FACULDADES INTEGRADAS DE CARATINGA

FACULDADE DE CIÊNCIA DA COMPUTAÇÃO

O REÚSO DE CÓDIGO-FONTE COMO METODOLOGIA DE APRIMORAMENTO DA PRODUÇÃO DE *SOFTWARES*

MAICON VINÍCIUS RIBEIRO

Caratinga

2013

Maicon Vinícius Ribeiro

O REÚSO DE CÓDIGO-FONTE COMO METODOLOGIA DE APRIMORAMENTO DA PRODUÇÃO DE *SOFTWARES*

Monografia apresentada à banca examinadora da Faculdade de Ciência da Computação das Faculdades Integradas de Caratinga, como requisito parcial para obtenção do título de bacharel em Ciência da Computação, sob orientação da professora Msc. Fabrícia Pires Souza Tiola.

Caratinga

2013

Maicon Vinícius Ribeiro

O REÚSO DE CÓDIGO-FONTE COMO METODOLOGIA DE APRIMORAMENTO DA PRODUÇÃO DE *SOFTWARES*

Monografia submetida à Comissão examinadora designada pelo Curso de Graduação em Ciência da Computação como requisito para obtenção do grau de Bacharel.

_____

Prof. Msc. Fabrícia Pires Souza Tiola
Faculdades Integradas de Caratinga

_____

Prof. Msc. Glauber Luis da Silva Costa
Faculdades Integradas de Caratinga

_____

Prof. Wanderson Miranda Nascimento
Faculdades Integradas de Caratinga

Caratinga, 02/12 /2013

AGRADECIMENTOS

Primeiramente agradeço a Deus, que durante esses anos não me desamparou por um segundo sequer. Ajudou-me nas decisões difíceis, foi paciente com minhas falhas e me enviou pessoas prontas a me auxiliar.

Agradeço a minha mãe que me proporcionou de muitas maneiras chegar até aqui, sendo diariamente um exemplo a seguir. Ao meu pai que me ensinou muito mais do que pode imaginar. Ao meu irmão que foi sempre muito sincero comigo e sempre zelou por mim. A minha namorada por ter sido paciente comigo tantas vezes quanto foram necessárias. A toda a minha família que foi um alicerce seguro para os momentos difíceis.

Aos meus amigos que não me abandonaram mesmo quando não pude ser o melhor dos amigos e aos meus professores que me deram valorosos ensinamentos nesses anos, principalmente aqueles que destinaram-se a serem exemplos de vida. Em especial a professora Fabrícia Pires pelas suas inúmeras contribuições para que esse trabalho se realizasse além do apoio constante durante toda a caminhada até este momento.

“Não te deixes vencer do mal, mas vence o mal com o bem.”

Romanos 12:21

RESUMO

Com o rápido avanço tecnológico das últimas décadas, sistemas computacionais tornaram-se indispensáveis. Se outrora computadores eram artigos encontrados apenas em instalações militares ou em grandes empresas, atualmente é comum que em uma residência tenha-se mais de um computador ou aparelhos celulares com capacidade de processamento considerável. Por esse motivo, os *softwares* passaram a ser produzidos em grande escala, tornando-se presentes nesses tipos de aparelhos eletrônicos.

Em diversos segmentos industriais, a palavra reusabilidade tornou-se bastante comum. Embora nesse caso o reúso tenha se tornado uma metodologia muito eficaz, a reusabilidade de *software* ainda não é muito comum tendo passado a ser preocupação recente dos fabricantes de *software.*

Pela necessidade de aprimorar os processos de produção de *software*, por meio de estudo realizado, constatou-se que a reusabilidade pode ser utilizada como método capaz de beneficiar a produção de *software*, foi então realizado um levantamento sobre o reúso de *software* e sobre como utilizá-lo.

Através de pesquisas realizadas sobre o reúso de código-fonte, notou-se que as opiniões de vários estudiosos do assunto convergem para diversas vantagens de se reutilizar *software*. Contudo, observou-se também que alguns fatores na realidade tem impossibilitado a aplicação da reusabilidade.

Visualizando a divergência entre as muitas vantagens inerentes a reusabilidade de *software* e a sua aplicação prática, foi elaborado um questionário destinado a profissionais da área de Tecnologia da Informação baseado na opinião de vários autores sobre a reusabilidade.

Por meio das respostas de 172 profissionais foi possível conhecer elementos mais relevantes capazes de possibilitar e dificultar o reúso de código-fonte na produção de *software* além de aferir a aceitabilidade do reúso como metodologia capaz de beneficiar a fabricação de *software* segundo a opinião geral dos entrevistados. Assim, constatou-se que o reúso pode ser utilizado para aperfeiçoar a produção de *software* aumentando a qualidade e confiabilidade e reduzindo o tempo de fabricação.

Palavras-chave: Reusabilidade; *Software*; Código-fonte;

ABSTRACT

With the fast technological advances of the last years, computer systems have become indispensable. If computers were once items found only at military installations or large companies, is now common in a home has become more of a computer or mobile devices with processing capacity considerably. For this reason, the software began to be produced on a large scale, making it present in these types of electronic devices.

In many industries, the word reusability has become quite common. Although in this case the reuse has become a very effective method, the reusability of software is still not very common having become a recent concern of software manufacturers.

By the need to improve the processes of software production, through study, it was found that the reusability can be used as a method to benefit the production of software, was then carried out a survey on software reuse and how to use it.

Through research on the reuse of source code, it was noted that the opinions of various scholars of the subject converge to several advantages of software reuse. However, it was also observed that some factors actually have prevented the implementation of reusability.

Viewing the divergence between the many advantages inherent reusability of software and its practical application, a questionnaire was designed for professionals in the field of Information Technology based on the opinion of several authors on the reusability.

Through the responses of 169 professionals were able to learn the most relevant elements able to enable and hinder the reuse of source code in software production beyond assess the acceptability of water reuse methodology capable of benefiting manufacturing software according to the general opinion of respondents. Thus, it was found that reuse may be used to improve the software production enhancing the quality and reliability and reducing the manufacturing time.

Keywords: Reusability, Software, Source Code;

LISTA DE SIGLAS

API – *Application Programming Interface* (Interface de programação de aplicativos)

TI – Tecnologia da Informação

XP – *Extreme Programming*

PHP – *Hypertext Preprocessor*

SOAP – *Simple Object Access Protocol*. (Protocolo Simples de Acesso a Objetos)

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1. INTRODUÇÃO

O processo de fabricação de softwares tem se revelado um processo bastante custoso. Pelo fato de que os softwares tornaram-se insumos indispensáveis para a sociedade atual, é desejável que os softwares possam ser fabricados requerendo um investimento menor resultando diretamente em queda de preço de um sistema informatizado. É cada vez mais desejável fabricar softwares de melhor qualidade com menor custo (PRESSMAN, 1995).

Atualmente os softwares são encontrados facilmente em produtos distintos. Os carros, motocicletas e aparelhos domésticos são bons exemplos de produtos nos quais há algumas décadas atrás, não se imaginava encontrar softwares. Hoje o cenário mudou drasticamente e os softwares não estão presentes apenas nos computadores ou aparelhos celulares.

A partir do momento em que objetos antes considerados simples tornaram-se mais complexos e mais tecnológicos, houve uma grande concentração de investimento no desenvolvimento de novas tecnologias. Cientistas conseguiram criar hardwares com circuitos integrados mais complexos em larga escala com custo de fabricação reduzido. Entretanto, o custo de fabricação de software segue a tendência contrária, correspondendo a um percentual cada vez maior no custo total de um sistema informatizado.

Construir softwares confiáveis, com qualidade e com custo de fabricação reduzido é um desafio para os projetistas ou engenheiros. Os usuários esperam que os softwares sejam inteligentes, tenham um visual simples e agradável, e que estejam direcionados às suas necessidades. Segundo Rezende (2005), o processo de fabricação de um software, é complexo e demanda tempo, investimento, criatividade, concentração e mão-de-obra de várias pessoas. É seguro afirmar que os softwares não são apenas ferramentas dispensáveis ou alternativas, mas ferramentas fundamentais, sobretudo do ponto de vista econômico. Portanto, produzir softwares confiáveis com qualidade e custos de fabricação bem administrados torna-se cada vez mais importante (SOMERVILLE, 2007).

No atual contexto, esforços no sentido de aperfeiçoar a produção de softwares são extremamente aprazíveis. Por meio desse estudo, foi possível

compreender quão vantajosa a reusabilidade pode ser na produção de softwares, além de determinar fatores capazes de auxiliar ou dificultar a aplicação do reúso na prática.

Para a aplicação da reusabilidade fazem-se necessárias algumas mudanças nas metodologias empregadas durante o processo de fabricação do software desde a fase inicial do projeto no momento de definir o problema a ser resolvido com a criação de um sistema, até o processo final de manutenção ou melhoria do sistema. Isso implica em várias mudanças, o que significa que empregar a reusabilidade não é uma tarefa simples. A fim de servir de auxílio às empresas fabricantes de sistemas, este estudo deseja conhecer algumas metodologias de fabricação de softwares que sejam voltadas à reusabilidade ou que possam simplificar a aplicação da reusabilidade como metodologia orientada a aperfeiçoar a produção e melhorar os softwares fabricados.

Visualizando o cenário atual de produção de *softwares* esse estudo visou descobrir se a reusabilidade é eficaz para aprimorar a fabricação de *software* de maneira geral. Tendo sido realizado um estudo sobre a reusabilidade, foi aplicado um questionário composto de 27 questões sobre o assunto a profissionais relacionados à produção de *software.*

Por meio das respostas obtidas foi possível realizar comparações entre a opinião dos entrevistados e o que foi escrito por diversos escritores sobre o assunto. Ainda foi possível conhecer como a reusabilidade é aplicada diariamente. Foram obtidas 172 respostas e na maioria das respostas foi perceptível a aceitabilidade dos profissionais em relação a reusabilidade, além da convergência das diversas opiniões sobre o assunto, afirmando as diversas vantagens do reúso de *software*, ferramentas e metodologias capazes de facilitar a reusabilidade e os problemas intrínsecos a seu uso.

Primeiramente será necessário compreender todo o contexto de fabricação de softwares atualmente, verificar como empregar o reúso de código-fonte identificando fatores que possam dificultar ou facilitar essa prática. As próximas seções desse estudo tratam dos temas citados anteriormente além de toda metodologia empregada no estudo.

2. REFERENCIAL TEÓRICO

2.1. O QUE É UM *SOFTWARE*

Se na década de 1970 fosse realizada uma pesquisa entre cidadãos comuns, uma parcela muito pequena dos entrevistados seria capaz de definir o que é um *software*. Hoje esse percentual seria muito maior, porque os *softwares* estão presentes em locais comuns no dia-a-dia das pessoas. Um *software* pode ser definido como instruções que quando executadas são capazes de fornecer características, funções e desempenho desejado. Poderia ser definido também como uma estrutura de dados que permite aos programas manipular de maneira adequada certas informações ou ainda como documentos que descrevem a operação e uso de programas (PRESSMAN, 2006).

