Introdução à Engenharia de Software Experimental

2 authors, including:

Guilherme Horta Travassos
Federal University of Rio de Janeiro
299 PUBLICATIONS **3,358** CITATIONS

SEE PROFILE

RELATÓRIO TÉCNICO

RT–ES–590/02

Introdução à Engenharia de Software Experimental

Guilherme Horta Travassos
ght@cos.ufrj.br

Dmytro Gurov

Edgar Augusto Gurgel do Amaral

Programa de Engenharia de Sistemas e Computação

COPPE / UFRJ

Rio de Janeiro, 2002

Conteúdo

1. INTRODUÇÃO

Experimentação é o centro do processo científico. Somente experimentos verificam as teorias. Somente experimentos podem explorar os fatores críticos e dar luz ao fenômeno novo para que as teorias possam ser formuladas e corrigidas. Experimentação oferece o modo sistemático, disciplinado, computável e controlado para avaliação da atividade humana. Novos métodos, técnicas, linguagens e ferramentas não deveriam ser apenas sugeridos, publicados ou apresentados para venda sem experimentação e validação. Portanto, é preciso avaliar novas invenções e sugestões em comparação com as existentes.

Quando falamos de Engenharia de Software, a discussão concentra-se em se podemos considerá-la ciência ou engenharia. Esta questão refere o duplo caráter do software. Por um lado, a Engenharia de Software considera o processo de criação do produto (software) e desse ponto de vista ela possui as características explicitas de produção ou engenharia. Por outro lado, os aspectos relacionados a *time-to-market* e competição exigem a melhoria contínua e seqüencial da qualidade do processo e do produto. É neste contexto que a parte cientifica de Engenharia de Software se apresenta.

Portanto, as metodologias específicas são necessárias para ajudar a estabelecer uma base de engenharia e de ciência para a Engenharia de Software. Existem quatro métodos relevantes para condução de experimentos na área de Engenharia de Software: científico, de engenharia, experimental, e analítico [Wohlin00].

O método científico observa o mundo, sugere o modelo ou a teoria de comportamento, mede e analisa, verifica as hipóteses do modelo ou da teoria. Isto é um paradigma indutivo. Esse método pode ser utilizado quando se tenta entender, por exemplo, o processo, o produto de software, ambiente. Ele tenta extrair do mundo algum modelo que possa explicar um fenômeno, e avaliar se o modelo é realmente representativo para o fenômeno que está sob observação. Isto é uma abordagem para construção de modelos.

O método de engenharia observa as soluções existentes, sugere as soluções mais adequadas, desenvolve, mede e analisa, e repete até que nenhuma melhoria adicional seja possível. Isto é uma abordagem orientada à melhoria evolutiva que assume a existência de algum modelo do processo ou produto de software e modifica este modelo com propósito de melhorar os objetos do estudo.

O método experimental sugere o modelo, desenvolve o método qualitativo e/ou quantitativo, aplica um experimento, mede e analisa, avalia o modelo e repete o processo. Isto é uma abordagem orientada à melhoria revolucionária. O processo se inicia com o levantamento de um modelo novo,

não necessariamente baseado em um modelo já existente, e tenta estudar o efeito do processo ou produto sugerido pelo modelo novo.

O método analítico (ou matemático) sugere uma teoria formal, desenvolve a teoria, deriva os resultados e se possível, compara-a com as observações empíricas. Isto é um método dedutivo que não precisa de um projeto experimental no sentido estatístico, mas oferece uma base analítica para o desenvolvimento de modelos.

Supõe-se que a abordagem mais apropriada para a experimentação na área de Engenharia de Software seja o método experimental que considera a proposição e avaliação do modelo com os estudos experimentais. Entretanto, existe a possibilidade da utilização de outros métodos. Às vezes é necessário aplicá-los para a resolução dos problemas de Engenharia de Software. Por exemplo, o método científico pode ser utilizado para compreender a maneira como o software está sendo construído por uma organização para ver se a ferramenta pode ser utilizada para automatizar o processo; o método de engenharia ajuda a demonstrar que uma ferramenta possui um desempenho melhor do que outra; o método analítico pode provar modelos matemáticos para conceitos, como o crescimento da confiabilidade, a complexidade do software, o projeto ou código propenso a erro.

É importante notar que experimentos não provam nada. É verdade que nenhum experimento oferece prova com certeza absoluta. Os experimentos verificam a previsão teórica de encontro à realidade. A comunidade aceita uma teoria se todos os fatos conhecidos dentro de seu domínio possam ser deduzidos da teoria, possui uma verificação experimental extensa e prediz o novo fenômeno corretamente.

Uma informação mais detalhada a respeito do papel da experimentação na área de Engenharia de Software pode ser encontrada em [Basili96].

Com a finalidade de exemplificar conceitos básicos relacionados à experimentação este relatório técnico apresenta a definição e a análise de um estudo que foi conduzido durante o primeiro curso de Engenharia de Software Experimental no terceiro bloco aulas de 2001 na COPPE / UFRJ, Programa de Engenharia de Sistemas e Computação.

O objetivo principal do estudo é definir se o curso básico de Engenharia de Software oferece as competências necessárias ao desenvolvimento de software pessoal do ponto de vista dos seus alunos. Os participantes escolhidos para o estudo foram os alunos da pós-graduação do curso de Engenharia de Software. Os questionários utilizados para a coleta de dados são baseados nas competências do currículo mínimo da SBC (http://www.sbc.org.br).

2. OBJETIVOS DA EXPERIMENTAÇÃO

Os objetivos relacionados à execução de experimentos em Engenharia de Software são a caracterização, avaliação, previsão, controle e melhoria a respeito de produtos, processos, recursos, modelos, teorias entre outros. A importância e o esforço aumentam de um experimento com o objetivo "caracterização" a um experimento com o objetivo "melhoria". Isso significa que é bastante simples conduzir um experimento com a finalidade de caracterização respondendo questões do tipo "o que está acontecendo?". É mais difícil medir algo, por exemplo, um processo ou produto e defini-lo "quão bom é isto?". Os experimentos com a finalidade de previsão além da medição precisam de meios de estimativa para mostrar a possibilidade de:"posso estimar algo no futuro?". Para atender a finalidade de controle deve existir a possibilidade de gerenciar os atributos de um processo ou produto e dar a resposta a "posso manipular o evento?". Finalmente, a finalidade da melhoria supõe que possamos caracterizar, avaliar, predizer e controlar, e há os objetivos da melhoria de um processo ou produto que possam ser atingidos respondendo a última questão "posso melhorar o evento?".

Um exemplo da definição dos objetivos de um estudo experimental se encontra no Anexo 1 (1.1 – 1.3).

A literatura oferece algumas outras definições dos objetivos da experimentação em Engenharia de Software. Em [Conradi01] são listados:

- 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.
- A observação e experimentação podem levar a novos e úteis meios da introspecção, e abrir novas áreas de investigação. A experimentação pode encontrar novas áreas onde a engenharia age lentamente.
- A experimentação pode acelerar o processo eliminando abordagens inúteis e suposições errôneas. A experimentação ajuda também a orientar a engenharia e a teoria nas direções promissoras de pesquisa.
- Os experimentos podem ser custosos, mas um experimento significativo geralmente pode se encaixar no orçamento de um pequeno laboratório. Por outro lado, um experimento caro

pode valer a pena muito mais do que seu custo e, por exemplo, oferecer à companhia liderança de três, quatro ou cinco anos sobre a competição.

- O crescimento do número de trabalhos científicos com uma validação empírica significativa possui a boa chance de acelerar o processo de formação da Engenharia de Software como ciência. As idéias duvidosas serão rejeitadas mais rapidamente e os pesquisadores poderão concentrar-se nas abordagens promissoras.
- A tecnologia vem se modificando rapidamente. As mudanças sempre trazem ou eliminam as suposições. Os pesquisadores devem então antecipar as mudanças nas suposições e aplicar os experimentos para explorar as conseqüências dessas mudanças.

3. DESCRIÇÃO GERAL DA EXPERIMENTAÇÃO

3.1 Vocabulário da experimentação

Os elementos principais do experimento são as variáveis, os objetos, os participantes, o contexto do experimento, hipóteses, e o tipo de projeto do experimento.

Há dois tipos de variáveis do experimento: dependentes e independentes. As variáveis independentes referem-se à entrada do processo de experimentação. Essas variáveis também se chamam "fatores" e apresentam a causa que afeta o resultado do processo de experimentação. O próprio valor de um fator se chama "tratamento". As variáveis dependentes referem-se à saída do processo de experimentação. Essas variáveis apresentam o efeito que é causado pelos fatores do experimento. O próprio valor de uma variável dependente se chama "resultado".

A Figura 1 apresenta os relacionamentos entre os conceitos descritos acima.

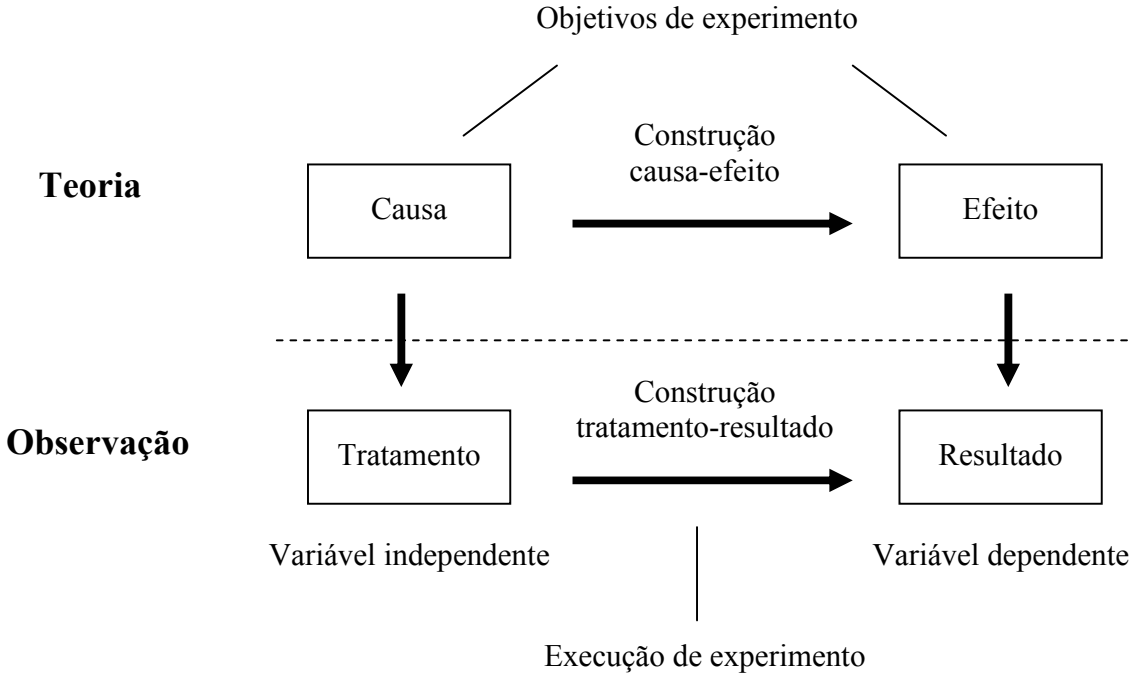


Figura 1. Os conceitos de um experimento [Wohlin00]

O anexo 1 (2.5) contém um exemplo da descrição das variáveis.

O objeto é uma ferramenta usada para verificar o relacionamento causa-efeito numa teoria. Durante a execução do experimento os tratamentos estão sendo aplicados ao conjunto dos objetos e assim o resultado está sendo avaliado.

Os objetos junto com o sistema de medição e diretrizes da execução do experimento compõem a instrumentação do experimento. O anexo 1 (2.2 e 3.1-3.2) apresenta um exemplo de instrumentação.

Os participantes são os indivíduos que foram especialmente selecionados da população sob interesse para conduzir o experimento. Isso significa que para generalizar os resultados de um experimento a uma população desejada, o conjunto de participantes deve ser representativo para aquela população. Para atingir a meta mencionada os parâmetros como o modo de seleção dos participantes e o tamanho do conjunto selecionado que influem o resultado do experimento, devem ser considerados. A princípio, quanto maior é a variedade da população tanto maior deve ser o tamanho do conjunto de participantes.

