linhas de código em uma linguagem que o participante já conheça. É necessário, também, verificar se os instrumentos e ferramentas estão preparados, bem como se os participantes estão aptos para utilizar tais instrumentos.

A segunda etapa é a de **execução**. O experimento pode ser executado em várias reuniões ou apenas em uma, sendo elas curtas ou longas, com os participantes agrupados ou isolados ou com formação de grupos isolados; pode ser executado no formato presencial ou não presencial (as maneiras como a execução será realizada depende do objetivo do experimento). A coleta dos dados executados pode ser realizada diretamente pelos participantes ou pelo experimentador, de maneira manual ou por meio de ferramentas informatizadas.

A terceira etapa é a de **validação dos dados**. Nesta o experimentador tem a função de verificar se houve participação efetiva dos participantes por meio de verificação e validação dos dados informados por eles, de maneira que os resultados do experimento sejam válidos. Isto é muito importante, pois se algum participante teve algum problema de interpretação do que fazer, ou não participou com motivação ou não foi bem preparado na primeira etapa, os resultados do experimento podem ser distorcidos e até invalidados (o experimentador deve prover uma boa preparação dos participantes com o máximo de esclarecimentos possíveis).

Em alguns casos, após a coleta dos dados, o experimentador pode apresentá-los aos participantes de maneira que eles possam verificar se concordam com o que foi coletado e podem até indicar mudanças na maneira da aplicação do experimento.

2.2.7 Atividade 4 - Análise e Interpretação

Existem muitos tipos de modelos experimentais. É útil conhecer e compreender os vários tipos de projetos que possam vir a utilizar na pesquisa de engenharia de software, uma vez que o tipo de projeto restringe o tipo de análise que pode ser feita e, portanto, os tipos de conclusões que podem ser tiradas. A amostragem também reforça o projeto e restringe a análise. A mai-

oria dos projetos de pesquisa em engenharia de software são baseados em duas relações simples entre os fatores: cruzamento (*crossing*) e nidificação. O projeto de um experimento pode ser expresso em uma notação que reflete o número de fatores e como eles se relacionam com os diferentes tratamentos, expressando assim o projeto em termos de fatores, chamado de planejamento fatorial, que fornece quantas diferentes combinações de tratamentos são necessárias (Pfleeger, 1995).

Na atividade de análise e interpretação, os dados coletados na atividade de execução são interpretados pelo pesquisador para que se possa chegar a uma conclusão. Para tal conclusão, os dados podem ser interpretados de maneira quantitativa por meio de três técnicas: estatística descritiva, redução de conjunto de dados e testes de hipóteses, porém, seguindo a relação de cruzamento entre os fatores.

Na Figura 2.7, é ilustrada a atividade de análise e interpretação dos dados e as três técnicas. Na primeira técnica, a estatística descritiva tem a função de caracterizar os dados informando tendências, dispersões, desvios e outros dados estatísticos. Na segunda técnica, a redução de conjunto de dados tem a função de reduzir os dados para um conjunto de dados válidos, excluindo dados falsos ou que apresentem alguma anomalia. E na terceira técnica, os testes de hipóteses avaliam estatisticamente as hipóteses do experimento (Wohlin et al., 2012).

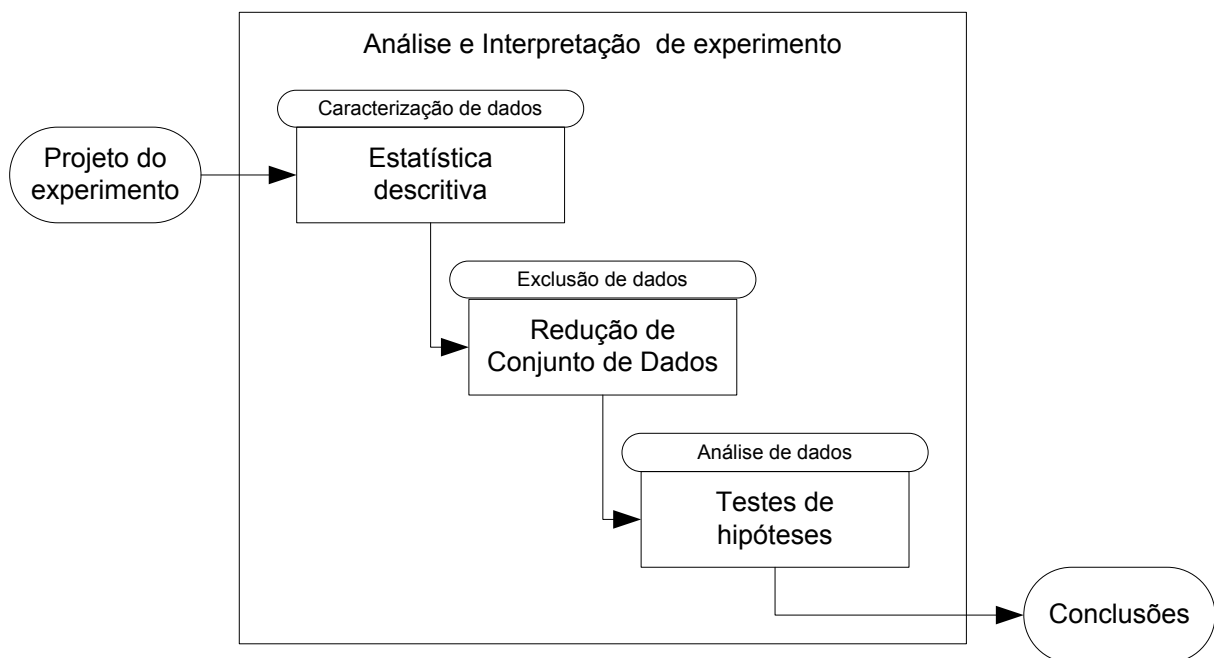


Figura 2.7: Visão das três etapas de uma análise e interpretação quantitativa, adaptado de Wohlin et al. (2012)

2.2.7.1 Estatística descritiva

Conseguir demonstrar como os dados estão distribuídos é um dos principais objetivos da técnica estatística descritiva, além de expor e representar graficamente alguns aspectos dos dados (como exemplo, em que escala os dados são positivos e o quanto concentrados ou espalhados eles estão). Dentro desta técnica existem ainda alguns tipos de medidas: medidas de tendência central, medidas de dispersão e medidas de dependência, que estão mais detalhadas em (Wohlin et al., 2012).

2.2.7.2 Redução de Conjunto de Dados

Aplicados os métodos estatísticos, a etapa seguinte dentro da atividade de análise e interpretação é a necessidade de fazer uma verificação que confirme se algum dado apresentado estatisticamente realmente representa uma informação correta e ou esperada.

Erros sistemáticos em um conjunto de dados podem ocorrer, ou também podem ocorrer discrepâncias (*outlier*), que significa que o ponto de dados é muito maior ou muito menor do que se poderia esperar comparando graficamente com os outros pontos de dados. Para melhor ilustrar e uma boa maneira de identificar tais discrepâncias, é utilizar gráficos de dispersão como segue ilustrado na Figura 2.8 ou até utilizar os *box plots* (diagramas de caixa).

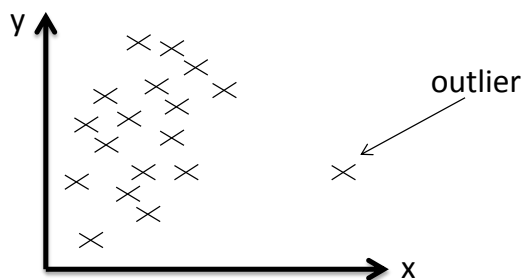


Figura 2.8: Gráfico de dispersão - detecção de um *outlier*, adaptado de Wohlin et al. (2012)

Segundo Wohlin et al. (2012), após a Identificação das distorções, é necessário verificar se ela foi causada por um evento raro (de maneira que não se deve deixar acontecer novamente no experimento), podendo então removê-la do conjunto de dados. Porém, se a distorção foi causada por uma variável não avaliada anteriormente, pode-se decidir em removê-la ou não, levando em conta os cálculos e modelos a serem refeitos. Em outros casos, se for identificado redundâncias dos dados, às vezes se tornará ineficaz analisar tais dados caso a redundância seja grande demais.

2.2.7.3 Testes de hipóteses

O objetivo dos testes de hipóteses é verificar a possibilidade de rejeitar uma certa *hipótese nula* (H_0), baseado em uma amostra de alguma distribuição estatística. A *hipótese nula* descreve algumas propriedades de distribuição na qual a amostra é extraída e o experimentador deseja rejeitar com uma certa significância que ela é verdadeira.

Conforme [Wohlin et al. \(2012\)](#), as hipóteses compreendem as distribuições estatísticas e, a partir disso, é necessário escolher um tipo de teste que será executado com o objetivo de verificar se é possível rejeitar a *Hipótese Nula* apresentada no experimento. Para tal escolha, o experimentador tem a sua disposição diversos tipos de testes, técnicas, métodos e cálculos estatísticos como, por exemplo, *I-Test*, *Mann-Whitney*, *F-test*, *T-Test pareado*, *Wilcoxon*, *Teste de sinal*, *ANOVA*, *Kruskal-Wallis* e *Chi-2*.

Ainda nos testes de hipóteses, existe a possibilidade de o experimentador elaborar outras hipóteses para desenvolver outras conclusões conforme outros resultados, isso independentemente de rejeitar a *hipótese nula* ou não. Se as hipóteses são rejeitadas, pode-se registrar conclusões com relação à influência das *variáveis independentes* sobre as *variáveis dependentes*, uma vez confirmada a validade do experimento. Porém, se em contrapartida o experimento não rejeitou a *hipótese nula*, não se pode concluir que as *variáveis independentes* influenciaram sobre as *dependentes*. Assim, a única coisa que fica evidente é que não há diferenças estatisticamente significativas entre os tratamentos.

Portanto, nesta atividade é indicado se o experimento executado para provar que um novo método é “melhor” quando comparado com um outro método existente e, portanto, pode ser utilizado em projetos de desenvolvimento de software futuros, ou, se mais experimentos serão necessários para provar que sim. ([Wohlin et al., 2012](#)).

2.2.8 Atividade 5 - Apresentação e Empacotamento de um Experimento

Após a realização da atividade de análise e interpretação do experimento, é importante apresentar as descobertas de uma maneira adequada, por exemplo, artigos em conferência, relatórios de tomada de decisão, empacotando para replicação do experimento, como material educacional, entre outros. É essencial não esquecer aspectos importantes ou informações necessárias, a fim de permitir que outros reproduzam ou aproveitem a experiência e os conhecimentos adquiridos por meio do experimento conduzido.

Porém, [Amaral e Travassos \(2003\)](#) afirmam ser melhor que ela não seja executada apenas como etapa final, e sim ao longo do processo de experimen-

tação, aumentando com isso a fidelidade do pacote de laboratório à condução do experimento (Garcia, 2006). Na Figura 2.9 é ilustrado esse esquema.

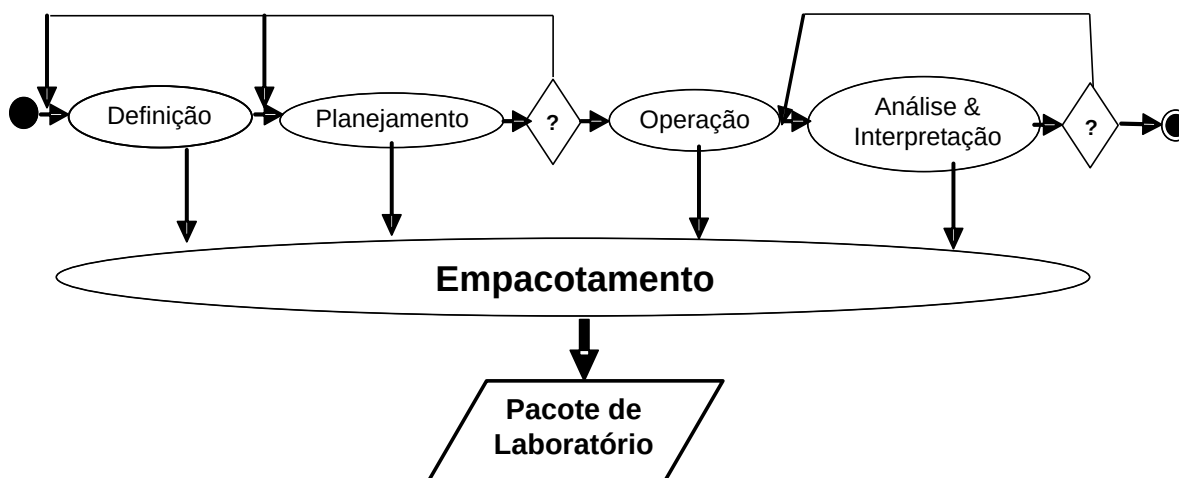


Figura 2.9: Fase de empacotamento do processo de experimentação, adaptado de Amaral e Travassos (2003)

Wohlin et al. (2012) propõem um esquema para um relatório de ensaio a ser utilizado na preparação de, por exemplo, um artigo para publicação ou pacote de laboratório, contendo:

- Introdução: um texto curto descritivo contendo uma visão e motivação do experimento conduzido.
- Declaração do problema: uma descrição detalhada do problema, incluindo informações sobre como o problema foi observado e a razão da tal investigação, além de descrever os trabalhos envolvidos.
- Planejamento do experimento: uma descrição detalhada do *contexto* do experimento incluindo as hipóteses, as *variáveis independentes* e as *variáveis dependentes* e o que elas mediram propriamente dito, os participantes, o projeto do experimento, e uma explanação da estratégia para medir e analisar os dados.
- Operação do experimento: uma descrição detalhada de como a operação (execução) do projeto foi preparada, desde formulários, ambientes, treinamento com os participantes. Em seguida, também relatar como foi a execução do experimento e como os dados foram coletados e validados.
- Análise dos dados: neste item ocorre a apresentação da análise dos dados, em que os cálculos são descritos juntamente com as suposições para a utilização de algum modelo de análise específico. Informações sobre tamanhos de amostras, níveis de significância e aplicação de testes também são pré-requisitos para a análise.

- Interpretação dos resultados: resultados crus de uma análise não são suficientes para fornecer compreensão dos resultados e conclusões do experimento, e daí então a ideia de se ter uma interpretação, que inclui a rejeição das hipóteses ou a incapacidade para rejeitar as *hipóteses nulas*. Uma interpretação resume como os resultados de um experimento podem ser usados.
- Discussões e conclusões: apresentar discussões a respeito dos resultados e conclusões do experimento inteiro juntamente com os resultados, problemas, desvios dos planos e outros. Os resultados também devem ser relacionados com trabalhos relatados anteriormente. Além disso, é importante abordar as semelhanças e as diferenças entre os resultados descobertos e ideias futuras para replicação do experimento.
- Apêndices: informações não vitais para a apresentação podem ser incluídas em apêndices. Se a intenção é produzir um pacote de laboratório, todo o material usado no experimento pode ser incluído no apêndice.

2.3 Considerações Finais

Neste capítulo foram apresentados assuntos relacionados à Experimentação em Engenharia de Software, tais como, definição dos principais conceitos e termos utilizados na execução de estudos experimentais em Engenharia de Software, abordagem sobre experimentos controlados, abordagem sobre o processo de experimentação e suas etapas tendo como base na teoria proposta por [Wohlin et al. \(2012\)](#). Os conceitos e processo experimental apresentado são apoiados pela ferramenta proposta e desenvolvida neste trabalho de mestrado.

No capítulo seguinte é apresentada uma contextualização sobre a transferência de conhecimento e replicações de experimentos por meio de pacotes de laboratórios, uma visão geral sobre ferramentas e ambientes de apoio à Experimentação em Engenharia de Software já desenvolvidos por grupos de pesquisas em Engenharia de Software e o conceito de ontologias que apoiam a Engenharia de Software Experimental, mais especificamente a *ExperOntology* elaborada por [Garcia et al. \(2008\)](#) que visa a empacotar os estudos de acordo com um padrão, para viabilizar a integração dos resultados e facilitar a revisão dos pacotes de laboratório.

Empacotamento de Experimentos Controlados

3.1 Considerações Iniciais

A execução de experimentos em Engenharia de Software gera um grande volume de dados no decorrer de cada atividade desenvolvida. Uma ferramenta computacional para prover apoio às fases de um experimento, proporciona auxílio tanto para pesquisadores quanto para os participantes no registro de dados.

Os dados utilizados e coletados em cada fase do processo de experimentação são registrados, gerando como produto o que é chamado de **pacote de laboratório**, por meio de uma fase denominada empacotamento. O pacote de laboratório tem como finalidade facilitar o acesso de outros pesquisadores aos resultados do estudo nele armazenados. Tem, também, o objetivo de facilitar a replicação de experimentos.

Neste capítulo, uma revisão bibliográfica é apresentada para caracterizar a necessidade do empacotamento (criação de Pacotes de Laboratório) e propostas existentes na literatura. Assim, inicialmente na Seção 3.2 são descritos conceitos sobre a transferência de conhecimento por meio de pacotes de laboratórios gerados nos experimentos executados, bem como as replicações dos mesmos. Na Seção 3.4 são relatados estudos similares sobre o desenvolvimento de ferramentas e ambientes de apoio à experimentação por outros grupos de pesquisas. Na Seção 3.3 é apresentado o uso de ontologias em Ciência da Computação, mais especificamente na Engenharia de Software Experimental, sendo descrita uma ontologia para experimentos controlados.

3.2 A transferência de conhecimento e as replicações de experimentos

Com a execução de vários estudos independentes sobre um determinado tópico, é gerado conhecimento sobre sistemas de domínios diferentes. A partir disso, surge a necessidade de gravar dados e transferir tais conhecimentos (Garcia et al., 2008). A experimentação pode ser utilizada como uma ferramenta para suporte de transferência de tecnologia, e para tal o uso de pacotes de laboratório podem ser vistos como um pacote de conhecimento sobre estudos controlados. Contudo, tais pacotes devem estar habilitados para suportar alterações para evolução dos experimentos por meio de estudos múltiplos. Um pacote de laboratório possui a descrição de um experimento, além disso fornece material para a replicação, destaca oportunidades para a variação e constrói um contexto combinando resultados de diferentes tipos de tratamentos experimentais (Linkman e Rombach (1997)). Segundo Garcia (2006), em um pacote de laboratório é necessário estar registrado de maneira correta o procedimento de condução do experimento por completo, para com isso possibilitar a condução de replicações e a utilização dos conhecimentos adquiridos ali registrados.

De acordo com Travassos et al. (2002), com a replicação, os pesquisadores adquirem conhecimento adicional a respeito dos conceitos estudados. Para que se possa replicar experimentos, é necessário que seu empacotamento seja realizado apropriadamente. Uma replicação em contextos diferentes sujeita o experimento a variações, entre elas a experiência e a habilidade de cada participante selecionado, bem como o ambiente cultural onde o experimento é executado (Shull et al., 2003, Mendonça et al., 2008), possibilitando a verificação da validade de uma hipótese com um nível maior de confiança. Baseando-se em tais variações, reserva-se a generalização das conclusões do estudo, pois as condições direcionadas durante a execução limitam a população à qual se aplicam as conclusões. Com isso, a influência dessas variações deve ser estudada ou os resultados dos experimentos devem ser reproduzidos externamente. Brooks et al. (2008) citam que a confiança nos resultados de um experimento aumenta a medida que pesquisadores vão reproduzindo-o com sucesso.

Scatalon et al. (2011a) citam que pesquisadores enfrentam dificuldades ao rever o pacote de laboratório, como problemas no compartilhamento de conhecimento entre os grupos de pesquisa, falta de padronização para a integração do conhecimento de um estudo isolado com um conhecimento comum. Mendonça et al. (2008) propuseram um *framework* denominado FIRE (*Framework for Improving the Replication of Experiments*), definido como uma estrutura

composta de atividades que visa a indicar aspectos que possam facilitar a replicação. Essa estrutura é composta por dois ciclos, conforme ilustrado na Figura 3.1.

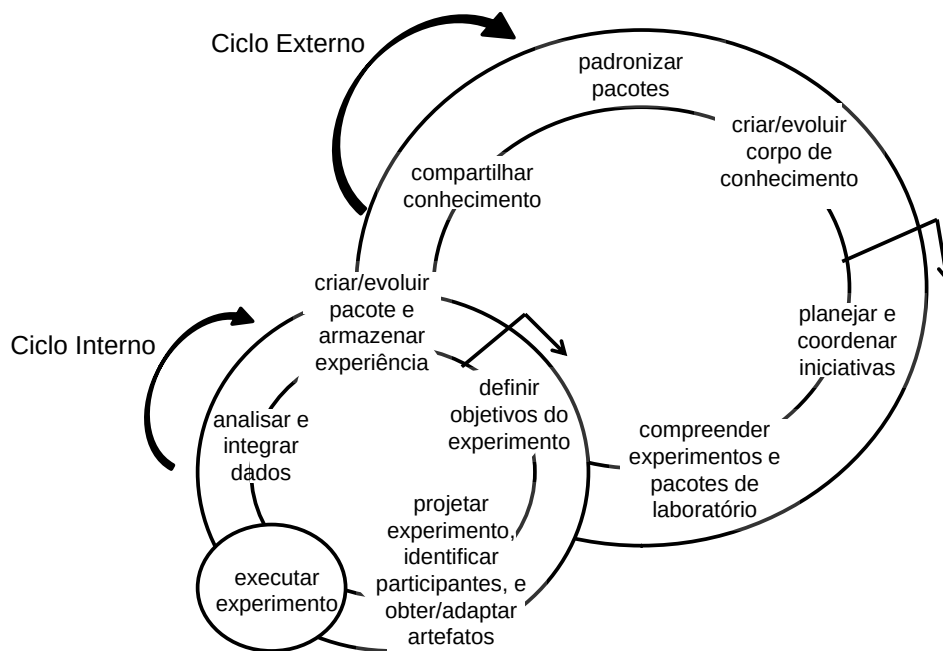


Figura 3.1: Ciclos do FIRE, adaptado de [Mendonça et al. \(2008\)](#)

O ciclo interno contém atividades como planejamento, execução, empacotamento, que se referem à execução de um experimento isolado e o processo de aprendizagem dentro do grupo de pesquisadores do experimento. O ciclo externo refere-se à aprendizagem entre grupos de pesquisa e tem por objetivo a construção de um amplo corpo de conhecimento por meio de replicações planejadas com cuidado. Este ciclo lida com questões de entendimento de pacotes de laboratório, compartilhamento de conhecimento adquirido e sua evolução experimental, que servem para contribuir com a transferência de conhecimento para outros grupos com contextos diferentes e que queiram replicar o estudo.

O principal objetivo do ciclo externo é a aprendizagem entre grupos por meio de trocas de experiências. As replicações a serem executadas podem ocorrer em diferentes contextos, culturas e níveis de experiências diversificados, tornando muito importante a troca de informações sobre dificuldades encontradas entre os grupos de pesquisas, bem como os pontos positivos e negativos em suas execuções. Ainda outros pesquisadores podem utilizar em outros ambientes os dados e experiências registradas, por isso a necessidade de seguir os padrões de empacotamento, bem como a sua evolução na direção desse padrão (também uma atividade do FIRE).

O ponto chave para executar replicações é compartilhar conhecimento, e este se configura nas atividades de interface entre os ciclos do FIRE, no qual

é preciso tomar cuidado para o foco não ficar apenas no ciclo interno tornando os estudos isolados e sim contrário, integrá-los em um corpo comum de conhecimento ([Mendonça et al., 2008](#)).