Definir um *software* apenas como um programa de computador ou um aplicativo para um aparelho celular é uma maneira injusta de defini-lo. Um *software* pode ser visto não apenas como um programa de computador, mas também como o conjunto de toda sua documentação e dados de configuração que façam com que o sistema funcione corretamente. Segundo Sommerville (2004), os *softwares* podem ser classificados quanto a sua finalidade em dois grupos, os produtos genéricos e os fabricados por encomenda. Como descrito a seguir:

- Os produtos genéricos são *softwares* criados para atender necessidades gerais de um grande número de usuários. Geralmente os *softwares* que se enquadram nesse grupo são fabricados e colocados à venda para qualquer usuário que deseje adquiri-lo, como um *player* de música ou um editor de texto que pode ser usado por usuários muito diferentes entre si.
- Existem também os *softwares* fabricados por encomenda. Esses por sua vez, são fabricados com o objetivo de atender às exigências de um grupo menor de usuários, como por exemplo, um *software* que gerencie as transferências financeiras de uma rede de agências bancárias ou um *software* que auxilie nas vendas de passagens de uma companhia aérea.

A diferença fundamental entre os dois grupos está nos requisitos. Os *softwares* genéricos têm seus requisitos especificados pelos próprios fabricantes dos sistemas, os *softwares* feitos por encomenda são fabricados de modo a atender aos requisitos especificados pelos usuários que os encomendaram (SOMMERVILLE, 2004).

A definição de *software* pode parecer bastante simples, entretanto, é fundamental que a conheçamos para compreender como um *software* é fabricado. Embora seja possível separar os *softwares* em apenas dois grupos, isso influencia em algumas mudanças durante o processo de fabricação. Portanto, essas mudanças serão consideradas, mas não serão objetivos desse estudo.

2.2. A PRODUÇÃO DE *SOFTWARE*

Para compreender como um *software* é fabricado, é importante conhecer características que o fazem diferente de outras coisas produzidas pelos seres humanos e mais simples de serem compreendidas. Um *software* é um elemento lógico e não físico o que o difere essencialmente de outras coisas fabricadas como tijolos que compõem uma casa ou até mesmo o *hardware* de um computador. Um *software* passa por um processo de engenharia, não é construído no sentido clássico. Ou seja, os *softwares* não são palpáveis. Embora os processos de fabricação de *softwares* dependam de pessoas assim como na produção de *hardware*, na produção de um *software* as pessoas investem seu intelecto e não apenas a mão-de-obra. Um *software* não se desgasta como objetos físicos sendo correto afirmar que um *software* não se desgasta por efeito climático, de tempo ou de uso como um *hardware* (PRESSMAN, 2006).

Não há apenas um método de fabricar *softwares*. Todavia, os métodos usados mais frequentemente por empresas fabricantes costumam ser parecidos. Em sua obra, Davis (1994) propôs um método de fabricação de *software* dividido em sete fases, sendo: Definição do problema, estudo de viabilidade, análise, projeto do sistema, projeto detalhado, implementação e manutenção. Conforme descrito a seguir:

- Ao final da fase de definição do problema, o analista ou o responsável deve ser capaz de responder qual é o problema a ser tratado e elaborar uma declaração dos objetivos e delimitação do problema.
- O estudo de viabilidade deve ser uma tarefa relativamente curta, não tem o objetivo de resolver o problema, mas de investigar o problema e saber se vale a pena resolvê-lo através de um sistema. Deverá ser feita uma análise geral da relação entre custo e benefício.
- A fase de análise parece ser a mais lógica. Ela visa determinar o que precisamente deve ser feito para resolver o problema em questão. Nessa fase deverá ser criado o modelo lógico do sistema, com diagrama de fluxo de dados, dicionário de dados e algoritmos.
- Sabendo o que deve ser feito para resolver o problema, na fase de projeto do sistema deverá ser planejada em forma de amplo contorno como será possível fazê-lo apresentando soluções alternativas.
- Uma vez definida de maneira geral como deverá ser feita a solução para o problema, a fase de projeto detalhado deverá descrever de maneira minuciosa como o sistema deverá ser construído, nessa fase deverão ser detalhadas as especificações de implementação, especificações de *hardware*, estimativas de custo, plano de teste preliminar e a programação de implementação.
- Feitas todas essas análises é hora de implementar. Nessa fase o sistema é fisicamente criado, os programas necessários serão codificados. Ao fim dessa fase, deverão estar prontos os códigos e a documentação, deverão ser feitos os procedimentos operacionais, procedimentos de segurança, procedimentos de auditoria e plano de teste. Então deverá ser feito o teste formal do sistema.
- A última fase denominada manutenção tem a finalidade de manter o sistema funcionando corretamente. Eventualmente alguns erros poderão não ser encontrados na fase anterior, logo, deverão ser corrigidos durante a manutenção, nessa fase deve-se dar apoio continuado aos usuários do sistema.

O método acima proposto por Davis (1994) assemelha-se aos demais métodos presentes nesse estudo. Isso se dá pelo fato de ser um método de simples compreensão, e suficientemente abrangente para ser utilizado na produção de *softwares* destinados a finalidades diferentes.

Outra metodologia muito utilizada é dividida em apenas quatro fases: Planejamento, projeto, codificação e teste, como descrito por Pressman (2006). Essa metodologia pode ser visualizada na Figura 1:

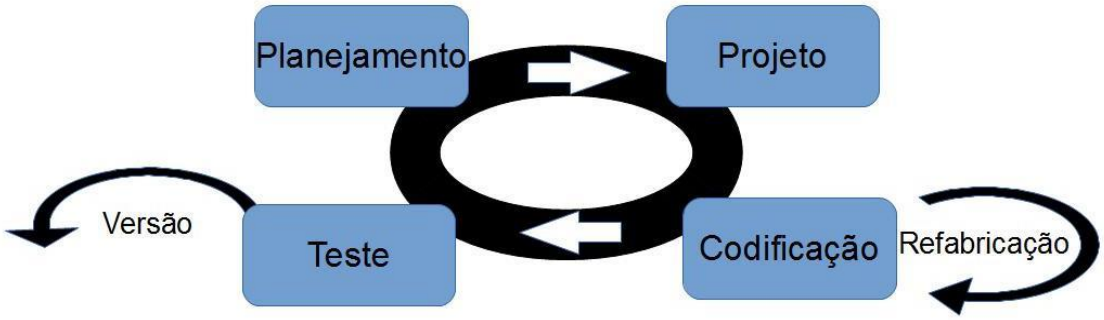

FIGURA 1: Extreme Programming (XP) (PRESSMAN, 2006)

Na Figura 1 é possível visualizar o funcionamento do método de fabricação de softwares através da interação entre as fases de produção de software (PRESSMAN, 2006):

- A fase de planejamento é destinada a documentar de maneira organizada as características e funcionalidades requeridas para software a ser produzido. Cada uma dessas características ou funcionalidades é chamada de histórias.
- Durante o momento denominado projeto, é criado um projeto simples de como alcançar os objetivos definidos através das histórias.
- No início da fase de codificação é ideal que sejam feitos testes unitários para cada história, a fim de que haja um entendimento maior do que se espera e de como fazê-lo. Depois disso, as equipes envolvidas no processo de fabricação deverão de fato criar o código-fonte do software.
- A fase de teste não se difere muito do rápido momento que se repete antes da codificação, nessa fase serão feitos testes mais detalhados e lançadas possíveis novas versões do sistema.

As duas últimas fases do método se repetem diversas vezes antes que uma versão seja disponibilizada para que o usuário a utilize. À medida que modificações

tornem-se necessárias, todo o ciclo deverá ser repetido e uma nova versão deve ser disponibilizada para os usuários (PRESSMAN, 2006).

Embora existam muitas outras metodologias de fabricação de *software*, na empreitada de exemplificar algumas entre as mais utilizadas, cabe destacar o modelo Cascata proposto por Sommerville (2004) que pode ser visualizado na Figura 2:

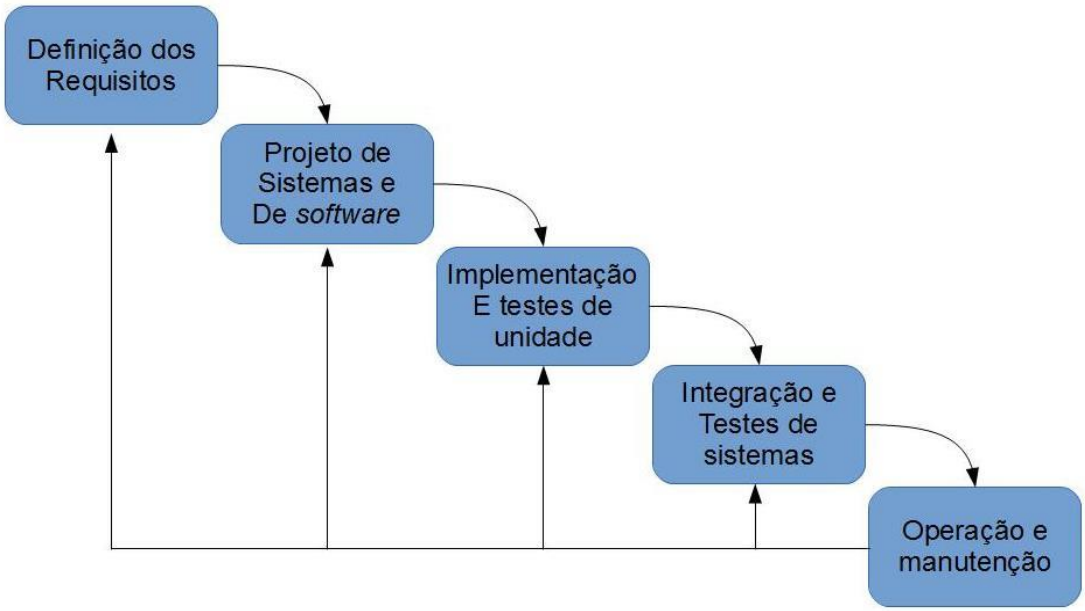

FIGURA 2: Modelo em Cascata. (SOMMERVILLE, 2004).

Na Figura 2 é possível visualizar o método Cascata, assim denominado por causa da sequência em cascata entre uma fase e outra do processo de produção de *software.* As cinco fases do método em Cascata descritas por Sommerville (2004) são detalhadas a seguir:

- Na fase de definição dos requisitos as funções, restrições e objetivos são definidos por meio de consulta aos usuários do sistema.
- O processo de projeto de sistemas de *software* envolve essencialmente a identificação e descrição das abstrações fundamentais do sistema e suas relações, e evidentemente o planejamento das atividades e processos de fabricação do sistema.
- Durante a fase de implementação e testes de unidade cada parte do sistema é subdivida em um componente que compreende uma parte menor do todo,

envolvendo além de sua própria implementação, uma série de testes específicos para cada um desses componentes.

- Na fase de Integração e testes de sistemas cada um dos componentes é acoplado aos demais, é feito com que tudo funcione em conjunto como um único programa e são realizados testes.
- A fase de operação e manutenção costuma ser a mais longa do ciclo total de fabricação, o sistema é instalado e colocado em funcionamento. A manutenção implica em corrigir possíveis erros que não tenham sido encontrados e corrigidos nos testes anteriores.

Os métodos de fabricação de *softwares* citados no texto são capazes de fornecer conhecimentos básicos sobre a fabricação de *software*. De maneira geral, muitos outros métodos utilizados em grande escala na produção de *software* assemelham-se a esses. Esse conhecimento torna-se necessário para que seja possível compreender alguns fatores relacionados aos altos custos de fabricação de sistemas.

2.3. PANORAMA ATUAL DA PRODUÇÃO DE *SOFTWARE*

Sendo os *softwares* insumos de grande importância atualmente, reduzir os custos de fabricação de *software* significa um lucro maior para o fabricante e economia para o cliente. Por isso, investimentos destinados a reduzir prazos e custos na produção de *software* são altamente desejáveis (PAULA, 2000).