O anexo 1 (2.4) contém um exemplo da descrição da seleção dos participantes.

O contexto do experimento é composto das condições em que o experimento está sendo executado. O contexto pode ser caracterizado de acordo às três dimensões:

In-vitro vs. *In-vivo*. .....O primeiro refere-se à experimentação no laboratório sob as condições controladas. O segundo considera o estudo de um projeto real.

Alunos vs. Profissionais. .....Define a equipe que vai executar o experimento.

Problema de sala de aula vs.

Problema real .....Mostra o tamanho do problema que está sendo estudado.

Específico vs. Geral. .....Mostra se os resultados do experimento são válidos para um contexto particular ou para o domínio da Engenharia de Software inteiro.

O anexo 1 (2.3) contém um exemplo da descrição do contexto do experimento.

Um experimento geralmente é formulado através de hipóteses. A hipótese principal se chama hipótese nula e declara que não há nenhum relacionamento estatisticamente significante entre a causa e o efeito. O objetivo principal do experimento é, então, rejeitar a hipótese nula a favor de uma ou algumas hipóteses alternativas. A decisão sobre rejeição da hipótese nula pode ser tomada baseado nos resultados da sua verificação utilizando um teste estatístico.

A verificação das hipóteses sempre lida com algum tipo de risco que implica que um erro pode acontecer. O erro do primeiro tipo (*type-I-error*) acontece quando o teste estatístico indica o relacionamento mesmo que não exista nenhum relacionamento real. A probabilidade do erro desse tipo pode ser avaliada como:

$$P(\text{type-I-error}) = P(H0 \text{ é rejeitada} \mid H0 \text{ é verdadeira})$$

O erro do segundo tipo (*type-II-error*) acontece quando o teste estatístico não indica o relacionamento mesmo que efetivamente exista um relacionamento. A probabilidade de erro desse tipo pode ser avaliada como:

P (type-II-error) = P (H0 não é rejeitada |H0 é falsa)

O tamanho do erro durante a verificação das hipóteses depende da potência do teste estatístico. A potência do teste implica a probabilidade de que o teste vai encontrar o relacionamento mesmo que a hipótese nula seja falsa. A potência pode ser avaliada como:

Potência = P (H0 é rejeitada |H0 é falsa) = 1 - P (type-II-error).

O teste estatístico com a maior potência deve ser escolhido. Informação mais detalhada a respeito de testes e potência dos testes pode ser encontrada em [Miller97].

O anexo 1 (2.1) apresenta a descrição da hipótese nula e das hipóteses alternativas do estudo.

O projeto do experimento determina a maneira como um experimento será conduzido. A decisão sobre alocação dos objetos e dos participantes é feita nesse momento. Também a maneira como os tratamentos serão aplicados aos objetos é definida.

A combinação dos objetos, participantes e tratamentos se chama teste experimental ou "*trial*". A quantidade e a seqüência dos testes experimentais definem o projeto do experimento. A informação mais detalhada a respeito do projeto do experimento pode ser encontrada em [Juristo98].

3.2 Princípios da organização do experimento

A experimentação na Engenharia de Software quanto a experimentação em geral, presume um conjunto dos princípios que devem ser considerados ao longo do processo de organização e execução do experimento tal como no tempo da análise e interpretação dos resultados (veja seção 5.2).

Os princípios gerais da organização do experimento são a aleatoriedade, o agrupamento (*blocking)*, e o balanceamento.

Todos os métodos estatísticos utilizados para a análise empírica requerem que a observação seja feita a partir das variáveis independentes e aleatórias. Para atingir essa exigência usa-se aleatoriedade. A aleatoriedade implica que a alocação dos objetos e dos participantes e a ordem da execução dos testes experimentais sejam aleatórias. A aleatoriedade é utilizada para evitar o efeito de algum fator que de outra maneira possa estar presente, e também para selecionar os participantes que sejam representativos para a população do interesse.

Se houver um fator no projeto do experimento que provavelmente tenha um efeito sobre o resultado, mas esse efeito não é interessante para os pesquisadores, o agrupamento deve ser utilizado.

O agrupamento sistematicamente elimina o efeito indesejado durante a comparação dos tratamentos. Dentro do bloco o efeito indesejado é indiferente e pode ser eliminado da consideração. O agrupamento aumenta a precisão do experimento.

Se o experimento está organizado de uma forma que todos os tratamentos têm o número dos participantes igual, o projeto do experimento é balanceado. O balanceamento é desejável porque isso simplifica e melhora a análise estatística dos dados experimentais.

Os princípios descritos na maior parte se referem ao projeto do experimento. Outro princípio importante da experimentação que se refere à potência dos resultados do experimento é a repetição do experimento. Um experimento deve ser adequadamente empacotado para permitir a outros investigadores reproduzir os resultados. A repetição é importante porque implica que as variáveis imprevistas não estão afetando os resultados, e assegura que não há nenhuma confusão entre dois efeitos. Os resultados dos experimentos não podem ser amplamente aceitados sem que a repetição interna ou externa seja aplicada. A informação mais detalhada a respeito da essência da repetição dos experimentos pode ser encontrada em [Brooks96].

Além disso, há alguns princípios que devem ser considerados a respeito das características particulares da experimentação na área de Engenharia de Software.

O controle local se refere à habilidade de modificação do tratamento aplicado a cada objeto. Por exemplo, o estudo de caso geralmente tem o controle fraco sobre os tratamentos. O controle local é o maior problema da experimentação na ciência de computação desde que a maioria dos tratamentos incorre em despesas de custo ou de tempo significativas.

Também é importante considerar o impacto que o dado projeto do experimento tem sobre os resultados do experimento. O método de investigação pode ser passivo, ou seja, o objeto está sendo estudado sem qualquer efeito sobre o objeto em si, ou ativo, ou seja, a interação com o objeto afeta o seu comportamento.

3.3 Medição

A medição é a parte central de um estudo experimental. A medição é definida como o mapeamento do mundo experimental para o mundo formal ou relacional. O objetivo principal desse mapeamento é caracterizar e manipular os atributos das entidades empíricas da maneira formal. Ao invés de fazer o julgamento diretamente a partir das entidades reais, os números ou símbolos estão atribuídos a essas entidades, e o julgamento é feito a partir desses números e símbolos. O número ou

símbolo atribuído à entidade por meio do mapeamento se chama medida. O atributo da entidade que está sendo medida se chama métrica.

O anexo 1 (1.4 e 2.2) apresenta um exemplo das métricas usadas durante a coleta dos dados experimentais.

O mapeamento pode ser feito de várias maneiras. Os mapeamentos diferentes do mesmo atributo se chamam escalas. A transformação de uma escala da medição para outra se chama *rescaling* ou transformação admissível. Se a afirmação é verdadeira mesmo após uma medida ser transformada em escala diferente, essa afirmação é chamada significativa, no caso contrário é chamada sem significado.

Existem quatro tipos de escalas: nominal, ordinal, intervalo, e razão. A escala nominal apresenta o atributo de uma entidade como o nome ou símbolo. A classificação das entidades pode ser feita a partir dos atributos nominalmente mapeados. A escala ordinal ordena as entidades segundo um critério definido. Nesse caso, as afirmações como "maior do que …" ou "mais complexo do que …" podem ser feitas. A escala do intervalo ordena os valores da mesma forma que a escala ordinal, mas acrescenta a noção da distância relativa entre as entidades. Na escala razão existe o valor do zero significativo e a razão entre medidas é significativa. A possibilidade de produzir as afirmações significativa, chamada também "a potência da escala" cresce da escala nominal à escala razão.

Há duas dimensões das medidas: objetiva e subjetiva, direta e indireta. Se o valor da medida depende somente do objeto em si a medida é objetiva. A medida objetiva pode ser tomada várias vezes e o mesmo valor será sempre recebido. Se o valor da medida depende ao mesmo tempo do objeto e da perspectiva na qual o valor foi tomado, a medida é subjetiva. A medida subjetiva pode tomar os valores diferentes se medir o objeto várias vezes. A medida direta é aquela que não envolva a medição de outros atributos. A medida indireta é derivada das medidas de outros atributos.

Além disso, a medição na Engenharia de Software é caracterizada em termos dos atributos internos e externos. Um atributo interno pode ser medido em termos do objeto em si. Esses atributos geralmente são das medidas diretas. Um atributo externo pode ser medido somente a respeito dos atributos de outros objetos. Os atributos externos são das medidas indiretas e devem ser derivados dos atributos internos.

A medição na Engenharia de Software na maioria dos casos utiliza as próprias métricas do software. As métricas do produto são usadas para medir o produto intermediário ou final. As métricas do processo medem as características do processo do desenvolvimento de software. As métricas do recurso são usadas para medir os objetos tais como equipe, hardware, ferramentas, etc. necessários ao desenvolvimento de software.

O objetivo principal da medição na Engenharia de Software é aumentar a compreensão do processo e do produto, controlá-los definindo antecipadamente as atividades corretivas e identificar as possíveis áreas de melhoria.

A medição é a base da abordagem *bottom-up* para melhoria do processo do desenvolvimento de software. Essa abordagem implica a análise detalhada das práticas de software, a seleção dos objetivos de melhoria derivados dessa análise, e a gerência das atividades de melhoria sustentadas pela medição. Um exemplo da abordagem descrita é o Paradigma da Melhoria da Qualidade (Quality Improvement Paradigm - QIP). QIP apresenta um *framework* do processo de melhoria sustentado pelos princípios do paradigma Goal/Question/Metric (GQM). A informação mais detalhada a respeito da aplicação da medição na melhoria do processo do desenvolvimento de software dirigida pelos objetivos pode ser encontrada em [Solingen99]. A informação geral a respeito da aplicação da medição na área de Engenharia de Software pode ser encontrada em [SEL95].

3.4 Validade

A questão fundamental a respeito dos resultados do experimento é quão validos são eles. Os resultados devem ser válidos para a população da qual o conjunto de participantes foi recebido. É interessante também generalizar os resultados para uma população mais ampla. Os resultados possuem a validade adequada se são válidos para a população a qual tendem ser generalizados.

Há quatro tipos da validade dos resultados do experimento: a validade de conclusão, a validade interna, a validade de construção, e a validade externa.

A validade de conclusão é relacionada à habilidade de chegar a uma conclusão correta a respeito dos relacionamentos entre o tratamento e o resultado do experimento. Durante a avaliação da validade de conclusão é necessário considerar os conceitos como a escolha do teste estatístico, a escolha do tamanho do conjunto dos participantes, a confiabilidade das medidas, e a confiabilidade da implementação dos tratamentos.

Alguns problemas podem acontecer quando se lida com a validade da conclusão. Se a potência do teste estatístico é baixa, existe o risco alto que conclusões errôneas sejam tiradas. Alguns testes têm condições especificas, por exemplo, a respeito da distribuição normal do conjunto dos participantes. Os pesquisadores podem influenciar os resultados tentando receber o resultado específico. As medidas podem envolver o julgamento humano e, assim, os resultados diferentes podem ser recebidos caso um objeto seja medido várias vezes.

A validade interna define se o relacionamento observado entre o tratamento e o resultado é causal, e não é o resultado da influência de outro fator que não é controlado ou mesmo não foi medido. Durante a avaliação da validade interna uma maior atenção deve ser prestada aos participantes, ou seja, à seleção da população, à maneira da divisão nas classes, ao modo da aplicação dos tratamentos, e aos aspectos sociais.

Os problemas ou riscos relacionados a esse tipo de validade são ligados aos participantes. Por exemplo, os participantes podem ficar cansados ou desanimados ou, ao contrário, podem aprender ao longo do estudo. A seleção dos voluntários pode formar o conjunto dos participantes que não são representativos para a população. Uma instrumentação inadequadamente preparada pode ocasionar um mal-entendido e afetar os resultados negativamente. Os grupos dos participantes podem produzir os resultados diferentes por causa do comportamento e as habilidades diferentes. Os participantes que receberam o tratamento menos interessante ou menos desejado, podem ser motivados a reduzir ou inverter os resultados do experimento. Além disso, aqueles que receberam o tratamento menos interessante, podem deixam de terminar o estudo ou executar o estudo com o desempenho pior do que sempre.