3.3 Ontologia e seu uso no empacotamento

Na Ciência da Computação, a ontologia foi introduzida na representação do conhecimento, popularmente definida inicialmente por [Gruber \(1995\)](#), que a definiu como “uma especificação explícita de uma conceitualização”. Posteriormente, [Borst \(1997\)](#) acrescentou nesta definição a perspectiva de colaboração, reformulando-a como: “uma ontologia é uma especificação formal e explícita de uma conceitualização compartilhada” ([Rautenberg et al. \(2009\)](#)). De uma maneira generalizada, [Rautenberg et al. \(2009\)](#) citam que uma ontologia é considerada um importante meio de representar, formalizar e compartilhar conhecimento. Sendo assim, quando representado de maneira formal, o corpo do conhecimento se baseia em uma conceitualização explícita ou implícita e é composto por objetos, conceitos, outras entidades e as suas relações existentes na área de interesse. Tal conceitualização é considerada uma visão simplificada e abstrata do mundo em que se deseja representar para atingir algum propósito.

Se o conhecimento de um domínio for representado por meio de um formalismo declarativo, o conjunto de objetos envolvidos e que podem ser representados será chamado de universo de discurso. As relações descritas do conjunto de objetos são refletidas na descrição do vocabulário de representações que representa o conhecimento por meio de um programa baseado em conhecimento ([Gruber, 1995](#)).

No universo do discurso, as definições associam nomes de entidades, por exemplo, classes, funções, relações ou outros objetos, com texto de fácil interpretação que descreve o que os nomes significam e também os axiomas formais que restringem o entendimento e o uso bem formado desses termos ([Gruber, 1995](#)).

Uma ontologia é uma especificação legível por máquina. Ela pode ser classificada em termos de tipos e formalidades. Por tipos, são elas: ontologia de domínio, ontologia de tarefa, ontologia de senso-comum e meta-ontologia; e por formalidades são: informal, semi-informal, semiformal e rigorosamente formal.

Para o desenvolvimento de ontologias é essencial o uso de linguagens, porém, elas devem atender os requisitos mínimos estipulados pelo *WebOntology* (*WebOnt*), grupo de trabalho do W3C (*World Wide Web Consortium* – consórcio internacional com a finalidade de estabelecer padrões para a criação e a interpretação de conteúdos para a *Web*). Tais requisitos indicam uma visão geral

das funcionalidades que devem ser seguidas pela linguagem, de maneira que se tornem úteis em uma aplicação da *web* semântica. Seguem alguns requisitos: o compartilhamento, a interoperabilidade, a evolução, a expressividade e a escalabilidade da ontologia, a detecção de inconsistência, a compatibilidade com outros padrões, a facilidade de uso e a internacionalização. Algumas linguagens de ontologias mais usadas são: *XML e XML Schema*, *SHOE (Simple HTML Ontology Extensions)*, *RDF (Resource Description Framework) (RDF)* e *o RDF Schema (RDFS)*, *DAML (DARPA Agent Markup Language)* e *DAML-ONT*, *OIL (Ontology Interchange Language)*, *OWL (Web Ontology Language)* (Moreira, 2006).

Além de uma linguagem de desenvolvimento, é importante seguir as seguintes técnicas e métodos:

- Especificação: identificar e especificar o porquê a ontologia deve ser construída, quais são as suas finalidades e quem são os usuários finais.
- Conceitualização: construir um modelo conceitual no qual descreva os problemas e as soluções.
- Formalização: transformar um modelo conceitual em um modelo formal.
- Implementação: codificar a ontologia em uma linguagem formal.
- Manutenção: Gerar um guia ou manual atualizado sobre as ontologias para possíveis inclusões ou alterações das definições (Moreira, 2006).

O aumento de utilização de ontologias em Engenharia de Software ocorre devido à possibilidade de padronização, compartilhamento, entendimento e reuso comum de algum domínio de conhecimento que possa haver interação entre pessoas e computadores (Scatalon et al., 2011b, Moraes e Ambrosio, 2007).

3.3.1 *Exper*Ontology - uma ontologia para experimentos controlados

As ontologias podem ser usadas para permitir que vários pesquisadores tenham acesso a fontes heterogêneas de informações que são expressas usando diversos vocabulários ou formatos inacessíveis (de A. Biolchini et al., 2007). Na experimentação, a troca de conhecimento e experiência entre pesquisadores também é dificultada por não utilizarem um vocabulário comum. A partir disso, a comunidade *ISERN (International Software Engineering Research Network)* deu início à construção da primeira versão de um glossário de termos. A evolução desse glossário foi então amplamente estimulada pelo grupo eSEE (Lopes e Travassos, 2009b).

Em outra perspectiva, [Garcia et al. \(2008\)](#) investigaram o estabelecimento de uma ontologia para Engenharia de Software Experimental, focando em experimentos controlados, denominada *ExperOntology*. Esta visa a descrever formalmente os conceitos que compõem um pacote de laboratório e pode ser usada para facilitar o reúso e partilha dos pacotes de laboratórios ([Garcia et al. \(2008\)](#)).

A *ExperOntology* favorece que pesquisadores entendam e compartilhem melhor o conhecimento gerado ao longo da execução das famílias de experimentos. Também deve ser usada para estabelecer uma arquitetura de referência que apoie o desenvolvimento de ambientes/ferramentas para automatizar atividades de testes de software e relacionados com estudos experimentais.

[Garcia et al. \(2008\)](#) citam que a *ExperOntology* aborda os principais conceitos dos experimentos controlados, desde a definição até a análise dos resultados. Afirmam ainda que é importante considerar as evoluções do experimento, que também são armazenadas no pacote de laboratório. A partir da primeira execução do experimento, pode ser criado um pacote de laboratório, descrevendo o experimento original e fornecendo materiais para replicação. É importante ressaltar que ao analisar os conceitos instanciados a partir de um pacote de laboratório existente, outras pesquisas podem identificar oportunidades de variação e construir um contexto para a combinação de resultados de diferentes tipos de tratamentos experimentais.

A *ExperOntology* é apresentada como uma ontologia para Engenharia de Software Experimental, construída com base nos conhecimentos de seus pesquisadores e experiências na condução de experimentos controlados, principalmente na avaliação das técnicas V&V (Validação e Verificação). Esta ontologia é composta por dois níveis de refinamento, sendo que o primeiro se refere aos principais conceitos do experimento controlado e o segundo trata do refinamento da validação dos conceitos e do pacote de laboratório. A notação UML é utilizada para representar tal ontologia. Na condução de um experimento original, é gerado um pacote de laboratório. Uma replicação utiliza um pacote de laboratório de experimentos anteriores também como base para motivação de geração de um novo pacote de laboratório, como representado na Figura 3.2.

Tanto o experimento original quanto a replicação devem ser avaliados quanto à sua validade. Um experimento original é criado por um pesquisador, que tem o seu perfil (experiência) relacionado ao experimento como um parâmetro para definir uma possível ameaça à validade. No mesmo sentido, um replicador também tem o seu perfil associado com a replicação. É importante ressaltar que tanto o perfil do pesquisador e o do replicador podem influenciar positivamente ou negativamente a condução do experimento e/ou replicação e por isso devem ser levados em conta durante a análise dos resultados como

uma influência da validade do experimento. Um exemplo de influência negativa é a falta de experiência, devido à possibilidade de dificuldade em isolar os fatores de risco para a definição de um experimento. Na replicação, a falta de experiência também pode influenciar a fidelidade da execução do experimento original. Por outro lado, uma grande experiência é uma influência positiva, de maneira que isto minimiza o esforço para a definição do experimento e ajuda a analisar o pacote de laboratório para identificar oportunidades e combinar os resultados de diferentes tratamentos experimentais. A avaliação da validade é uma questão a ser tratada em todas as fases do processo de experimentação.

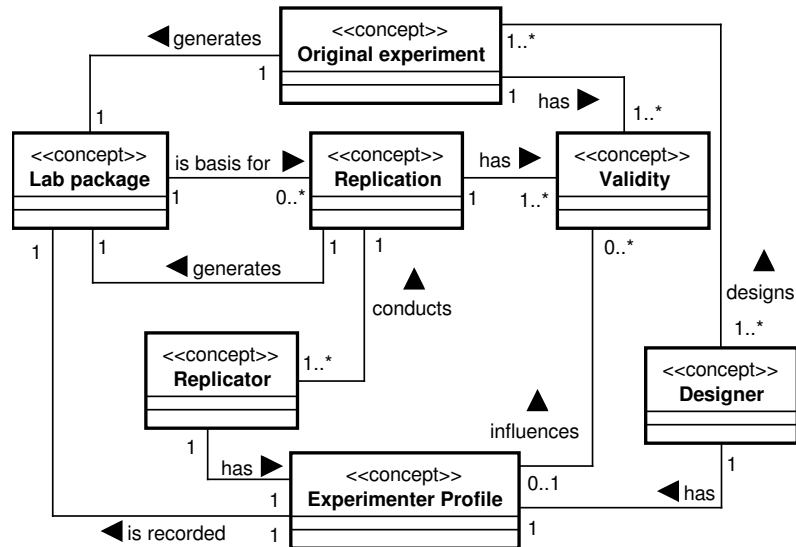


Figura 3.2: Ontologia para um Experimento Controlado, adaptado de [Garcia et al. \(2008\)](#)

Por meio do segundo nível é possível refinar a validação de todos os conceitos envolvidos na criação de um pacote de laboratório de um experimento controlado, como é representado na Figura 3.3.

Na ontologia apresentada na Figura 3.3 são definidos e instanciados os conceitos para um pacote de laboratório. A princípio, a *hipótese inicial* de um experimento controlado é definida. É constituída pelo objeto de estudo, de acordo com o propósito, sob o foco da qualidade e um contexto específico. Como exemplo de utilização de como um conceito instanciado é utilizado em todas as fases do processo de um experimento, será demonstrada a utilização do conceito *Hipótese Formalizada*.

Na atividade *Definição*, a medida que o objetivo do experimento é definido, a base para formulação das hipóteses também é estabelecida, incluindo a *Hipótese Inicial*, que em seguida na atividade *Planejamento* servirá de base para a criação das *Hipóteses Formalizadas* (*Hipótese Nula e Alternativa*). A partir das *Hipóteses Formalizadas*, o experimentador criará ainda na atividade *Planejamento*, as *Variáveis Independente e Dependente*. Durante a atividade *Operação*, os tratamentos são aplicados aos sujeitos, ou seja, o experimento

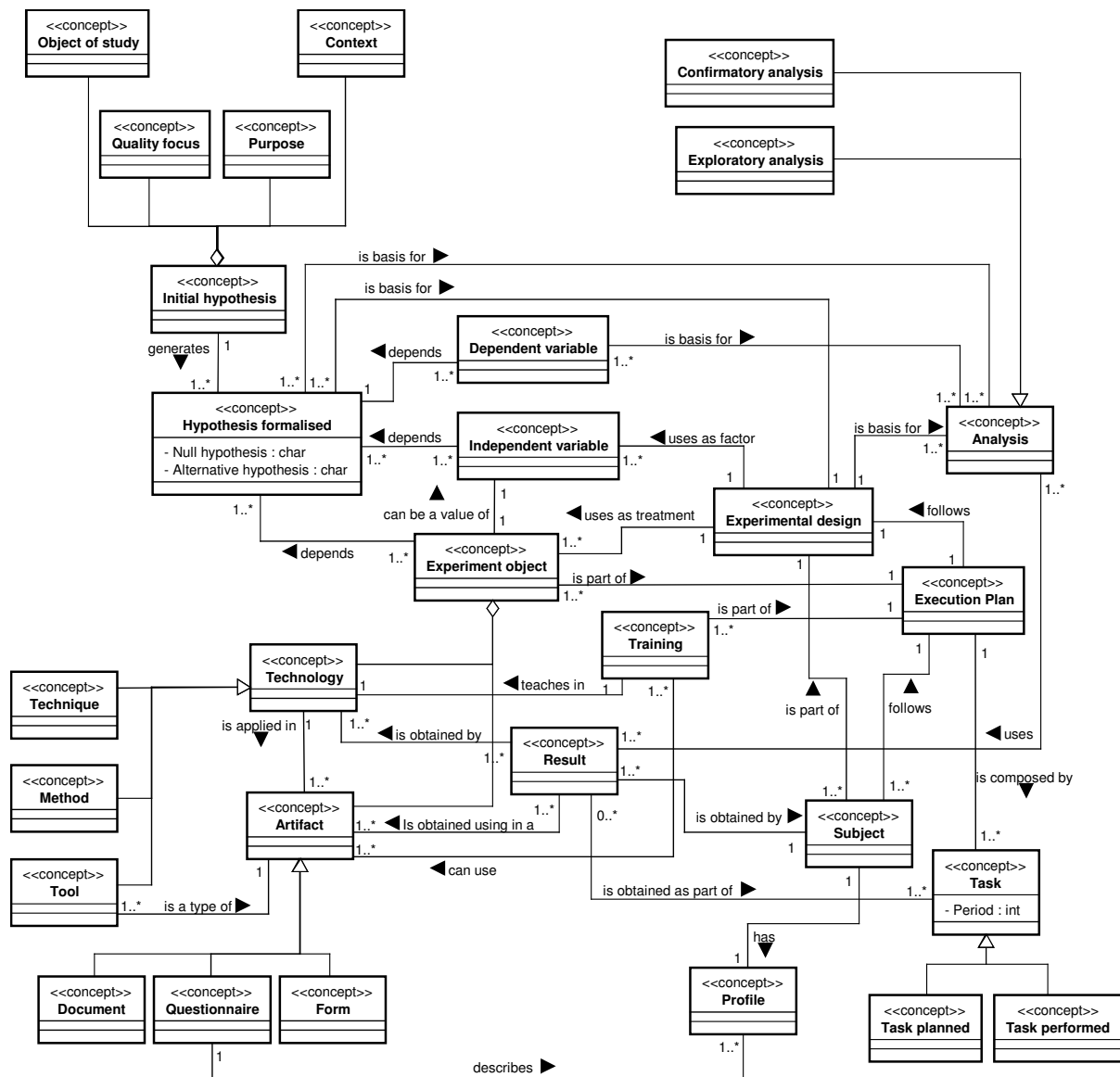


Figura 3.3: Ontologia para Pacotes de Laboratório, adaptado de [Garcia et al. \(2008\)](#)

é colocado em prática. Após o término da execução, mas ainda na atividade *Operação*, é conveniente que o experimentador aplique uma validação dos dados, que em geral verifica se os participantes agiram de maneira efetiva. Isso ocorre por meio da verificação e validação dos dados informados pelos participantes, para que os resultados do experimento sejam válidos, e para tanto, as *Hipóteses Formalizadas* devem ser consultadas.

Na atividade *Análise e Interpretação* no passo de Testes de Hipóteses, as *Hipóteses Formalizadas* são testadas para que se verifique a possibilidade de rejeitar uma determinada *hipótese nula*. Hipóteses e conclusões são gravadas no pacote de laboratório.

É importante ressaltar que se o experimento for mantido com o uso da *ExperOntology*, torna-se possível a utilização de outras ferramentas computaci-

onais (exemplo, os *reasoners* ¹) como meio de exploração dos pacotes de laboratório gerados, apoiando dessa maneira a transferência de conhecimento.

3.4 Ambiente de Apoio à Experimentação Aplicada à Engenharia de Software

Algumas aplicações têm sido desenvolvidas com o intuito de dar apoio à experimentação, como é o caso do ambiente eSEE (*experimental Software Engineering Enviroment*) do Grupo de Pesquisa do Programa de Engenharia de Sistemas e Computação da COPPE/UFRJ que é definido como uma infraestrutura computacional baseada em serviços *web* que visa facilitar a instanci-ção de ambientes de apoio a atividades de estudos experimentais, tais como, definição, planejamento, execução e empacotamento de estudos (dos Santos e Travassos, 2007).

Esse ambiente disponibiliza ferramentas e serviços para os diferentes tipos de estudos identificados em Engenharia de Software, bem como, processos de experimentação, pacotes de experiência, padrões de apresentação de dados (Lopes e Travassos, 2010).

Para apoiar a Experimentação em Engenharia de Software, Chapetta *et. al* (2004 *apud* Lopes (2010)) definiram algumas pré-condições para a definição de um ambiente, como o sistema ser baseado em serviços *web* (permitindo com isso o seu acesso em diferentes localidades), integrar ferramentas de suporte à experimentação, possuir mecanismos que sirvam de apoio à gestão do conhecimento e outros.

A partir dessas pré-condições, o grupo identificou algumas características que interessavam e capturou os requisitos desejáveis para então construir a arquitetura inicial para o eSEE, composta por três macrocomponentes: Meta-Configurador (neste, ocorrem a definição e configuração de todos os elementos básicos que serão utilizados pelos dois outros componentes), Ambiente de Instanciação (fornece meios para customizar e instanciar um ambiente, possibilitando assim a execução de um Processo de Experimentação) e o Ambiente de Execução (é responsável pela execução e acompanhamento de todas as atividades do Processo de Experimentação).

Outro segmento do ambiente eSEE é construir um repositório para o conhecimento adquirido por meio de experimentos realizados e desenvolver um protótipo de ferramenta que atue como uma prova de conceito para ilustrar como o conhecimento de um repositório pode ser utilizado. Esta ferramenta utiliza duas estruturas de representação do conhecimento: um glossário de

¹Um *reasoner* é um programa que infere conclusões lógicas a partir de um conjunto de fatos explicitamente afirmado ou axiomas e normalmente fornece suporte automatizado para tarefas de raciocínio, tais como classificação, depuração e consulta ((Dentler et al., 2010).)

termos que serve para o estabelecimento da terminologia comum na área de Engenharia de Software Experimental (tal glossário foi construído na plataforma web para disponibilizar o acesso aos pesquisadores); a outra estrutura são as ontologias, utilizadas para tornar explícito o conhecimento formal sobre o subdomínio de estratégia de estudo (Lopes e Travassos, 2009a).

Além disso, o próprio conhecimento sobre os pontos de decisão também será explicitado no repositório e disponibilizado pela ferramenta no formato de perguntas (com suas respectivas opções de resposta).

Outra funcionalidade desta é explicitar no repositório o conhecimento sobre os pontos de decisão e o disponibilizar na ferramenta por meio de perguntas e respectivas opções de respostas. A partir das decisões tomadas, esta ferramenta ainda gera um *template* de documento de planejamento pré-preenchido. Esses documentos gerados e utilizados durante a execução do estudo experimental constituem o pacote do experimento.

Tal ambiente citado acima foi desenvolvido para experimentos controlados, seguindo assim a proposta de todas as fases de um processo.

3.5 Considerações Finais

Neste capítulo foram apresentados assuntos relacionados ao Empacotamento de Experimentos Controlados, o detalhamento de tal processo. Relatou-se a importância da transferência de conhecimento e as replicações de experimentos e como podem ser realizadas. Foram citadas pesquisas similares por outros grupos de pesquisadores que também desenvolvem um ambiente de Apoio à Experimentação Aplicada à Engenharia de Software; também abordou sobre ontologias e seu uso na Ciência da Computação, porém mais especificamente em Experimentação em Engenharia de Software; e por fim, foi contextualizada ontologia específica para experimentos controlados, denominada *ExperOntology*.

No próximo capítulo será apresentada a estrutura da ferramenta *OntoExpTool* contendo o *workflow* elaborado, a arquitetura mostrando o conceito de camadas que foi utilizado e os diagramas criados, e por fim, como ela foi implementada.

OntoExpTool - Ferramenta de Apoio à Experimentação

4.1 Considerações Iniciais

Neste capítulo são apresentados detalhes da estrutura, arquitetura e implementação da ferramenta *OntoExpTool*, ferramenta de apoio à execução de experimentos controlados em Engenharia de Software Experimental que utiliza uma ontologia para o empacotamento e a replicação de experimentos.

Seguindo o processo de experimentação citado na Seção 2.2.3, foi proposto um *workflow* descrito também neste capítulo. A ferramenta desenvolvida *OntoExpTool* serve de apoio ao pesquisador na condução de um experimento controlado, contribuindo para superar dificuldades que pesquisadores enfrentam ao ler, analisar e interpretar as informações contidas nos pacotes de laboratório. A *OntoExpTool* utiliza a ontologia *ExperOntology*.

Neste capítulo são apresentados conceitos sobre a *OntoExpTool*, sendo que na Seção 4.2 são descritos detalhes da estrutura da ferramenta, como o *workflow* elaborado e o diagrama de Casos de Uso. Na Seção 4.3 são descritos detalhes da arquitetura da ferramenta e apresentado diagrama que representa a arquitetura. Na Seção 4.4 são apresentados e descritos os diagramas de sequência gerados a partir do *workflow*. Na Seção 4.5 são descritos detalhes de como foi executada a implementação da ferramenta.

4.2 Estrutura da *OntoExpTool*

Na Figura 4.1 é ilustrado o *Workflow* proposto neste trabalho e que serve como base para operacionalizar o desenvolvimento da ferramenta *OntoExpTool*. Ele consiste em uma sequência de atividades para condução e empaco-

tamento de experimentos controlados. A cada fase do experimento, dados são armazenados e ao término da atividade de *análise e interpretação* é fornecida uma opção de criação do pacote de laboratório do experimento no formato XML ou OWL, instanciando-o por meio da ontologia *ExperOntology*, conforme apresentado na Seção 3.3.1.

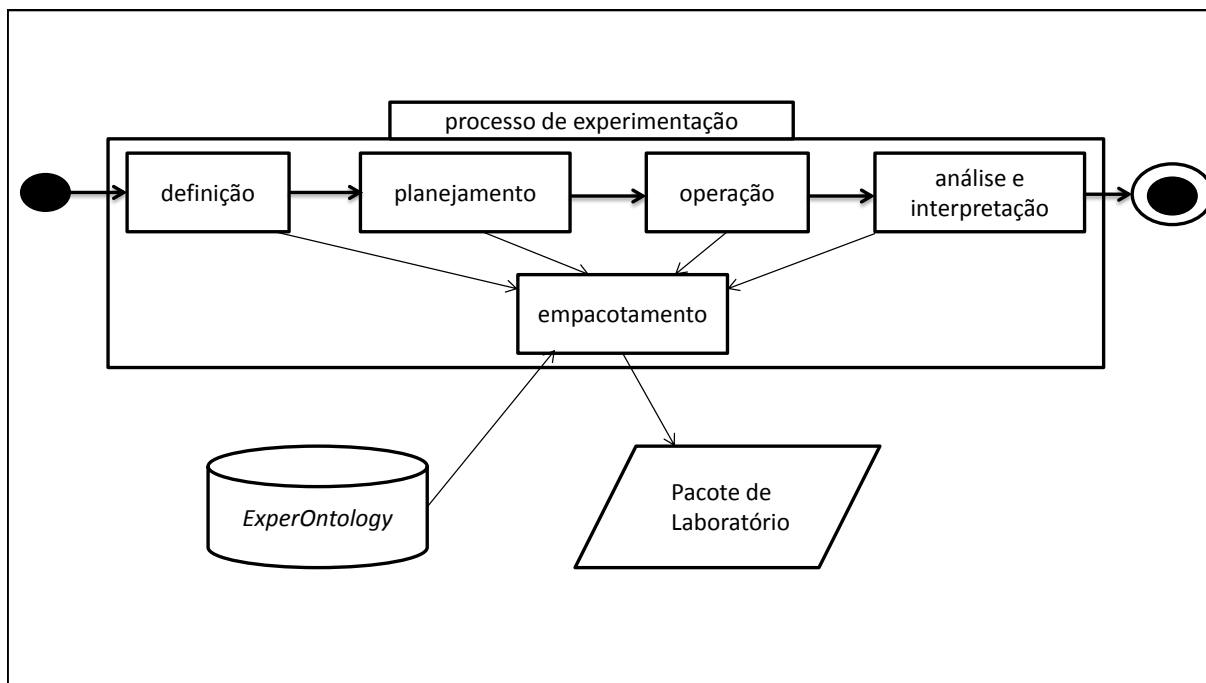


Figura 4.1: *Workflow* de sequência de atividades de condução e empacotamento de experimentos controlados

A partir do *Workflow*, foi desenvolvido o Diagrama de Casos de Uso ilustrado na Figura 4.2. Nesta figura é possível verificar as atividades em que o experimentador interage com o sistema e a única atividade em que o participante interage com o sistema.

Para cada atividade do Diagrama de Casos de Uso, foi gerado um Diagrama de Sequência, apresentados e discutidos na Seção 4.4 e outros no Apêndice .