Atualmente nos projetos de produção de *softwares*, falhas na gestão de prazos e custos são problemas relativamente frequentes. Um percentual muito significativo do total de sistemas que deveriam ser produzidos não é entregue dentro do prazo desejado ou até mesmo não é finalizado. Isso pode ser verificado através do Gráfico 1:

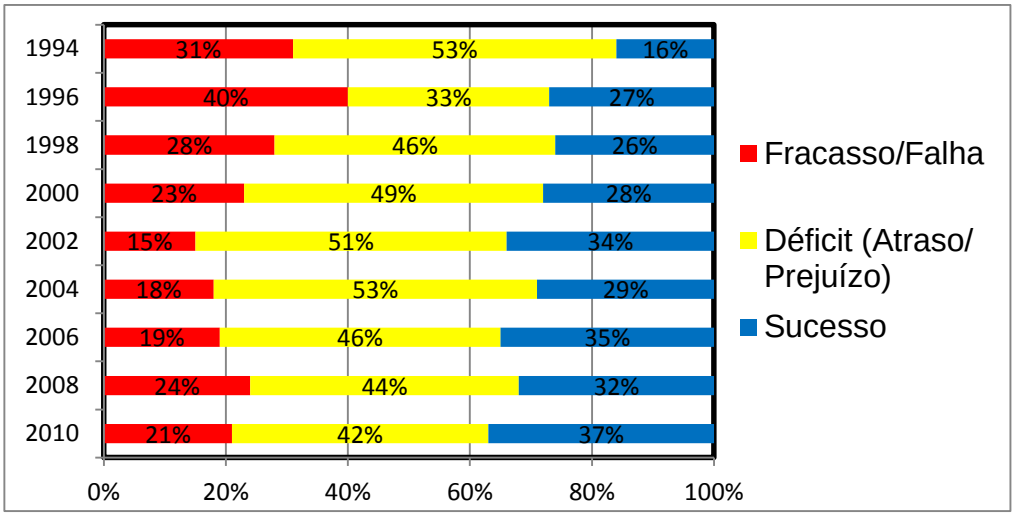

Gráfico 1 - *Chaos* Manifesto 2011 - Fonte :*The Standish Group, Chaos Summary for* 2010, *Extreme Chaos* 2001

No Gráfico 1 é possível visualizar o quadro atual da produção de *software*. Através de dados percentuais é possível perceber o alto índice de insucesso nos processos de fabricação de *softwares*. O que está caracterizado pelo pequeno percentual de projetos de *softwares* finalizados com sucesso, ou seja, dentro do prazo e do custo planejado.

Os altos custos de produção de *software* é uma das causas ligadas ao grande percentual de insucesso dos projetos de produção de *softwares*. Em sua obra, Davis (1994) mostra um gráfico capaz de demonstrar que o gerenciamento de custos nos projetos de produção de *softwares* é mais crítico na fase de implementação, o que significa dizer que a diminuição dos custos nessa fase acarretará redução do custo total de fabricação de um *software* por consequência:

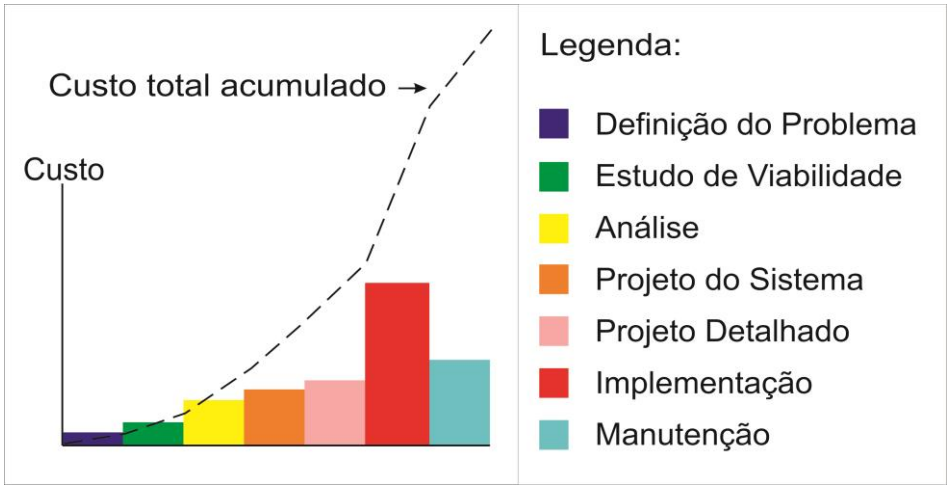

Gráfico 2 - O custo do sistema no deslocamento de uma fase para outra (DAVIS 1994)

No Gráfico 2 é notável o alto custo de fabricação concentrado na etapa de implementação, Davis (1994). O reúso de código-fonte pode não anular as fases anteriores, ou até mesmo dificultá-las pela necessidade de encontrar mudanças necessárias a serem efetuadas no código reutilizável. Entretanto, é desejável que um código reutilizável necessite do mínimo possível de modificações. Deste modo ele seria capaz de reduzir o tempo despendido nessa fase e por consequência a redução do custo total dessa fase e de todo o sistema.

2.4. VIABILIDADE DO REÚSO DE CÓDIGO-FONTE PARA REDUZIR PRAZOS E CUSTOS NA FABRICAÇÃO DE *SOFTWARES*

Os principais objetivos da reusabilidade são a qualidade, produtividade e a efetividade na fabricação e manutenção de *softwares*. Uma vez que partes criadas anteriormente e já testadas de um sistema forem reutilizadas, os processos de digitação, codificação, verificação e correção de erros tornam-se desnecessários, sendo necessário apenas o trabalho de adaptar as partes reutilizáveis ao novo sistema. Desta forma, a qualidade estará assegurada, o tempo de trabalho será reduzido e por consequência o custo de fabricação deste novo sistema será reduzido (REZENDE, 2005).

A reusabilidade permite principalmente atingir uma grande economia e um nível de qualidade satisfatório na produção de novos *softwares*. O ato de escrever código-fonte implica em aumentar os riscos de erros, uma vez que os desenvolvedores podem produzir *softwares* ou componentes com falhas. Contudo, um código-fonte produzido anteriormente e já testado, deve ter tido suas falhas identificadas e corrigidas anteriormente. Isto implica diretamente no fato de que a reusabilidade pode influir em outros fatores de qualidade importantes, tais como a correção e a robustez. Além evidentemente da redução de custo de fabricação (MAZZOLA, 2010).

Com a modernização da produção de *software* e o surgimento de técnicas de fabricação específicas para a criação de *softwares*, a reutilização tornou-se preocupação dos profissionais ligados ao desenvolvimento de *softwares*. Entretanto só muito recentemente os engenheiros de *software* puderam aplicar o reúso em larga escala (BRAUDE,2005).

2.4.1. Fatores dificultadores da reusabilidade

Embora o reúso de código-fonte possa ser visto como metodologia facilitadora da redução de custos nos processos de fabricação de *software* é necessário analisar alguns aspectos que podem torna-lo inviável. Sommerville (2007) descreve alguns fatores que podem dificultar ou inviabilizar o reúso de código-fonte:

- O custo de manutenção aumentado: estando indisponível o código-fonte ou algum componente de um sistema reutilizável, os custos serão aumentados uma vez que os elementos reusados serão cada vez mais incompatíveis com o novo sistema criado.
- A falta de apoio de ferramenta: ferramentas que auxiliam no processo de fabricação não são voltadas ao reúso, sendo difícil ou até impossível integrar essas ferramentas a um sistema de biblioteca de componentes.
- A síndrome do não-inventado-aqui: engenheiros de sistemas tendem a reescrever o código-fonte por acreditarem que são capazes de aprimorá-lo e porque a escrita de código original é vista como mais desafiadora do que reutilizar partes de códigos feitos por outras pessoas.
- A criação e manutenção de uma biblioteca de componentes: é muito custoso e demasiadamente difícil criar uma biblioteca e assegurar aos desenvolvedores que seu uso é seguro porque as atuais técnicas de recuperação de componentes de *software* ainda são imaturas.
- A necessidade de procurar, compreender e adaptar um componente reusável em uma biblioteca para uma nova funcionalidade demanda tempo, os engenheiros não terão esse trabalho sem a certeza de que encontrarão algum componente que satisfaça os novos requisitos de maneira eficaz, preferindo criar componentes novos.

Ainda, a reusabilidade de código-fonte tende a ser possível apenas se os componentes reusáveis já criados sejam em geral destinados à resolução de requisitos muito parecidos com os novos, sendo por tal motivo vista como impossível no caso de requisitos distintos entre os componentes reusáveis e os requisitos de um novo sistema em criação (SOMMERVILLE, 2007).

2.4.2. Fatores facilitadores da reusabilidade

Tudo aquilo que compõe um *software* pode ser reutilizado. Um processo, um módulo ou um objeto pode ser reutilizado várias vezes e em vários locais diferentes de um sistema. É comum que sistemas mesmo que criados com finalidades diferentes possuam atividades de validar datas, rotinas pré-montadas, esqueletos de programas como alteração, inclusão, exclusão, consulta ou impressão de informações. Entre outros, esses são exemplos de funcionalidades de um sistema que se repetem em muitos outros, portanto, podem ser reutilizáveis (REZENDE, 2005).

A reusabilidade de código-fonte pode ser uma boa alternativa para reduzir prazos e custos, entretanto é imprescindível que não haja queda na qualidade dos sistemas criados. Embora o código reutilizado já tenha sido testado anteriormente, é necessário saber com certeza se seus erros foram corrigidos, por esse motivo é necessário alto nível de maturidade e organização das equipes responsáveis pela construção de componentes reutilizáveis (SOMMERVILLE, 2007).

O custo de um componente reutilizável é conhecido, entretanto o custo de fabricação de um novo componente requer uma nova estimativa de custo, estimativa essa que será passível de erro. A padronização de componentes como a interface melhora a confiança do usuário que cometerá menos erros utilizando uma interface gráfica familiar, isso é simples de ser feito com o reúso. Com a reusabilidade a produtividade de especialistas será mais eficaz, pois fabricarão componentes reutilizáveis ao invés de fabricar componentes parecidos repetidas vezes (SOMMERVILLE, 2007).

O tempo total de produção de um *software* pode ser até mais importante que o custo total do mesmo, o reúso de código-fonte deve reduzir o tanto o tempo de desenvolvimento, quanto o tempo de validação do sistema criado (SOMMERVILLE, 2007).

Para que o reúso de código-fonte seja possível, é necessário que metodologias sejam empregadas com o intuito de reorganizar a maneira de trabalho da equipe responsável pelo desenvolvimento do sistema, desde a definição do problema até a manutenção do sistema (PETERS, 2001).

Peters (2001) apresenta alguns passos a serem observados para produzir um *software* reutilizando código-fonte:

- Estudar o problema e soluções disponíveis para ele.
- Explorar níveis de reúso reutilizando ideias e conhecimento ou reutilizando artefatos já produzidos.
- Desenvolver um plano de reúso baseado nas soluções analisadas anteriormente.
- Solucionar o problema visando o melhor ajuste com os componentes reutilizáveis de acordo com o plano de reúso.
- Adquirir, modificar os componentes reutilizáveis e desenvolver componentes necessários que não possam ser adquiridos para o reúso.
- Por fim, integrar componentes reutilizáveis com componentes novos e avaliar a funcionalidade de ambos para que o sistema funcione corretamente.

Uma das primeiras abordagens da reusabilidade para *softwares* veio com a documentação e publicação de algoritmos fundamentais por Knuth (1971). Sendo posteriormente tratada por Booch (1998), com a documentação de tipos de dados abstratos como pilhas, árvores e listas.