A validade de construção considera os relacionamentos entre a teoria e a observação, ou seja, se o tratamento reflete a causa bem e o resultado reflete o efeito bem. Durante a avaliação da validade da construção os aspectos relevantes ao projeto do experimento e os fatores humanos devem ser considerados. Os problemas surgem por causa do comportamento incorreto do lado dos participantes ou do experimentador. O experimento pode ser sub-representado e não oferecer a imagem completa da teoria. Os participantes possam ser envolvidos em vários estudos e, portanto, o efeito seja o resultado da combinação dos tratamentos. Os participantes possam basear o seu comportamento nas suposições sobre as hipóteses. O ser humano sempre está tentando parecer melhor quando está sendo avaliado. Os pesquisadores podem afetar os resultados (viés) projetando o estudo baseado naquilo que eles esperam do experimento.

A validade externa define as condições que limitam a habilidade de generalizar os resultados de um experimento para a prática industrial. Durante a avaliação da validade externa a interação do tratamento com as pessoas, o lugar e o tempo devem ser considerados. Os problemas podem acontecer devido à população dos participantes não ser representativa à população sob interesse, a instrumentação não ser adequada à prática industrial, e o experimento pode ser executado num dia ou tempo especial que venha afetar os resultados.

A prioridade dos tipos da validade é determinada segundo os objetivos da experimentação. Para verificação de uma teoria a ordem da importância dos tipos da validade é: interna, construção,

conclusão, e externa. Para os experimentos aplicados, que são o alvo da maioria dos experimentos na área de Engenharia de Software, a ordem da importância dos tipos da validade é: interna, externa, construção, e conclusão.

O anexo 1 (2.7) apresenta os exemplos de todos os tipos da validade. Informação adicional a respeito da validade pode ser encontrada em [Wohlin00].

4. TIPOS DE EXPERIMENTOS

Os diferentes tipos de classificação dos experimentos foram descritos por várias fontes. O grande número de classificações existe, provavelmente, porque a experimentação ainda é uma abordagem nova na área de Engenharia de Software. O tipo de experimento mais apropriado em uma situação concreta vai depender, por exemplo, dos objetivos do estudo, das propriedades do processo de software usado durante a experimentação, ou dos resultados finais esperados.

Ao mesmo tempo, a classificação dos experimentos está sempre relacionada aos conceitos das estratégias empíricas. Em síntese, a literatura descreve três principais estratégias experimentais:

- *Survey*;
- Estudo de caso;
- Experimento.

As características principais usadas para diferenciar essas estratégias são o controle de execução, o controle de medição, o custo de investigação e a facilidade de repetição. Além disso, o risco está sendo considerado como uma característica importante. É essencial escolher uma estratégia, que seja mais conveniente, baseado em custo e risco, e na maioria dos casos é melhor começar em uma escala menor acrescentando-a à medida que o conhecimento aumente e o risco diminua.

O *Survey* é uma investigação executada em retrospectiva. O *Survey* é conduzido quando algumas técnicas ou ferramentas já tenham sido utilizadas. Os objetivos do *survey* são:

- descritivo, por exemplo, determinar a distribuição de atributos ou características;
- explanatório, por exemplo, explicar porque os desenvolvedores escolheram uma das técnicas;
- explorativo, por exemplo, um estudo preliminar para uma investigação mais profunda.

Os meios principais para coletar a informação quantitativa e qualitativa preliminar são os questionários. O *Survey* possui habilidade de levantar o grande número de variáveis a serem avaliadas. Entretanto, é necessário visar-se na recuperação de um volume maior da compreensão de um número menor das variáveis porque isso vai simplificar o trabalho analítico. O survey não oferece nenhum controle sobre a execução ou medição e é sempre impossível manipular as variáveis.

Estudo de caso é utilizado para monitorar os projetos, atividades e atribuições. Os estudos de caso visam observar um atributo específico e estabelecer o relacionamento entre atributos diferentes. Há três maneiras de arranjar o estudo de caso:

- a comparação dos resultados de usar os métodos novos de encontro a *baseline* da companhia que está expressa em valores médios dos projetos da companhia;
- Projetos semelhantes (*sister projects*), que implicam a comparação dos resultados de projetos similares um dos quais usa uma tecnologia nova e o outro usa uma tecnologia atual;
- aplicação de um método aleatoriamente atribuído a um projeto da companhia e não atribuído aos outros.

O nível de controle num estudo de caso é baixo, mas ao contrário de *survey*, o estudo de caso possui o controle sobre a medição das variáveis. O maior problema do estudo de caso é a possibilidade de fatores de confusão (*confounding factors*), ou seja, é difícil diferenciar o efeito proveniente de um fator do efeito proveniente de outro fator.

O experimento geralmente é realizado em laboratório e oferece o maior nível de controle. O objetivo do experimento é manipular uma ou algumas variáveis e manter as outras fixas medindo o efeito do resultado. Os experimentos podem ser feitos *in-vitro* sob condições de laboratório ou *in-vivo* sob condições normais (veja Seção 3.1). Os experimentos são apropriados para confirmar as teorias, confirmar o conhecimento convencional, explorar os relacionamentos, avaliar a predição dos modelos, ou validar as medidas. A maior força do experimento encontra-se no controle total sobre o processo e as variáveis e na possibilidade de ser repetido.

A comparação das estratégias experimentais está na Tabela 1.

Tabela 1. A comparação das estratégias empíricas.

Fator	*Survey*	Estudo de caso	Experimento
O controle da execução	Nenhum	Nenhum	Tem
O controle da medição	Nenhum	Tem	Tem
O controle da investigação	Baixo	Médio	Alto
Facilidade da repetição	Alta	Baixa	Alta
Custo	Baixo	Médio	Alto

De acordo com as estratégias experimentais existem três principais métodos para coleta de dados:

- histórico;

- de observação;
- controlado.

O método histórico é utilizado para coletar os dados experimentais dos projetos que já tenham sido terminados. Os dados já existem e é preciso analisá-los. O método de observação coleta os dados relevantes enquanto o projeto está sendo executado. O método oferece o controle fraco sobre o processo de desenvolvimento. O método controlado provê as instâncias múltiplas de uma observação oferecendo a validade estatística dos resultados do estudo.

Na literatura estes métodos estão ainda divididos em subtipos com a finalidade de uma maior especialização e precisão.

Por exemplo, existe a seguinte classificação dos métodos [Zelkowitz98]:

- O método histórico: pesquisa bibliográfica, lições aprendidas, análise estática;
- O método de observação: o monitoramento do projeto, estudo de caso, afirmação, estudo de campo;
- O método controlado: repetição, sintético, análise dinâmica, simulação.

Outra classificação considera as características do contexto do experimento, ou seja, diferenciação dos tipos de experimentos dependendo da quantidade dos objetos e participantes envolvidos no estudo. Nesse caso, a classificação apresenta uma matriz 2x2 (Tabela 2).

Tabela 2. A classificação de estudos experimentais dependendo da quantidade dos objetos e participantes [Wohlin00].

		Número de objetos	
		1	2 ou mais
Número de participantes	1	Estudo de único objeto	Estudo da variação de objetos
	2 ou mais	Estudo de objeto com vários testes	Estudo agrupado por objetos e participantes

A classificação pode também considerar como os dados experimentais foram medidos. Há nove tipos de estudos agrupados em três categorias gerais:

- O estudo qualitativo;
- O estudo quantitativo;
- *Benchmarking*.

O estudo qualitativo está relacionado à pesquisa sobre os objetos quando os resultados são apresentados em termos naturais. Informações mais detalhadas e alguns exemplos sobre estudos qualitativos na área de Engenharia de Software podem ser encontrados em [Seaman99]. O estudo quantitativo geralmente é conduzido através um experimento controlado. Uma das vantagens do estudo controlado é que os dados qualitativos promovem a comparação e a análise estatística. *Benchmarking* é utilizado para a medição do desempenho dos diferentes produtos de software.

5. PROCESSO DE EXPERIMENTAÇÃO

5.1 Metodologia

Um experimento deve ser tratado como um processo da formulação ou verificação de uma teoria. A fim de que o processo ofereça os resultados válidos, ele deve ser propriamente organizado e controlado ou, pelo menos, acompanhado. Com o propósito de atingir estes objetivos várias metodologias de organização dos experimentos foram elaboradas. Para serem comparadas as metodologias da experimentação possuem diferentes características como, por exemplo, as fases do processo de experimentação, a maneira da transformação dos conceitos abstratos do domínio às métricas concretas, o objetivo principal da experimentação, as ferramentas do empacotamento, etc.

Um bom exemplo da metodologia da experimentação avançada é o Paradigma da Melhoria da Qualidade (Quality Improvement Paradigm - QIP) [Basili94]. A essência dessa metodologia está na melhoria contínua do processo do desenvolvimento de software.

A metodologia define os seis passos que terminam resultando em um ciclo da melhoria do processo completo. O ciclo se inicia com a caracterização do processo de negócio necessária para a compreensão do ambiente e a definição dos objetivos básicos. A seguir os objetivos quantitativos são estabelecidos com a propósito de demonstrar as expectativas razoáveis da experimentação. Baseado na caracterização e nos objetivos definidos o processo da melhoria apropriado é escolhido tomando em consideração a consistência entre os objetivos. O processo do desenvolvimento de software oferece, além do próprio software, o *feedback* do projeto que representa a informação recolhida. Essa informação serve como a base para a análise, ou seja, a avaliação das práticas atuais, a determinação dos problemas, a proposição da melhoria futura. Finalmente, toda informação relevante está empacotada para utilização futura.

O QIP é ligado ao conceito da Fábrica da Experiência [Basili94] que é o conjunto das ferramentas para o armazenamento, a modificação, e a retirada da informação do projeto empacotada. Isso significa que além do armazenamento passivo dos dados experimentais, a Fábrica de Experiência pode processar os pedidos do projeto atual oferecendo a informação relevante dos projetos semelhantes. A Figura 2 apresenta a estrutura da Fábrica da Experiência.

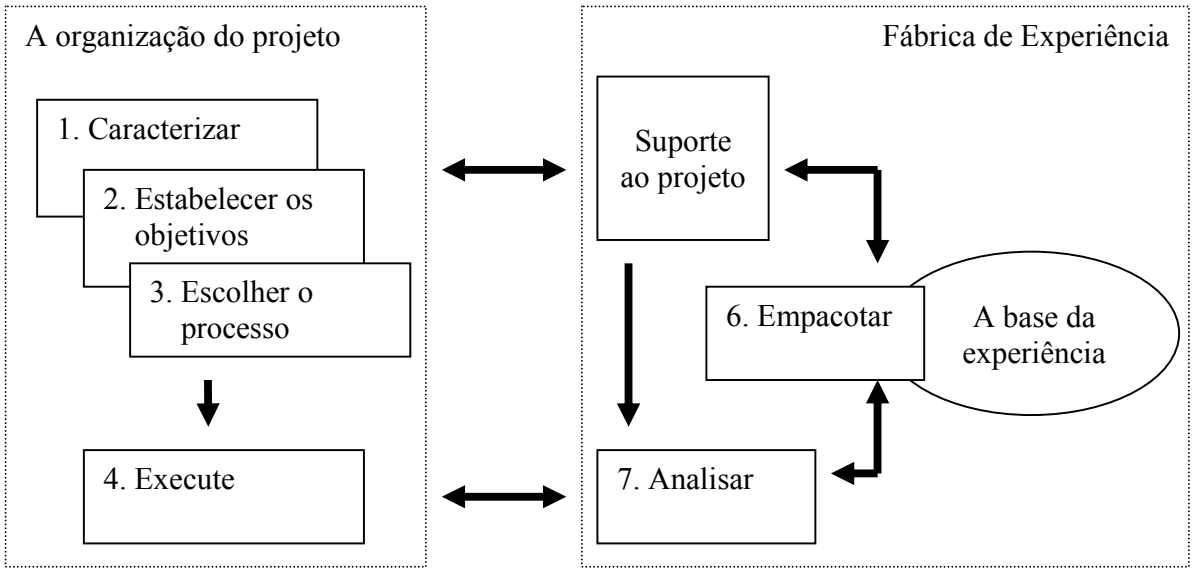


Figura 2. Fábrica de Experiência [Wohlin00].