4.3 Arquitetura da *OntoExpTool*

Com o intuito de melhor organizar a estrutura e a codificação da ferramenta, foi utilizado o conceito de camadas *MVC* (*Model View Controller*), ilustrado na Figura 4.3.

Essa estrutura foi dividida em quatro pacotes de códigos fontes, de acordo com as suas características e funções. A organização de toda a ferramenta foi feita em quatro camadas que trocam dados entre si: camada *Model* (inclui todas as classes Java padrão), camada das persistências (classes que interagem com a base de dados), camada *View* inclui todos os JSPs (*JavaServer Pages*), e a camada *Controller* (inclui os *servlets*). Elas foram separadas de acordo com o nível e a função de cada uma delas.

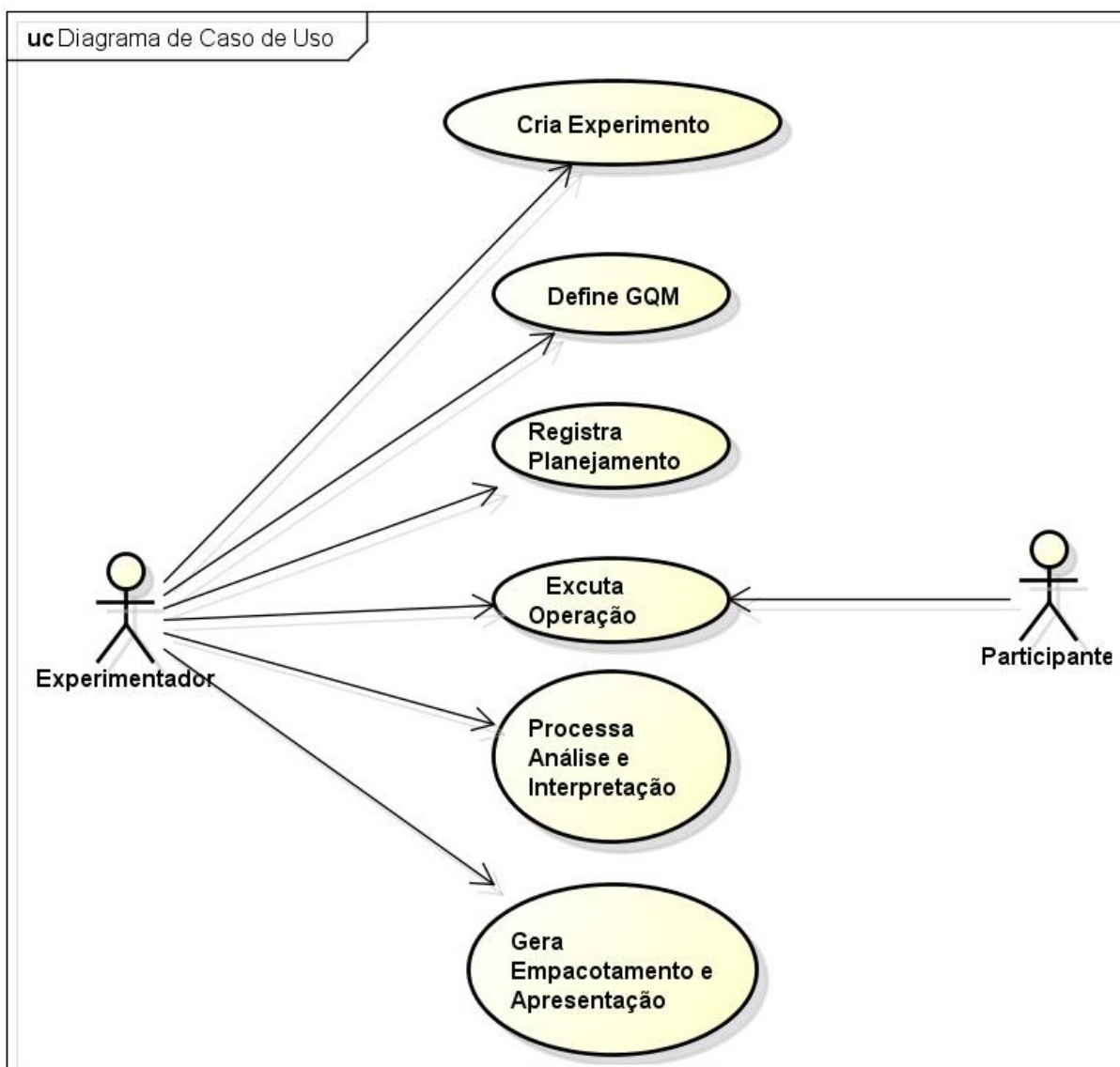


Figura 4.2: Diagrama de Caso de Uso - Visão geral

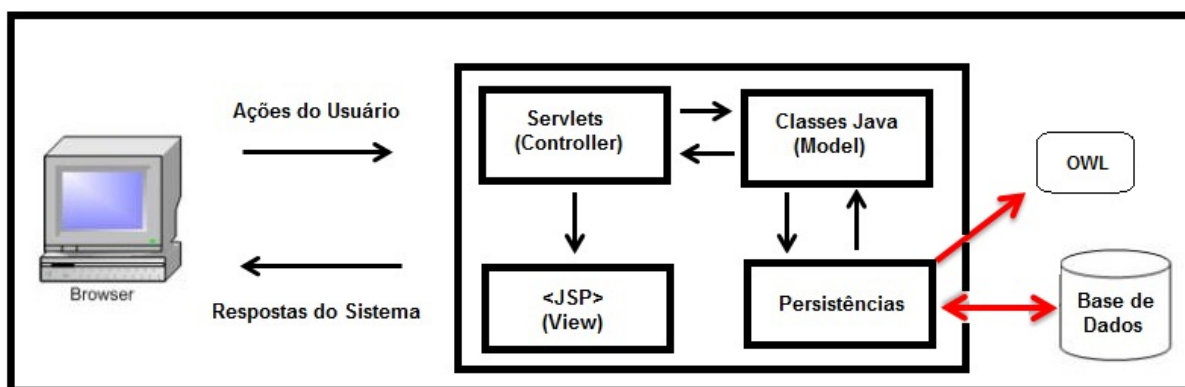


Figura 4.3: Arquitetura – OntoExpTool

O módulo que contém as JavaServer Pages (JSPs) estão na primeira camada e são responsáveis pelo contato direto com o usuário, sendo compiladas no servidor e exibidas no navegador, seja ele experimentador ou participante do experimento. Os JSPs recebem dados, trocam mensagens com os *Servlets*

e mostram os resultados que serão exibidos ao usuário.

Na segunda camada, estão os *servlets* que possuem a função de receber e tratar os dados enviados pelas JSPs, utilizando-se de instâncias das Classes do experimento, e a função de retornar resultado para a requisição feita. A troca de dados é feita entre os *Servlets* e as *JavaServer Pages* por meio dos protocolos *POST* e *GET*. Foi utilizada a tag `<form>` do HTML para envio de informações entre cliente e servidor, utilizando também a tecnologia *AJAX*, que com o auxílio de *JavaScript*, faz requisições e trocas de dados sem a necessidade de recarregar a página no navegador. De acordo com os propósitos, os *Servlets* foram divididos em pacotes, tornando-se alguns responsáveis por processar informações relativas ao processo específico do experimento. O *servlet incluirArtefato*, por exemplo, recebe as informações sobre os artefatos que serão utilizados no experimento e as processa com o auxílio de uma instância da classe *Artefato*.

Na terceira camada estão as classes Java (descritas na Seção A.2) que foram criadas baseadas nos conceitos de Programação Orientada a Objetos, sendo que, cada uma abstrai do mundo real conceitos e ideias pertinentes à experimentação e definem seu comportamento através dos métodos criados. A classe *Validade*, por exemplo, define todas as características e funções dos tipos de validade de um estudo experimental. Existem também algumas classes auxiliares pertencentes ao pacote "Classes" e que executam tarefas secundárias, como a classe *Mail*, utilizada para o envio de *e-mails*. Outro pacote é o "Banco", que contém a classe responsável pela conexão com o banco de dados e pelas atividades *CRUD*. Na quarta camada estão as persistências que são os objetos que tem que ser gravados em um meio de armazenamento persistente, no caso neste projeto, em uma base de dados e no pacote de laboratório *OWL*.

4.4 Diagramas de Sequência – *OntoExpTool*

4.4.1 Caso de Uso – Cria Experimento

Para a primeira atividade do Caso de Uso – Cria Experimento – foram criados dois Diagramas de Sequência. Na Figura 4.4 é apresentado o Diagrama de Sequência – Cria Experimento, que possui eventos de acesso ao sistema pelo experimentador para criar ou acessar experimentos e incluir novos experimentadores na ferramenta, bem como conduzir um experimento criado. O segundo, Diagrama de Sequência – Associar Experimentador – é apresentado na Figura 4.5, e possui eventos de consulta a experimentadores e experimentos cadastrados para, então, associar um experimentador a um experimento.

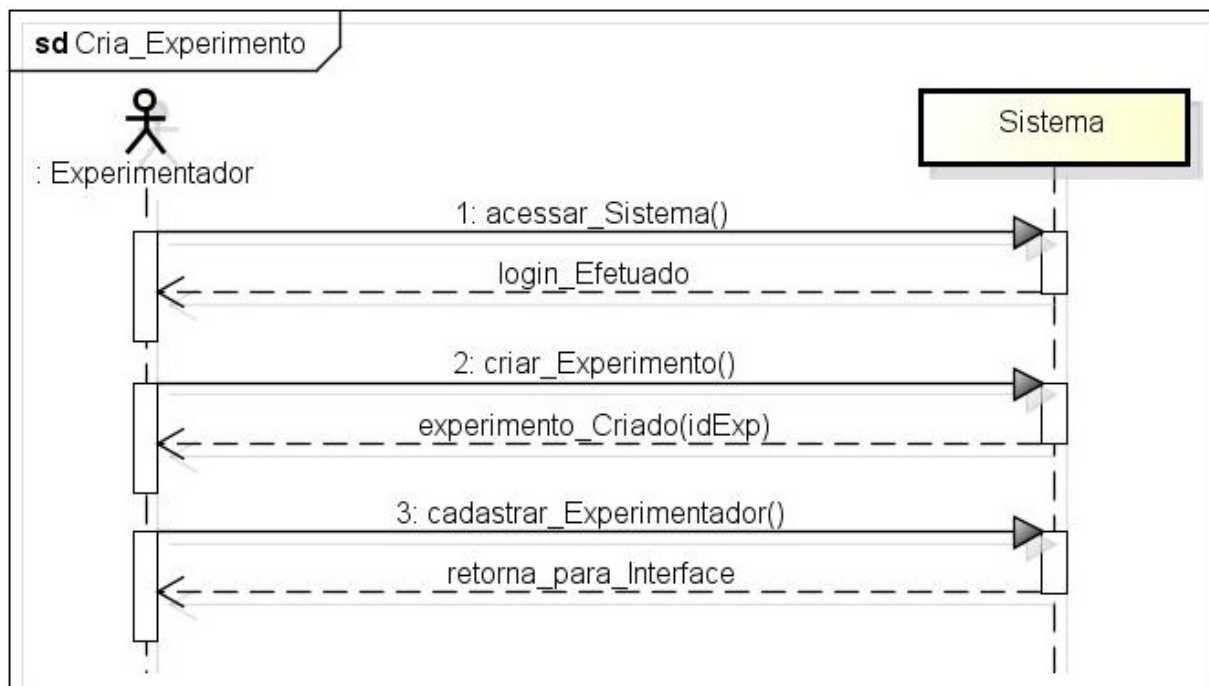


Figura 4.4: Diagrama de Sequência – Cria Experimento

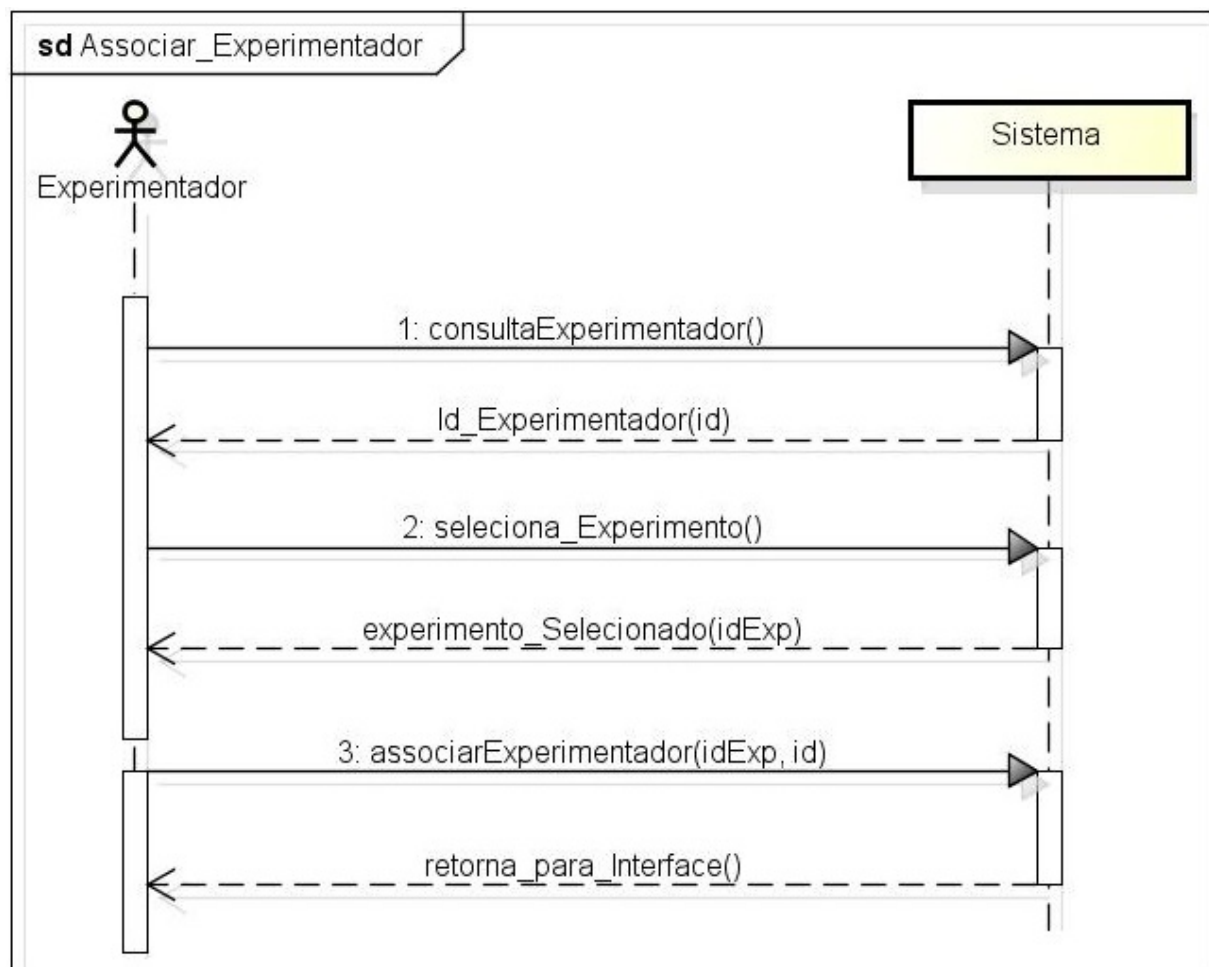


Figura 4.5: Diagrama de Sequência – Associar Experimentador

4.4.2 Caso de Uso – Definição

Para este caso de uso foi criado o Diagrama de Sequência – *Incluir Definição* – apresentado na Figura 4.6, que possui os eventos de selecionar um experimento para, então, incluir os dados referentes à atividade de definição do experimento (vide Seção 2.2.4.1).

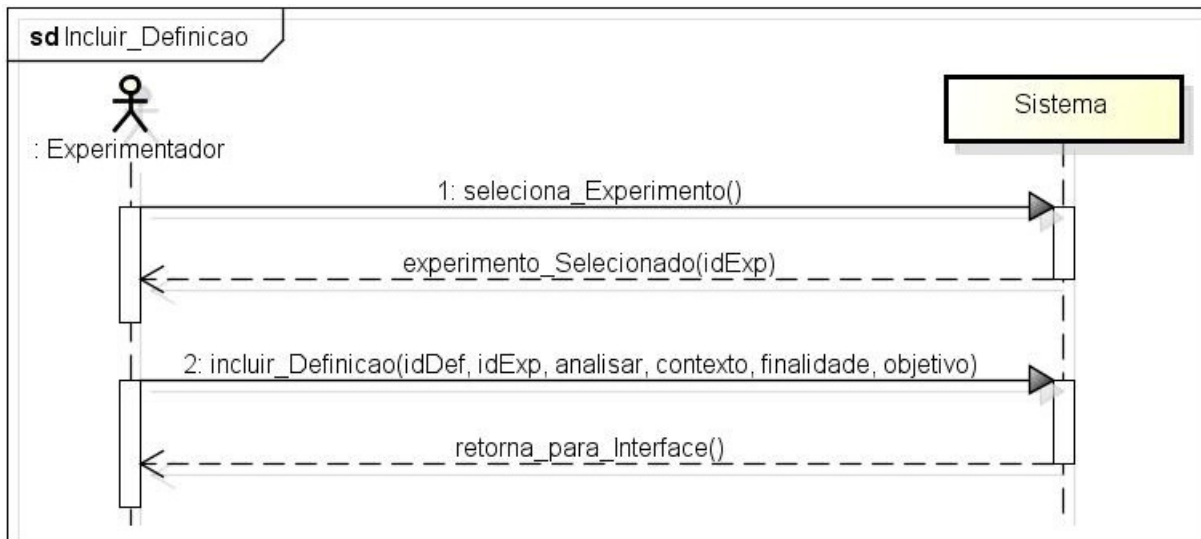


Figura 4.6: Diagrama de Sequência – Incluir Definição

4.4.3 Caso de Uso – Planejamento

Para a atividade – Registra Planejamento – terceira do Caso de Uso, foram criados alguns Diagramas de Sequência. Na Figura 4.7 é apresentado o Diagrama de Sequência – *Seleção Contexto* – que possui evento que proporciona a seleção de um experimento para registrar os dados do contexto (vide Seção 2.2.5.1). Os demais diagramas são apresentados no Apêndice A.

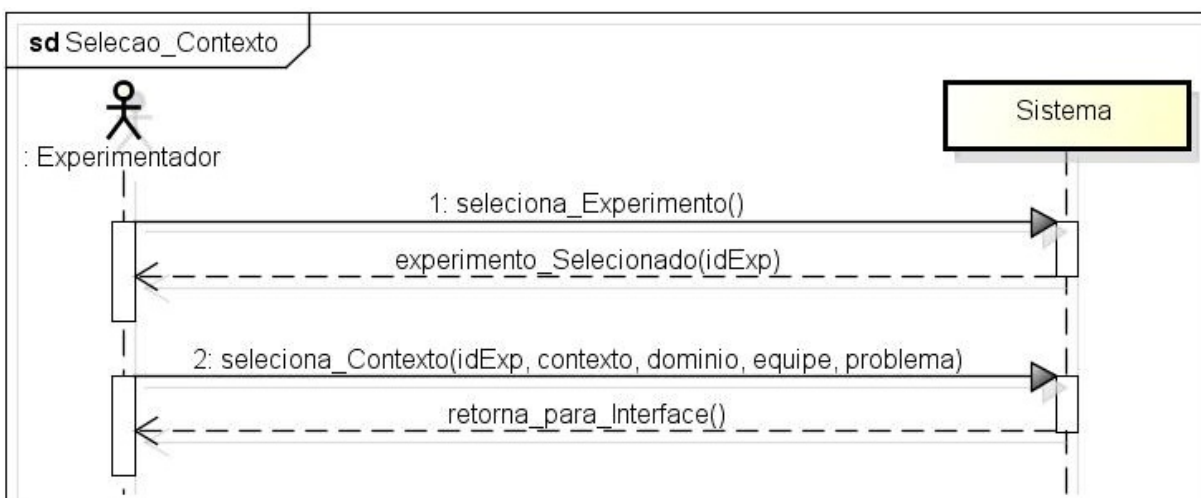


Figura 4.7: Diagrama de Sequência – Seleção de Contexto

4.4.4 Caso de Uso – Operação

A quarta atividade do Diagrama de Caso de Uso – Executa Operação – é acessada tanto pelo experimentador como pelo participante do experimento selecionado. Após finalizar as atividades anteriores de *Definição* e *Planejamento*, o experimentador pode disponibilizar a atividade de *Operação* para o participante então realizar as atividades planejadas. Na Figura 4.8 é apresentado o Diagrama de Sequência – Processa Dados Operação – que possui evento que solicita *login* por parte de cada participante e solicita que o mesmo cadastre alguns dados pessoais, selecione o experimento disponível e após realizar as atividades propostas, registre-as no sistema, conforme o treinamento que lhe foi aplicado. (vide Seção 2.2.6).



Figura 4.8: Diagrama de Sequência – Processa Dados Operacao – Participante

4.4.5 Caso de Uso – Análise

Na quinta do Caso de Uso atividade – Processa Análise e Interpretação – foi criado o Diagrama de Sequência – Realiza Análise Interpretação. Este diagrama é ilustrado na Figura 4.9 e possui evento em que o experimentador seleciona um experimento e carrega os arquivos enviados pelos participantes após a finalização da atividade de operação, possibilitando, assim, a sua análise e registro das interpretações geradas (vide Seção 2.2.7).

4.4.6 Caso de Uso – Empacotamento

A última atividade do Caso de Uso é – Gera Empacotamento e Apresentação. Na Figura 4.10 é ilustrado o Diagrama de Sequência – Realiza Apresen-

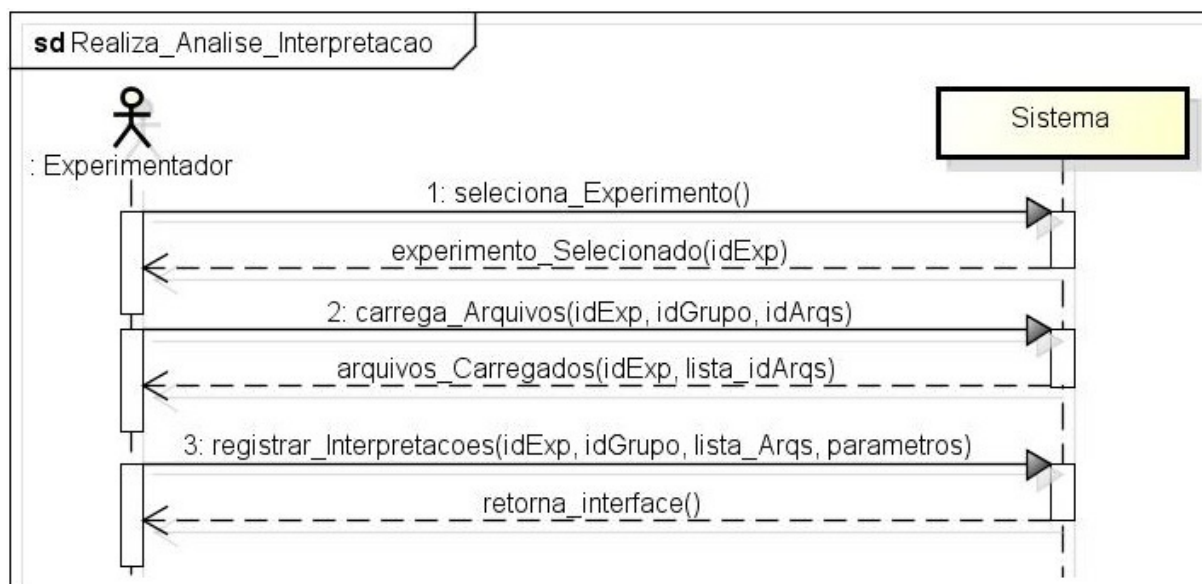


Figura 4.9: Diagrama de Sequência – Realiza Analise e Interpretacao

tação Empacotamento – que possui evento *seleciona experimento* que disponibiliza o evento escolha de opção para empacotamento em um formato: XML ou OWL (vide Seção 2.2.8, ou ainda gerar um pacote ZIP contendo os arquivos XML, OWL e outros arquivos que fazem parte do experimento.