Atualmente existem outras técnicas, ferramentas ou métodos que embora não sejam necessariamente destinados a fabricação de *software* com reúso, podem ser muito úteis. Entre elas as mais destacam-se a programação orientada a objetos, engenharia de *software* baseada em componentes, a programação orientada a serviços, a programação orientada a aspectos, frameworks, padrões de projeto (*Design Patterns*) e *Web Services*. Como descrito a seguir:

2.4.2.1. Programação orientada a objetos

Um objeto é algo distinguível que contém atributos ou propriedades e se comporta de maneira específica. Cada objeto tem sua própria identidade e é distinguível de outro mesmo que seus atributos sejam idênticos. Um objeto possui limites nítidos (REZENDE, 2005).

Existem definições distintas sobre o que é o desenvolvimento de *software* orientado a objetos. Contudo, a ideia geral sobre o que é a orientação a objetos segundo Mazzola (2010), limita-se a:

- Uma abordagem de modelagem e desenvolvimento que tem por objetivo à construção de sistemas complexos a partir de pequenos componentes individuais;
- Uma técnica de construção de *software* condicionada a uma coleção estruturada de tipos abstratos de dados;
- Um estilo de desenvolvimento de aplicações com alto nível de abstração, com intuito de construir de forma econômica o que imita o mundo real mais fielmente;
- Uma forma de organizar o *software* como um repositório de objetos discretos que incorporam estrutura de dados e comportamentos.

A fabricação de *softwares* orientados a objetos está fortemente acoplada à abstração, encapsulamento, herança e polimorfismo (MAZZOLA, 2010).

Mais precisamente a herança e o polimorfismo são grandes auxiliadores no reúso de objetos. Através deles é possível reutilizar métodos de uma determinada classe de maneira facilmente adaptável. Nesse caso, é possível obter um maior nível de reúso de modo abstrato. Ou seja, é possível utilizar um método sem compreender detalhadamente seu funcionamento (PAULA, 2000).

2.4.2.2. Engenharia de *software* baseada em componentes

A Engenharia de *Software* Baseada em Componentes traz a possibilidade de criar componentes de *softwares* de maneira funcional e lógica e que possuam interfaces capazes de estabelecer a comunicação entre componentes diferentes.

A Engenharia de *Software* Baseada em componentes surgiu na década de 1990, para desenvolvimento de sistemas com base no reúso de componentes e *software*. Componentes de *Software* são elementos de *software* que encapsulam uma série de funcionalidades. Um componente é uma unidade independente, que pode ser utilizado com outros componentes para formar um sistema mais complexo (HEINEMAN, 2001).

Os componentes são abstrações de nível mais alto do que objetos e são definidos por sua interface. Todos os detalhes de implementação de um componente devem ser escondidos dos demais componentes e cada componente costuma ser maior que um objeto. A Engenharia de *Software* Baseada em Componentes está

fundamentada nos processos de definir, implementar, integrar ou compor componentes independentes ou pouco acoplados em sistemas (OLIVEIRA, 2009).

Para tornar os componentes reutilizáveis, através da Engenharia de *Software* Baseada em Componentes, faz-se necessário que durante a criação desses componentes sejam removidos métodos específicos da aplicação, alteração de nomes dos componentes tornando-os mais genéricos, adicionar métodos que tornem os componentes mais abrangentes, tornar o tratamento de exceções eficiente para todos os métodos, criar uma interface que facilite a adaptação do componente aos demais e integrar outros componentes para aumentar a independência. Dessa forma, o uso da Engenharia de *Software* Baseada em Componentes facilita consideravelmente o reúso dos componentes (SOMERVILLE, 2011).

2.4.2.3. Programação orientada a serviços

A programação orientada a serviços é um caminho para a criação de sistemas distribuídos nos quais os componentes desses sistemas são serviços dedicados. Os serviços podem ser executados em computadores distribuídos em lugares diferentes. Protocolos padronizados necessitaram ser projetados para apoiar troca de serviços de comunicação e de informações. Um serviço pode ser definido simplesmente como uma abstração reusável. É possível defini-lo como uma parte de software reusável, com funcionalidade genérica, que engloba a funcionalidade que pode ser distribuída e acessada por meio de programas (VALENTE, 2008).

O processo de desenvolvimento orientado pode ser basicamente descrito da forma seguinte: primeiramente é necessário definir o serviço e fornecer detalhes necessários para quem quiser utiliza-lo o faça de maneira apropriada. Posteriormente, o provedor de serviços deve publicar detalhes dos serviços de que dispõe, de modo que os interessados possam saber como agir para acessar os serviços desejados. Por fim, os interessados no serviço têm de ter alguma forma para determinar quais os serviços que satisfazem as suas necessidades (NGOLO, 2009).

Os conceitos da orientação a serviços permitem implementar aplicações sob a forma de conjuntos de serviços que interagem entre si, permitindo que outros programas ou outros componentes possam reutilizar os serviços sem a necessidade

de conhecer detalhes de seu funcionamento. Por isso a orientação a serviços tem sido usada em larga escala atualmente, a comunicação entre serviços diferentes de maneira bem definida é grande facilitadora da reusabilidade (COELHO, 2006).

2.4.2.4. Programação orientada a aspectos

Durante a produção de sistemas, surgem partes do sistema que não se enquadram como componentes funcionais, ou componentes não utilizados diretamente pelos usuários, tais como tratamento de exceções ou restrições de tempo real. Como geralmente esses pequenos componentes não-funcionais são implementados várias vezes e em lugares diferentes, é complexo separá-los das funcionalidades básicas do sistema. Esse fenômeno denomina-se entrelaçamento de código (PIVETA, 2001).

Se for possível separar tais funcionalidades das demais, será mais fácil compreender como elas agem no sistema como um todo, principalmente porque não necessário procurá-la em diferentes lugares. Ainda, será trivial analisá-la, modificá-la, estendê-la e reusá-la (CZARNECKI & EISENECKER, 2000).

Como várias funções ou métodos de um sistema costumam requerer o uso de tais funcionalidades, é necessário um alto nível de abstração para poder construí-los de maneira bem localizada com a intenção de reutilizá-los tantas vezes quanto for possível. Contudo, técnicas de produção orientada a aspectos são especialistas em desenvolver requisitos não-funcionais como exemplificados acima de maneira bastante coesa favorecendo o reúso de tais funcionalidades sem a necessidade de recriá-las (SILVA, 2006).

2.4.2.5. Framework's

Um *framework* pode ser definido como um conjunto de componentes tais como classes, métodos, funções ou até mesmo funcionalidades cooperativas que utilizam um desenho de reutilização para um tipo específico de *software*. Por exemplo, um *framework* pode ser utilizado para facilitar a construção de compiladores para diferentes linguagens de programação ou diferentes máquinas, também pode ser criada para auxiliar na construção de aplicações de gestão financeira ou páginas *Web*. Os *frameworks* podem ser personalizados para uma

aplicação em particular criando subclasses específicas a partir das classes abstratas do *framework* (COSTA, 2010).

Frameworks são atualmente utilizados em larga escala. São também grandes auxiliadores do desenvolvimento de sistemas com reúso. A possibilidade de reutilizar um componente, funcionalidade, método ou função criada primordialmente com objetivo genérico faz dos *frameworks* ferramentas muito úteis, o que é evidenciado pela facilidade de usá-los e pela quantidade relativamente satisfatória de *frameworks* disponíveis (MOORE, 2007).

2.4.2.6. Padrões de projeto (*Design Patterns)*

Um padrão de projeto descreve um problema que ocorre diversas vezes em um determinado ambiente e a sua solução. Essa descrição é feita de maneira bem organizada, possibilitando que possa ser reutilizada várias vezes sem a necessidade de ser reescrita (MADEIRA, 2001).

Padrões de projetos são utilizados com muita frequência. Segundo Johnson (1997), os padrões de projetos costumam estar presentes no desenvolvimento de *software* orientado a objetos por facilitarem a reutilização de informações de projetos.

Segundo Gamma (2002), geralmente um padrão de projeto é composto por quatro elementos essenciais: O nome do padrão, o problema, a solução e as consequências. Como descrito a seguir:

- O nome do padrão deve fornecer uma referência descritiva sobre o padrão. Desse modo, é possível compreendê-lo com um nível mais alto de abstração.
- O problema deve descrever a situação na qual o padrão deve ser aplicado. Ou seja, explica o contexto do problema.
- A solução descreve todos os elementos tais como relacionamentos responsabilidades e colaborações que compõem o padrão.
- As consequências são resultados de análises realizadas visando enxergar vantagens e desvantagens da aplicação do padrão.

Embora os padrões de projeto não sejam geralmente destinados ao código fonte de uma aplicação diretamente, eles favorecem muito o reúso. Uma vez que

determinada resolução de um problema já tenha sido aplicada anteriormente, ela pode ser reaplicada na resolução de um problema semelhante, diminuindo o risco e o tempo de desenvolvimento de nova solução (GAMMA, 2002).

2.4.2.7. *Web Services*

Um *Web Service* é um aplicativo servidor capaz de fornecer serviços a seus clientes. Geralmente um *Web Service* comunica-se com seus clientes através de um protocolo conhecido por SOAP (*Simple Object Access Protocol*. Em português, Protocolo Simples de Acesso a Objetos) (SAMPAIO, 2006).

Através dos *Web Services* os clientes são capazes de utilizar um método ou funcionalidade implementada em um servidor através da internet. Desse modo o cliente utiliza o serviço fazendo uma requisição ao servidor solicitando um serviço específico e o servidor responde fornecendo o serviço (PESSOA, 2002).

Segundo Abinader (2006), a utilização de *Web Services* fornece algumas vantagens, como descrito abaixo:

- Encapsulamento: A implementação do serviço não pode ser vista pelo lado do cliente.
- Baixo Acoplamento: A mudança na implementação do serviço não requer mudança no cliente do serviço.
- Serviço contratável: A descrição do comportamento do serviço, como se ligar a ele, quais são os seus parâmetros de entrada e quais são as suas saídas, estão publicamente disponíveis.

Os *Web Services* são capazes de permitir a colaboração entre aplicações criadas com propósitos diferentes, criadas por pessoas diferentes e dispostas em locais diferentes. Segundo Hansen (2002) esses são motivos suficientes para tornar o uso de *Web Services* conveniente na criação de aplicações.

Em termos práticos, o uso de *Web Services* acelera o desenvolvimento de aplicações, permite um alto grau de reusabilidade de serviços, e facilita o desenvolvimento de sistemas complexos (ANDREWS. Et al, 2004).

Conhecendo fatores básicos inerentes à produção de *software* tais como metodologias de fabricação, além de compreender como a produção

de *softwares* acontece na prática atualmente, deverá ser possível definir se a reusabilidade é viável durante os processos de produção. Assim sendo, fez-se necessário compreender fatores que dificultassem a aplicação do reúso na fabricação de *softwares,* bem como fatores que pudessem auxiliar essa aplicação.

Embora pareça evidente a viabilidade aplicar o reúso nos processos de fabricação de *software*, alguns fatores como os citados anteriormente podem torná-la complexa de ser realizada. Por tal motivo, este estudo deve ajudar a definir se a reusabilidade é um fator realmente vantajoso além de analisar fatores que do ponto de vista prático podem auxiliar ou dificultar a aplicação do reúso.

A próxima seção descreve a metodologia utilizada nesse estudo para alcançar os objetivos citados acima.

3. METODOLOGIA

O presente trabalho teve o objetivo de analisar as vantagens da reusabilidade na produção de *softwares*. Com essa finalidade foram realizadas pesquisas a fim de conhecer como funciona a produção de *software* e como a reusabilidade se aplica no processo de fabricação.