Um outro instrumento ligado a QIP é a abordagem Goal/Question/Metric (GQM) [Solingen99]. A abordagem oferece o processo da melhoria com o modelo da medição baseado em camadas. A definição e a interpretação do processo da experimentação são divididas em camadas conceitual, operacional, e quantitativa. A definição usa a abordagem *top-down*: o estabelecimento dos objetivos, a formulação das questões, e elaboração das métricas. A interpretação usa a abordagem *bottom-up*: a medição para receber os dados experimentais, a formulação das respostas para as questões baseada nos dados experimentais, e o agrupamento das respostas para demonstrar o grau do de sucesso dos objetivos estabelecidos.

A Figura 3 apresenta o processo principal da abordagem GQM.

Os objetivos principais definidos por GQM são compreender, controlar, e melhorar. Esses objetivos são focados nos quatro fatores: o custo, o risco, o tempo, e a qualidade. Juntando os objetivos e os fatores os investigadores recebem o aumento da compreensão do produto e do processo de software, o produto e o processo se tornam controlados, e, finalmente, as atividades de melhoria do produto e do processo de software estão sendo definidas.

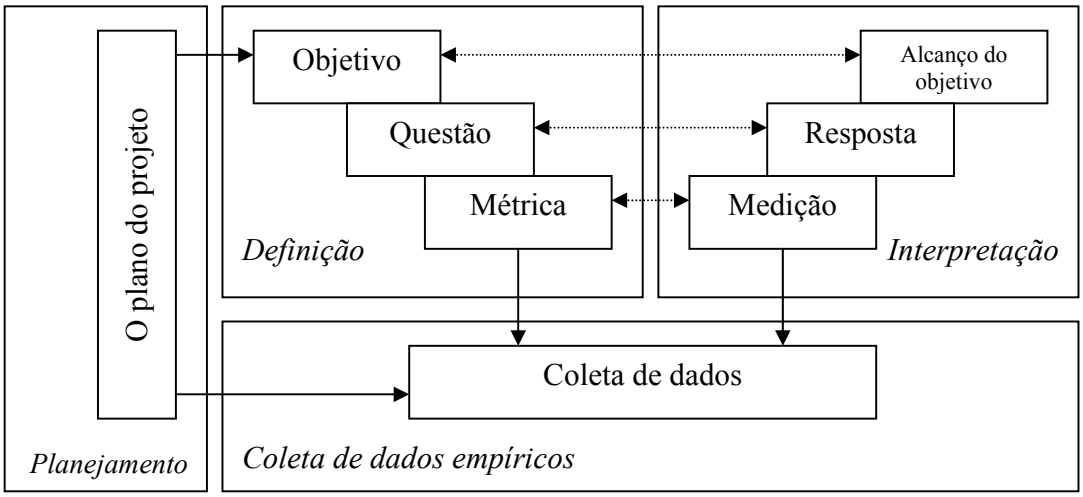


Figura 3. O processo principal da abordagem GQM [Solingen99]

A abordagem GQM é composta de quatro fases:

- A fase do planejamento, quando o projeto da medição está selecionado, definido, caracterizado, e planejado, resultando em o plano do projeto;
- A fase da definição, quando o programa da medição conceitualmente preparado, ou seja, os objetivos, as questões, as métricas e as hipóteses são estabelecidos;
- A fase da coleta de dados, quando a coleta de dados experimentais é efetivamente feita resultando em conjunto de dados prontos para a interpretação;
- A fase da interpretação, quando os dados são processados a respeito das métricas, questões, e objetivos definidos.

Uma parte essencial da abordagem GQM é a análise "custo-benefício" que está utilizada para avaliar se o benefício estimado supera o custo total.

O anexo 1 (1.3 e 1.4) apresenta um exemplo da definição de um estudo realizada de acordo com os princípios da abordagem GQM.

5.2 Fases do experimento

O processo da execução de um experimento presume a realização de diferentes atividades. O número e a complexidade dessas atividades podem variar de acordo com as características do estudo. A literatura apresenta cinco fases gerais que sempre estão presentes num processo da experimentação.

A definição é a primeira fase onde o experimento é expresso em termos dos problemas e objetivos. A fase de planejamento vem em seguida onde o projeto do experimento é determinado, a

instrumentação é considerada e os aspectos da validade do experimento são avaliados. A execução do experimento segue o planejamento. Nesse momento os dados experimentais são coletados para serem analisados e avaliados na fase de análise e interpretação. Finalmente, os resultados são apresentados e empacotados durante a fase da apresentação e empacotamento.A negligência de qualquer dessas fases acarreta em resultados errôneos e precisa de modificações no trabalho que já tinha sido feito, o que é difícil ou, às vezes, impossível de realizar. Por exemplo, é possível fazer as modificações nos objetivos e no plano do experimento somente antes da execução do experimento. Senão, o mesmo experimento pode ser conduzido só com os participantes novos. De qualquer maneira, as modificações trazem a perda do tempo, de esforços, e de recursos.

A fase de definição descreve os objetivos, o objeto do estudo, o foco da qualidade, o ponto de vista, e contexto. Como resultado, a fase de definição fornece a direção geral do experimento, o seu escopo, a base para a formulação das hipóteses e as notações preliminares para a avaliação da validade. A definição dos objetivos pode ser apresentada de acordo com a seguinte estrutura:

Analisar .....*<Objeto do estudo>*

Especifica entidades que serão estudadas ao longo do experimento.

Com o propósito de.....*<Objetivo>*

Define a intenção da realização do experimento. Geralmente é escolhida da lista enumerada: caracterizar, avaliar, predizer, controlar, ou melhorar.

Com respeito a .....*<O foco da qualidade>*

Indica o principal aspecto da qualidade que está sendo estudado, por exemplo, eficiência, confiabilidade, produtividade.

Do ponto de vista .....*<Perspectiva>*

Especifica o ponto de vista que os resultados do experimento serão interpretados, por exemplo, desenvolvedor, consumidor, gerente.

No contexto de .....*<Contexto>*

Especifica o ambiente onde o experimento está sendo executado.

O anexo 1 (1.3) apresenta essa estrutura da definição preenchida.

A fase de planejamento implementa a fundação do experimento. Nesse momento acontecem a seleção do contexto, a formulação das hipóteses, a seleção das variáveis, a seleção dos participantes, o

projeto do experimento, preparação conceitual da instrumentação, a consideração da validade do experimento. O resultado dessa fase apresenta o experimento totalmente elaborado e pronto para execução.

O anexo 1 (2.1 – 2.7) contém a descrição dos resultados da fase do planejamento.

O aspecto mais importante da fase da execução é que a parte humana entra no jogo nesse momento. Os participantes devem ser preparados para a experimentação do ponto de vista moral e metodológica para evitar os resultados errôneos devido ao mal-entendido ou falta de interesse. A coleta de dados deve ser realizada de maneira que não cause efeito significativo ao processo sendo estudado. Finalmente, a validação preliminar dos dados experimentais se realiza.

Os resultados da fase de análise e interpretação oferecem as conclusões sobre a possibilidade da rejeição da hipótese nula usando a estatística descritiva, a redução do conjunto de dados, e a verificação das hipóteses. Nesse momento, os aspectos mais importantes são eliminar dados fora da distribuição normal (*outliers)*, escolher o teste estatístico apropriado, explicar os resultados considerando os aspectos da validade, realizar a análise custo-benefício, e interpretar corretamente os resultados negativos.

O anexo 1 (4.1 – 4.6) apresenta um exemplo da análise e da interpretação dos resultados do estudo experimental.

A fase menos elaborada da metodologia da experimentação na área de Engenharia de Software é a fase da apresentação e empacotamento. Os aspectos gerais a respeito de empacotamento serão discutidos na seção seguinte.

5.3 Empacotamento

Como mencionado em seção 4.2 uma das características mais importantes do experimento é a necessidade da sua repetição. Com a repetição os pesquisadores adquirem o conhecimento adicional a respeito dos conceitos estudados, e recebem os resultados que são iguais ou diferentes dos resultados do experimento original. De qualquer maneira, o aumento das repetições traz o aumento do aprendizado dos conceitos investigados e, também, a calibração das características do experimento.

O pré-requisito necessário para a repetição do experimento da alta qualidade é o seu empacotamento propriamente realizado.

O empacotamento padronizado dos dados experimentais pode servir como base para a criação das bibliotecas de experimentação. Para uma organização concreta, banco de dados com a informação empírica organizada dessa forma pode abrir a possibilidade de armazenar os artefatos diferentes desde

as idéias e as hipóteses até os resultados e experiências finais dos projetos realizados. Isso sem dúvida vai ajudar a reutilizar os as descobertas nos estudos futuros, providenciar os meios para a classificação dos dados experimentais e a criação dos relatórios detalhados com os resultados confiáveis.

Um exemplo da metodologia que considera a construção da base do conhecimento foi mencionado na seção 5.1 e é descrita com o maior detalhamento em [Basili94].

Uma idéia evolucionária do banco de dados experimentais é a organização dos servidores distribuídos que devem armazenar os dados experimentais a respeito da área da experimentação particular ou da família dos experimentos. Isso vai motivar o surgimento dos novos aspectos para a padronização do processo de empacotamento, porque o conteúdo e os formatos da apresentação da informação empírica podem ser bastante distintivos mesmo para domínios semelhantes.

É possível também a criação dos servidores regionalmente orientados. Esses servidores podem focar-se no armazenamento de informação a respeito dos experimentos conduzidos dento de uma região particular. Mas esses tipos de servidores vão trazer novos desafios por causa das diferenças de língua, cultural, política etc., requerendo o ajustamento das normas. A metodologia que trata da organização dos servidores distribuídos foi elaborada recentemente e é descrita com o detalhamento maior em [Conradi01].

O estado de arte atual a respeito de empacotamento de experimentos indica a ausência de normas internacionais aprovadas. De um lado, isso estimula a evolução dessa parte da Engenharia de Software experimental fornecendo as boas oportunidades e atraindo a atenção dos pesquisadores. Por outro lado, o grande número de caminhos possíveis da evolução se transforma numa certa anarquia trazendo a discordância sobre a interpretação dos conceitos relevantes ao empacotamento, o conteúdo ótimo do pacote, e os meios da apresentação do pacote.

A tendência comum e fundamental para a maioria das práticas de investigação nessa área é a necessidade da existência de um modelo elaborado para tratar os seguintes aspectos:

– A comunidade do processo do experimento.

Essa comunidade inclui não só os pesquisadores que preparam e executam os experimentos, analisam e apresentam os resultados, mas também os participantes que realizam as atribuições, oferecidas ao longo da execução do experimento, e os possíveis usuários da informação empírica que não participam diretamente no processo do experimento, mas usam os resultados do experimento para os seus estudos, pesquisas ou necessidades industriais.

– A organização do experimento.

A organização do experimento inclui o conjunto total das informações a respeito do projeto do experimento, a preparação dos participantes, a preparação da instrumentação, as diretrizes para a execução do experimento, etc.

– Os artefatos do experimento.

Os artefatos incluem a descrição da instrumentação usada para a coleta dos resultados puros durante a execução do experimento. Dependendo do tipo e dos objetivos do experimento, o papel do artefato do experimento pode ser cumprido pela documentação do projeto do desenvolvimento de software, código fonte, módulos executáveis, ou algo outro que possa ajudar a coletar a informação.

– Os resultados do experimento.

Os resultados incluem a descrição detalhada dos resultados recebidos. Os resultados são apresentados como: dados puros, dados refinados sem *outliers*, e dados analisados (depois da aplicação da estatística descritiva e os testes estatísticos). Os dados analisados podem ser utilizados para verificar as hipóteses e fazer as conclusões a respeito do atendimento dos objetivos. Além disso, os resultados devem ser apresentados de acordo com o conjunto de forma que incluam a informação geral sobre os resultados do experimento, a informação sobre o processo de experimentação, a lista dos problemas e as questões que devem ser resolvidas nos próximos estudos.

Os conceitos mencionados acima possibilitaram a criação do modelo de empacotamento dos estudos experimentais que ajuda a organizar a informação dos experimentos conduzidos de uma maneira estruturada e unificada. O anexo 2 apresenta os conceitos envolvidos num pacote de experimento. Detalhes adicionais do modelo podem ser encontrados em [Amaral02].