Figura 4.10: Diagrama de Sequência – Realiza Apresentação e Empacotamento

Algumas atividades existentes na *OntoExpTool* foram suprimidas na geração dos Diagramas de Sequência por possuírem apenas os métodos *GETS* e

SETS.

4.4.7 Diagrama de Classes - *OntoExpTool*

É possível verificar na Figura 4.11 o Diagrama de Classes simplificado da ferramenta.

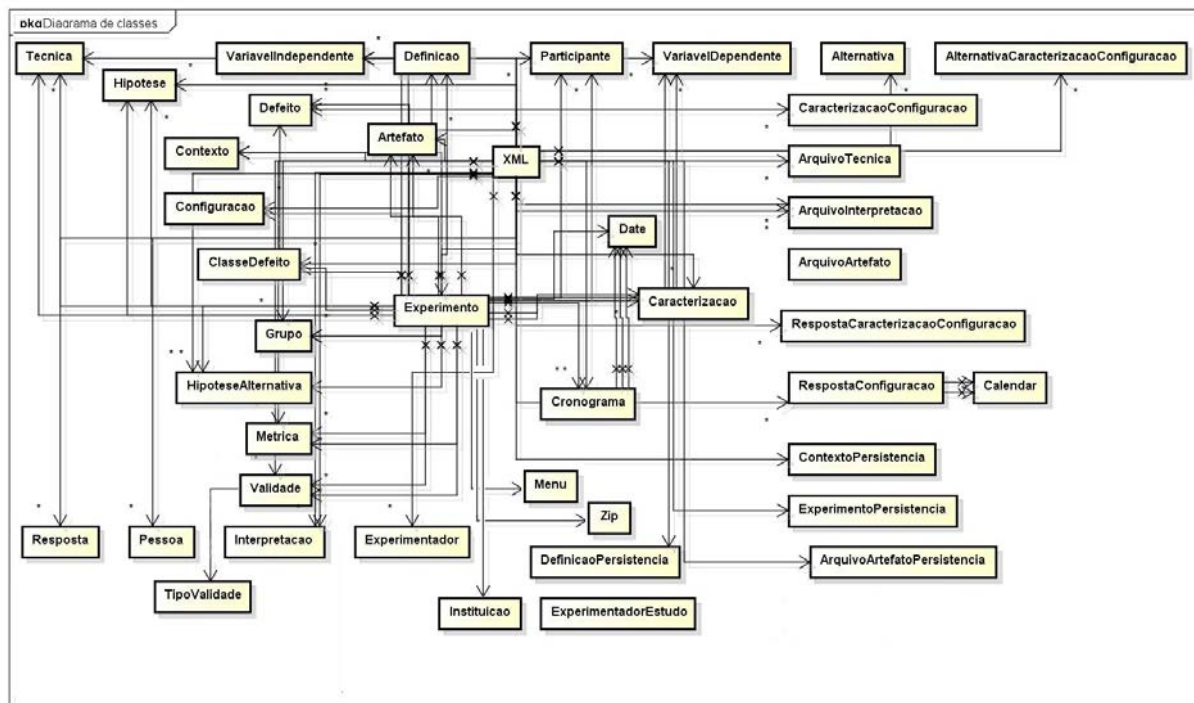


Figura 4.11: Diagrama de Classes – Simplificado

4.5 Implementação da ferramenta

A ferramenta *OntoExpTool* foi implementada utilizando a linguagem de programação Java, direcionada para a criação de aplicações para a *web*. utilizar recursos *web* como apoio na execução de estudos experimentais pode auxiliar no sentido de que a experiência fique mais próxima das situações encontradas na indústria, bem como ajudar a solucionar alguns problemas de logística, como a execução de uma tarefa por equipes geograficamente disseminadas, facilitando com isso o recrutamento, seleção e participação dos participantes, projetando a diminuição da falta de representatividade de amostras e consequentemente aumentar o grau de confiabilidade dos resultados do experimento (dos Santos e Travassos (2007)). Pode ser utilizada em qualquer computador que tenha acesso à Internet e que possua um navegador atualizado. Indêpende de plataforma, podendo ser acessada tanto em ambiente *Linux* como *Windows*.

Na implementação da ferramenta foram criados os pacotes *Servlets* e páginas *JSP* (*JavaServer Page*) para o tratamento e exibição dos dados, assim como o pacote com as classes Java padrão, que desempenham atividades base

da aplicação, seguindo o modelo de camadas citado na Seção 4.3. O ambiente NetBeans 7.2.1 foi utilizado no desenvolvimento de toda a aplicação. A execução da ferramenta é feita por meio de um servidor Apache Tomcat 7.0.

Tecnologias auxiliares como alguns pacotes e bibliotecas voltados para *web* foram utilizados para proporcionar uma melhor usabilidade por parte do usuário e atender os objetivos planejados, tais como:

- *Ajax*: é possível fazer requisições e troca de informações entre o navegador (cliente) e o servidor *web* sem a necessidade do recarregamento total da página HTML. Tecnologia utilizada no projeto para simplificar a experiência do usuário na utilização da ferramenta;
- *jQuery*: biblioteca *JavaScript* utilizada para simplificar o desenvolvimento e codificação de HTML, evitando incompatibilidades no desenvolvimento para os diversos navegadores existentes. Disponibiliza várias funcionalidades para a aplicação, como efeitos visuais e uma interface para requisições Ajax mais simples e intuitiva;
- *Apache Commons fileUpload*: pacote de classes que têm como principal objetivo o *upload* de arquivos do cliente para armazenamento no servidor *web*. Foi aplicada na ferramenta para possibilitar o envio dos arquivos e documentos de artefatos, técnicas e outros documentos que são utilizados no experimento;
- *XStream*: é uma biblioteca Java utilizada na serialização de objetos para o formato XML, bem como também o processo contrário. Gera um arquivo bem legível e de fácil entendimento. Utilizada na atividade de apresentação e empacotamento do experimento e replicação. Foi utilizada também nas atividades de Definição e Planejamento para exportar um experimento no formato XML, possibilitando assim a importação por outro pesquisador para a continuação do mesmo;
- *opencsv*: ferramenta que possibilita a geração de arquivos CSV (*Comma-separated values*). Utilizada na atividade de Análise e Interpretação do experimento, gerando a exportação dos dados da fase de Operação, como questionários e observações relativas às atividades executadas pelos participantes, possibilitando ao experimentador acesso mais fácil a essas informações;
- *iReport* e *JasperReports*: ferramentas destinadas à geração de relatórios. *iReport* possibilita criação de templates que serão utilizados pelo *JasperReports* para a exibição dos relatórios. Utilizados na fase de Análise e Interpretação para gerar arquivos no formato PDF que ajudam o experimentador em algumas tarefas.

4.5.1 Módulo de controle do tempo de execução de tarefas

Para cada atividade proposta no experimento, é determinado um tempo limite para sua execução, sendo necessário ao participante cumprir esse prazo. Existe um módulo que faz o controle do tempo de execução de cada tarefa por participante, permitindo ou não o envio dos artefatos de resultado das atividades para o servidor. Por meio da ferramenta, o pesquisador tem o controle de quem já iniciou as atividades na fase de Operação, bem como o mesmo pode consultar as atividades que foram encerradas e as que ainda não foram iniciadas.

4.5.2 Módulo de segurança

Como o ambiente da ferramenta é *web*, considerou-se a preocupação com a segurança, para isso foi utilizado o sistema de *Sessions*, no qual se estabelece uma sessão única entre o navegador (cliente) e o servidor enquanto o usuário está logado no sistema, seja ele um experimentador ou um participante. Existe a possibilidade de registro das variáveis ou objetos no servidor para poder identificar a sessão criada. O *servlet login* faz a autenticação e validação do usuário consultando e validando as informações de *login* enviadas, e em seguida um objeto do tipo Experimentador ou Participante é registrado na sessão. A cada acesso às *JavaServer Pages* e aos *Servlets*, ocorre a verificação se há o objeto registrado na sessão estabelecida, e, se não houver, o usuário é redirecionado para a página de *login* novamente. Algumas questões de segurança, como ataques do tipo *SQL Injection*, não foram tratados nesta ferramenta.

4.5.3 Armazenamento dos dados

Todos os dados coletados e gerados pela ferramenta são armazenadas em uma base de dados administrada pelo sistema gerenciador de banco de dados *MySQL*. Essa base de dados possui várias tabelas para melhor organizar o registro dos dados dos experimentos. Uma parte do esquema de tabelas e atributos dessa base de dados pode ser visualizado na Figura 4.12. Os demais esquemas estão particionados e são apresentados no Apêndice A.

4.5.4 Pacote OWL gerado

Com o propósito de aplicar a ontologia no empacotamento de experimentos controlados em Engenharia de Software, foi desenvolvido um módulo para instanciar as informações contidas em um pacote de laboratório baseada nos conceitos da *ExperOntology*, e, para tanto, foi utilizada a linguagem *OWL (Web Ontology Language)* ¹.

¹<http://www.w3.org/TR/owl-guide>

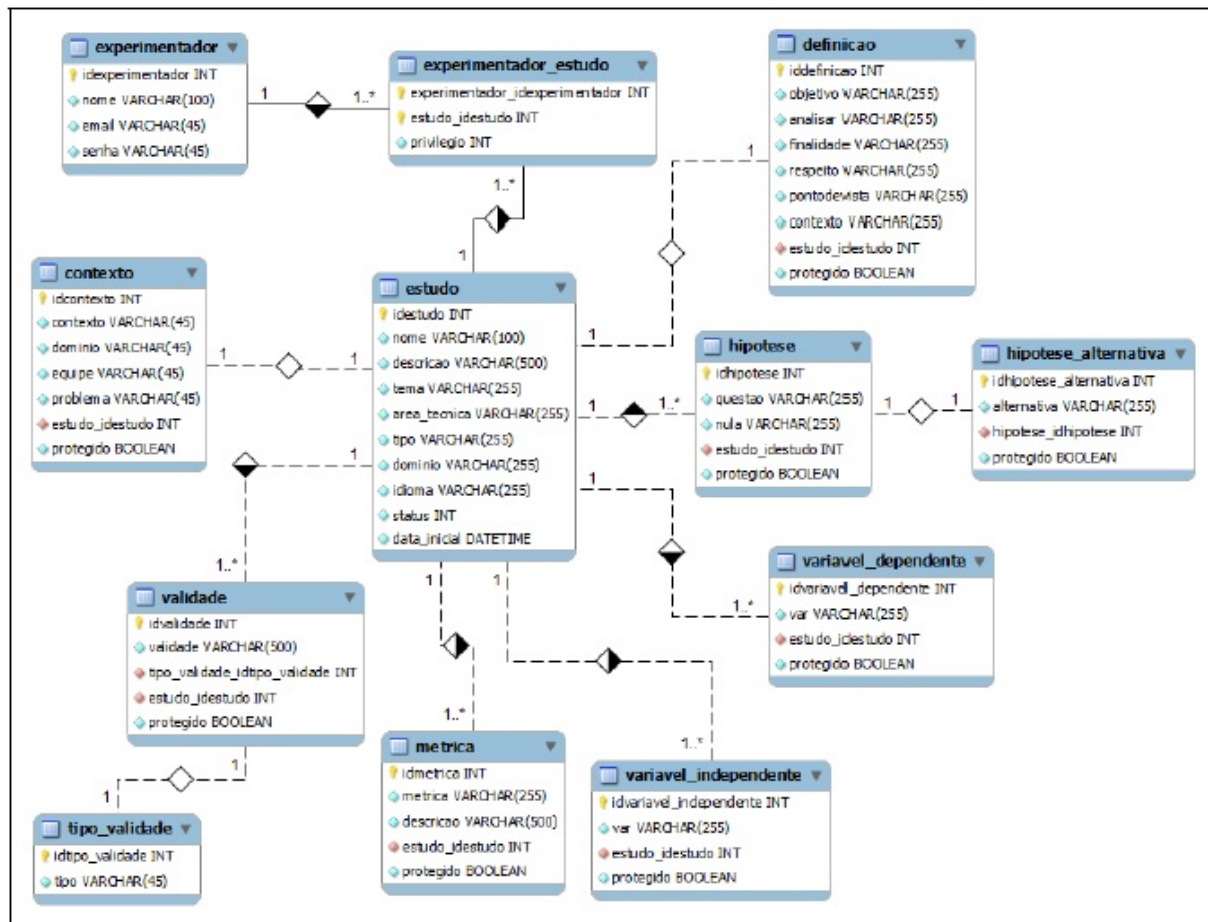


Figura 4.12: Esquema da Base de Dados I

Para implementar a linguagem *OWL* foi utilizada a *OWL API Java*, uma interface de programação de aplicativos de alto nível para trabalhar com ontologias *OWL*. A *OWL API* está estreitamente alinhada com a especificação estrutural *OWL 2*, oferecendo suporte a: análise e renderização em sintaxes definido na especificação *W3C*², manipulação de estruturas ontológicas e à utilização de motores de raciocínio. A utilização de *APIs* permite aos desenvolvedores trabalhar em um nível mais alto de abstração, e também isolar algumas das questões problemáticas relacionadas com a serialização e análise de estruturas de dados. A *OWL API* procura apresentar um componente altamente reutilizável para a construção de diferentes aplicações, tais como editores, ferramentas de anotação e agentes de consulta.

Na Figura 4.13 é ilustrado um trecho mais completo do pacote de laboratório gerado em *OWL* instanciado pela *ExperOntology*.

²<http://www.w3.org/>



Figura 4.13: Pacote de Laboratório gerado em OWL

4.6 Considerações Finais

A arquitetura, implementação e organização da ferramenta, bem como sua base de dados, foram apresentados brevemente neste capítulo. As tecnologias utilizadas foram descritas, destacando a importância de cada uma no contexto geral da aplicação. Detalhes importantes como o controle do tempo das atividades e questões de segurança foram considerados e explicados. No próximo capítulo é apresentado um exemplo de uso da ferramenta, detalhando cada uma de suas funcionalidades.

Empacotamento com uso da OntoExpTool

5.1 Considerações Iniciais

Neste capítulo é demonstrada o empacotamento de um experimento controlado utilizando a ferramenta *OntoExpTool* com o intuito de ilustrar o apoio que a ferramenta provê ao experimentador em cada atividade do processo experimental descrito no *workflow* deste trabalho. Também é ilustrada a criação de um Pacote de Laboratório contendo as informações do experimento de maneira organizada e padronizada baseado nos conceitos estruturados da *ExperOntology* – facilitando a análise e a interpretação das informações contidas no Pacote de Laboratório por outro pesquisador.

A apresentação do empacotamento é demonstrada passo a passo, com o uso da ferramenta em cada atividade do *workflow*: *Definição, Planejamento, Operação, Análise e Empacotamento*. Para a apresentação do uso da ferramenta no apoio ao empacotamento, foi escolhido o experimento conduzido para avaliar a eficácia e a eficiência da Ferramenta de Visualização de Software *SofVis_{OA}H* (d'Arce, 2012) sobre uma técnica *Ad-hoc* em determinado contexto. Este capítulo encontra-se organizado como segue: na Seção 5.2 é apresentado o experimento controlado usado para empacotamento; na Seção 5.3 é demonstrado o empacotamento do experimento em todas atividades do processo experimental (*Definição, Planejamento, Operação, Análise e Empacotamento*).

5.2 Experimento utilizado como demonstração: Avaliação da *SofVis_{OA}H*

O experimento registrado na ferramenta e que serviu como demonstração de utilização da *OntoExpTool*, é do experimento executado no trabalho de [d'Arce \(2012\)](#), que contextualiza que uma maneira alternativa de auxílio à área de Engenharia de Software é a Visualização de Software que lida com a complexidade estrutural, apoiando tarefas de Compreensão de Programas por meio de representações visuais, e que permite interagir com essas representações gráficas, provendo auxílio ao processo de entendimento. Uma ferramenta de Visualização de Software proporciona suporte ao usuário para obter informações por meio de análises dos artefatos visuais disponíveis, gerando assim, auxílio ao processo de entendimento de programas.

O principal objetivo do experimento de [d'Arce \(2012\)](#) é avaliar a utilização da ferramenta *SofVis_{OA}H* no apoio ao entendimento de Programas Orientados a Aspectos, em comparação com o entendimento de código convencional (ad hoc – sem o auxílio da ferramenta). Alguns fatores foram avaliados no experimento: verificação se a ferramenta auxilia na localização de defeitos (tarefa de informar onde os defeitos estão localizados no código fonte) e se auxilia a compreensão (abstração estrutural e entrecorte de aspectos) de um Programa Orientado a Aspectos.

Com a condução do experimento, os participantes puderam fornecer, como *feedback*, além das medições (dados coletados) que permitiram derivar métricas quantitativas relativas ao uso da ferramenta, uma avaliação qualitativa da mesma. Após a condução do experimento e avaliação dos resultados, foi possível constatar estatisticamente que a *SofVis_{OA}H* proveu auxílio às tarefas de Localização de Defeitos e Compreensão de Programa ([d'Arce, 2012](#)).

5.3 Perspectiva do Experimentador

5.3.1 Definição do Experimento

Na execução do experimento utilizado como estudo de caso, primeiro foram definidos os objetivos do experimento seguindo o paradigma de melhoria contínua GQM (Goal/-Question/Metric). Os itens são descritos a seguir:

- Objeto de Estudo – Ferramenta de Visualização de Software *SofVis_{OA}H*;
- Propósito – avaliar e comparar a utilização da ferramenta *SofVis_{OA}H* como apoio a tarefas de Revisão de Programa em relação à Revisão de Programa ad hoc, conforme as seguintes atividades:
 - _ Revisão de Programa ad hoc: entendimento de programa e localização de defeitos (indicação do local do defeito no código fonte) sem o apoio

da ferramenta *SofVis_{OA}H*, apenas com o apoio do ambiente de desenvolvimento Eclipse SDK;

_ Revisão de Programa com visualização: entendimento de programa e localização de defeitos com o apoio da ferramenta *SofVis_{OA}H*, além do apoio do ambiente de desenvolvimento Eclipse SDK.

- Enfoque de Qualidade – Tempo, Quantidade de defeitos localizados e Entendimento do programa;
- Perspectiva – Do experimentador;
- Contexto – Laboratório de informática da Universidade Estadual Paulista Júlio de Mesquita Filho, Unesp, no campus da Faculdade de Ciências e Tecnologia.

Dados estes que fazem parte da primeira atividade do processo experimental denominado *Definição*, conforme descrito no Capítulo 2 e ilustrado na cor amarelo na Figura 5.1 da *ExperOntology*.

Portanto, o objetivo definido, consiste em analisar a ferramenta de Visualização de Software *SofVis_{OA}H* com o propósito de avaliar a sua utilização (comparando a Revisão de Programa ad hoc e a Revisão de Programa apoiada pela ferramenta) com respeito ao tempo e à quantidade de defeitos localizados, da perspectiva do experimentador, no contexto de um laboratório de pesquisa. Neste experimento a localização de defeitos refere-se à atividade de encontrar trechos de código fonte em que um dado defeito se encontra. Os participantes não precisam realizar tarefas de detecção de defeitos¹, pois existem outras ferramentas com o objetivo de auxiliar tal tarefa. Para efetuar a localização de defeitos, uma lista de defeitos inseridos no código fonte é fornecida aos participantes, que devem indicar onde o defeito está localizado.

5.3.2 Planejamento do Experimento

Na atividade *Planejamento* do experimento em estudo, foram planejadas três questões e para cada questão duas métricas conforme ilustrado na Figura 5.2.

Também foram planejadas três *Hipóteses Nulas* e uma *Hipótese Alternativa* para cada uma, ilustradas na Figura 5.3. Ainda na atividade *Planejamento* após o planejamento das hipóteses, foram criadas duas *Variáveis Independentes* conforme ilustradas na Figura 5.4 e cinco *Variáveis Dependentes* ilustradas na Figura 5.5.

Outra etapa da atividade de *Planejamento* é a *Seleção de Participantes* do experimento. foram escolhidos mediante seleção determinística, envolvendo alunos de graduação em Ciência da Computação (estágio avançado) e de mestrado em Ciência da Computação. Tais alunos têm conhecimento suficiente

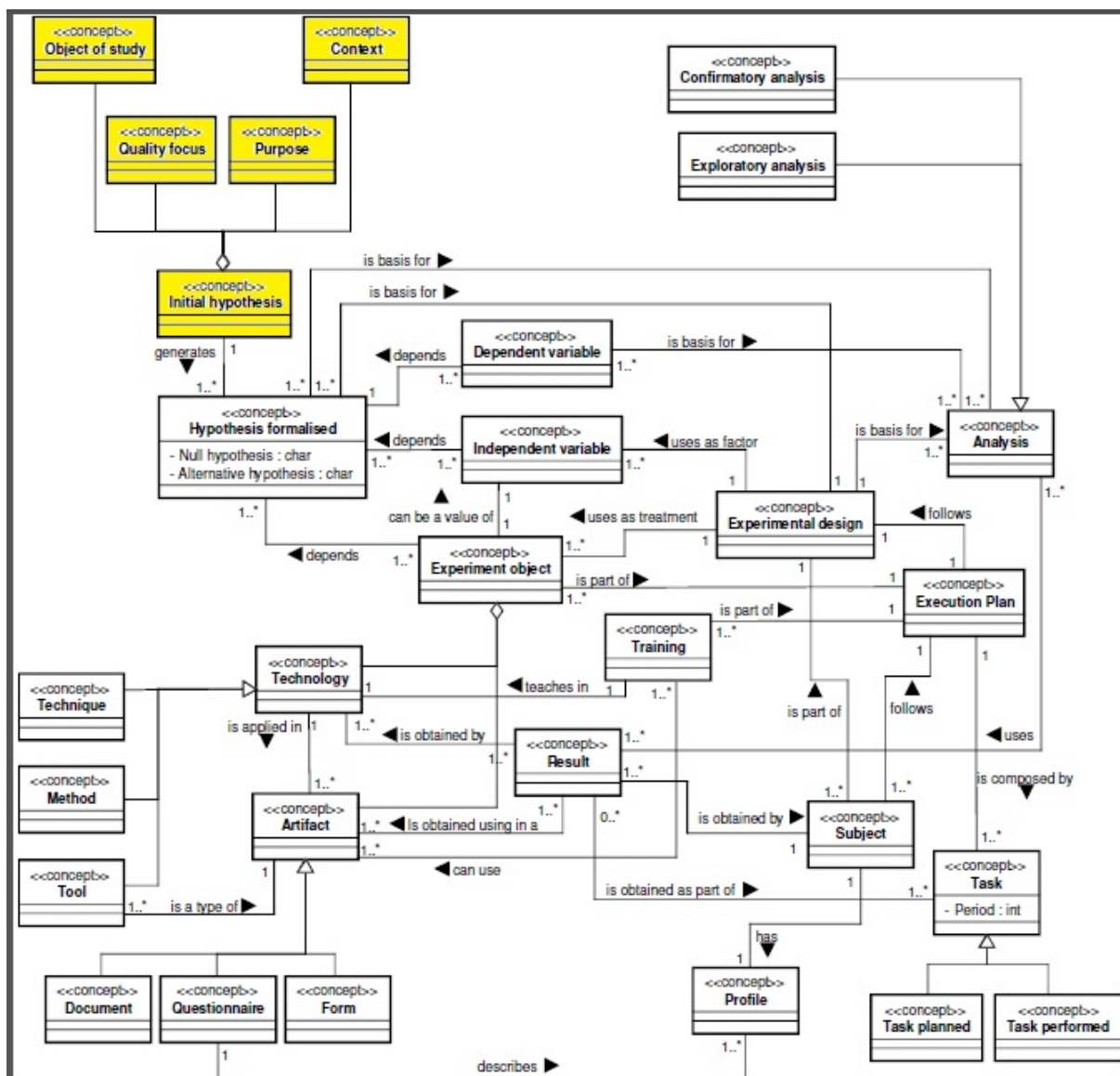


Figura 5.1: Conceitos organizados na *ExperOntology* da Atividade *Definição*, adaptado de [Garcia et al. \(2008\)](#)

QUESTÕES	MÉTRICAS
Q1: A utilização da ferramenta no apoio à Compreensão de Programa auxilia o entendimento do programa?	– M1.1: Quantidade de adendos (de todos os aspectos) contidos no código fonte. – M1.2: Quantidade de adendos descritos corretamente.
Q2: A utilização da ferramenta no apoio à Localização de Defeitos torna a localização mais eficaz (maior quantidade)?	– M2.1: Quantidade de defeitos previamente inseridos no código fonte. – M2.2: Quantidade de defeitos localizados por cada revisor de programa.
Q3: A utilização da ferramenta no apoio à Localização de Defeitos torna a localização de defeitos mais eficiente (mais rápida)?	– M3.1: Tempo médio (em minutos) despendido em cada localização de defeitos. – Métricas M2.1 e M2.2.