Após a realização do estudo sobre os processos de fabricação de *software*, através de pesquisa bibliográfica foi analisada a viabilidade de aplicação da reusabilidade de código-fonte na fabricação de *software*, além de fatores capazes de auxiliar e dificultar essa aplicação.

Foi constatado que embora existam diversos fatores que aparentemente façam da reusabilidade uma metodologia capaz de melhorar a produção de *softwares* através da redução do tempo de fabricação, redução do custo de fabricação e principalmente do aumento da qualidade do *software* produzido, existem também elementos que em casos específicos possivelmente inviabilizam-na, tornando a manutenção dos sistemas mais demorada ou mais complexa, exigindo dos profissionais um alto nível de maturidade.

Dessa forma, foi realizada por meio de um questionário (Anexo I) uma pesquisa com diversos profissionais relacionados à produção de *software* a fim de conhecer suas experiências sobre o assunto e definir a viabilidade do uso da reusabilidade como metodologia capaz de melhorar de maneira geral o processo fabricação de *softwares.* Detalhes sobre a construção do questionário serão descritos na subseção seguinte.

3.1. PÚBLICO ALVO DO QUESTIONÁRIO

Foi escolhido como público respondente do questionário, profissionais relacionados à produção de *softwares,* organizados segundo o papel deles dentro da instituição onde trabalham: Apoio Estratégico (Diretor, CEO, CIO), Apoio Gerencial (Gerente de projetos, gerente de qualidade, gerente de teste) e Apoio Operacional (Desenvolvedor, analista de sistemas, webdesigner, analista de testes).

Além destes, foram entrevistados também estudantes da área de Tecnologia da informação. Desse grupo esperava-se extrair a opinião sobre a reusabilidade gerada durante o processo de aprendizagem e eventuais construções de *software*.

Para ambos os grupos de entrevistados, o questionário foi apresentado por meio de um e-mail, contendo um endereço de acesso ao questionário ou por intermédio de postagens em *blogs* especializados em programação. Desse modo, profissionais relacionados a produção de *software* puderam conhecer o presente estudo e responder ao questionário.

3.2. ELABORAÇAO DO QUESTIONÁRIO

As questões que compuseram o questionário foram construídas fundamentadas na opinião de autores sobre o assunto, tais como Davis (1994), Gamma (2002), Mazzola (2010), Peters (2001) e Sommerville (2011). O questionário foi composto por um total de vinte e sete questões organizadas estrategicamente em três seções, visando simplificar o processo de resposta por parte dos entrevistados.

Com o objetivo de facilitar a compreensão do questionário por parte do público respondente, foi realizada uma breve introdução sobre o intuito do questionário e sobre a metodologia de criação das questões. Além de uma sucinta apresentação do autor do questionário.

Sequencialmente no questionário estão dispostas as questões divididas em três seções, Sobre você, Sobre a reusabilidade no seu dia-a-dia e Sobre como você enxerga a reusabilidade. Dessa maneira, os entrevistados visualizaram apenas uma seção de cada vez e foram direcionados para as seções seguintes de acordo com as respostas assinaladas, visando tornar o processo de resposta menos cansativo.

Cada uma das seções do questionário tem o objetivo de conhecer um tipo específico de informação. A primeira seção, denominada “Sobre você” contém questões sobre o entrevistado, como experiência profissional e formação acadêmica. A segunda seção, “Sobre a reusabilidade no seu dia-a-dia” tem o objetivo de conhecer como a reusabilidade de *software* influencia a produção de *softwares* segundo a opinião de profissionais. A terceira seção, “Sobre como você enxerga a reusabilidade” questiona a opinião dos entrevistados em relação à reusabilidade. As subseções seguintes fazem referência às seções do questionário.

3.2.1. Primeira seção: Sobre você

Nesta primeira seção estão dispostas questões que visam conhecer informações básicas sobre o entrevistado.

A primeira e quarta questão como descritas a seguir, tem o objetivo de conhecer a experiência dos entrevistados:

- 1 - Qual seu nível de formação?
- 4 - Há quanto tempo aproximadamente você trabalha com produção de softwares?

A partir dessas questões, foi possível conhecer a experiência de trabalho do entrevistado com a produção de *software*. As demais questões desta seção visa conhecer outros fatores preponderantes sobre a vivência dos entrevistados quanto a produção de *softwares*:

- 2 - Qual seu papel na instituição onde trabalha?
- 3 - Diariamente, você está envolvido com a produção de softwares de qual natureza?
- 5 - Na instituição onde você trabalha aproximadamente quantas pessoas estão envolvidas com a produção de softwares?

Estas questões por sua vez, destinam-se a traçar o perfil do entrevistado com base em informações práticas do seu dia-a-dia e consequentemente contribuem na separação entre as perguntas que lhe serão feitas nas seções seguintes.

3.2.2. Segunda Seção: Sobre a reusabilidade no seu dia-a-dia

As questões dessa seção tem o objetivo de conhecer do ponto de vista prático, como a reusabilidade acontece segundo a vivência dos entrevistados. Por esse motivo, entrevistados que não trabalham diretamente com a produção de *softwares* de modo operacional foram dispensados dessa seção.

Após pesquisa sobre profissões relacionadas à produção de *software*, foi decidido direcionar essa seção apenas aos entrevistados que trabalham diretamente com a produção de *softwares*, entre eles, desenvolvedores, analistas de sistemas, *webdesigner* e analista de testes. Desse modo, entrevistados que não se encaixaram nesse grupo foram direcionados a próxima seção.

A princípio fez-se necessário conhecer melhor o ambiente de trabalho do entrevistado. As duas primeiras questões desta seção tem esse objetivo:

- 6 - Quais paradigmas de programação você utiliza diariamente?
- 7 - Quais linguagens de programação você utiliza diariamente?

As opções de resposta da questão de número 7 foram selecionadas a partir das vinte linguagens de programação mais pesquisadas no *Google* no primeiro semestre de 2013.

As demais questões dessa seção foram concebidas baseadas nas informações coletadas durante as pesquisas relacionadas à reusabilidade, principalmente quanto aos elementos auxiliadores e dificultadores da reusabilidade. De modo geral, essas questões visam extrair do entrevistado informações sobre como o reúso de código-fonte influencia seu trabalho diariamente.

3.2.3. Terceira seção: sobre como você enxerga a reusabilidade.

As questões presentes nessa seção foram direcionadas a todos entrevistados. Foram fundamentadas na opinião de autores pesquisados sobre a reusailidade como Davis (1994), Rezende (2005), Mazzola (2010), Sommerville (2007) e Peters (2001), além de outros autores que contribuíram indiretamente na concepção do assunto de maneira geral. Conhecendo a opinião dos entrevistados, espera-se confrontar a percepção dos autores sobre o assunto e definir os pontos onde ambos concordam e discordam com a intenção de definir se a reusabilidade é afinal uma metodologia capaz de melhorar a produção de *software* de maneira geral.

Segundo Davis (1994) e Rezende (2005), o reúso de código-fonte é capaz de reduzir o tempo e o custo de criação de *softwares*. Essas afirmações são aferidas nas seguintes questões:

- 15 - Você concorda que a reutilização de código-fonte reduz o tempo necessário para a criação de uma solução?
- 23 - Como você julga o impacto da reusabilidade sobre o custo de fabricação de software?
- 25 - Em sua opinião qual impacto a reusabilidade exerce sobre o tempo de fabricação de software?

- 17 - Você concorda que reutilizar um componente é uma forma de readquirir lucro sobre o investimento de criação do software?
- 16 - Na sua opinião, qual impacto a reusabilidade exerce sobre o software produzido?

Segundo Mazzola (2010) a reusabilidade é capaz de influenciar novos *softwares* produzidos agregando qualidade, economia e confiabilidade aos mesmos. Isso foi abordado nas seguintes questões:

- 26 - Você concorda que se um software for constituído de partes já testadas é menos provável que o mesmo apresente falhas?
- 18 - Você concorda que é desejável que um software seja constituído de partes novas e partes reutilizadas funcionando em conjunto?
- 27 - Na sua opinião, qual impacto a reusabilidade exerce sobre a confiabilidade do novo sistema produzido?

Segundo Sommerville (2007), a reusabilidade é muito compensatória. Contudo ele levanta alguns elementos que podem inviabilizar seu uso. Tais elementos são analisados nas questões:

- 20 - Você acha que embora seja necessário compreender e adaptar um código-fonte reutilizável, a reusabilidade é capaz de reduzir o custo de fabricação?
- 19 - Na sua opinião, qual impacto a reusabilidade exerce sobre a produtividade nos processos de produção de software?
- 21 - Criar um componente reutilizável implica em testá-lo e documentá-lo de maneira detalhada. Você acha que o tempo investido nesses processos mantém a reusabilidade compensatória?

Além disso, o próprio Sommerville (2007) fala sobre a estimativa de custo de produção de *software,* sendo confrontado em:

- 22 - Você acha mais fácil estimar o custo de reutilizar um componente do que estimar o custo de fabricar um novo?

Peters (2001) cita alguns elementos a serem observados durante a produção de *software* com reúso de código-fonte. Tais elementos são objeto de análise nas questões:

- 24 - Reutilizar um componente, naturalmente fará com que o novo sistema se pareça com o reutilizado. Na sua opinião, a semelhança entre o sistema novo e o reutilizado é um fator positivo?
- 19 - Você acha que embora seja necessário compreender e adaptar um código-fonte reutilizável, a reusabilidade é capaz de reduzir o custo de fabricação?
- 20 - Criar um componente reutilizável implica em testá-lo e documentá-lo de maneira detalhada. Você acha que o tempo investido nesses processos mantém a reusabilidade compensatória?

As questões dessa seção tem o intuito de prover uma comparação entre a opinião de autores estudados e respondentes do questionário. Assim sendo, as questões de maneira geral foram construídas fundamentadas na opinião geral dos autores estudados.

3.3. COLETA DE DADOS

Foi utilizado o *Google Docs* para a criação de um formulário *online.* Por meio dessa ferramenta, foi possível a criação e o compartilhamento do formulário, além de possibilitar a resposta das questões através de um terminal conectado à internet.

A pesquisa foi divulgada através de um *link* capaz de permitir o acesso às questões. Esse *link* foi enviado por *e-mail* a profissionais da área de TI, alunos e ex-alunos do curso de Ciência da Computação das Faculdades Integradas de Caratinga. Além de grupos de discussão sobre desenvolvimento na internet, principalmente fóruns de contribuição sobre programação e através da rede social *Facebook* a profissionais relacionados à programação, banco de dados, redes de computadores, criação e manutenção de *hardware.*

O *link* foi disponibilizado no dia 27 de setembro de 2013 às 01h00min, ficando disponível para resposta até às 01h00min do dia 21 de outubro de 2013. Nesse período foram coletadas 172 (cento e setenta e duas) respostas.

3.4. TRATAMENTO DE DADOS

As repostas foram enviadas automaticamente para uma planilha *online* criada pelo *Google Docs*. Através dela foi possível visualizar cada resposta

separadamente. Após ser exportada como uma planilha a ser visualizada através do *LibreOffice Calc,* os dados foram organizados de acordo com o perfil do respondente. Desse modo, foi possível a análise das respostas do grupo de respondentes que se afirmaram como profissionais operacionais da área de desenvolvimento de *softwares.*

Os resultados foram exibidos através de gráficos com a intenção de facilitar a compreensão. Eles podem ser visualizados na próxima seção.