6. CONCLUSÕES

O relatório técnico oferece discussão sobre os conceitos básicos da organização de estudos experimentais na área de Engenharia de Software. Experimentação oferece o modo sistemático, disciplinado, computável e controlado para avaliação da atividade humana. Novas invenções e sugestões não devem ser apenas sugeridas, publicadas ou apresentadas para venda, mas pelo menos devem ser comparadas com as existentes.

Existem quatro métodos relevantes para condução de experimentos na área de Engenharia de Software: científico, de engenharia, experimental e analítico. A abordagem mais apropriada para a experimentação na área de Engenharia de Software é o método experimental que considera a proposição e avaliação do modelo com os estudos experimentais.

Os objetivos relacionados à execução de experimentos em Engenharia de Software são a caracterização, avaliação, previsão, controle e melhoria a respeito de produtos, processos, recursos, modelos e outros resultados e atividades do processo de desenvolvimento de software.

Há três estratégias empíricas básicas que descrevem três modos gerais de organização e execução dos estudos experimentais: *survey*, estudo de caso e experimento. De acordo com as estratégias empíricas existem três principais métodos de coleta de dados: histórico, de observação, e controlado. Um estudo experimental completo, na verdade, deve ser realizado considerando todas as estratégias e utilizar os diferentes métodos de coleta de dados, por exemplo, adquirir a informação inicial utilizando um *survey*, elaborar uma teoria através de um experimento controlado e verificar a teoria proposta em condições reais por meio de um estudo de caso.

Qualquer estudo experimental presume um relacionamento causa (representado por tratamentos) - efeito (representado por resultados). Os conceitos envolvidos num estudo experimental independentemente da estratégia incluem variáveis dependentes e independentes, objeto(s), participantes, contexto, hipóteses nula e alternativa(s), e o projeto de estudo.

Três princípios básicos: aleatoriedade, agrupamento, e balanceamento, são utilizados durante realização de um estudo experimental. Além disso, alguns princípios adicionais também podem ser considerados dependendo das características do estudo.

Ao longo do processo de realização de um estudo experimental um conjunto de parâmetros deve ser medido.A medição é a parte importante de qualquer estudo incluindo os estudos qualitativos porque oferece os meios para obtenção de dados necessários para verificação de hipóteses. Uma

característica importante a respeito da medição é a escala da medição. São quatro tipos de escala: nominal, ordinal, intervalo e razão.

Os resultados de estudo devem ser válidos. Isso significa que para um estudo ter uma importância científica ou industrial, seus resultados devem ser considerados segundo quatro tipos de validade: de conclusão, interna, externa e de construção, que apresentam, respectivamente, os aspectos a respeito da análise estatística, o relacionamento tratamento-resultado, a generalização dos resultados a uma população maior, a relação entre a teoria e observação. Os aspectos de validade são considerados logo na fase de planejamento do estudo e depois são utilizados durante a análise dos resultados.

Com o propósito de atingir estes objetivos várias metodologias de organização dos experimentos foram elaboradas. Um exemplo da metodologia da experimentação avançada é o Paradigma da Melhoria da Qualidade (QIP). A essência dessa metodologia está na melhoria contínua do processo do desenvolvimento de software. O QIP é ligado ao conceito da Fábrica de Experiência que representa o conjunto das ferramentas para armazenamento, modificação, e retirada da informação empacotada do projeto. Um outro instrumento ligado a QIP é a abordagem Goal/Question/Metric (GQM) que apresenta um modelo da medição baseado em: objetivos, questões e métricas.

O número e a complexidade de atividades ao longo do processo de realização de um estudo experimental podem variar de acordo com as características do estudo. A literatura apresenta cinco fases gerais que sempre estão presentes num processo da experimentação: definição, planejamento, execução, análise e apresentação e empacotamento.

A fase menos tratada na literatura é a fase de empacotamento. Entretanto essa fase é importante porque a repetição de um estudo experimental só pode acontecer caso todos os aspectos incluindo o projeto, a instrumentação, as observações sobre a execução, os resultados, estejam adequadamente empacotados.

A aplicação da abordagem empírica em grande parte determina o futuro da ciência de Engenharia de Software. O grande número de descobertas sem validação empírica perde seu significado e não pode entrar como um elemento inerente na estrutura da ciência e ainda pode causar a demora no desenvolvimento da ciência direcionando os pesquisadores nos caminhos que na verdade não têm futuro.

REFERÊNCIAS

[Amaral02] AMARAL, E. A. G.; TRAVASSOS, G. H. "Em busca de uma abordagem para Empacotamento de Experimentos em Engenharia de Software". In. Anais da 2a JIISIC - JORNADA IBERO-AMERICANA DE ENGENHARIA DE SOFTWARE E ENGENHARIA DE CONHECIMENTO, 2002, Salvador, 2002.

[Basili93] Basili, V., "The Empirical Paradigm in Software Engineering", Experimental Software Engineering Issues: Critical Assessment and Future Directives, Spring-Verlag, #706, Lectural Notes in CS, University of Maryland, 1993.

[Basili94] Basili, V., Caldeira, G., Rombach, H., "Experience Factory", Encyclopedia of Software Engineering, ed. J.J. Marciniak, Vol.I, Wiley, pp. 469-476, 1994.

[Basili96] Basili V., "The Role of Experimentation in Software Engineering: Past, Present, Future", Proceedings of the 18th International Conference on Software Engineering, 1(2), pp. 133-164, 1996.

[Brooks96] Brooks, A., Dali, J., Miller, J., Roper, M., Wood, M., "Replication of Experimental Results in Software Engineering", Technical Report ISERN-96-10, International Software Engineering Research Network (ISERN), University of Strathclyde, 1996.

[Conradi01] Conradi,R., Basili, V., Carver, J., Shull, F., Travassos, G., "A Pragmatic Document Standards for an Experience Library: Roles, Documents, Contents and Structure.", Technical Report CS-TR-4235, University of Maryland, April 2001.

[Juristo98] Juristo, N., Moreno, A., "An Adaptation of Experimental Design to the Empirical Validation of Software Engineering Theory", 23nd Annual NASA Software Engineering Workshop, USA, December 1998.

[Miller97] Miller, J., Dali, J., Wood, M., Roper, M., Brooks, A., "Statistical power and its Subcomponents – Missing and Misunderstood Concepts in Empirical Software Engineering Research", Information and Software Technology, Vol. 39, No. 4, pp. 285-295, 1997.

[Seaman99] Seaman, C., "Qualitative Methods in Empirical Studies of Software Engineering", IEEE Computer, Vol. 25, No. 4, July/August 1999.

[SEL95] "Software Measurement Guidebook. Revision 1", Software Engineering Laboratory, University of Maryland, 1995

[Shull01] Shull, F., Carver, J., Travassos, G., "An Empirical Methodology for Introducing Software Processes", Proceedings of the ESEC/FSE-9, Austria, September 2001.

[Solingen99] Solingen, R., Berghout, E., "The Goal/Question/Metric Method: a Practical Guide for Quality Improvement of Software Development", the McGraw-Hill Companies, UK, 1999.

[Tichy98] Tichy, W., "Should Computer Scientists Experiment More?", IEEE Computer, Vol. 31, No 5, pp. 32-39, 1998.

[Wohlin00] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., Wesslén, A., "Experimentation in Software Engineering: an introduction", Kluwer Academic Publishers, USA, 2000.

[Zelkowitz98] Zelkowitz, M., Wallace, D., "Experimental Models for Validating Technology", IEEE Computer, Vol.31, No.5, pp. 23-31, 1998.

GLOSSÁRIO DE PALAVRAS EM INGLÊS

Baseline.....é o conjunto de medidas que descrevem as características usuais ou típicas. Em relação aos estudos experimentais a *baseline* descreve, por exemplo, o projeto médio de uma organização e é utilizado com a finalidade de comparação com outros projetos dessa organização.

Benchmarking.....é o processo de execução de dois ou mais sistemas com a finalidade de comparar as características desses sistemas, principalmente, o desempenho.

Blocking.....é uma técnica de organização do estudo experimental utilizado quando um fator no projeto do experimento provavelmente tem um efeito sobre o resultado, mas esse efeito não é interessante para os pesquisadores. *Blocking* presume agrupamento de participantes e atribuição de tratamentos que sistematicamente eliminam o efeito indesejado durante a comparação dos tratamentos.

Bottom-up .....é uma técnica da organização de alguma atividade como estudo, planejamento, desenvolvimento, entre outros, quando o processo começa no mais baixo nível continuamente aumentando o nível até chegar ao mais alto nível.

Confounding factor .....é o aspecto que provavelmente causa o efeito em vez de fator(es) considerado(s) no projeto de estudo experimental.

Feedback .....é o conjunto de informações sobre o processo ou resultados de estudo que serão utilizados para calibração do projeto de estudo.

Framework .....é uma estrutura de sistema ou processo que deve ser instanciada. O processo de instanciação presume especialização, parametrização, implementação dos elementos da estrutura entre outros.

Outlier .....são os dados recebidos durante o estudo experimental que representam a informação falsa ou extremamente diferente do que foi esperado e dos resultados médios do estudo, por exemplo, devido a mal-entendimento.

Rescaling.....é a transformação de uma escala de medição para outra.

Survey .....é uma das estratégias empíricas que realiza uma pesquisa retrospectiva sobre um assunto que já tinha acontecido. Os meios principais de *survey* são entrevistas ou questionários. A tradução possível para português é "Pesquisa de Campo".

Time-to-market .....é um dos fatores chaves que mudaram o conceito do desenvolvimento de software. Implica que o desenvolvimento de software deve ser organizado a produzir os produtos ou artefatos de software considerando as forças do mercado.

Top-down.....é uma técnica de organização de alguma atividade como estudo, planejamento, desenvolvimento, entre outros, quando o processo começa no mais alto nível e vem diminuindo o nível de detalhamento até chegar ao nível desejado.

Toy problem.....é o modelo de sistema utilizado num estudo experimental para tentar concluir sobre alguns aspectos de sistema real. Um exemplo é a organização do projeto universitário com a finalidade de estudar um sistema real (industrial).

Type-I-error.....é o tipo de erro que pode acontecer durante a verificação de hipótese nula quando o teste estatístico indica o relacionamento mesmo que não exista nenhum relacionamento real.

Type-II-error .....é o tipo de erro que pode acontecer durante a verificação de hipótese nula quando o teste estatístico não indica o relacionamento mesmo que efetivamente exista um relacionamento.

ANEXO 1: O ESTUDO EXPERIMENTAL

O ROTEIRO DO ESTUDO

1. DEFINIÇÃO DOS OBJETIVOS

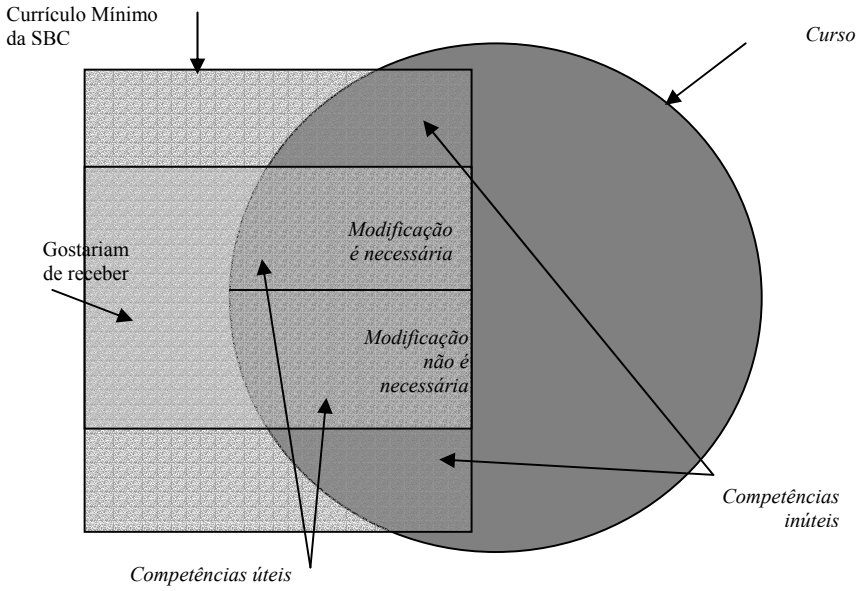
1.2 Objetivo global:

Definir se o curso básico de Engenharia de Software (ES) oferece competências necessárias para desenvolvimento de software pessoal do ponto de vista dos seus alunos.