Figura 5.2: Questões e Métricas elaboradas na Atividade *Planejamento* ([d'Arce, 2012](#))

em Programação Orientada a Objetos (elementos e definições básicas, como objeto, herança, classe, método e associação) e conhecimento teórico em atividades de revisão e testes de programas. A maioria deles tem conhecimento suficiente em Java (implementação dos elementos e dos conceitos da Orienta-

HIPÓTESE NULA	HIPÓTESE ALTERNATIVA
Hipótese Nula (H ¹ ₀): A utilização da ferramenta SofV isOAH não provê auxílio à Compreensão de Programa.	Hipótese Alternativa (H ¹ ₁): A utilização da ferramenta SofV isOAH provê auxílio à Compreensão de Programa.
Hipótese Nula (H ² ₀): A utilização da ferramenta SofV isOAH não proporciona maior eficácia que a localização de defeitos ad hoc.	Hipótese Alternativa (H ² ₁): A utilização da ferramenta SofV isOAH proporciona maior eficácia que a localização de defeitos ad hoc.
Hipótese Nula (H ³ ₀): A utilização da ferramenta SofV isOAH não proporciona maior eficiência que a localização de defeitos ad hoc.	Hipótese Alternativa (H ³ ₁): A utilização da ferramenta SofV isOAH proporciona maior eficiência que a localização de defeitos ad hoc.

Figura 5.3: Hipóteses elaboradas na Atividade *Planejamento* (d'Arce, 2012)

VARIÁVEIS INDEPENDENTES
_ Quantidade de defeitos previamente inseridos no código-fonte. Essa quantidade permite verificar o índice de eficácia e de eficiência da Revisão de Programa com apoio da ferramenta SoftV isOAH e da Revisão de Programa ad hoc.
– Quantidade de adendos de aspectos contidos no código fonte. Essa quantidade permite verificar o nível de auxílio provido à Compreensão de Programa pela Revisão de Programa com apoio da ferramenta SoftV isOAH e pela Revisão de Programa ad hoc.

Figura 5.4: Variáveis Independentes elaboradas na Atividade *Planejamento* (d'Arce, 2012)

VARIÁVEIS DEPENDENTES
– Quantidade de defeitos localizados por meio da Revisão de Programa ad hoc e da Revisão de Programa com apoio da ferramenta <i>SoftVisOAH</i>. Cada participante deve localizar o máximo de defeitos para a avaliação de ambas as revisões de programa.
– Tempo (em minutos) para a localizar os defeitos por meio da Revisão de Programa ad hoc e da Revisão de Programa com apoio da ferramenta <i>SoftVisOAH</i>. Cada participante deve localizar cada defeito em um menor tempo para a avaliação de ambas as revisões de programa.
– Nível de auxílio provido à Compreensão de Programa da Revisão de Programa com apoio da ferramenta <i>SoftVisOAH</i> e da Revisão de Programa ad hoc. Esse nível é utilizado na análise de a partir de qual revisão de programa os participantes obtiveram um maior auxílio ao entendimento do programa por meio da descrição de cada adendo contido em cada aspecto do artefato de software.
– Índice de eficácia da Revisão de Programa com apoio da ferramenta <i>SoftVisOAH</i> e da Revisão de Programa ad hoc. Esse índice é utilizado na análise da eficácia (quantidade de defeitos localizados) de ambas as revisões de programa.
– Índice de eficiência da Revisão de Programa com apoio da ferramenta <i>SoftVisOAH</i> e da Revisão de Programa ad hoc. Esse índice é utilizado na análise da eficiência (tempo despendido na localização de cada defeito) de ambas as revisões de programa.

Figura 5.5: Variáveis Dependentes elaboradas na Atividade *Planejamento* (d'Arce, 2012)

ção a Objetos) para participar nas atividades do experimento. Como a maioria não tem conhecimento em Programação Orientada a Aspectos, em JUnit e em técnicas de Visualização de Software, que também são necessários, foram realizados treinamentos para prover condições mínimas para a realização das

atividades.

Ainda na atividade *Planejamento* foram definidos os instrumentos utilizados no experimento, sendo eles de dois tipos: *Tecnologia* (Técnica, Método ou Ferramenta) e *Artefato* (Documento, Questionário ou Formulário). Os instrumentos de *Tecnologia* utilizados na condução do experimento foram definidos de acordo com os requisitos tecnológicos para a execução de ambas as revisões de programa e com os requisitos de conhecimento prévio para a realização das mesmas. A instrumentação é dividida nos seguintes grupos: Ferramentas Informatizadas (identificadas pelas letras FI e descritas na Figura 5.6), Materiais de Realização de Testes (identificados pelas letras RT e descritos na Figura 5.7), Materiais de Treinamento (identificados pelas letras MT e descritos na Figura 5.8), Questionários (identificados pelas letra Q e descritos na Figura 5.9) e Formulários (identificados pelas letra F e descritos na Figura 5.10).

FI01	Ambiente Java – necessário para as revisões de programa e execução da ferramenta <i>SoftVis_{OA}H</i> .
FI02	Eclipse SDK – necessário para as localizações de defeitos nos artefatos de software.
FI03	Plugin <i>AJDT</i> (<i>AspectJ Development Tool</i>) para o <i>Eclipse</i> – Programação Orientada a Aspectos em <i>AspectJ</i> para o <i>Eclipse</i> .
FI04	<i>SoftVis_{OA}H</i> – Ferramenta de Visualização de Software.

Figura 5.6: Instrumentação – Ferramentas Informatizadas – *Planejamento* (d'Arce, 2012)

RT01	Artefato de software <i>GanttProject</i> com defeitos pré-inseridos – escolhido pelo fato de ser um projeto de código aberto construído no <i>Eclipse</i> (o qual permite a instalação do <i>AspectJ</i>).
RT02	Artefato de software <i>Memoranda</i> com defeitos pré-inseridos – escolhido pelo fato de ser um projeto de código aberto construído no <i>Eclipse</i> (o qual permite a instalação do <i>AspectJ</i>).
RT03	Relação de defeitos inseridos no artefato <i>RT01</i> – cada participante deve informar a localização de cada defeito contido nessa relação.
RT04	Relação de defeitos inseridos no artefato <i>RT02</i> – cada participante deve informar a localização de cada defeito contido nessa relação.
RT05	Artefatos de software para treinamento na ferramenta – é necessário um treinamento prático na ferramenta <i>SoftVis_{OA}H</i> , onde cada participante visualizará alguns artefatos de software.

Figura 5.7: Instrumentação – Materiais de Realizacao de Testes – *Planejamento* (d'Arce, 2012)

Dois *Artefatos* de *Software* foram utilizados (o *GanttProject* e o *Memoranda*), ambos são projetos de código aberto desenvolvidos em Java. O *GanttProject*

MT01	Slides de apresentação do experimento.
MT02	Slides de treinamento em Visualização de Software e Compreensão de Programa.
MT03	Informativo de preenchimento de questionários e formulários – especifica como preencher devidamente os questionários e formulários utilizados.
MT04	Slides de treinamento em Programação Orientada a Aspectos.
MT05	Slides de treinamento em <i>JUnit</i> .
MT06	Slides de treinamento da ferramenta <i>SoftVis_{OA}H</i> .

Figura 5.8: Instrumentação – Materiais de Treinamento – *Planejamento* (d'Arce, 2012)

Q01	Levantamento de perfil – para caracterização dos participantes.
Q02	<i>Feedback</i> do treinamento em Programação Orientada a Aspectos.
Q03	<i>Feedback</i> do treinamento em <i>JUnit</i> .
Q04	<i>Feedback</i> do treinamento em Visualização de Software e Compreensão de Programa.
Q05	<i>Feedback</i> do treinamento na ferramenta <i>SoftVis_{OA}H</i> .
Q06	<i>Feedback</i> da revisão de programa <i>ad hoc</i> .
Q07	<i>Feedback</i> da revisão de programa com o apoio da ferramenta <i>SoftVis_{OA}H</i> .

Figura 5.9: Instrumentação – Questionários – *Planejamento* (d'Arce, 2012)

F01	Localização de defeitos e Compreensão de Programa – os participantes devem informar cada defeito localizado, o horário de cada localização e a descrição do funcionamento de cada <i>adendo</i> contido em cada <i>aspecto</i> do artefato de software.
F02	Termo de consentimento de participação no experimento – cada participante deve preencher este termo de consentimento, elaborado com o auxílio de um advogado.

Figura 5.10: Instrumentação – Formulários – *Planejamento* (d'Arce, 2012)

consiste em um sistema de gerenciamento de projetos e está disponível em <http://www.ganttproject.biz>. O Memoranda consiste em um sistema de controle de agenda e de apontamentos pessoais e encontra-se disponível a partir de <http://memoranda.sourceforge.net>. Eles foram escolhidos devido ao fato de os participantes selecionados terem conhecimento prévio a respeito de suas funcionalidades, assim os participantes não têm dificuldades em entender a descrição do defeito para a localização. Originalmente escritos em Java, os sistemas foram modificados (fatorando-se algumas funcionalidades para *AspectJ*) para serem utilizados como artefatos para a avaliação da ferramenta *SoftVis_{OA}H*. Após a fatoração das funcionalidades e desconsiderando os subprojetos não utilizados no experimento, a estrutura do *GanttProject* resultou em 45 pacotes, 482 classes e 5 aspectos. Já o *Memoranda* resultou em 24

pacotes, 128 classes e 7 aspectos. No artefato GanttProject, foram inseridos: 1 defeito para Fato Incorreto, 1 defeito para Falha em estabelecer pós-condições esperadas, 1 defeito para Falha em preservar estados invariantes e 1 defeito para Alterações incorretas em dependências de controle. No artefato Memoranda, foram inseridos: 1 defeito para Informação estranha, 1 defeito para Força incorreta nos padrões de Pointcuts, 1 defeito para Falha em preservar estados invariantes e 1 defeito para Fato Incorreto.

As informações incluídas na atividade *Planejamento* estão associadas aos conceitos destacados na cor azul na Figura 5.11 da *ExperOntology*. Ainda na mesma figura é destacada na cor amarela as informações da atividade anterior (*Definição*) para ilustrar que existe uma ligação entre ambas e que para a elaboração da atividade de *Planejamento* os itens que fazem parte da *Definição* são necessários.

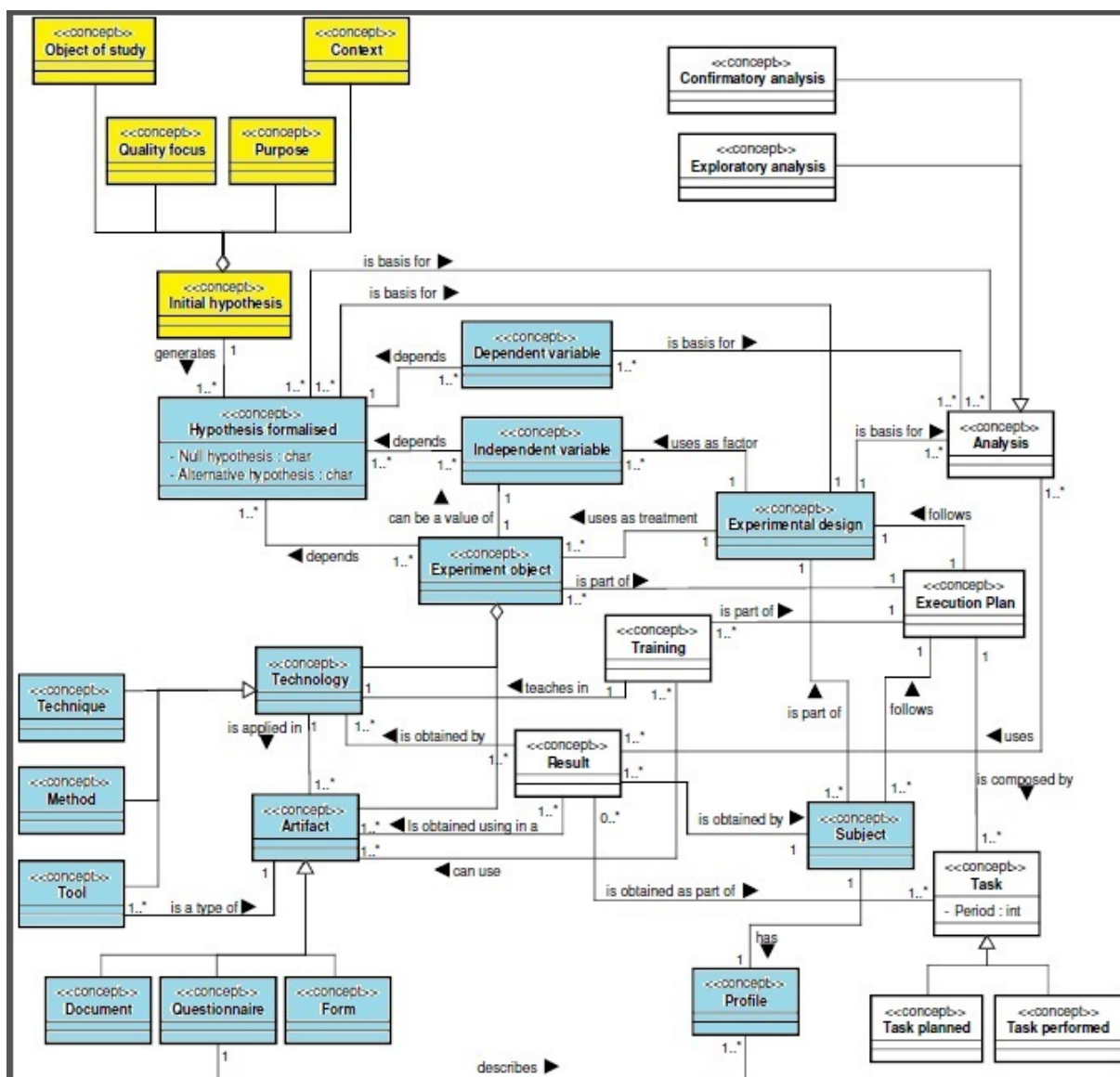


Figura 5.11: Conceitos organizados na *ExperOntology* da Atividade *Planejamento*, adaptado de Garcia et al. (2008)

5.3.3 Operação do Experimento

Para a execução deste experimento, dezessete participantes foram aleatoriamente divididos em dois grupos: os participantes de código ímpar foram alocados para o Grupo A e os participantes de código par foram alocados para o Grupo B. Cada participante teve seu perfil caracterizado por meio de um questionário de levantamento de perfil diferentemente dos formulários e questionários distribuídos em planilhas eletrônicas, o questionário de levantamento de perfil foi disponibilizado na *OntoExpTool*. Os participantes realizaram os mesmos tipos de atividade, revisões de programa ad hoc e com o apoio da ferramenta *SoftVis_{OA}H*, exercendo entendimento do programa e a localização de defeitos no código fonte.

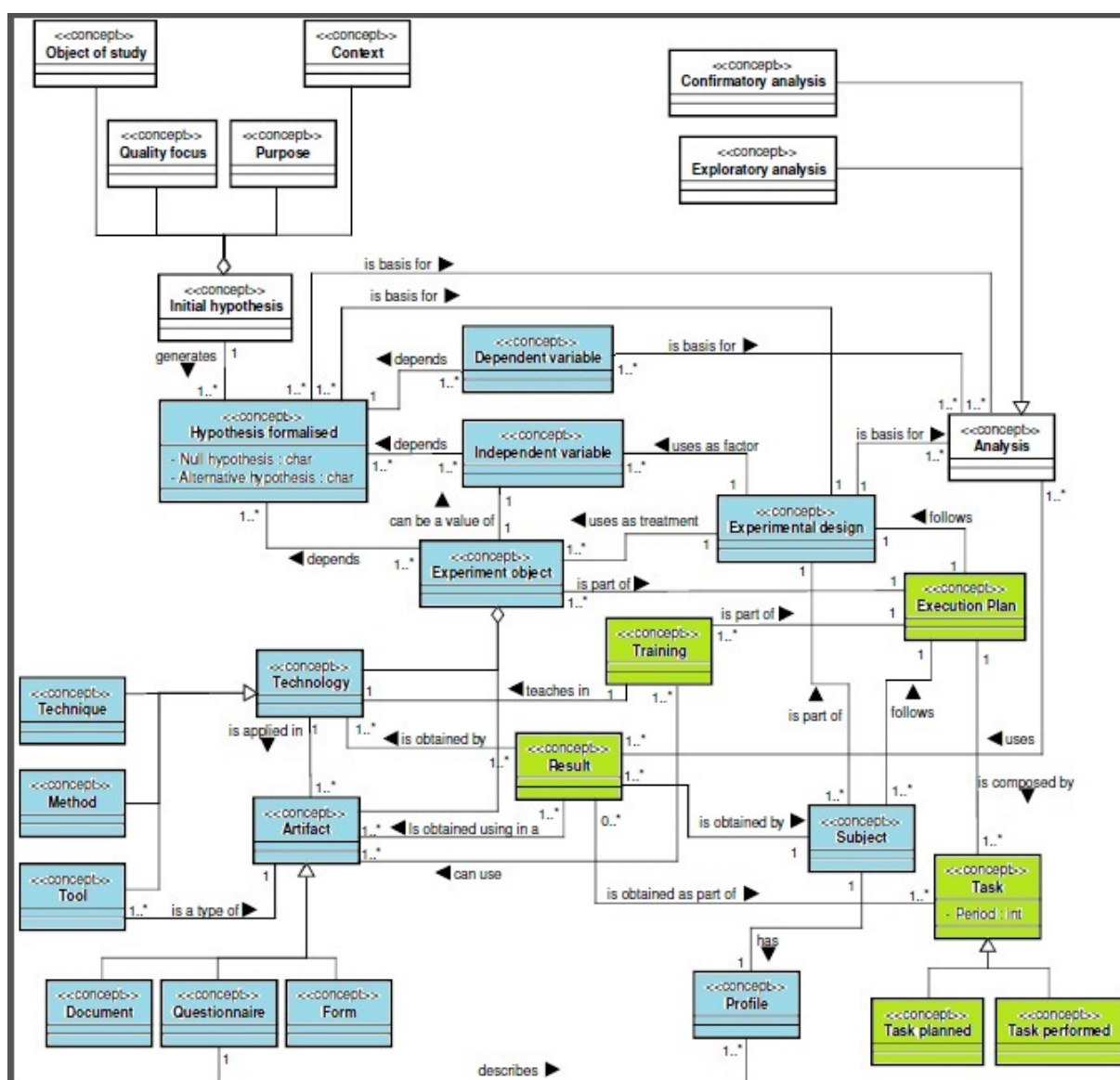


Figura 5.12: Conceitos organizados na *ExperOntology* da Atividade Operação, adaptado de [Garcia et al. \(2008\)](#)

As informações incluídas na atividade *Planejamento* estão associadas aos

conceitos destacados na cor verde na Figura 5.12 da *ExperOntology*. Na mesma figura ainda é destacada na cor azul as informações da atividade anterior (*Planejamento*) para ilustrar que existe uma ligação entre ambas e que para a elaboração e execução da atividade de *Operação* os itens que fazem parte do *Planejamento* são necessários.

5.3.4 Análise e Interpretação dos dados do Experimento

De acordo com os dados referentes à análise executada no experimento de d'Arce (2012) os participantes descreveram corretamente mais adendos de aspectos utilizando a *SofVis_{OA}*. Em relação ao quesito auxílio à Compreensão de Programa, referente à Hipótese Nula (H_{10}), foi aplicado o Teste-T Bicaudal com Variação Desigual (Wohlin et al., 2012) e considerando o nível padrão de significância de 5%, foi possível rejeitar a Hipótese Nula (H_{10}). Portanto, constata-se que a utilização da ferramenta *SofVis_{OA}* provê auxílio à Compreensão de Programa. Esta é a descrição de uma dentre outras análises e interpretações executadas no experimento demonstrado.

As informações incluídas na atividade *Análise e Interpretação* estão associadas aos conceitos destacados na cor cinza na Figura 5.13 da *ExperOntology*. Ainda na mesma figura é destacada na cor verde as informações da atividade anterior (*Operação*) para ilustrar que existe uma ligação entre ambas e que para a elaboração e execução da atividade de *Análise* os itens que fazem parte da *Operação*, bem como o item "Projeto Experimental" (*Experimental design*) da atividade de *Planejamento*, são necessários.

5.3.5 Empacotamento e Apresentação dos dados do Experimento

No decorrer da execução do experimento por meio da *OntoExpTool* os dados foram armazenados em uma base de dados e após a conclusão da atividade *Análise e Interpretação* o Pacote de Laboratório deste experimento foi criado em dois formatos: XML e OWL que podem ser visualizados na própria ferramenta, bastando clicar sobre o link de cada conforme demonstrado na Figura 5.14. Dentro dos arquivos gerados em XML e OWL, existe um link de um servidor no qual o pesquisador interessado poderá baixar os arquivos (.CSV ou .PDF) que contém os resultados da análise do experimento.

Também por meio da *OntoExpTool* foi criado um pacote compactado no formato ZIP contendo além dos arquivos XML e OWL, todos os artefatos e documentos provenientes do estudo e que pode ser baixado, clicando no respectivo link que se encontra na atividade de Empacotamento, ficando disponível para análises e futuras replicações. Nas Figuras 5.15 e 5.16 são ilustrados trechos do código gerado no pacote de laboratório em XML e OWL, respectivamente.