4. RESULTADOS

Nesta sessão serão apresentados os resultados obtidos por meio das respostas do questionário. Um total de 172 pessoas responderam ao questionário das quais à maioria dos entrevistados (107 pessoas, aproximadamente 63% de todas as respostas) declararam trabalhar na área operacional de suas empresas.

Para facilitar a compreensão dos resultados, as respostas serão exibidas através de gráficos. A organização das respostas bem como os gráficos serão descritos nas subseções seguintes.

4.1. DISCUSSÃO DOS RESULTADOS

Durante o processo de análise de respostas, uma das entrevistas realizadas foi considerada irrelevante ao estudo. Embora o entrevistado em questão tenha respondido de maneira coerente todas as questões propostas de acordo com a sua própria experiência de trabalho, ele não compreende o público alvo da pesquisa por ter respondido a questão sobre sua formação acadêmica como “Técnico de Gás”. Desse modo, sua entrevista foi desconsiderada no tratamento de respostas a seguir. Portanto, todas as respostas analisadas a seguir não contêm as respostas do entrevistado em questão.

Além do entrevistado citado acima, duas outras pessoas responderam o questionário sem assinalar nenhuma das opções de resposta, portanto foram desconsideradas das respostas obtidas. Assim sendo, as respostas tratadas nessa seção são referentes aos 169 respondentes restantes.

As respostas exibidas a seguir serão organizadas pelas três seções do questionário, sendo: Primeira seção: Sobre você, Segunda seção: Sobre a reusabilidade no seu dia-a-dia e Terceira seção: Sobre como você enxerga a reusabilidade.

4.1.1. Primeira seção: Sobre você

Nesta seção serão descritas as respostas obtidas na primeira seção do questionário. As respostas de cada questão serão exibidas em um gráfico próprio.

Por meio do estudo realizado, concluiu-se que diversos elementos influenciam na maneira de cada profissional produzir *softwares*. Dentre eles, a formação acadêmica do profissional, a área de trabalho, o tipo de *software* produzido, a experiência do profissional e o tamanho das equipes de desenvolvimento são os fatores mais preponderantes.

As questões dessa seção permitem conhecer o entrevistado. Elas tem o objetivo de definir o perfil do entrevistado, com perguntas relacionadas ao profissional respondente. Cada um dos subtópicos desta seção faz referência a uma questão do questionário.

4.1.1.1. Questão 01: Qual seu nível de formação?

Esta primeira questão, visou conhecer a formação acadêmica do entrevistado. Por meio dela, será possível compreender qual tipo de formação pode ser mais propiciadora da reusabilidade.

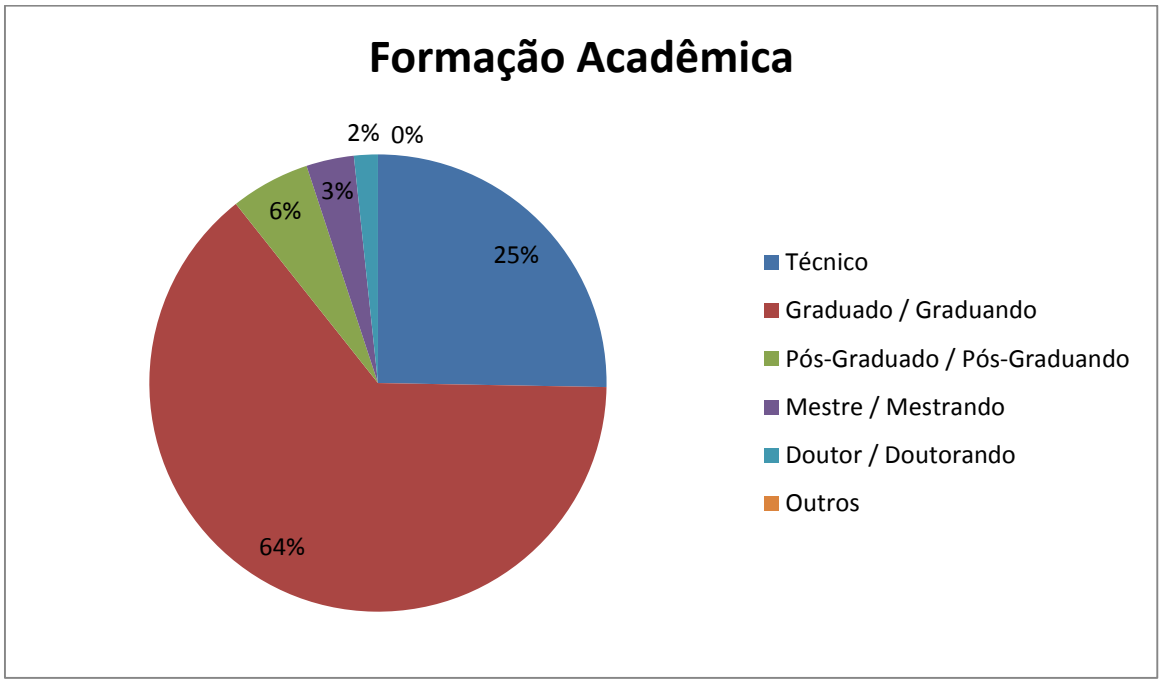

Gráfico 3 - Formação acadêmica

Aproximadamente 25% dos entrevistados, precisamente 45 pessoas declararam ter formação de cursos Técnicos. 64% declararam terem formação de graduação em andamento ou finalizada, exatamente 114 pessoas. 10 entrevistados, aproximadamente 6% são pós-graduados ou estão fazendo uma pós-graduação. 3%, ou seja, 6 pessoas são mestres ou mestrandos. 3 pessoas, cerca de 2% declararam serem doutores ou doutorandos.

Tendo em vista que um profissional pode ter mais de um tipo de formação acadêmica, foi possível ao entrevistado assinalar mais de uma resposta para esta questão.

Fundamentado nas respostas das demais questões do questionário, de modo geral, os entrevistados demonstraram um alto nível de aceitabilidade em relação a reusabilidade de *software.* Tendo a ampla maioria dos entrevistados formação de graduação já finalizada ou em andamento, é visível que esse grupo de entrevistados possui grande maturidade em relação a produção de *softwares* com reúso de código-fonte.

4.1.1.2. Questão 02: Qual seu papel na instituição onde trabalha?

Essa questão especificamente tem o papel de separar os grupos de entrevistados. Por meio da resposta dessa questão, foi possível definir quais entrevistados responderiam a segunda seção de perguntas.

Essa questão visa compreender qual a ligação real entre o entrevistado e a produção de *softwares.*

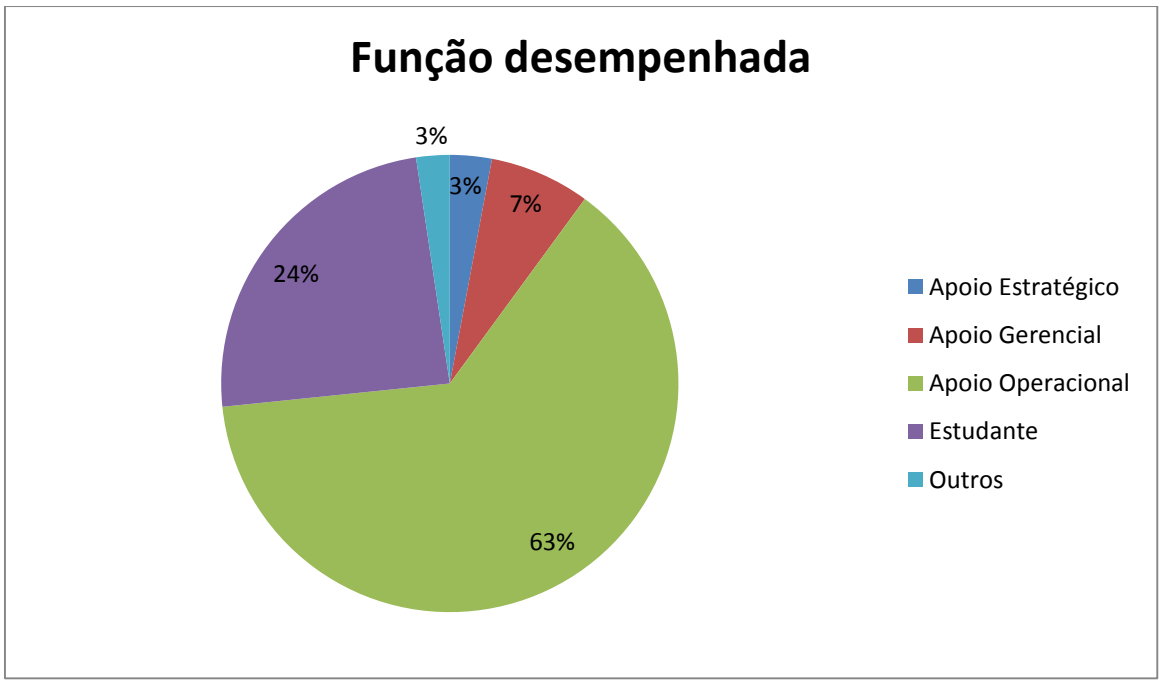

Gráfico 4 - Função desempenhada

Aproximadamente 3% dos entrevistados responderam desempenhar funções estratégicas, exatamente 5 pessoas. 12 pessoas desempenham funções gerenciais, precisamente 7%. 107 pessoas (63%) trabalham de forma operacional nas suas empresas. 24% dos entrevistados são estudantes (41 pessoas).

Os demais entrevistados, aproximadamente 3% correspondentes a um total de 4 pessoas, declararam desempenhar funções diferentes das disponibilizadas. Estes responderam: “Analista coordenador de TI”, “Elaborador e executor de projetos”, “Técnico de informática” e “Gestor de TI” como sendo suas funções desempenhadas diariamente.

Com a análise das respostas desta questão através do Gráfico 4, é seguro afirmar que o questionário utilizado no presente estudo alcançou o público desejado. Várias questões foram destinadas apenas para profissionais relacionados ao apoio operacional na produção de *software,* esse grupo de entrevistados corresponde a 63% de todos os entrevistados.

4.1.1.3. Questão 03: Diariamente, você está envolvido com a produção de *softwares* de qual natureza?

O tipo de *software* produzido influencia no processo de fabricação. Por esse motivo, esta questão visa conhecer qual tipo de *software* o entrevistado produz no dia-a-dia.