1.3 Objetivo da medição:

Tendo como base as competências do currículo mínimo da SBC, caracterizar:

1. Quais são as competências que os alunos recebem:
   - quais são as competências oferecidas pelo curso básico de ES que os alunos consideram úteis para o desenvolvimento de software pessoal;
   - quais são as competências oferecidas pelo curso básico de ES que os alunos consideram inúteis para o desenvolvimento de software pessoal.
2. Quais são as competências que os alunos recebem com conteúdo inadequado:
   - quais são as competências que necessitam melhor detalhamento;
   - quais são as competências que apresentam o detalhamento excessivo.
3. Quais são as competências que os alunos gostariam de receber além das oferecidas pelo curso básico de ES.



1.3 Objetivo do estudo:

Analisar as competências oferecidas aos alunos

Com o propósito de caracterizar

Com respeito à interseção com as competências do currículo mínimo da SBC

Do ponto de vista do desenvolvedor de software pessoal

No contexto de alunos de curso básico de ES.

1.4 Questões:

Q1: Existem competências de ES listadas no currículo mínimo da SBC que não fazem parte do curso básico de ES?

Métrica: A lista de competências de ES do currículo mínimo da SBC que não fazem parte do curso básico de ES.

Q2: Existem competências de ES listadas no currículo mínimo da SBC e oferecidas pelo curso básico de ES que são consideradas inúteis pelos alunos?

Métrica: A lista de competências de ES do currículo mínimo da SBC que fazem parte do curso básico de ES e são consideradas inúteis pelos alunos desse curso

Q3: Existem competências listadas no currículo mínimo da SBC, oferecidas pelo curso básico de ES e consideradas úteis pelos alunos, cujo detalhamento deve ser modificado?

Métrica: A lista de competências de ES do currículo mínimo da SBC que fazem parte do curso básico de ES e são consideradas úteis pelos alunos desse curso cujo detalhamento deve ser modificado

Q4: Existem competências de ES listadas no currículo mínimo da SBC que não fazem parte do curso básico de ES, mas que os alunos gostariam de receber porque consideram úteis para desenvolvimento de software pessoal?

Métrica: A lista de competências de ES do currículo mínimo da SBC que não fazem parte do curso básico de ES.

2. PLANEJAMENTO

2.1 Definição das Hipóteses

Hipótese nula (H0): As competências oferecidas para os alunos do curso básico de ES são similares às competências de ES que o currículo mínimo da SBC considera fundamental para o desenvolvimento de software

C_a – competências oferecidas para os alunos do curso básico de ES;

C_c – competências de ES do currículo mínimo da SBC.

H0: $C_c - (C_a \cap C_c) = \varnothing$

Hipótese alternativa (H1) A lista de competências oferecidas para os alunos do curso básico de ES é diferente da lista de competências de ES que o currículo mínimo da SBC considera fundamental para o desenvolvimento de software.

C_a – competências oferecidas para os alunos do curso básico de ES;

C_c – competências de ES do currículo mínimo da SBC.

H1: $C_c - (C_a \cap C_c) \neq \varnothing$

Hipótese alternativa (H2): Na lista de competências oferecidas para os alunos do curso básico de ES e que fazem parte do currículo mínimo da SBC existem competências que os alunos consideram inúteis para o desenvolvimento de software pessoal.

C_a – competências oferecidas para os alunos do curso básico de ES e que fazem parte do currículo mínimo da SBC;

C_{au} – competências oferecidas para os alunos do curso básico de ES e que fazem parte do currículo mínimo da SBC, que os alunos consideram úteis para desenvolvimento de software pessoal.

H2: $C_a - (C_a \cap C_{au}) \neq \varnothing$

Hipótese alternativa (H3): Na lista de competências oferecidas para os alunos do curso básico de ES, que fazem parte do currículo mínimo da SBC e consideradas úteis para desenvolvimento de software pessoal, existem competências cujo detalhamento deve ser modificado para atingir o nível esperado pelos desenvolvedores de software pessoal.

C_{au} – competências oferecidas para os alunos do curso básico de ES, que fazem parte do currículo mínimo da SBC e consideradas úteis para desenvolvimento de software pessoal;

C_{aun} – competências oferecidas para os alunos do curso básico de ES, que fazem parte do currículo mínimo da SBC e são consideradas úteis para o desenvolvimento de software pessoal, cujo detalhamento não precisa de modificação.

H3: $C_{au} - (C_{au} \cap C_{aun}) \neq \varnothing$

Hipótese alternativa (H4): Na lista de competências não oferecidas para os alunos do curso básico de ES existem competências que os alunos gostariam de receber.

C_{na} – competências não oferecidas para os alunos do curso básico de ES ($C_{na} \equiv C_c - (C_a \cap C_c)$)

C_g – competências não oferecidas para os alunos do curso básico de ES que os alunos gostariam de receber.

H4: $C_{na} - (C_{na} \cap C_g) \neq \varnothing$

2.2 Descrição da instrumentação

Para cada competência que o currículo mínimo da SBC considera fundamental para o desenvolvimento de software oferecer escolha:

Presença da competência (P)	Utilidade da competência (U)	Adequação do nível de detalhamento da competência (A)
1. Não é oferecida pelo curso e não gostaria de receber. 2. Não oferecida, mas gostaria de receber. 3. Oferecida, parcialmente. 4. Oferecida.	1. Não é útil. 2. Provavelmente é útil, mas ainda não apliquei. 3. É útil e já apliquei em diferentes projetos.	1. O detalhamento deve ser aumentado. 2. O detalhamento não precisa ser modificado. 3. O detalhamento deve ser diminuído.

Para cada competência aplicar teste estatístico Chi-2 para definir:

se pode considerar que essa competência é fornecida;

se pode considerar que essa competência é útil.

se pode considerar que o detalhamento da competência não precisa de modificação.

Resultado: N competências com valores (P;U;A)

onde P – presença {0 – não oferecida;1 - oferecida}; U – utilidade {0 – não é útil;1 - é útil}; A – adequação {0 – o nível é adequado,1 – o nível não é adequado}.

Métricas

Nº	P	U	A	Descrição da competência	Questões
1	0	0	0	não é oferecida, não é útil, a modificação não é necessária.	Q1, Q4
2	0	0	1	não é oferecida, não é útil, a modificação é necessária.	N/A
3	0	1	0	não é oferecida, é útil, a modificação não é necessária.	Q1, Q4
4	0	1	1	não é oferecida, é útil, a modificação é necessária.	Q1, Q4
5	1	0	0	É oferecida, não é útil, a modificação não é necessária.	Q2
6	1	0	1	É oferecida, não é útil, a modificação é necessária.	Q2
7	1	1	0	É oferecida, é útil, a modificação não é necessária.	Q3
8	1	1	1	É oferecida, é útil, a modificação é necessária.	Q3

2.3 Seleção do contexto

O contexto pode ser caracterizado conforme quatro dimensões:

- o processo: on-line / off-line;
- os participantes: alunos / profissionais;
- realidade: o problema real / modelado;
- generalidade: especifico / geral.

Nosso estudo supõe o processo off-line porque os alunos não estão sendo entrevistados durante todo o tempo do curso, mas em um certo instante. Os participantes são os alunos que estão realizando o curso. O estudo é modelado porque as competências dos alunos não são caracterizadas durante a resolução do problema real, mas utilizando as notas subjetivas. As competências dos alunos do certo curso são comparadas com as competências listadas no currículo da SBC, então, o contexto possui o caráter especifico.

2.4 Seleção dos indivíduos

Como participantes para o estudo se propõe utilizar os alunos de pós-graduação, graduação ou alunos de 2o grau da área de Engenharia de Software. Assume-se que esses indivíduos estão disponíveis para o estudo e a maioria deles desenvolve software pessoal.

Seria conveniente utilizar para o estudo os alunos que estão realizando alguns de cursos básicos na área de Engenharia de Software. Nesse caso, dependendo do tamanho da turma, é possível usar uma das técnicas (probabilística ou não-probabilística) para escolha dos indivíduos.

Supõe-se propor aos participantes o questionário que tem como objetivo caracterizar sua formação do ponto de vista acadêmico, experiência, tipo de curso entre outros para analisar os dados e reduzir o viés.

2.5 Variáveis

Variável independente:

A lista de competências de ES do currículo mínimo da SBC.

Variáveis dependentes:

1. A similaridade entre competências oferecidas para os alunos do curso básico de ES e competências de ES do currículo mínimo da SBC.

Pode receber os valores:

Igual, quando todas as competências têm o valor PUA = {1, X, X} (métricas 5-8);

Diferente, quando todas as competências têm o valor PUA = {0, X, X} (métricas 1-4).

Similar, quando não se cumprem as condições de "Igual" e "Diferente". O grau de similaridade pode ser avaliado como:

$$\{1, X, X\} / (\{0,X,X\} + \{1,X,X\}) * 100\%$$

2. A utilidade de competências similares. Mostra a parte útil das competências oferecidas pelo curso básico de ES:

Parte útil: $\{1, 1, X\} / \{1, X, X\} * 100\%$

Parte inútil: $\{1, 0, X\} / \{1, X, X\} * 100\%$

3. A adequação de competências similares. Mostra a parte adequada das competências oferecidas pelo curso básico de ES:

Parte adequada: {1, X, 0} / {1, X, X} * 100%
Parte inadequada: {1, X, 1} / {1, X, X} * 100%

2.6 Análise qualitativa

Para analisar a informação referente às competências não oferecidas para os alunos do curso básico de ES, mas que os alunos gostariam de receber, se propõe aplicar a análise qualitativa. Essa análise deve apresentar a lista de competências de ES do currículo mínimo da SBC, que não são oferecidas para os alunos do curso básico de ES, mas que os alunos consideram necessárias para desenvolvimento de software pessoal e gostariam de receber realizando o curso.

Assim, essa análise deve considerar competências com valor PUA = {0, X, X} (métricas 1-4) e a opção "Não oferecida, mas gostaria de receber" para "presença da competência".

2.7 Validade

Validade interna: como mencionado no parte "Seleção dos indivíduos" para o estudo se propõe a utilizar os alunos da área de Engenharia de Software, que geralmente costumam desenvolver software pessoal. Assim, assume-se que eles são representativos para a população dos desenvolvedores de software pessoal.

Além disso, para redução da influência dos fatores que não são interesse do nosso estudo e, portanto, para aumento da validade interna do estudo supõe-se utilizar os dados do questionário para divisão dos participantes em grupos conforme a suas características individuais.

Validade de conclusão: para receber os valores da presença, utilidade e conformidade o teste binomial será utilizado. A verificação de hipótese será feita por meio de simples demonstração de presença ou não de competências nas listas que representam os variáveis independentes.

Validade de construção: esse estudo está caracterizado pela conformidade das competências listadas no currículo mínimo da SBC com as competências reais necessárias para desenvolvimento de software pessoal. O currículo mínimo da SBC representa a lista de competências que os especialistas na área de computação devem possuir para mostrar o desempenho adequado do ponto de vista da SBC. As competências, que têm o maior relacionamento com desenvolvimento de software pessoal do ponto de

vista dos pesquisadores, foram escolhidas do conjunto total de competências do currículo mínimo da SBC.

Validade externa: como foi mencionado nas partes "Seleção dos indivíduos" e "Validade interna" os participantes do estudo em geral podem ser considerados representativos para a população dos desenvolvedores de software pessoal. Para avaliação do nível de envolvimento no processo de desenvolvimento de software pessoal os dados do questionário conforme a experiência dos participantes podem ser analisados.

Os materiais utilizados no estudo podem ser considerados representativos e "em tempo" para o problema sob análise, porque se compõem das competências do currículo mínimo atual da SBC relacionadas ao desenvolvimento de software.

As características temporárias não devem ser o problema, porque os materiais dão a possibilidade de conduzir o estudo durante a meia-hora. Além disso, o estudo será realizado quando os participantes estiverem realizando algum curso e, geralmente, estiverem envolvidos no desenvolvimento de software.

3. OPERAÇÃO

3.1 Questionário do Perfil do Participante

Marque um X sobre a opção que melhor representar sua formação.