Para a replicação de um experimento, é necessário que haja estudos concluídos, e assim utilizar um pacote de laboratório existente, entretanto, inde-

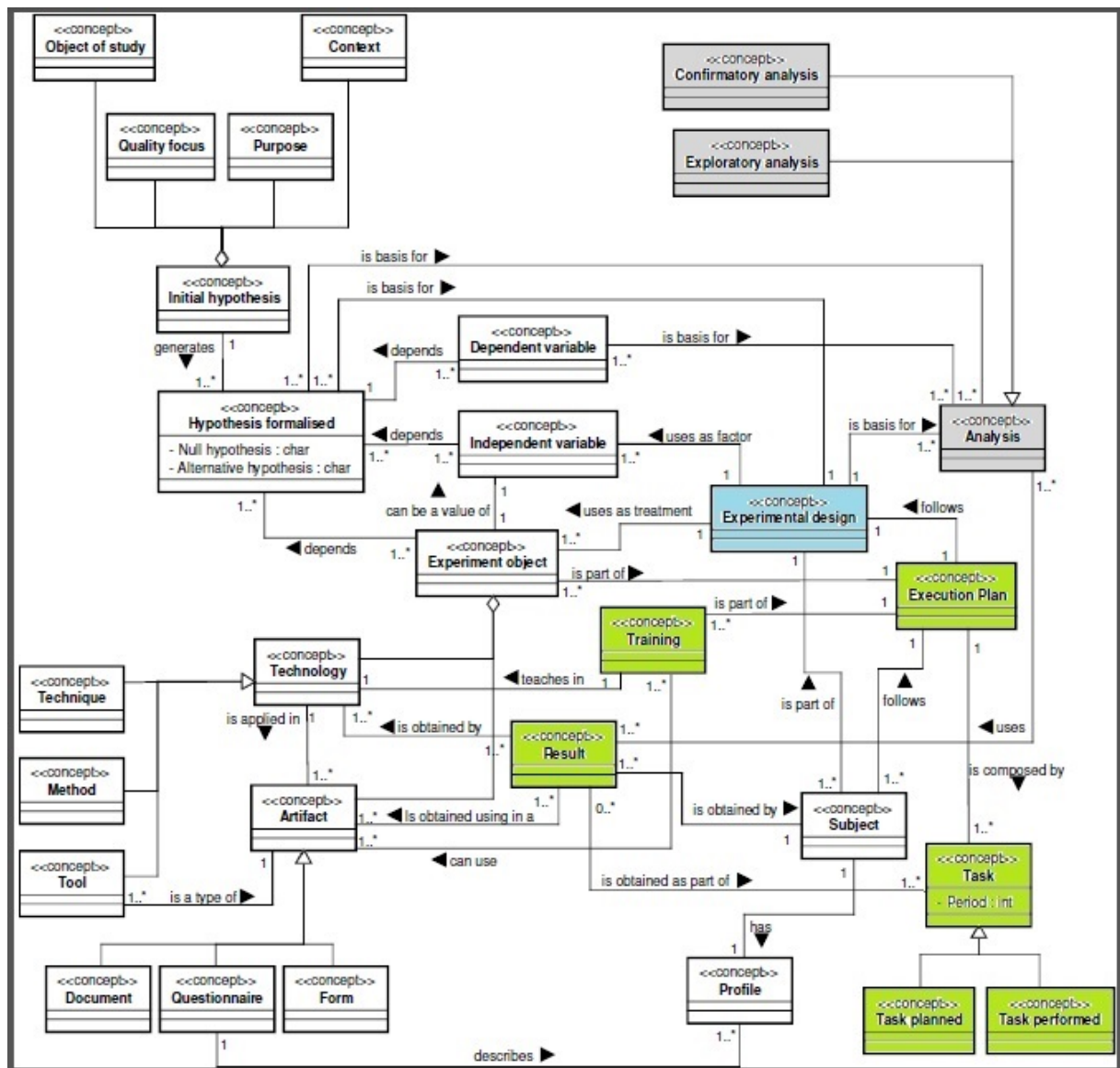


Figura 5.13: Conceitos organizados na *ExperOntology* da Atividade *Análise*, adaptado de [Garcia et al. \(2008\)](#)

pendente de como a replicação será efetuada, seja externa ou interna, a ferramenta protege os dados das atividades de *Definição* e *Planejamento* no novo experimento, tornando possível apenas adicionar novos parâmetros conforme necessário.

No Apendice B estão ilustradas figuras com instantâneos das interfaces da ferramenta *OntoExpTool* para cada atividade do processo experimental com as informações do experimento empacotado.



Figura 5.14: Instantâneo da Interface para a Atividade de *Empacotamento*



Figura 5.15: Arquivo XML - Pacote de Laboratório gerado



```
<?xml version="1.0"?>
<Ontology xmlns="http://www.w3.org/2002/07/owl#"
  xml:base="http://www.semanticweb.org/ontologies/2011/8/ExperOntology.owl"
  >
  <ClassAssertion>
    <Class IRI="#ConclusionValidity"/>
    <NamedIndividual IRI="#DeConstrucao"/>
  </ClassAssertion>
  <ObjectPropertyAssertion>
    <ObjectProperty IRI="#has"/>
    <NamedIndividual IRI="#Exp1"/>
    <NamedIndividual IRI="#Externa"/>
  </ObjectPropertyAssertion>
  <ObjectPropertyAssertion>
    <ObjectProperty IRI="#has"/>
    <NamedIndividual IRI="#Exp1"/>
    <NamedIndividual IRI="#Interna"/>
  </ObjectPropertyAssertion>
</Ontology>
```

Figura 5.16: Arquivo OWL - Pacote de Laboratório gerado

5.4 Considerações Finais

Neste capítulo foi apresentado a demonstração de um experimento controlado executado por meio da ferramenta *OntoExpTool*, ilustrando o ponto de vista do experimentador seguindo as atividades do processo experimental proposta no *workflow* deste projeto, associando e destacando as informações registradas em cada atividade com os conceitos organizados contidos na ontologia *ExperOntology*.

Conclusões

O foco principal da Engenharia de Software Experimental é avaliar e medir a performance de modelos e técnicas em contextos práticos, com o intuito de estabelecer uma base de conhecimento para apoiar a tomada de decisão. A execução de um estudo experimental gera muitas informações que compõem a base de conhecimento que deve ser registrada em Pacote de Laboratório.

Conforme a literatura indicada, experimentadores encontram dificuldades para compreender as informações registradas ao revisar um pacote de laboratório, principalmente por falta de organização e padronização. Com o intuito de auxiliar experimentadores na condução de experimentos controlados bem como a criação de pacotes de laboratório organizados e padronizados, foi desenvolvido o *workflow* descrito no Capítulo 4. O *workflow* é baseado no processo de experimentação apresentado no Capítulo 2 e provê uma sequência de tarefas para a condução de um estudo controlado, coletando informações essenciais, como: *hipóteses, variáveis, objetos, participantes, contexto do experimento e projeto do experimento*, utilizados nas atividades *definição, planejamento, operação, análise e interpretação e apresentação e empacotamento*. Este *workflow* foi implementado em uma ferramenta de apoio à condução de experimentos controlados, chamada *OntoExpTool*. Seus diagramas de Casos de Uso, de Sequência e de classe foram apresentados no Capítulo 4.

A principal contribuição deste trabalho é a criação de pacotes de laboratório contendo as informações de cada fase da execução de um experimento controlado, de maneira organizada e padronizada, baseado nos conceitos existentes na *ExperOntology*. Espera-se com isso facilitar a interpretação e o entendimento do pacote de laboratório por parte de um pesquisador interessado em analisar o experimento. O pacote de laboratório gerado pela ferramenta e que é baseado na ontologia *ExperOntology* é no formato OWL, conforme descrito no Capítulo 5.

Para enfatizar a contribuição deste trabalho, três elementos isolados presentes na literatura são apresentados na Figura 6.1: em A é apresentado o *FIRE*; em B é apresentado o processo experimental, deixando o pacote de laboratório fora do escopo deste; e em C a *ExperOntology*. Como mencionado, tais elementos estão presentes na literatura, mas de modo isolado, não havendo a interligação entre eles. A contribuição deste trabalho é representada na Figura 6.1 como as setas que interligam os elementos: a execução de um experimento (ciclo interno do *FIRE* – A) segue o processo experimental (B) que instancia o Pacote de Laboratório usando a organização da *ExperOntology* (C).

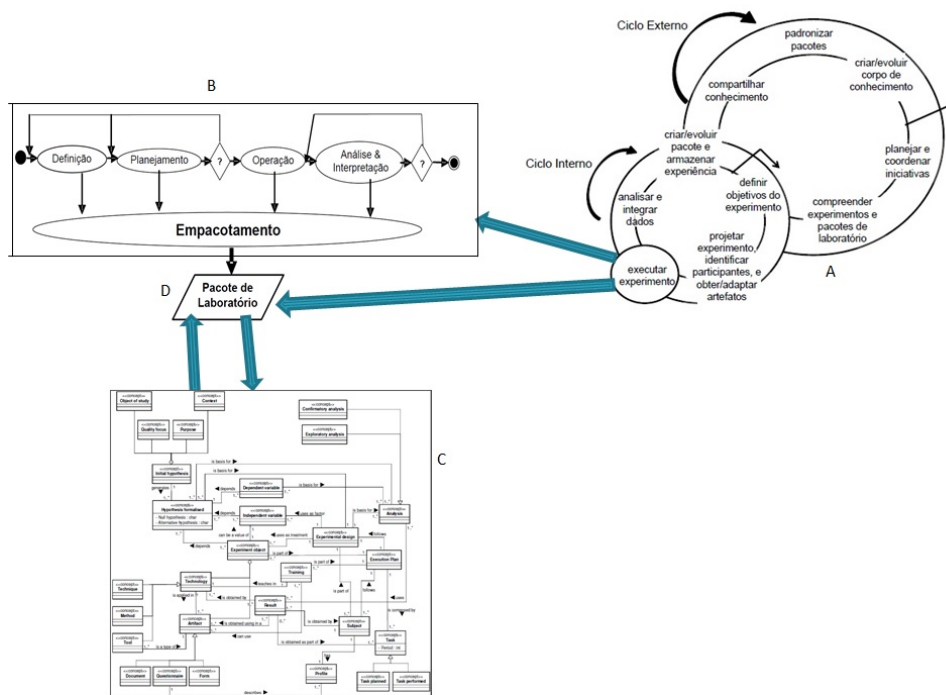


Figura 6.1: Detalhes de um processo experimental com a criação de um Pacote de Laboratório baseado na *ExperOntology*

Por meio do pacote de laboratório criado pela ferramenta no formato OWL e que foi instanciado e organizado baseado nos conceitos da *ExperOntology*, outro pesquisador ou grupo de pesquisa pode utilizar, por exemplo, um *software* com a função de *reasoner*, para apoiá-lo no entendimento dos conceitos registrados no Pacote de Laboratório.

Como outras contribuições deste trabalho, um artigo foi publicado na *SERP'14 - The 2014 International Conference on Software Engineering Research and Practice*, denominado *ExpTool: a Tool to Conduct, Package and Replicate Controlled Experiments in Software Engineering*. Além disso, será disponibilizado o acesso à ferramenta *OntoExpTool* via *web* para utilização pública por qualquer pesquisador ou grupo de pesquisa (por meio de contato com os administradores deste projeto).

Como ponto fraco deste trabalho, cita-se a limitação de apenas ter sido

registrado o experimento controlado concluído, citado no Capítulo 5, como maneira de avaliar o funcionamento da ferramenta, e, portanto, não ter sido possível a execução de um experimento na íntegra, bem como replicações, ficando como proposta de serem realizados futuramente e publicados por meio de artigos.

Como trabalhos futuros também são propostas as seguintes atividades:

- Adaptação da ferramenta para que possam ser executados outros tipos de experimentos além dos controlados;
- Modificação da ferramenta para que ela possa editar e configurar o *work-flow* de acordo com as necessidades específicas de cada experimento, seguindo a vontade do experimentador.
- Criação de um módulo de comunicação entre o pesquisador e os participantes do experimento, disponibilizando troca de *e-mails*, *chat* e videoconferência para o auxílio das atividades de experimentação, como o caso de ser necessária alguma reunião para esclarecimento de dúvidas ou para treinamento;
- Estabelecer um comparativo entre experimentos realizados manualmente e estudos feitos com o auxílio da ferramenta, verificando assim a sua eficácia e como consequência a publicação de artigos.

Diagramas da Ferramenta

A.1 Diagramas de Sequência

Na Figura [A.1](#) é apresentado o Diagrama de Sequência – Formulacao Hipoteses, que possui evento que proporciona a seleção de um experimento para registrar as hipóteses Nula e Alternativa(s) (vide Seção [2.2.5.2](#)). O Diagrama de Sequência – Selecao Variaveis, é ilustrado na Figura [A.2](#), e possui evento que proporciona a seleção de um experimento para registrar as Variáveis Dependentes e Independetes (vide Seção [2.2.5.3](#)). O Diagrama de Sequência – Selecionar Participante, é ilustrado na Figura [A.3](#), e possui evento que proporciona a seleção de um participante por vez e um experimento para registrá-lo (vide Seção [2.2.5.4](#)). Na Figura [A.4](#) é apresentado o Diagrama de Sequência – Caracterizacao Participante, que possui evento que proporciona a seleção de um experimento para registrar os itens de caracterização, como questões, a serem aplicadas à um participante em outra fase do experimento (vide Seção [2.2.5.5](#)). Na Figura [A.5](#) é apresentado o Diagrama de Sequência – Associar Tecnica Artefato, que possui evento que proporciona a seleção de um experimento, seleciona artefato(s) e técnica a serem utilizadas no experimento para registrar a associação desses itens ao experimento (vide Seção [2.2.5.5](#)). O Diagrama de Sequência – Definir Cronograma, é ilustrado na Figura [A.6](#), e possui evento que proporciona a seleção de um experimento para registrar os dados referentes ao cronograma de execução do mesmo por parte dos participantes (vide Seção [2.2.5.3](#)). Por fim na atividade Registra Planejamento, foi criado o Diagrama de Sequência – Configurar Estudo, que é ilustrado na Figura [A.7](#), e possui evento que proporciona a seleção de um experimento, e os itens associados à ele, como técnica, artefato, cronograma e participantes, para assim criar os grupos de execução do experimento com as respectivas tarefas e de-

terminações, fazendo as devidas configurações para preparar o experimento para a próxima atividade de operação.

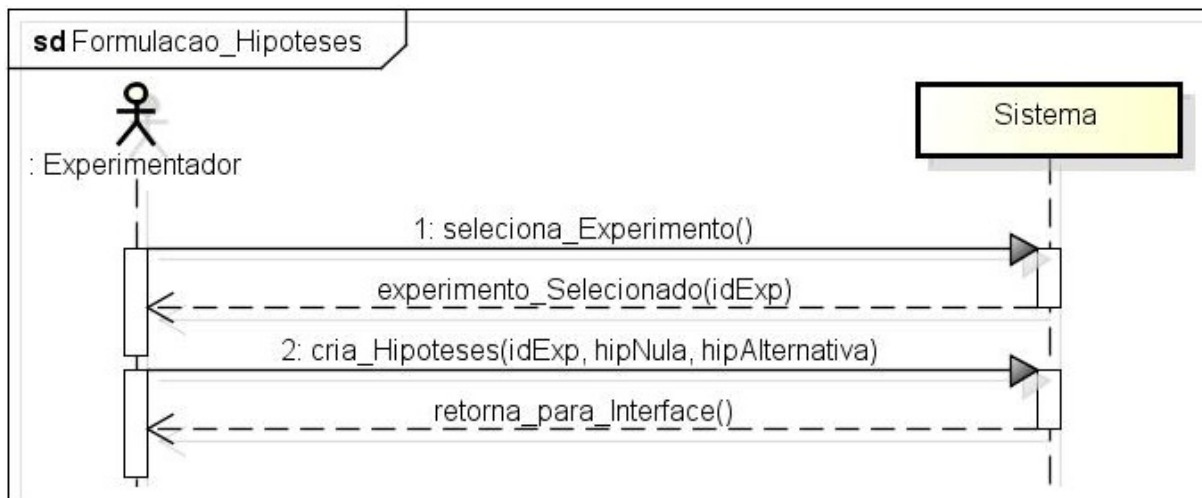


Figura A.1: Diagrama de Sequência – Formulação de Hipoteses – Nula e Alternativa

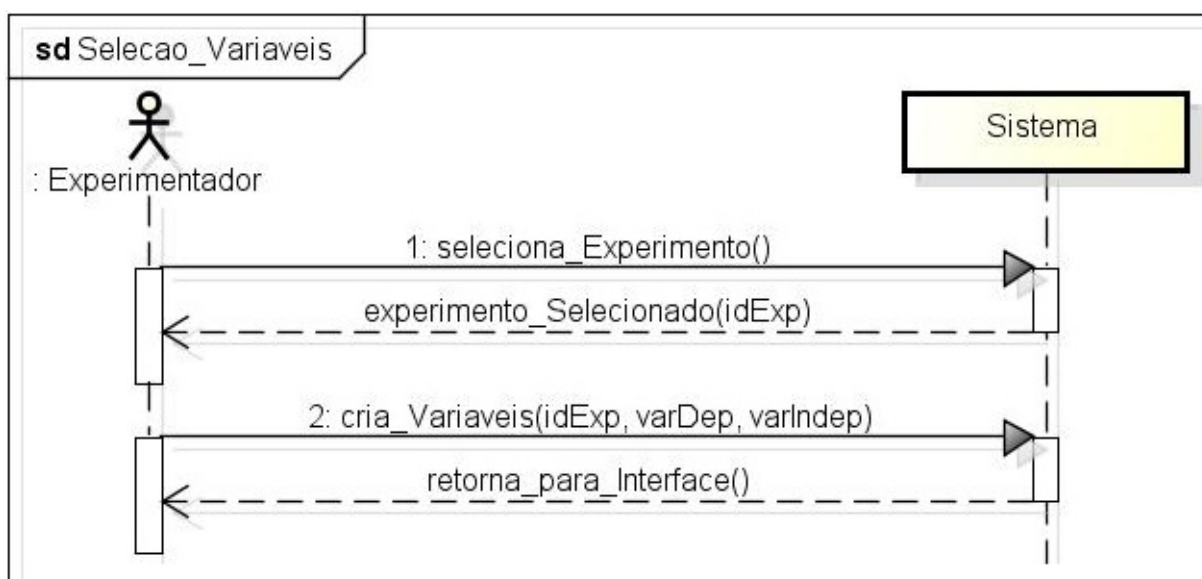


Figura A.2: Diagrama de Sequência – Seleção de Variáveis – Dependentes e Independentes

A.2 Diagramas de Classes - Particionados

Nas Figuras A.8, A.9, A.10 e A.11, são ilustrados os diagramas de classes particionados com maiores detalhes, como, os métodos de cada classes, atributos, parâmetros e tipos de retorno.

A.3 Estruturas da Base de Dados da ferramenta

Nas Figuras A.12, A.13, A.14, A.15 e A.16, são ilustrados as outras estruturas da Base de dados, particionadas com maiores detalhes, como, tabelas,

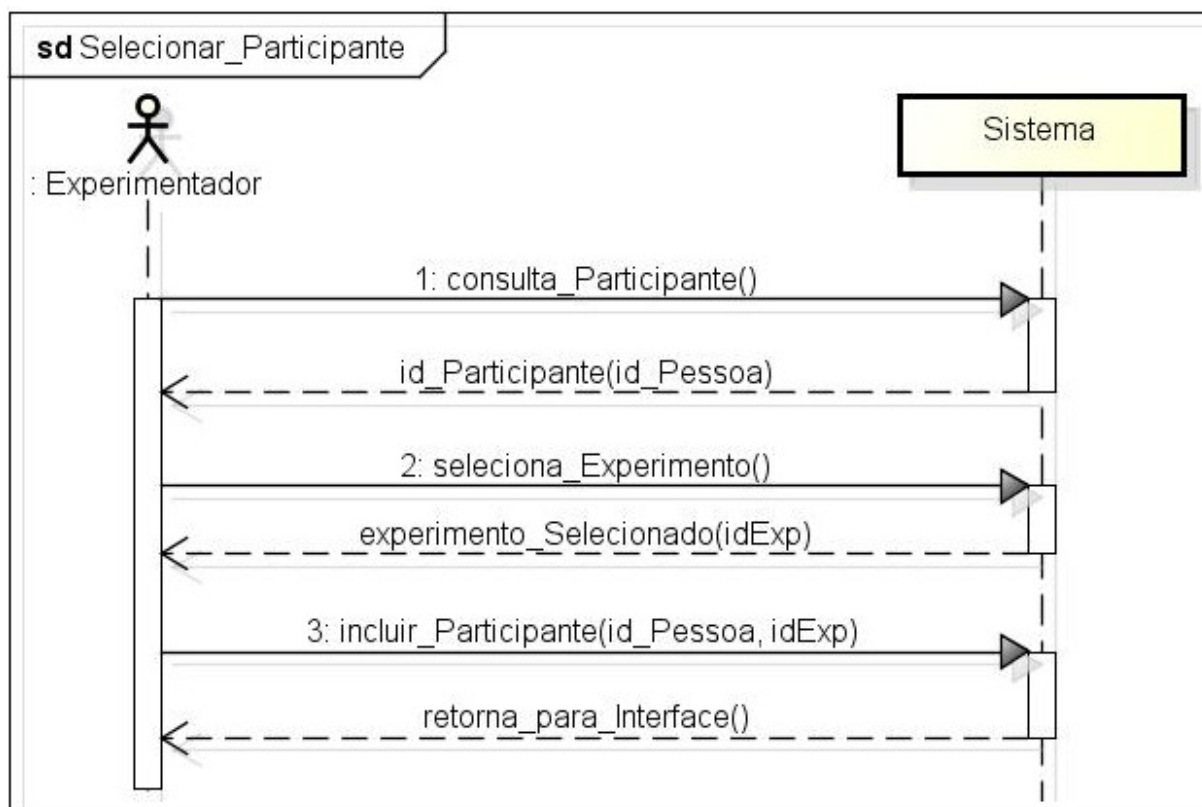


Figura A.3: Diagrama de Sequência – Selecionar Participantes

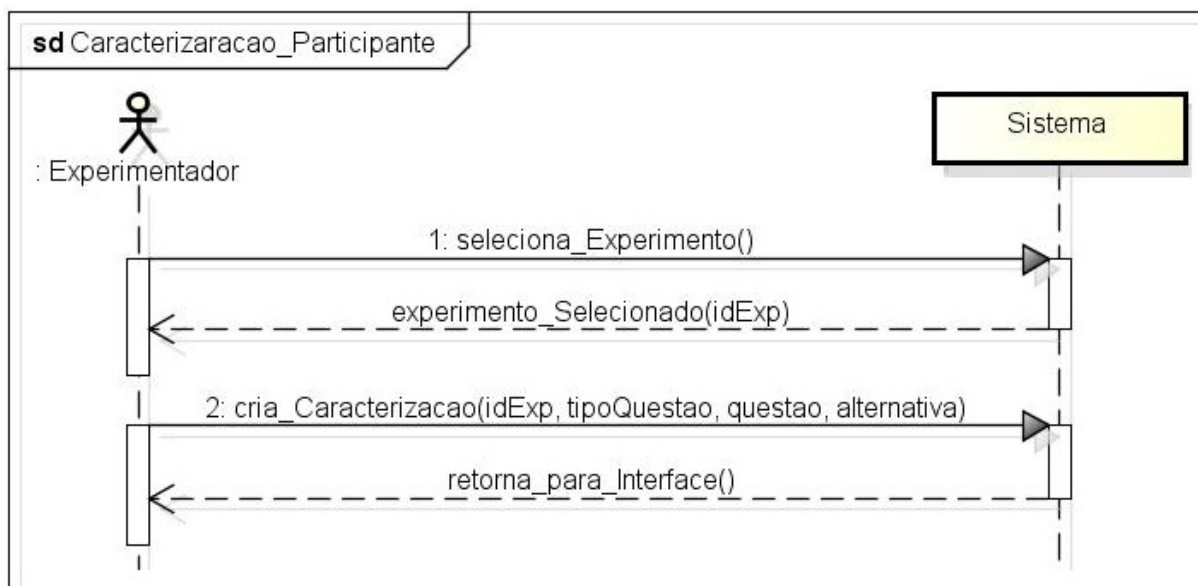


Figura A.4: Diagrama de Sequência – Caracterização de Participantes

atributos e relacionamentos.