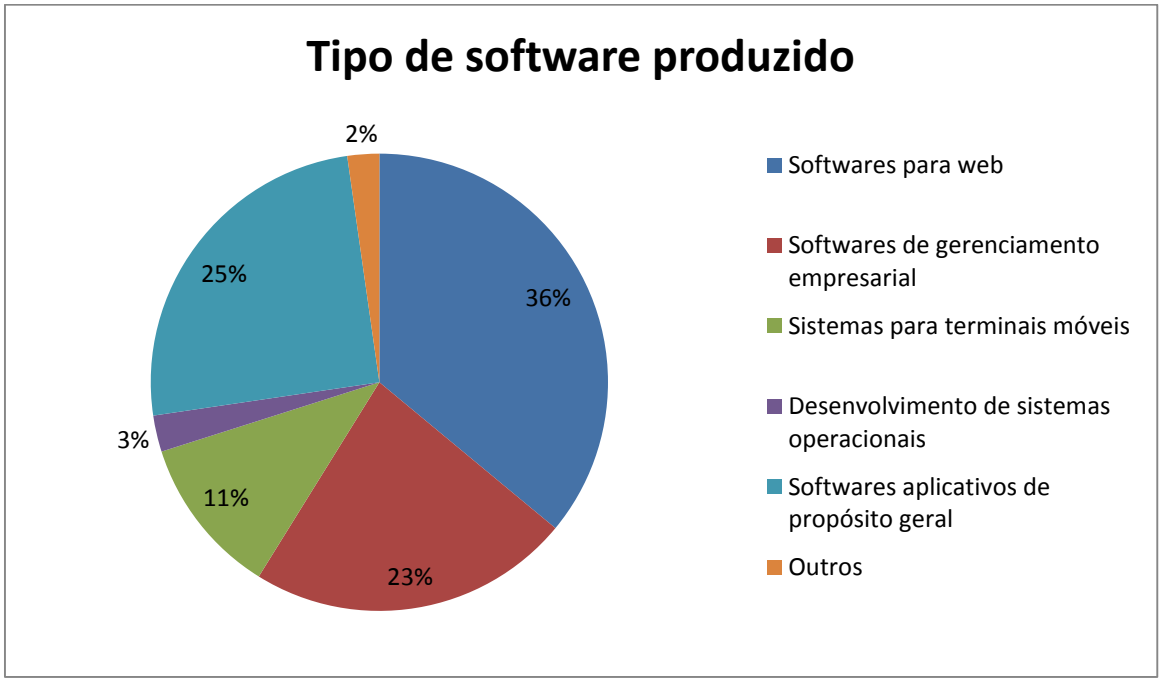

Gráfico 5 - Tipo de *software* produzido

A maioria dos entrevistados, exatamente 112 pessoas correspondendo a 36% produzem *softwares* para internet. 71 pessoas produzem *softwares* de gerenciamento empresarial, cerca de 23% dos entrevistados. 35 entrevistados produzem *softwares* para terminais móveis, correspondendo a 11% dos entrevistados. 3% dos entrevistados desenvolvem sistemas operacionais,

exatamente 8 pessoas. 78 pessoas (25%) trabalham com a fabricação de *softwares* de propósito geral.

As 7 pessoas restantes assinalaram a opção “Outros” e justificaram com “*Softwares* educacionais” e “Nenhum no momento”. Algumas pessoas assinalaram mais de uma resposta, portanto a soma das respostas dessa questão ultrapassa a quantidade total de respondentes.

De modo geral e de acordo com o Gráfico 5, é possível inferir que a reusabilidade de *software* pode ser aplicada com maior frequência na produção de *softwares* para *web,* aplicativos de propósito geral e de gerenciamento empresarial.

4.1.1.4. Questão 04: Há quanto tempo aproximadamente você trabalha com a produção de *softwares?*

Como a experiência é um fator preponderante na capacidade de produção de um profissional, essa questão visa compreender quão experiente é o entrevistado.

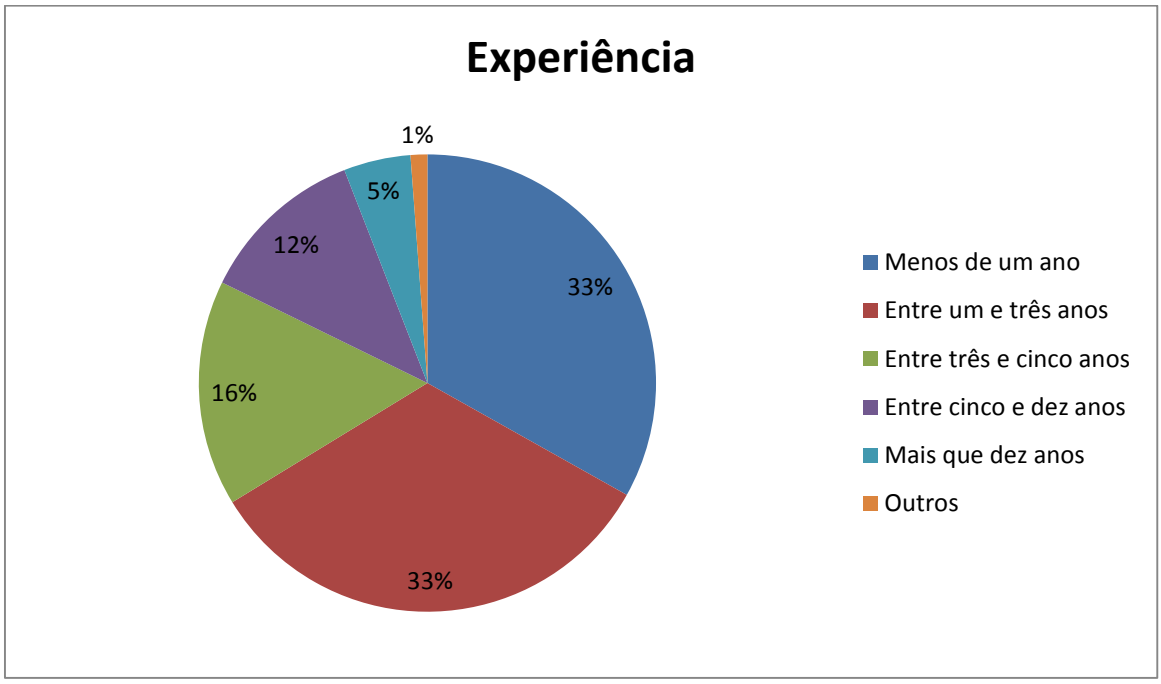

Gráfico 6 - Experiência

A maioria dos entrevistados respondeu trabalhar com a produção de *softwares* há até três anos, sendo 33% há menos de um ano e 33% entre um e três anos, ambos com 56 pessoas cada. 27 pessoas (16% aproximadamente) trabalham entre três e cinco anos.12% sendo precisamente 20 pessoas, trabalham entre cinco e dez anos. Apenas 5%, exatamente 8 pessoas, trabalham há mais de dez anos.

Outro entrevistado respondeu: “Ainda estudo para entrar no mercado” e apenas um entrevistado não respondeu a questão. Assim sendo, é possível afirmar que pelo fato de a maioria dos entrevistados tenham afirmado trabalhar com a produção de *softwares* há no máximo 5 anos, os entrevistados não possuem longa experiência embora demonstrem grande maturidade sobre o assunto em questão.

4.1.1.5. Questão 05: Na instituição onde você trabalha, aproximadamente quantas pessoas estão envolvidas na produção de *softwares?*

A quantidade de pessoas envolvidas na produção de *software* é um fator relevante pelo fato de que é possível ter mais componentes reutilizáveis e o auxílio de mais pessoas durante a produção diariamente.

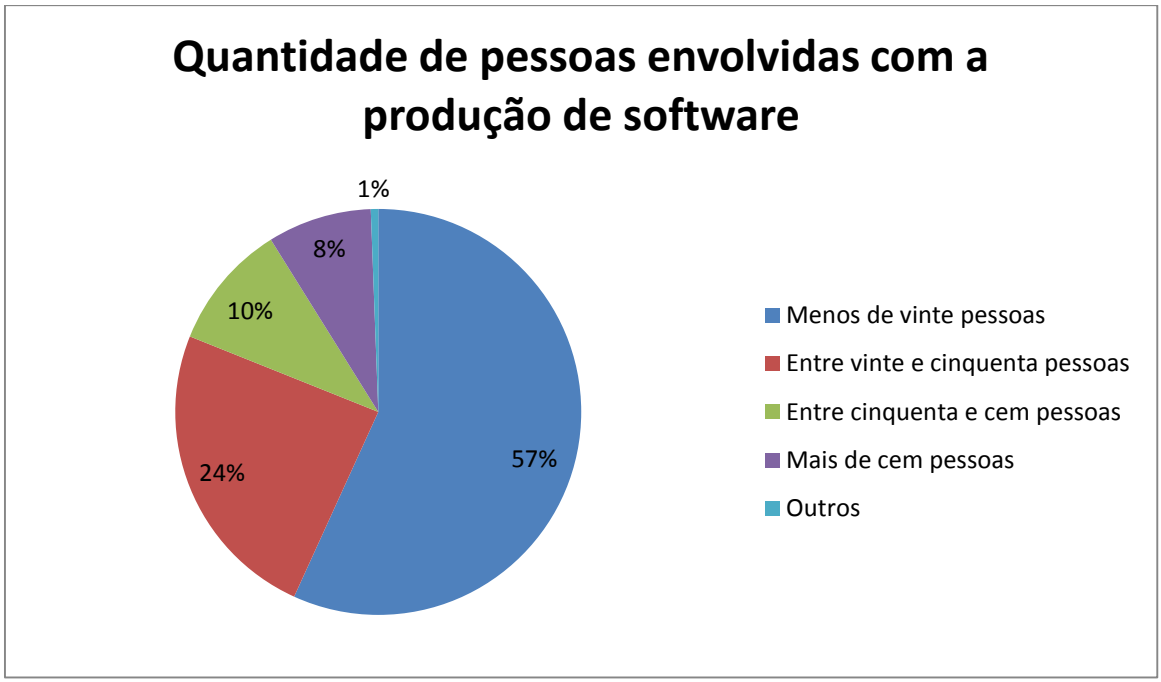

Gráfico 7 - Quantidade de pessoas envolvidas com a produção de *software*

A maioria dos entrevistados (57%, 96 pessoas) trabalham em empresas onde menos de vinte pessoas estão envolvidas com a produção de *softwares.* 41 pessoas, aproximadamente 24% responderam entre vinte e cinquenta pessoas. A opção “Entre cinquenta e cem pessoas” foi assinalada por 17 pessoas, correspondendo a 10% das respostas. Apenas 14 pessoas (8%) trabalham em empresas onde mais de cem pessoas estão envolvidas com a produção de *softwares.*

Apenas uma pessoa não respondeu à pergunta, essa foi contabilizada como a opção “Outros”.

De acordo com o Gráfico 7, é visível que a ampla maioria dos entrevistados trabalham em instituições de pequeno porte ou que possuam pouco pessoal envolvido com a produção de *softwares*. Contudo, não é possível assegurar que isso influencia na propensão de um profissional a reutilizar código-fonte.

O principal objetivo desta seção do questionário era conhecer o perfil dos entrevistados. De acordo com os gráficos das respostas, visualizou-se que a maioria dos respondentes são profissionais com graduação finalizada ou em andamento, desempenham funções operacionais onde trabalham, desenvolvem *softwares* com finalidades diversas, possuem pouca experiência e trabalhem em médias empresas. Visivelmente, o questionário alcançou o público desejado, composto por profissionais que lidam com o desenvolvimento de sistemas diariamente.

4.1.2. Segunda Seção: Sobre a reusabilidade no seu dia-a-dia

Nesta seção serão descritas as respostas das questões presentes na segunda seção do questionário. Apenas os entrevistados que assinalaram a opção “Apoio Operacional” na segunda questão do questionário (um total de 107 pessoas, correspondendo a 63% dos entrevistados), onde perguntava-se sobre o papel desempenhado pelo entrevistado no local de trabalho, responderam essa seção.

Cada uma das subseções seguintes trata de uma questão específica do questionário.

4.1.2.1. Questão 06: Quais paradigmas de programação você utiliza diariamente?

De acordo com o estudo realizado, concluiu-se que a facilidade de reutilizar código-fonte é influenciada pelo estilo de desenvolvimento utilizado. No gráfico a seguir serão visualizadas as respostas dos entrevistados.