FORMAÇÃO

Instituição.

- O Pública
- O Particular

Tipo de curso.

- O Engenharia.
- O Informática / Ciência da computação.
- O Matemática
- O Outros.

Acadêmica:

- O Ensino Médio Técnico
- O Universitária
- O Pós-graduação

EXPERIÊNCIA PROFISSIONAL

Tempo de experiência:

- O Seis meses.
- O Entre seis meses e dois anos.
- O Entre 2 e 4 anos
- O Entre 4 e 6 anos
- O Acima de 6 anos

Como você classificaria a sua experiência em desenvolvimento de software.

- O Baixa
- O Media
- O Alta

3.2 Questionário de Competências

Sob o ponto de vista de *desenvolvimento de software pessoal* e considerando o tipo de curso que você indicou acima, por favor, avalie e marque as colunas correspondentes segundo as escalas abaixo, a presença, utilidade e adequação quanto ao detalhamento que lhe foi apresentado durante o curso, das competências listadas no questionário:

Presença	Utilidade	Adequação do nível de detalhamento
1 Não oferecida durante os cursos e não gostaria de receber. 2 Não oferecida, mas gostaria de receber. 3 Oferecida, parcialmente. 4 Oferecida.	1 Não se demonstrou útil. 2 Provavelmente é útil, mas ainda não apliquei. 3 É útil e apliquei em muitos projetos.	1 O detalhamento deve ser aumentado. 2 O detalhamento não precisa ser modificado. 3 O detalhamento deve ser diminuído.

N	Competência	Descrição	Presença				Utilidade			Adequação		
			1	2	3	4	1	2	3	1	2	3
	Processo											
1	Qualidade do produto	é capaz de avaliar a qualidade de um produto de software utilizando-se de normas apropriadas; é capaz de empregar os modelos de medição.										
2	Interface homem-máquina	é capaz de avaliar e propor mudanças na interface de um produto de software, sob o ângulo da usabilidade.										
	Processo											
3	Modelos de ciclo de vida	é capaz de identificar situações e de empregar adequadamente os diversos tipos de modelos de ciclo de vida …										
4	Gerenciamento de configuração	é capaz de definir atividades técnicas e organizacionais relacionadas a identificação, controle, …										
5	Garantia da qualidade	é capaz de estabelecer processos, métricas e avaliações que possibilitem e garantam a qualidade de software.										
6	Melhoria de processo	possui conhecimento dos modelos de melhoria de qualidade de software (ex: CMM e SPICE).										
7	Padrões	possui conhecimento sobre padrões de desenvolvimento de software e consegue empregá-los.										
8	Engenharia reversa	é capaz de analisar um sistema de forma a identificar seus componentes e suas inter-relações; e de criar representações de maior nível de abstração.										
9	Reengenharia	é capaz de examinar um sistema e de reconstituí-lo em uma nova forma considerando sua respectiva redocumentação, redefinição de objetivos.										
10	Planejamento do desenvolvimento	é capaz de definir um planejamento de software utilizando-se de técnicas apropriadas (ex: técnicas de estimativas e de medição, PERT, CPM,…)										
	Tecnologia											
11	Programação	é capaz de identificar os diversos tipos de paradigmas de programação (ex: procedural, funcional, orientado a objetos).										
12	Ferramentas CASE	é capaz de utilizar uma ferramenta CASE para auxílio na construção de software										

3.3 Resultado do estudo

Dados de estudo crus

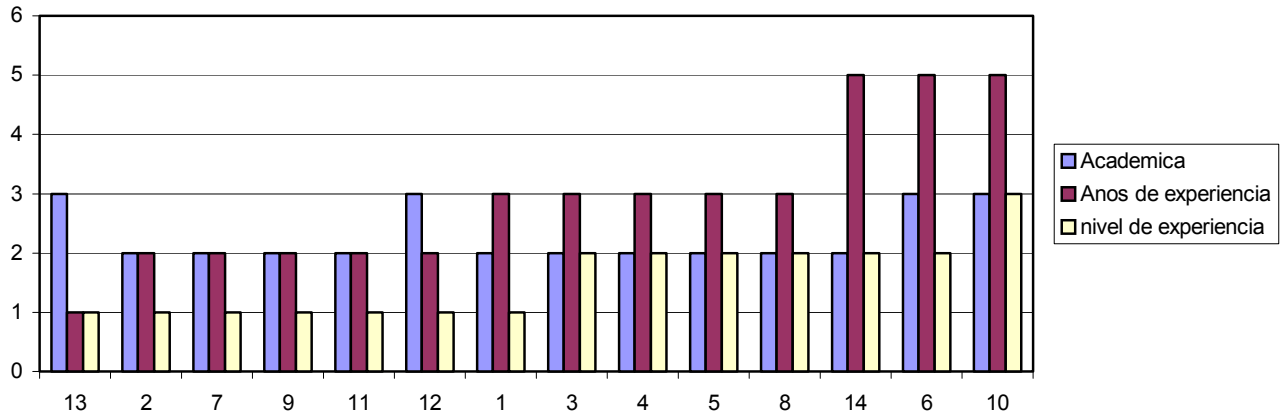
N	Competência	Descrição	Presença				Utilidade			Adequação		
			1	2	3	4	1	2	3	1	2	3
	Processo											
1	Qualidade do produto	é capaz de avaliar a qualidade de um produto de software utilizando-se de normas apropriadas; é capaz de empregar os modelos de medição.										
2	Interface homem-máquina	é capaz de avaliar e propor mudanças na interface de um produto de software, sob o ângulo da usabilidade.										
	Processo											
3	Modelos de ciclo de vida	é capaz de identificar situações e de empregar adequadamente os diversos tipos de modelos de ciclo de vida …										
4	Gerenciamento de configuração	é capaz de definir atividades técnicas e organizacionais relacionadas a identificação, controle, …										
5	Garantia da qualidade	é capaz de estabelecer processos, métricas e avaliações que possibilitem e garantam a qualidade de software.										
6	Melhoria de processo	possui conhecimento dos modelos de melhoria de qualidade de software (ex: CMM e SPICE).										
7	Padrões	possui conhecimento sobre padrões de desenvolvimento de software e consegue empregá-los.										
8	Engenharia reversa	é capaz de analisar um sistema de forma a identificar seus componentes e suas inter-relações; e de criar representações de maior nível de abstração.										
9	Reengenharia	é capaz de examinar um sistema e de reconstituí-lo em uma nova forma considerando sua respectiva redocumentação, redefinição de objetivos.										
10	Planejamento do desenvolvimento	é capaz de definir um planejamento de software utilizando-se de técnicas apropriadas (ex: técnicas de estimativas e de medição, PERT, CPM,…)										
	Tecnologia											
11	Programação	é capaz de identificar os diversos tipos de paradigmas de programação (ex: procedural, funcional, orientado a objetos).										
12	Ferramentas CASE	é capaz de utilizar uma ferramenta CASE para auxílio na construção de software										

Perfis dos participantes

Numero do participante	Instituição	Curso	Acadêmica	Experiência	
				anos	nível
	(1-2)	(1-4)	(1-3)	(1-5)	(1-3)
1	1	2	2	3	1
2	1	2	2	2	1
3	1	2	2	3	2
4	1	2	2	3	2
5	1	2	2	3	2
6	1	2	3	5	2
7	1	2	2	2	1
8	1	2	2	3	2
9	1	1	2	2	1
10	1	2	3	5	3
11	1	2	2	2	1
12	1	2	3	2	1
13	1	1	3	1	1
14	1	2	2	5	2

Legenda:

Instituição		Curso		Acadêmica		Experiência (anos)		Experiência (nível)	
1	Publica	1	Engenharia	1	Técnico	1	Menos 6 meses	1	Baixo
2	Particular	2	Informática	2	Universitária	2	6meses - 2anos	2	Médio
		3	Matemática	3	Pós-graduação	3	2 anos 4 anos	3	Alto
		4	Outros			4	4 anos 6 anos		
						5	6 anos e mais		



4. ANÁLISE E INTERPRETAÇÃO DOS RESULTADOS

4.1 Validação dos dados

Algumas respostas dos participantes estão erradas do ponto de vista dos valores válidos de PUA.

Participante	Resposta	Valor PUA
5	4	{0, 0, 1}
7	6, 10	{0, 0, 1}
8	4	{0, 0, 1}

O participante 14 não respondeu a pergunta 10 e seus dados foram eliminados da análise estatística.

4.2 Estatística descritiva

Medidas de tendência central. Como os valores "Presença", "Utilidade" e "Adequação" são da escala ordinal, é possível definir somente as métricas de "moda" e "mediana".

Presença

	Competências											
	1	2	3	4	5	6	7	8	9	10	11	12
mediana	3	3	4	2	3	2	2	2	2	3	4	3
moda	3	4	4	1	3	{1,3}	{2,3}	2	2	4	4	3

Utilidade

	Competências											
	1	2	3	4	5	6	7	8	9	10	11	12
mediana	2	3	3	2	2	2	2	2	2	2	3	3
moda	2	2	3	2	2	2	2	2	2	2	{3,4}	3

Adequação

	Competências											
	1	2	3	4	5	6	7	8	9	10	11	12
mediana	1	1	2	1	1	1	1	1	1	2	2	1
moda	1	1	2	1	1	1	1	1	1	{1,2}	2	{1,2}

Considerando respostas recebidas durante o estudo e utilizando os resultados de estatística descritiva podemos dividir as perguntas nos 3 grupos:

(valores nas tabelas significam: P - presente:não_presente; U - útil:inútil; A - adequado:inadequado)

1. Competências aprendidas e não utilizadas

Nº	Competências	P	U	A	Característica
1	Qualidade do produto	11:2	12:1	4:9	– As competências são recebidas parcialmente ou plenamente durante o curso; – As competências são consideradas úteis, mas a maioria (12, 10, 8) ainda não utilizou para desenvolvimento de software pessoal; – O detalhamento deve ser modificado, provavelmente para entender onde elas podem ser utilizadas.
5	Garantia da qualidade (processo)	9:4	10:3	4:9	
10	Planejamento do desenvolvimento	7:6	8:5	6:7	

2. Competências mal-entendidas

Nº	Competências	P	U	A	Característica
4	Gerenciamento de configuração	4:9	9:2	3:10	– A maioria dos participantes não recebeu as competências durante o curso, mas gostaria de receber; – As competências são consideradas úteis, mas quase todos (7, 10, 11, 12, 8) ainda não utilizaram para desenvolvimento de software pessoal; – O detalhamento deve ser modificado, porque mesmo que os participantes receberam alguma informação sobre essas competências o detalhamento não é adequado para utilização.
6	Modelos de melhoria de processo	6:7	12:1	5:8	
7	Padrões	6:7	13:0	4:9	
8	Engenharia reversa	2:11	13:0	3:10	
9	Reengenharia	4:9	11:2	5:7	

3. Competências aprendidas

Nº	Competências	P	U	A	Característica
2	Interface homem-máquina	8:5	13:0	4:9	– Todas as competências são recebidas durante o curso e maioria dos participantes recebeu essas competências plenamente; – As competências são consideradas úteis e a maioria dos participantes (7, 8, 12, 9) já utilizou essas competências para desenvolvimento de software pessoal; – O detalhamento quase não precisa de modificação à exceção de Interface homem-máquina, cujo detalhamento deve ser aumentado.
3	Modelos de ciclo de vida	12:1	13:0	8:5	
11	Programação	13:0	13:0	8:5	
12	Ferramentas CASE	11:2	13:0	6:7	

4.3 Aplicação do teste estatístico

Para cada competência foi aplicado o teste estatístico Chi-2 para definir:

se pode considerar que essa competência é fornecida;

se pode considerar que essa competência é útil.

se pode considerar que o detalhamento da competência não precisa de modificação.

Como o teste Chi-2 considera comparação das distribuições podemos comparar a distribuição de respostas dos participantes com a distribuição ideal, i.e. quando todos participantes responderam igualmente que competência é recebida, é útil e não precisa de modificação.

No início precisamos encontrar os valores Chi-2 para todas possíveis distribuições de respostas para poder tentar concluir sobre presença, utilidade e adequação de cada competência.