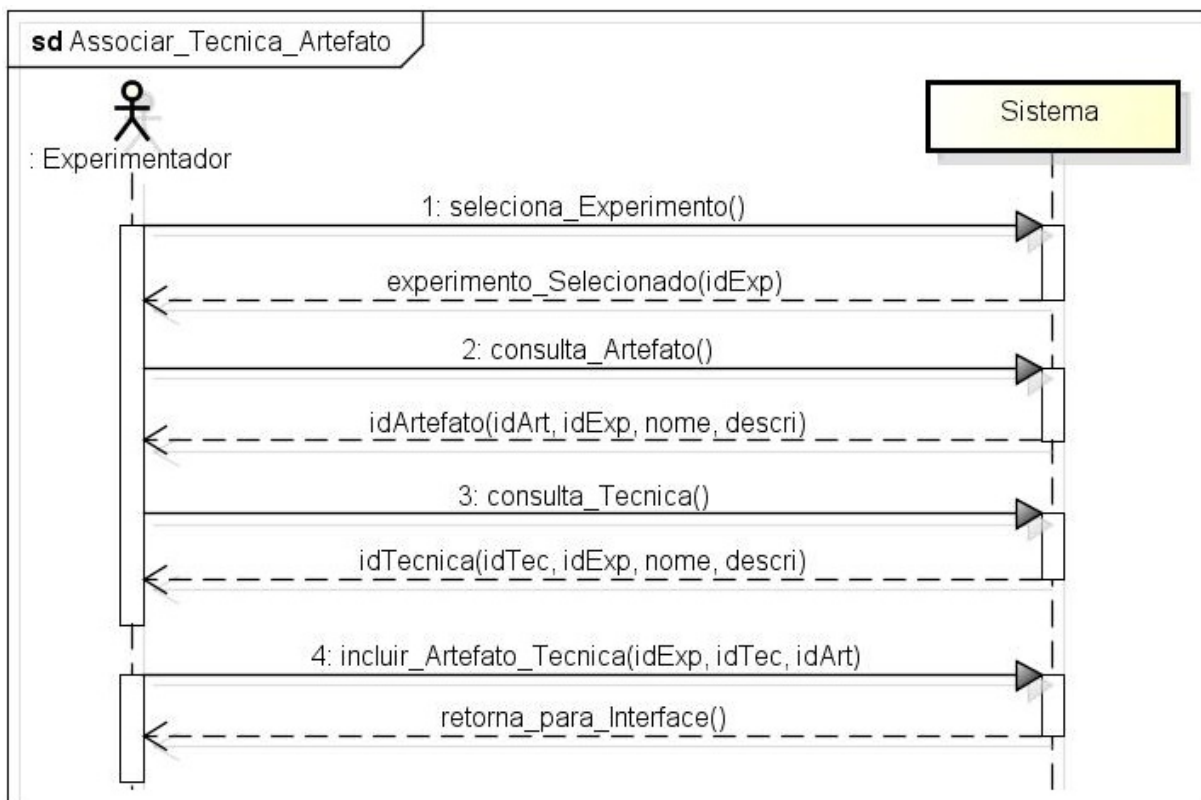


Figura A.5: Diagrama de Sequência – Associar Técnica-Artefato

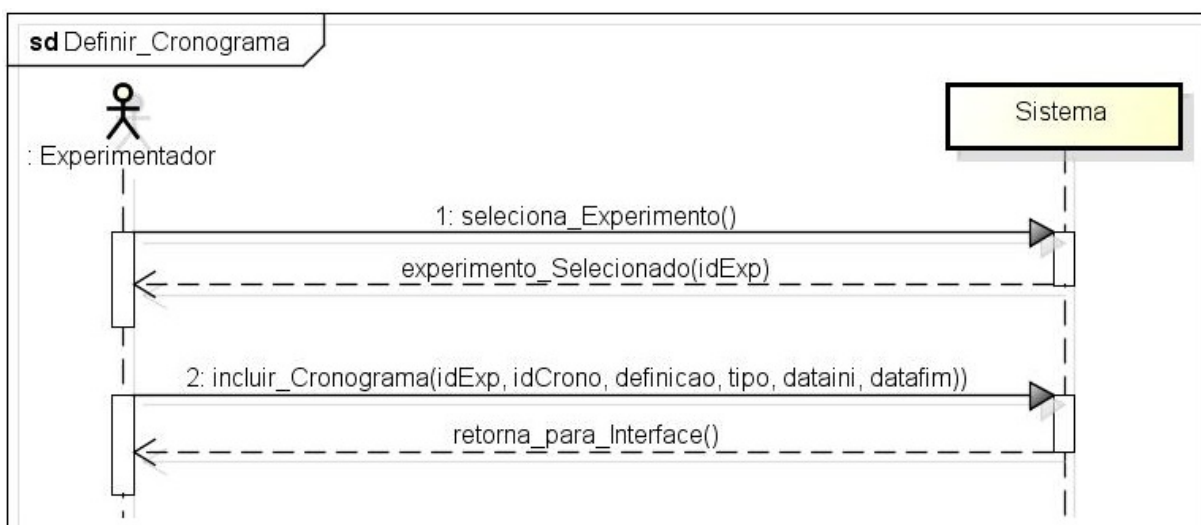


Figura A.6: Diagrama de Sequência – Definir Cronograma

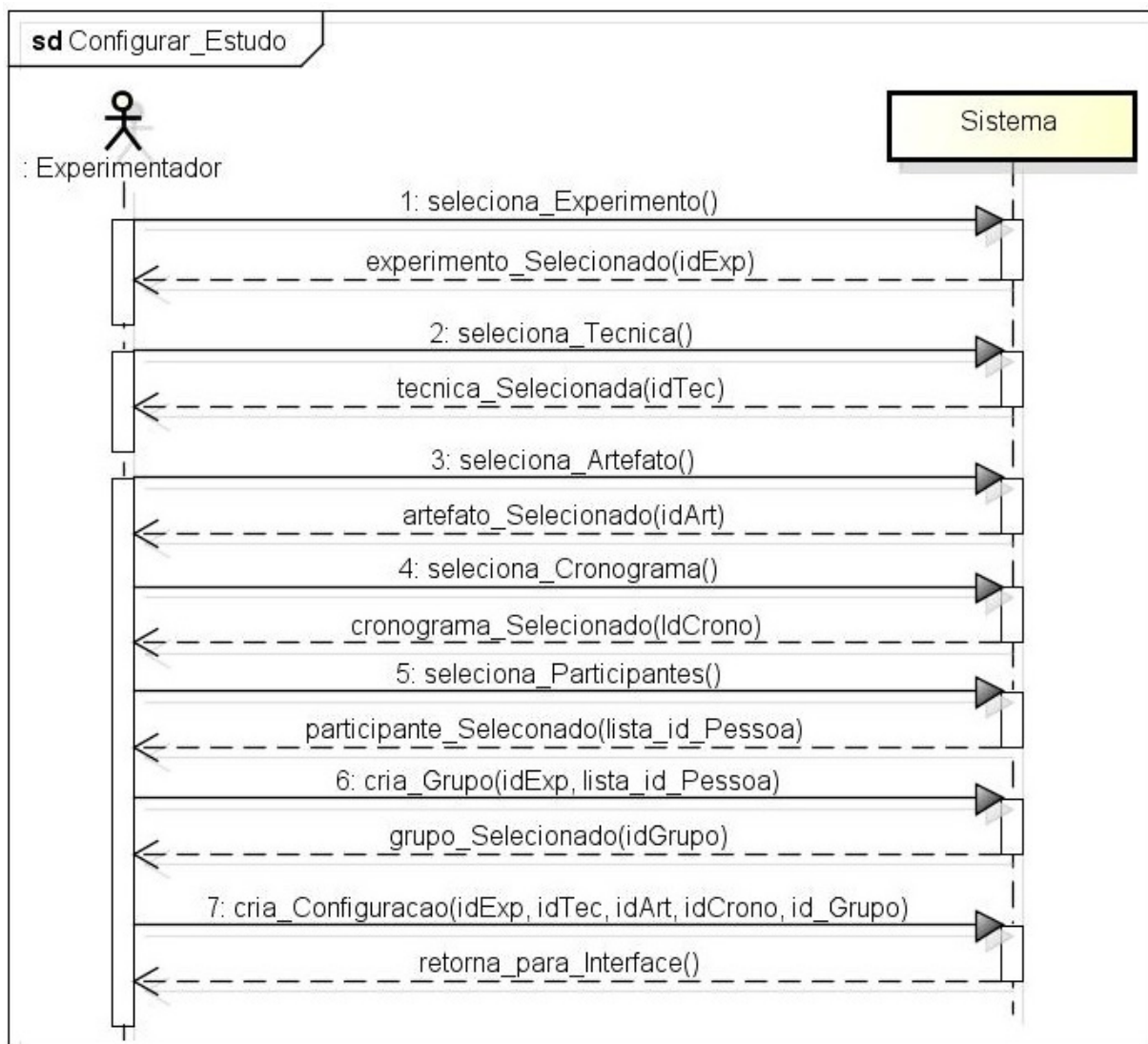


Figura A.7: Diagrama de Sequência – Configurar Estudo

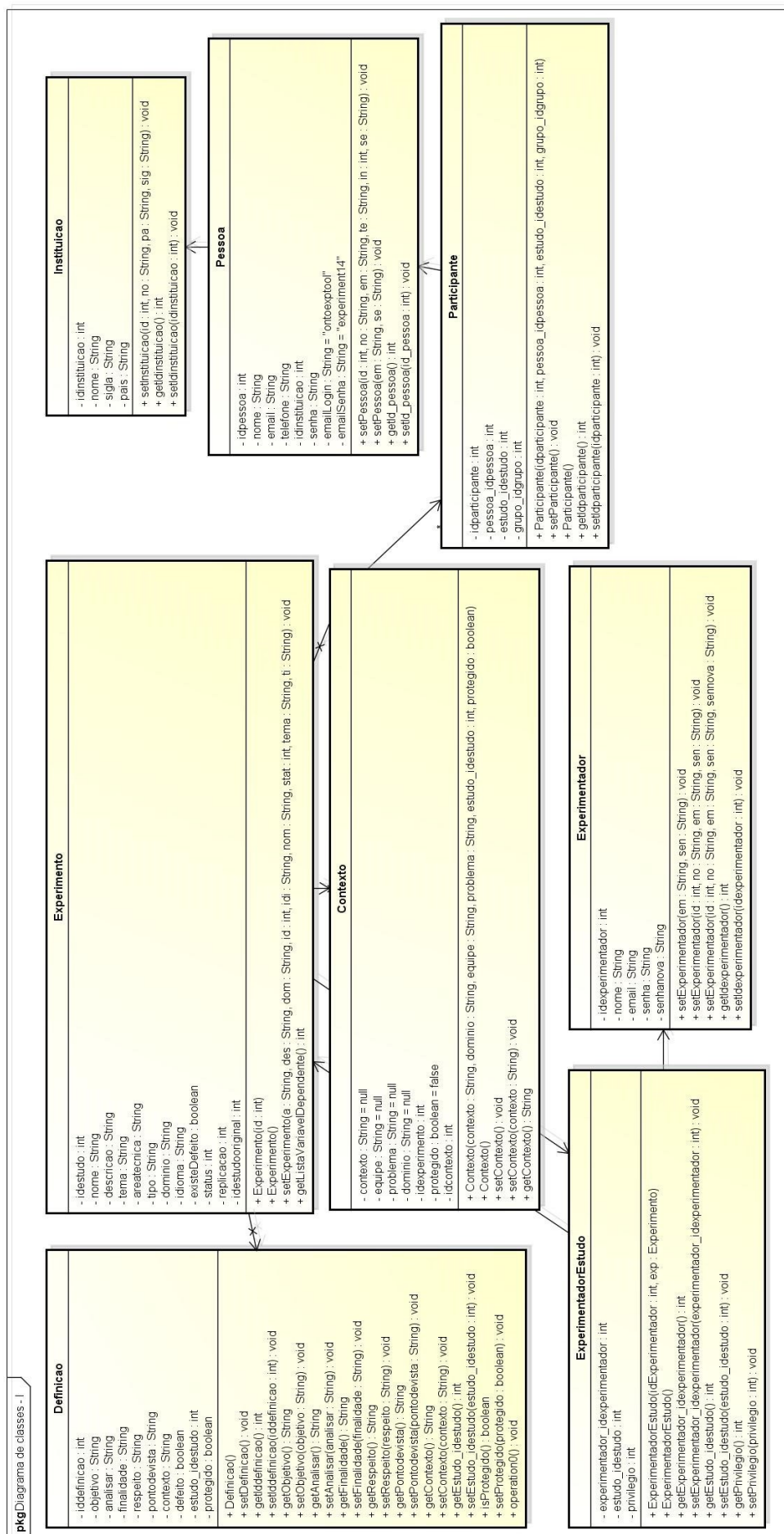


Figura A.8: Diagrama de Classes I

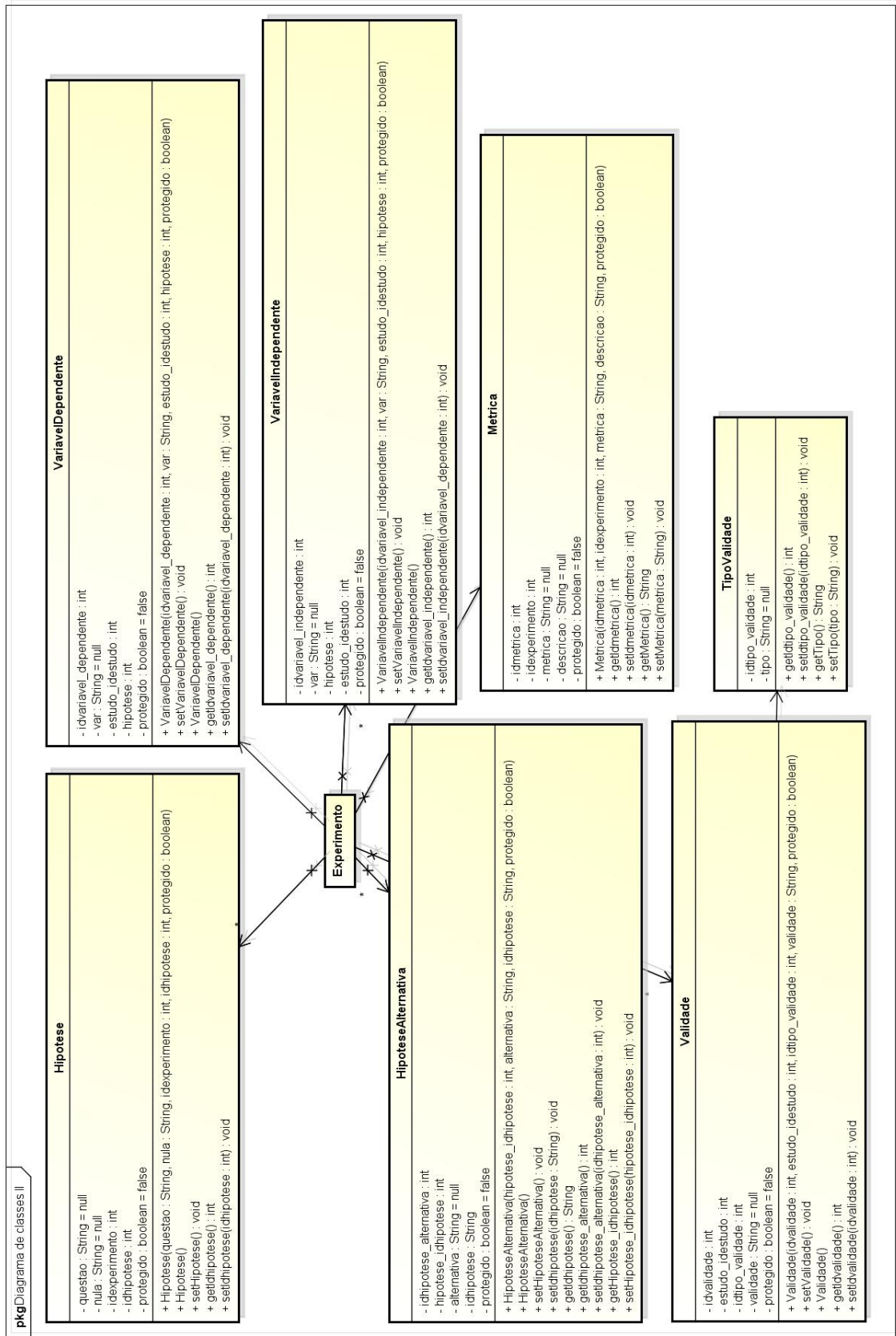


Figura A.9: Diagrama de Classes II

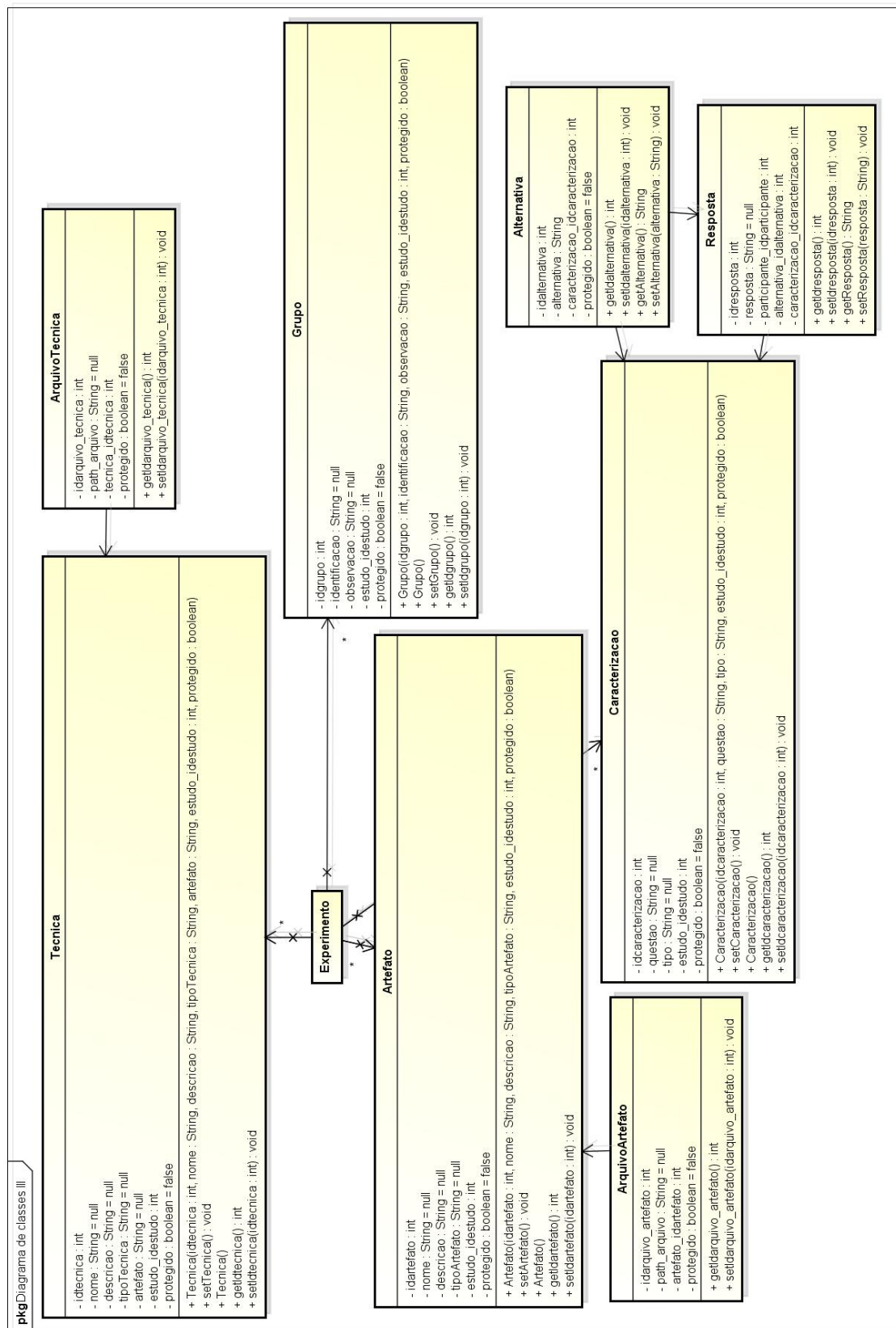


Figura A.10: Diagrama de Classes III

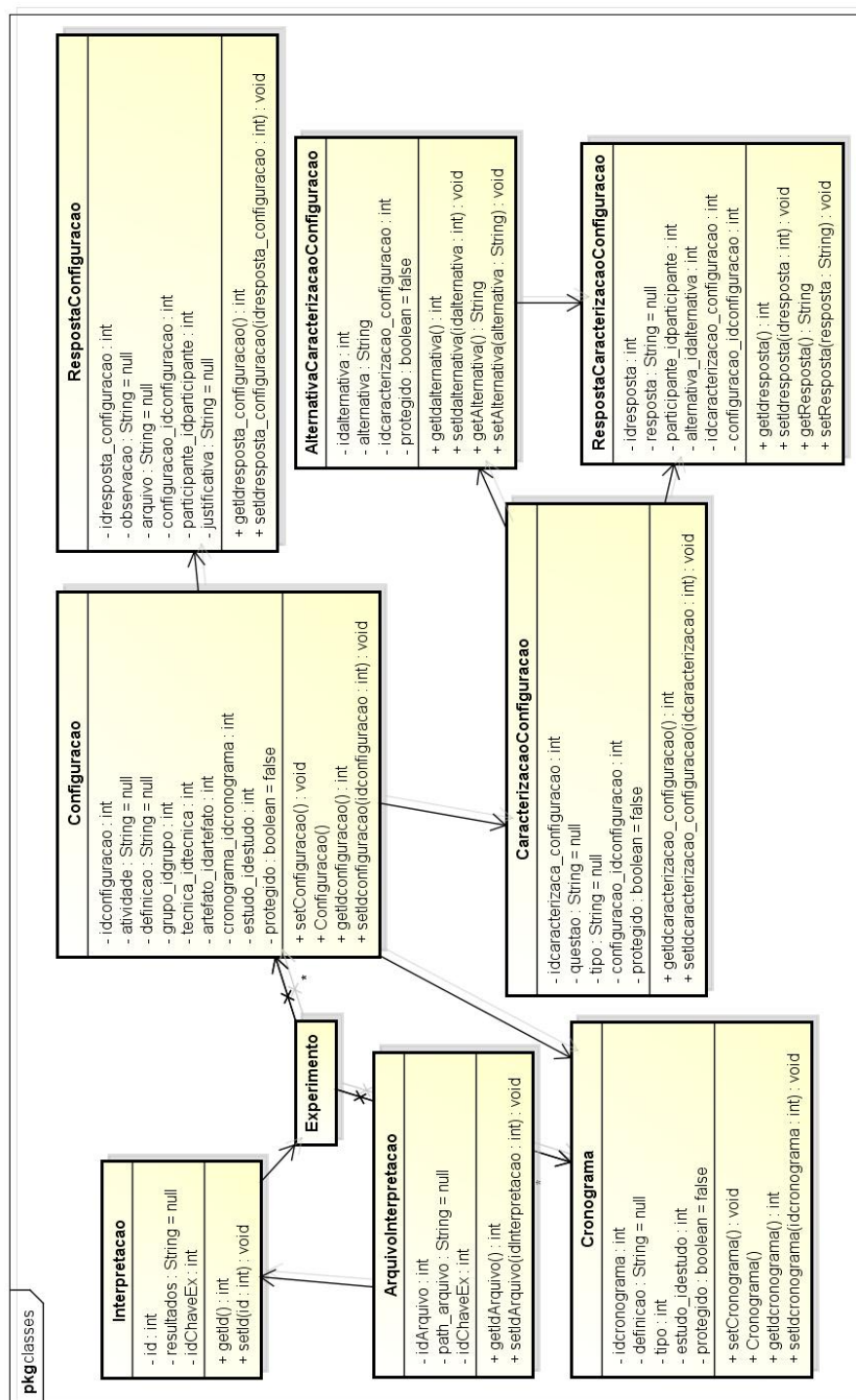


Figura A.11: Diagrama de Classes IV

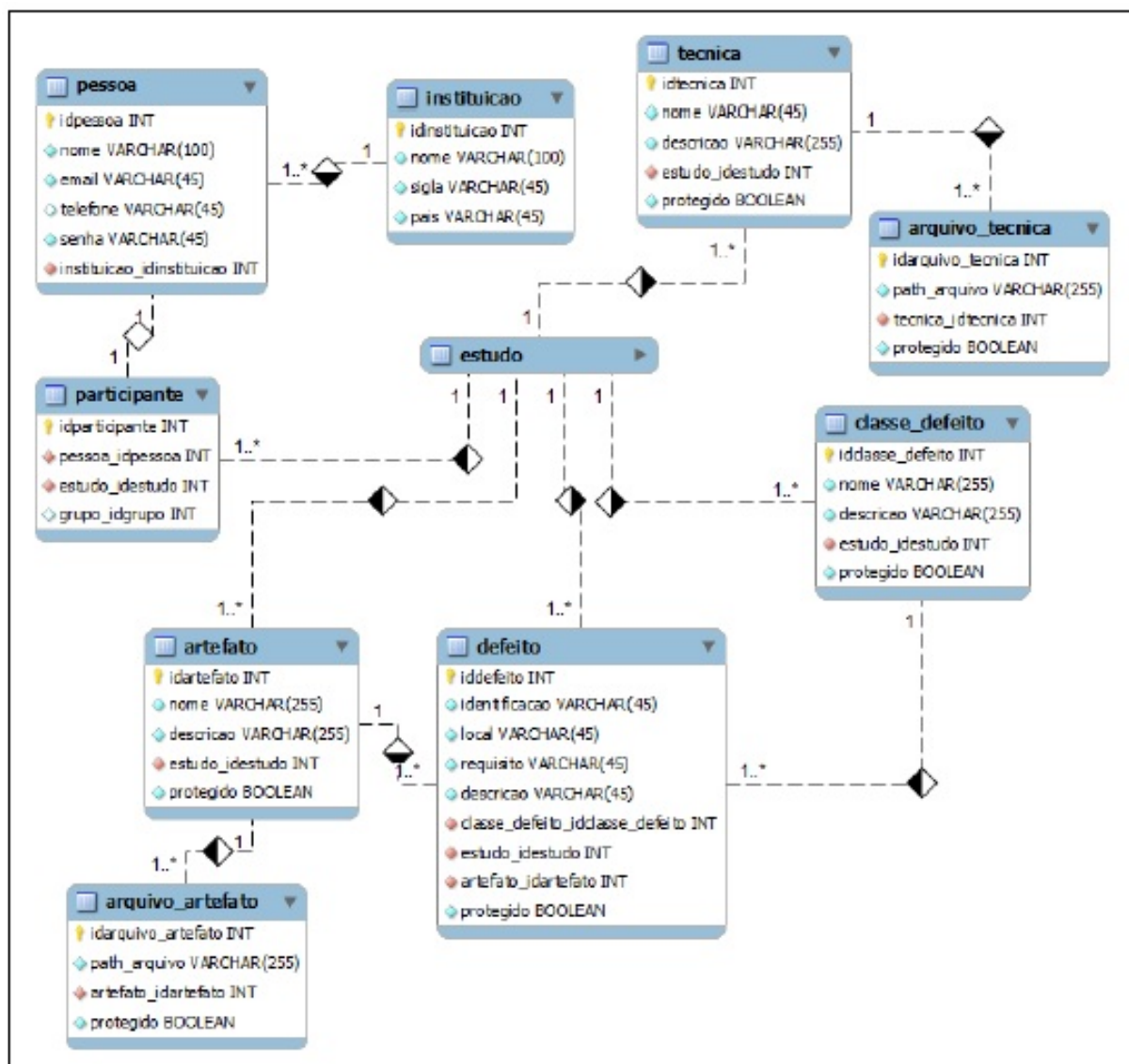


Figura A.12: Esquema da Base de Dados II

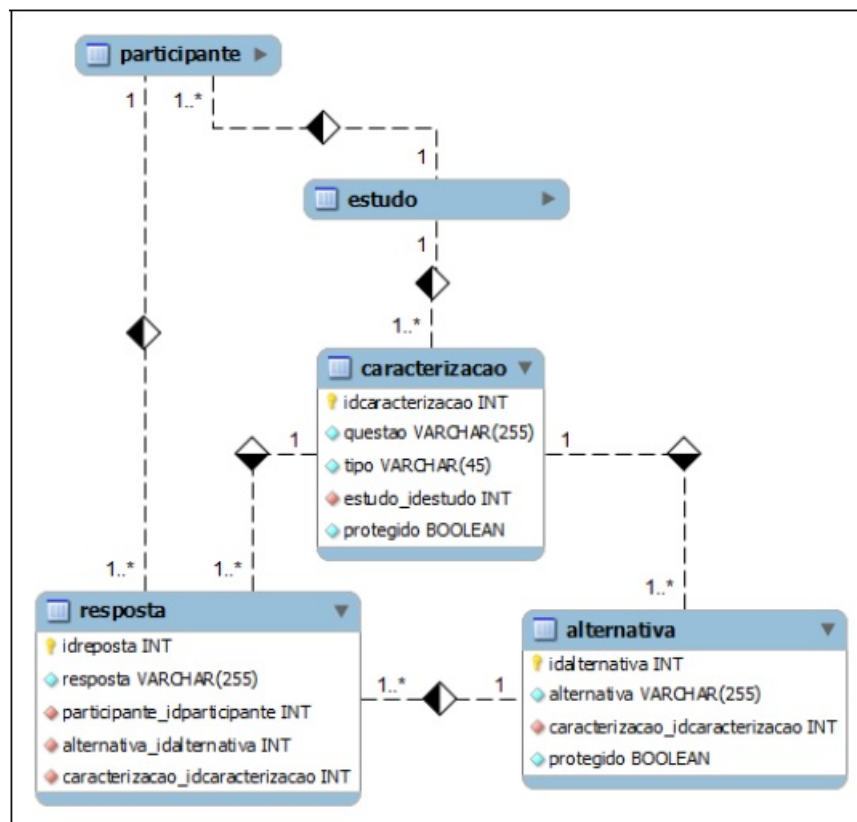


Figura A.13: Esquema da Base de Dados III

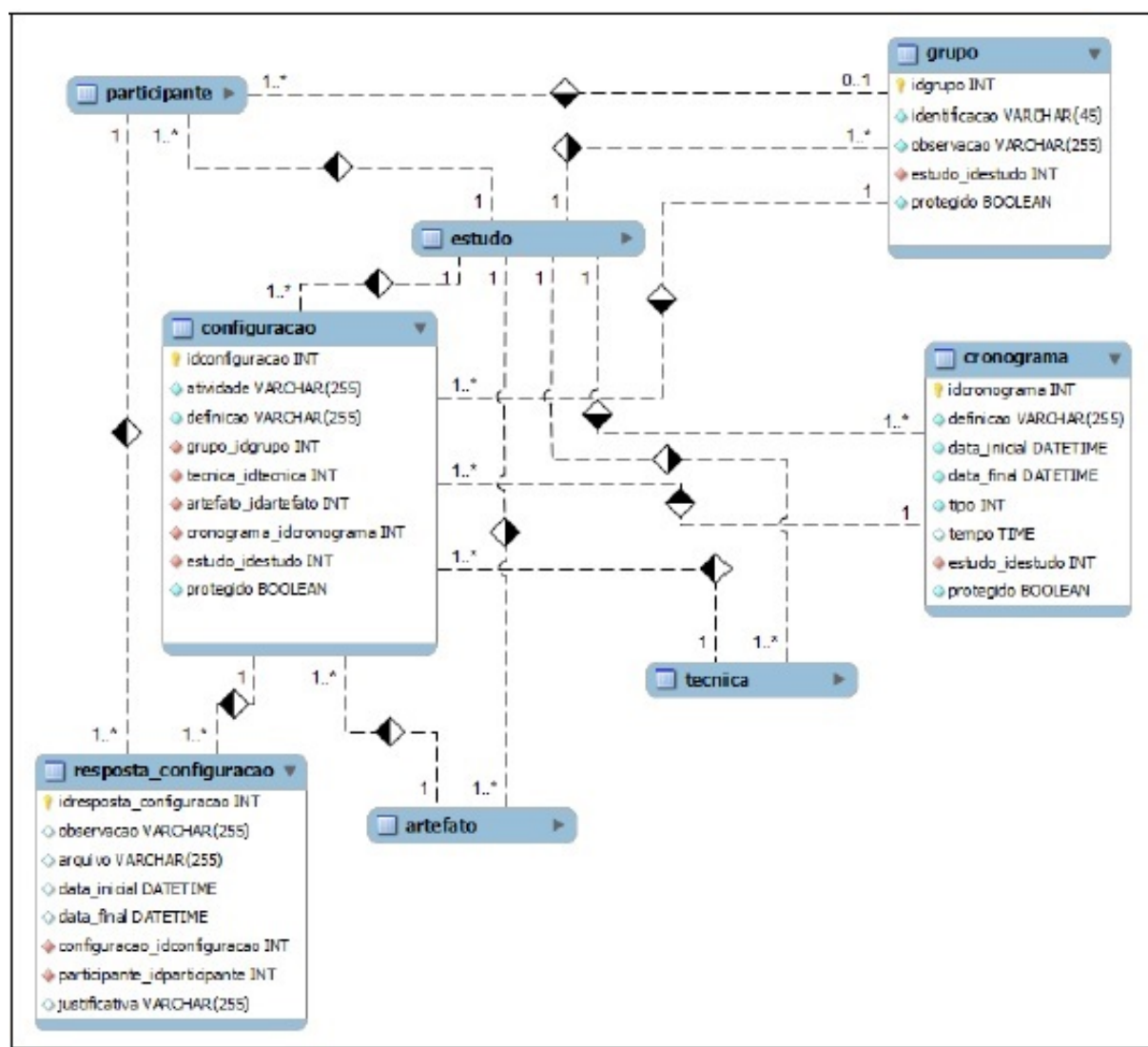


Figura A.14: Esquema da Base de Dados IV

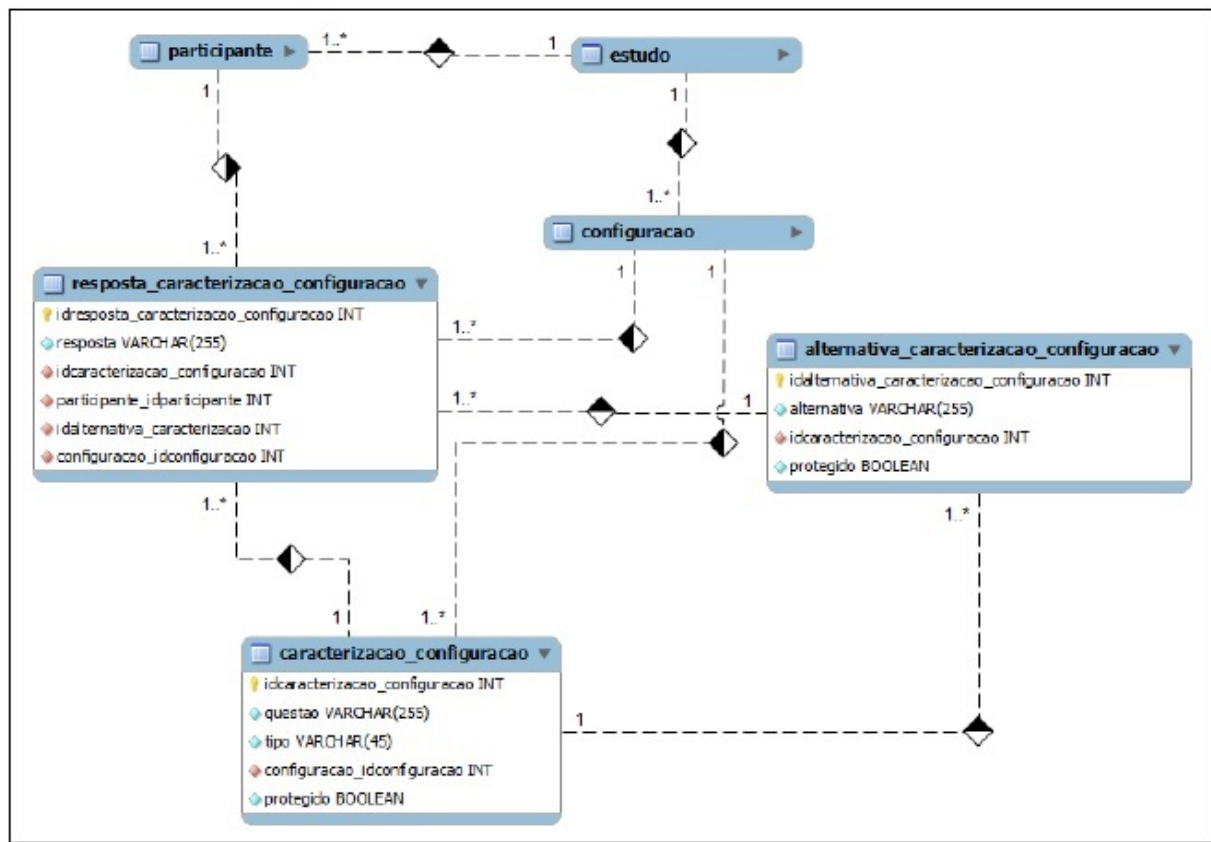


Figura A.15: Esquema da Base de Dados V

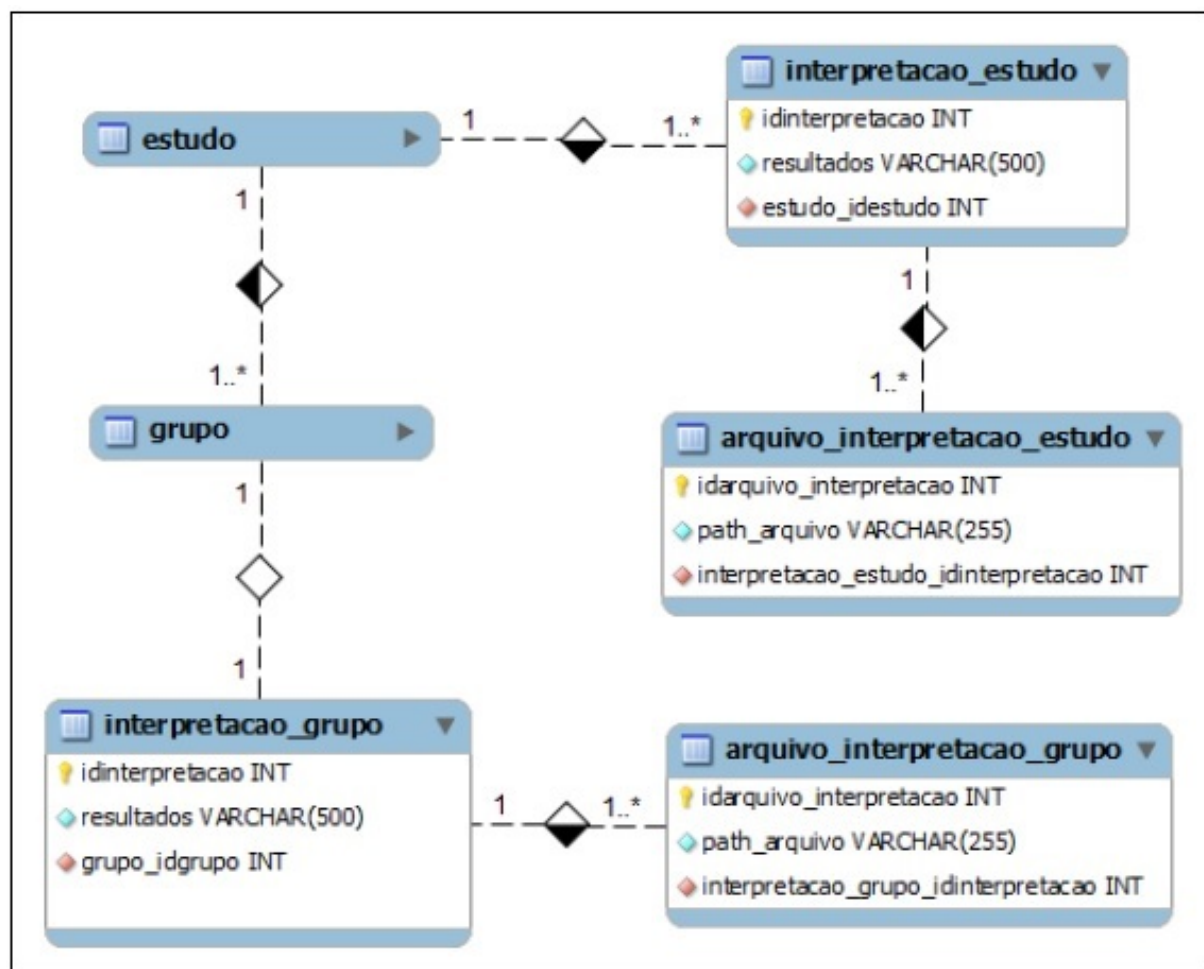


Figura A.16: Esquema da Base de Dados VI

Referências Bibliográficas

DE A. BIOLCHINI, J. C.; MIAN, P. G.; NATALI, A. C. C.; CONTE, T. U.; TRAVASSOS, G. H. Scientific research ontology to support systematic review in software engineering. 2007.

AMARAL, E. A. G. G.; TRAVASSOS, G. H. A package model for software engineering experiments. In: *Proceedings of ISESE 2003 - International Symposium on Empirical Software Engineering*, 2003, p. 21–22.

BASILI, V. R.; SELBY, R. W. Comparing the effectiveness of software testing strategies. *IEEE Transaction 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 Transactions on Software Engineering*, v. 25, n. 4, p. 456–473, 1999.

BORST, W. N. Construction of engineering ontologies for knowledge sharing and reuse. In: *PhD thesis, University of Twente, P.O. Box 217 - 7500 AE Enschede - The Netherlands*, 1997.

BROOKS, A.; ROPER, M.; WOOD, M.; DALY, J.; MILLER, J. *Guide to advanced empirical software engineering*, cap. Replication's Role in Software Engineering Springer-Verlag London, p. 365–379, 2008.

CARVER, J. C. Towards reporting guidelines for experimental replications: A proposal. In: *International Workshop on Replication in Empirical Software Engineering Research (RESER)*, 2010.

D'ARCE, A. F. Avaliação da ferramenta de visualização de software *sofvis_{OA}H* como apoio à depuração de programas: um experimento controlado. In: *Dissertação de Mestrado, Universidade Estadual Paulista*, 2012.

DENTLER, K.; CORNET, R.; TEN TEIJE, A.; DE KEIZER, N. Comparison of reasoners for large ontologies in the owl 2 el profile. *Semantic Web - Interoperability, Usability, Applicability an IOS Press Journal*, 2010.

GARCIA, R. E. *Vidaese: processo de visualização exploratória para apoio a estudos experimentais em verificação, validação e teste de software*. Tese de Doutorado, Instituto de Ciências Matemáticas e de Computação (ICMC-USP), 2006.

GARCIA, R. E.; HÖHN, E. N.; BARBOSA, E. F.; MALDONADO, J. C. An ontology for controlled experiments on software engineering. In: *Proc. (Software Engineering & Knowledge Engineering)*, Knowledge Systems Institute Graduate School, 2008, p. 685–690.

GÜNTHER, H. Pesquisa qualitativa versus pesquisa quantitativa: Esta É a questão? *Psicologia: Teoria e Pesquisa*, v. 22, n. 2, p. 201–210, 2006.

GRUBER, T. R. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies - Special issue: the role of formal ontology in the information technology*, v. 43, p. 907–928, 1995.

JURISTO, N.; MORENO, A. M. *Basics of software engineering experimentation*. Kluwer, 2001.

KITCHENHAM, B. The role of replications in empirical software engineering—a word of warning. *Empirical Software Engineering*, v. 13, n. 2, p. 219–221, 2008.

KITCHENHAM, B. A.; PFLEEGER, S. L.; PICKARD, L. M.; JONES, P. W.; HOAGLIN, D. C.; EMAM, K. E.; ROSENBERG, J. Preliminary guidelines for empirical research in software engineering. *IEEE Transactions on Software Engineering*, v. 28, n. 8, p. 721–734, 2002.

KITCHENHAM, B. A.; TRAVASSOS, G. H.; VON MAYRHAUSER, A.; NIESSINK, F.; SCHNEIDEWIND, N. F.; SINGER, J.; TAKADA, S.; VEHVILAINEN, R.; YANG, H. Towards an ontology of software maintenance. *Journal of Software Maintenance*, v. 11, n. 6, p. 365–389, 1999.

LINKMAN, S.; ROMBACH, H. Experimentation as a vehicle for software technology transfer - a family of software reading techniques, p. 777–780. 1997.

LOPES, V. P. Repositório de conhecimento de um ambiente de apoio à experimentação em engenharia de software. In: *Dissetação de mestrado*, COPPE/UFRJ, 2010.

LOPES, V. P.; TRAVASSOS, G. H. Experimentação em engenharia de software: Glossário de termos. In: *ESELAW'09 - VI Experimental Software Engineering Latin American Workshop*, 2009a.

LOPES, V. P.; TRAVASSOS, G. H. Knowledge repository structure of an experimental software engineering environment. In: *XXIII Brazilian Symposium on Software Engineering*, Brazilian Symposium on Software Engineering, 2009b, p. 32–42.

LOPES, V. P.; TRAVASSOS, G. H. Uma ferramenta de apoio ao planejamento de estudos experimentais em engenharia de software. In: *ESELAW'2010 - VII Experimental Software Engineering Latin American Workshop*, 2010.

MENDONÇA, M. G.; MALDONADO, J. C.; DE OLIVEIRA, M. C. F.; CARVER, J.; FABBRI, S. C. P. F.; SHULL, F.; TRAVASSOS, G. H.; HOHN, E. N.; BASILI, V. R. A framework for software engineering experimental replications. In: *ICECCS*, 2008, p. 203–212.

MORAIS, E. A. M.; AMBROSIO, A. P. L. Ontologias: conceitos, usos, tipos, metodologias, ferramentas e linguagens. In: *Instituto de Informática - Universidade Federal de Goiás*, 2007.

MOREIRA, V. R. Ontologias em sistemas multi-agentes. In: *Universidade Estadual de Campinas - Faculdade de Engenharia Elétrica e de Computação*, 2006.

NOY, N. F.; MCGUINNESS, D. L. Ontology development 101: A guide to creating your first ontology. In: *Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880*, 2001.

PFLEEGER, S. L. Experimental design and analysis in software engineering, p. 219–253. 1995.

RAUTENBERG, S.; TODESCO, J. L.; GAUTHIER, F. A. O. Processo de desenvolvimento de ontologias - uma proposta e uma ferramenta. In: *Revista Tecnologia*, 2009, p. 133–144.

DOS SANTOS, P. S. M.; TRAVASSOS, G. H. esee - ambiente de apoio a experimentação em larga escala em engenharia de software. In: *1st Brazilian e-Science Workshop*, 2007.

SCATALON, L. P.; GARCIA, R. E.; CORREIA, R. C. M. Packaging controlled experiments using an evolutionary approach based on ontology. In:

International Conference on Software Engineering and Knowledge Engineering (SEKE 2011), 2011a, p. 408–413.

SCATALON, L. P.; GARCIA, R. E.; CORREIA, R. C. M. Uma ferramenta para instanciação de ontologia de experimentos controlados em engenharia de software. In: *Conferência Ibérica de Sistemas e Tecnologias de Informação*, 2011b.

SHULL, F.; BASILI, V. R.; CARVER, J.; MALDONADO, J. C.; TRAVASSOS, G. H.; MENDONÇA, M. G.; FABBRI, S. C. P. F. Replicating software engineering experiments: Addressing the tacit knowledge problem. 2002.

SHULL, F.; CARVER, J.; TRAVASSOS, G. H.; MALDONADO, J. C.; CONRADI, R.; BASILI, V. R. Replicated studies: building a body of knowledge about software reading techniques. In: *Lecture notes on empirical software engineering*, River Edge, NJ, USA: World Scientific Publishing Co., Inc., p. 39–84, 2003.

TRAVASSOS, G. H.; GUROV, D.; AMARAL, E. A. G. *Introdução à engenharia de software experimental*. Relatório Técnico, COPPE/UFRJ, 2002.

WOHLIN, C.; RUNESON, P.; HÖST, M.; OHLSSON, M.; REGNELL, B.; WESLÉN, A. *Experimentation in software engineering: An introduction*. Boston, USA: Kluwer Academic Publishers, 2012.