Resposta	Distribuição		Total
	real	ideal	
positiva	m	13	13+m
negativa	n	0	n
Total	13	13	26

Assim, tivemos

Distribuição	…	(+7):(-6)	(+8):(-5)	(+9):(-4)	(+10):(-3)	(+11):(-2)	…
Valor Chi-2		6.80	6.19	4.72	3.39	2.16	

Como valor Chi-2 para grau de liberdade 2 é 5.99, concluirmos que as competências com distribuição das respostas (+9):(-4), (+10):(-3), (+11):(-2), (+12):(-1) e (+13):(-0) podem receber os valores "Presença", "Utilidade" = {1} e o valor "Adequação" = {0}. Para as competências com outras distribuições das respostas os valores "Presença", "Utilidade" = {0} e o valor "Adequação" ={1}.

	Competências											
	1	2	3	4	5	6	7	8	9	10	11	12
Presença	1	0	1	0	1	0	0	0	0	0	1	1
Utilidade	1	1	1	1	1	1	1	1	1	1	1	1
Adequação	1	1	1	1	1	1	1	1	1	1	1	1

4.4 Análise quantitativa

Para verificar a similaridade entre competências oferecidas para os alunos do curso básico de ES e competências de ES do currículo mínimo da SBC é necessário calcular o valor de variável dependente 1. Os conjuntos de competências oferecidas para os alunos do curso básico de ES e competências de ES do currículo mínimo da SBC não podem ser considerados iguais (a quantidade de competências com o valor PUA {1, X, X} = 5 < 12), nem diferentes (a quantidade de competências com o valor PUA {0, X, X} é 7 <12). Assim, precisamos calcular o grau de similaridade segundo a formula do variável 1:

grau de similaridade = 5 / 12 * 100% = 41,7%

Para verificar a utilidade de competências similares, i.e. competências listadas no currículo mínimo da SBC e que são oferecidas pelo curso básico de ES, é necessário calcular o valor de variável dependente 2:

Parte útil das competências similares: 5 / 5 * 100% = 100%
Parte inútil das competências similares: 0 / 5 * 100% = 0%

Para verificar a adequação de competências similares, i.e. se o nível de detalhamento de competências precisa ser modificado, é necessário calcular o valor de variável dependente 3:

Parte adequada das competências similares: 0 / 5 * 100% = 0%
Parte inadequada das competências similares: 5 / 5 * 100% = 100%

4.5 Análise qualitativa

Para verificar se existem competências de ES listadas no currículo mínimo da SBC que não fazem parte do curso básico de ES, mas que os alunos gostariam de receber porque consideram úteis para desenvolvimento de software pessoal, foi feita a análise qualitativa.

A tabela abaixo mostra as competências não oferecidas pelo curso básico de ES:

Competência	2	4	6	7	8	9	10
A quantidade total de participantes que não receberam competência.	4	9	7	7	11	9	6
A quantidade de participantes que não receberam competência, mas gostariam de receber.	1	3	2	4	7	7	3

Assim podemos concluir, que as competências 8 (Engenharia reversa) e 9 (Reengenharia) provocam interesse dos alunos, que não receberam essas competências durante o curso básico de ES, mais do que as outras competências que não foram oferecidas pelo curso básico de ES. Isso pode ser explicado pelo conteúdo dessas competências. A informação sobre Engenharia reversa e Reengenharia, provavelmente, não está sendo oferecida em geral ou oferecida com pouco detalhamento pelo curso básico de ES e assim chama atenção dos alunos. Outras competências não oferecidas pelo curso básico de ES, provavelmente, são conhecidas aos alunos, mas a falta de exemplos ou experiência de aplicação dessas competências reduz a interesse dos alunos.

4.6 Verificação das hipóteses

Hipótese nula (H0): Para verificar a hipótese nula precisamos responder a questão Q1 utilizando a métrica M3. O resultado do teste Chi-2 mostra que 7 das 12 competências não podem ser considerados como oferecidas pelo curso básico de ES para os alunos. Assim, podemos fazer conclusão, que "a lista de competências oferecidas para os alunos do curso básico de ES é diferente da lista de competências de ES que o currículo mínimo da SBC considera fundamental para o desenvolvimento de software" (hipótese alternativa H1).

Não podemos dizer que "na lista de competências oferecidas para os alunos do curso básico de ES e que fazem parte do currículo mínimo da SBC existem competências que os alunos consideram inúteis para o desenvolvimento de software pessoal" (hipótese alternativa H2) porque, respondendo a questão Q2 (métrica M5) o resultado do teste Chi-2 mostra, que nenhuma das 12 competências pode ser considerada como inútil.

Finalmente, podemos fazer conclusão, que "na lista de competências oferecidas para os alunos do curso básico de ES, que fazem parte do currículo mínimo da SBC e consideradas úteis para desenvolvimento de software pessoal, existem competências cujo detalhamento deve ser modificado para atingir o nível esperado pelos desenvolvedores de software pessoal" (hipótese alternativa H3) porque , respondendo a questão Q3 (métrica M7) o resultado do teste Chi-2 mostra, que todas as 12 competências precisam de modificação.

Para fazer conclusões relevante hipótese alternativa (H4) a análise qualitativa foi feita. Os resultados da análise mostraram que 2 competências não oferecidas pelo curso básico de ES provocaram interesse dos alunos. Assim podemos concluir, que "na lista de competências não oferecidas para os alunos do curso básico de ES existem competências que os alunos gostariam de receber" (hipótese alternativa H4).

ANEXO 2: CONCEITOS ENVOLVIDOS NUM PACOTE DE EXPERIMENTO

Classe	Descrição
Aggregated Results	Define os objetivos e hipóteses para combinar os resultados de uma família de experimentos repetidos ou similares.
Aggregator	Combina os resultados de uma família de experimentos similares ou repetidos.
Analyze Documents	Descreve informações de Análise como os Casos de Uso da UML e comentários.
Assignment Description	Descreve informações sobre os exercícios usados através do experimento.
Code	Descreve informações sobre os programas, funções e interfaces usados no experimento.
Comments	Descreve os comentários sobre os resultados do experimento.
Component	Descreve informações sobre componentes como bibliotecas e "componentes de prateleira", usados no experimento.
Course Training Notes	Descreve conceitos, termos relevantes e informações de notas de aulas, cursos e treinamentos.
Data	Descreve informações sobre os dados resultantes de um experimento.
Defect List Template	È um *template* utilizado para descrever os defeitos encontrados em um experimento.
Defect Lists	Descreve os defeitos encontrados no experimento, através de uma lista.
Design Documents	Descreve informações de Projeto como Diagramas de Classe da UML.
Documentation	Descreve informações sobre os artefatos não executáveis.
Domain Business Model	Descreve informações sobre o Modelo de Negócios.
Domain Description	Descreve o domínio da aplicação para o qual os artefatos foram construídos.
Experiment Artifact	Descreve os artefatos do experimento usados em um experimento. Podendo ser de três tipos: Template, Measurement ou Guidelines.
Experiment Document	Representa os diferentes tipos de documentos utilizados no experimento. Pode ser de um dos seguintes tipos: Experiment Organization, Result Forms ou Instrumentos.
Experiment Organization	Representa os conceitos relacionados à descrição dos documentos de um experimento.
Experiment Package	Representa o pacote contendo as informações e documentos relativos ao experimento ou ao conjunto de experimentos.
Experiment Performers	Representa as pessoas que executam os experimentos. Eles podem ser de três tipos: Profissional, Student ou Industrial Developer.

Experimental Design	Descreve informações sobre o estudo experimental, os participantes, os artefatos de software, mecanismos do processo, requisitos e hipóteses, projeto experimental, variáveis dependentes e independentes, questões e métricas, questões não respondidas, questões em aberto e caminhos de validação.
Experimental Designer	Define, projeta e executa um novo experimento.
Experimental Procedure	Descreve os procedimentos e ações que devem ser realizados para execução de um experimento.
Experimental Raw Results	São os resultados crus, que podem estar armazenados em uma planilha ou em um banco de dados.
Experimental Refined Results	Descreve os dados analisados provenientes dos resultados crus e que são sintetizados em Aggregated Results.
Final Report	Explica o método utilizado no experimento, bem como a técnica utilizada e o que pode ser melhorado.
Function	Descreve as funções utilizadas em um artefato de software para produzir um experimento.
General Reader	Aprende ou se atualiza em técnicas de experimentos ou em um certo experimento.
Glossary	Descreve os termos e conceitos utilizados no contexto de um experimento específico. Pode também ser utilizado para genericamente descrever documentos, com uma pequena introdução.
Guidelines	Descreve diretrizes como processos, *checklists* e métodos usados em um experimento.
Industrial Developer	Descreve os profissionais da área industrial que podem participar de um determinado experimento.
Instruments	Descreve os instrumentos usados em um experimento. Podendo ser de dois tipos: Software Artifact ou Experiment Artifact.
Interface	Descreve as interfaces utilizadas em um artefato de software para produzir um experimento.
Librarian Coordinator	Tem como função coordenar as outras Bibliotecas.
Local Theme Librarian	Descreve o bibliotecário com responsabilidades por uma determinada localidade.
Master Librarian	Estabelece e mantém os documentos padrões do pacote.
Master Librarian Secretary	Tem como função ajudar o Master Librarian em suas atividades.
Measurement	Descreve as métricas usadas para validar os dados coletados de forma manual ou através de entrevistas.
Observational Note Template	È um *template* utilizado para descrever as observações relativas a um experimento.
Observational Notes	Descreve observações sobre o experimento.

Open Questions	Descreve as perguntas que representam futuras perspectivas de pesquisa.
Partner Description	Descreve informações sobre os parceiros que realizam um experimento.
Practitioner	Representa uma pessoa interessada em saber que tipos de tecnologias foram validadas empiricamente. Esta pessoa não está interessada em replicar um experimento, mas precisa ter livre acesso aos resultados de um experimento, avaliando se estes resultados preenchem os requisitos do seu ambiente de desenvolvimento.
Professional	Descreve os profissionais da indústria que fazem parte da classe de Engenheiros de Software.
Program Text	Descreve os programas utilizados em um artefato de software para produzir um experimento.
Questionnaire Template	È um *template* utilizado para descrever as questões de um experimento.
Questionnaires	Descreve as questões aplicadas através do experimento.
Questions Cannot Answer	Descreve as questões que não se consegue responder apenas realizando o experimento, mas que seria bastante interessante poder respondê-las, como por exemplo: não podemos ter o caso ideal, é impraticável testar isto, etc...
Readme Reminders	Descreve as informações freqüentemente perguntadas sobre o experimento.
Replicator	Responsável por repetir um experimento.
Report	Descreve informações sobre os resultados de um experimento através de relatórios.
Requirement Documents	Descreve as características que o sistema deve ter em acordo com os requisitos do usuário. Representa as características funcionais e não-funcionais.
Researcher	Representa os Engenheiros de Software que desenvolvem pesquisa na área.
Result Form	Descreve informações sobre os documentos resultantes de um experimento.
Software Artifact	Descreve os artefatos de software utilizados em um experimento.
Software Engineer	Representa os Engenheiros de Software participantes da comunidade. Eles podem ser pesquisadores.
Software Engineering Community Members	Representa toda a comunidade de Engenharia de Software participante de um experimento.
Software Process Model	Descreve como o processo de software foi modelado.
Steering Board	É o comitê gestor dos experimentos. Supervisiona e participa dos experimentos. Tem como função controlar a qualidade do material utilizado nos experimentos e nas bibliotecas. Tem ainda como função formular o “Quality Assurance Rules” e supervisionar a evolução dos

	documentos padrões.
Student	Descreve os estudantes que podem participar de um determinado experimento.
Subject	Descreve as pessoas que participam de um experimento planejado.
System Documentation	Responsável pelas informações sobre o sistema que servem de apoio para os pesquisadores.
Template	Descreve *templates* usados em um experimento. Podendo ser de três tipos: Defect List Template, Observational Note Template ou Questionnaire Template.
Test Data Case	Descreve os casos de teste definidos para o Sistema.
Thematic Area Description	Descreve informações sobre temas gerais e sobre áreas técnicas específicas.
User Documentation	Responsável pelas informações sobre o sistema que servem de apoio para os usuários.
Web Cruiser	Descreve as pessoas interessadas em Engenharia de Software que possam estar navegando pela Internet e lendo parte do conteúdo dos pacotes.