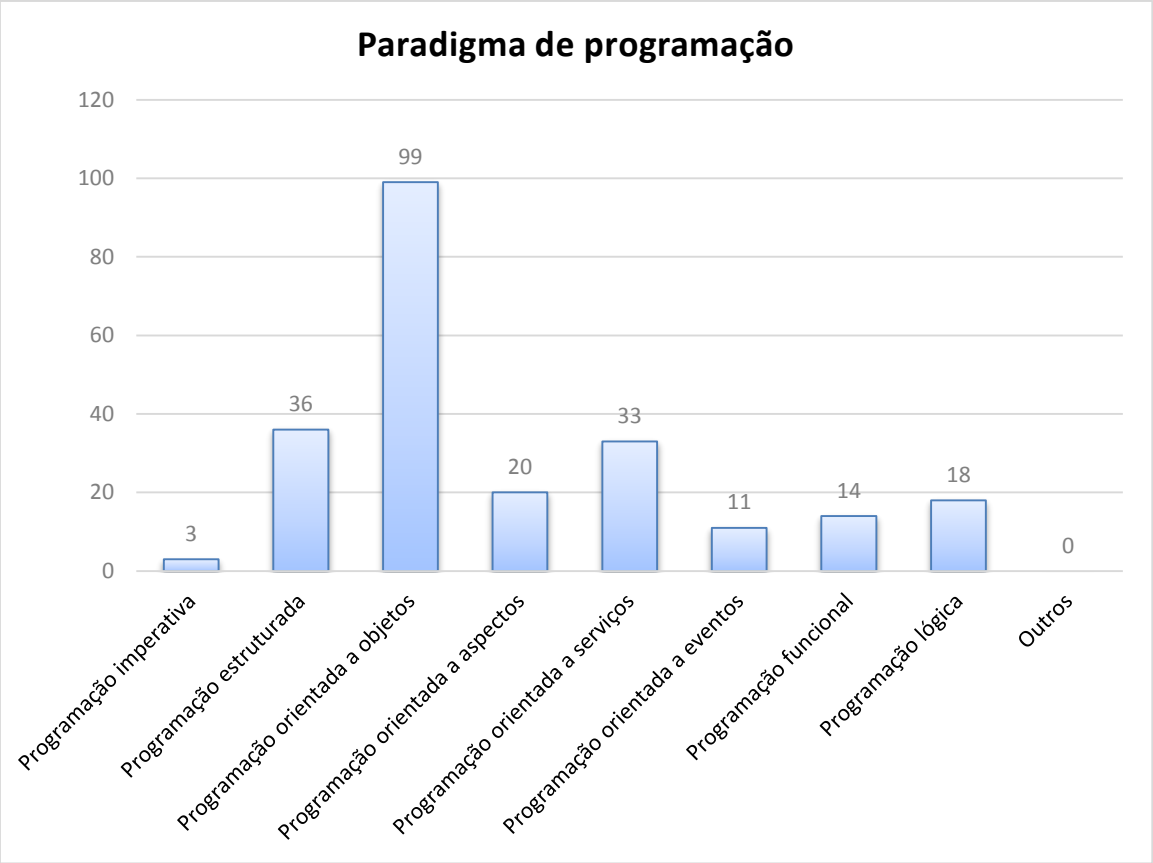

Gráfico 8 - Paradigma de programação

Diariamente, um desenvolvedor pode utilizar mais de um paradigma de programação. Portanto, permitiu-se que os entrevistados, selecionassem mais de uma opção de resposta para esta questão.

A maioria dos entrevistados, aproximadamente 42% utilizam a programação orientada a objetos. Assim sendo, é visível que a programação orientada a objetos é muito propícia à reusabilidade de *software* de acordo com as respostas dos entrevistados, concordando diretamente com a opinião de Paula (2000).

4.1.2.2. Questão 07: Quais linguagens de programação você utiliza diariamente?

Para essa pergunta, foi possível ao entrevistado escolher mais de uma resposta. Por esse motivo, a soma de linguagens de programação assinaladas pelos respondentes ultrapassa a quantidade de pessoas que responderam a esta questão.

As respostas dessa questão são visualizadas no Gráfico 9:

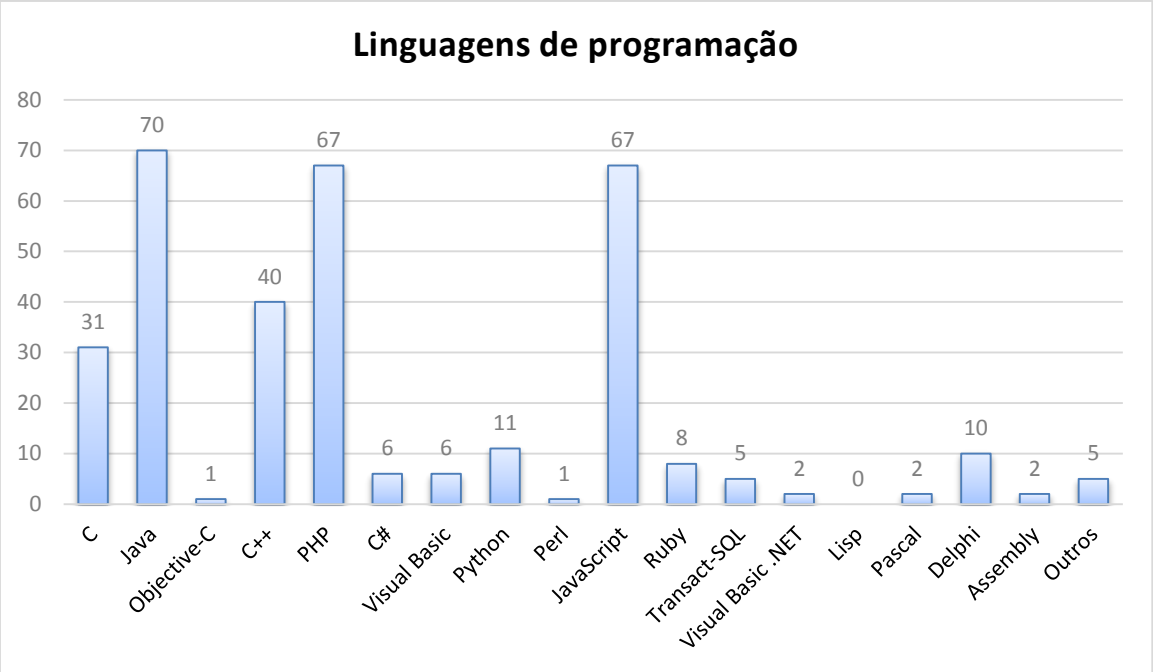

Gráfico 9 - Linguagens de Programação

A maior parte dos entrevistados assinalaram as linguagens Java, PHP e JavaScript como as principais linguagens de programação utilizadas diariamente.

Apenas cinco entrevistados responderam que trabalham diariamente com outras linguagens de programação, tendo especificado as linguagens: Lua, CoffeeScript, Clipper, Groovy e ASP como linguagens utilizadas.

Embora de acordo com as respostas obtidas por meio do questionário tenha sido visualizado um alto nível de aceitabilidade em relação a reusabilidade, não é possível aferir o quanto as linguagens de programação mais utilizadas pelos entrevistados. As linguagens de programação Java, PHP e JavaScript foram assinaladas por grande parte dos entrevistados e de maneira geral correspondem a respostas com alto nível de aceitabilidade em relação a reusabilidade. Ainda assim, seria necessário um estudo mais específico no sentido de aferir a influência das linguagens de programação em relação a reusabilidade.

4.1.2.3. Questão 08: De maneira geral, com qual frequência você reutiliza código-fonte?

Esta questão, bem como as seguintes desta sessão, tem o objetivo de compreender como o entrevistado julga sua propensão a reutilizar código-fonte de maneira geral.

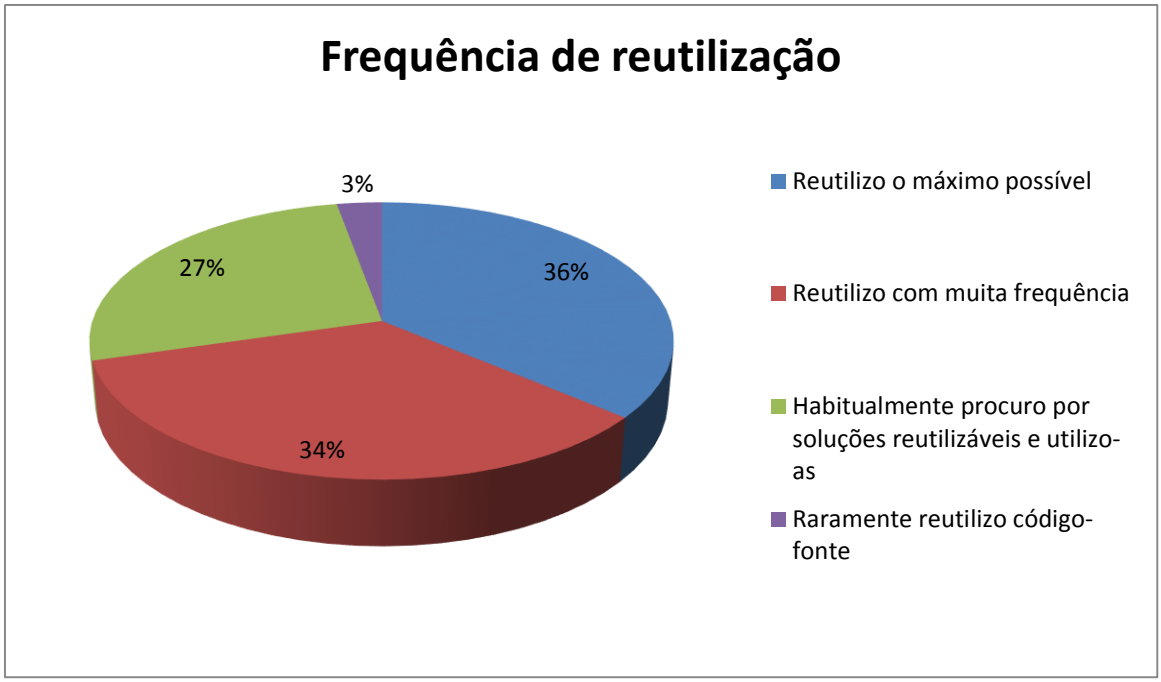

Gráfico 10 - Frequência de reutilização

A partir das respostas coletadas nessa questão, é possível concluir que a maioria dos entrevistados tende a reutilizar código-fonte o máximo possível ou com muita frequência. Separadas, essas duas alternativas correspondem a 36% e 34% de todas as respostas respectivamente. Somadas correspondem a 70% de todas as respostas.

Outros 27% dos entrevistados responderam que habitualmente procuram e utilizam soluções reutilizáveis. Apenas 3% dos entrevistados declararam jamais reutilizar código-fonte e uma pessoa não respondeu a questão.

Assim sendo, é visível que o reúso de código-fonte é objetivo da ampla maioria dos entrevistados durante o processo de desenvolvimento diariamente. Naturalmente, profissionais desejosos de reutilizar código-fonte em maior quantidade possível, tendem a fazê-lo de maneira mais aprimorada, resultando em maior eficiência na reutilização.

Embora, Heineman (2001) e Gamma (2002) tenham afirmado que a reusabilidade era preocupação recente dos projetistas de *softwares*, de acordo com a opinião dos entrevistados, esse cenário mudou. Atualmente, a maior parte dos profissionais relacionados a produção de sistemas tendem a reutilizar *softwares* com muita frequência.

4.1.2.4. Questão 09: Ao construir um componente de *software* você se preocupa em fazê-lo de maneira que facilite a reutilização posterior do mesmo?

Esta questão visa confrontar o grau de preocupação do entrevistado em criar componentes que sejam facilmente reutilizados posteriormente. As respostas são exibidas a seguir.

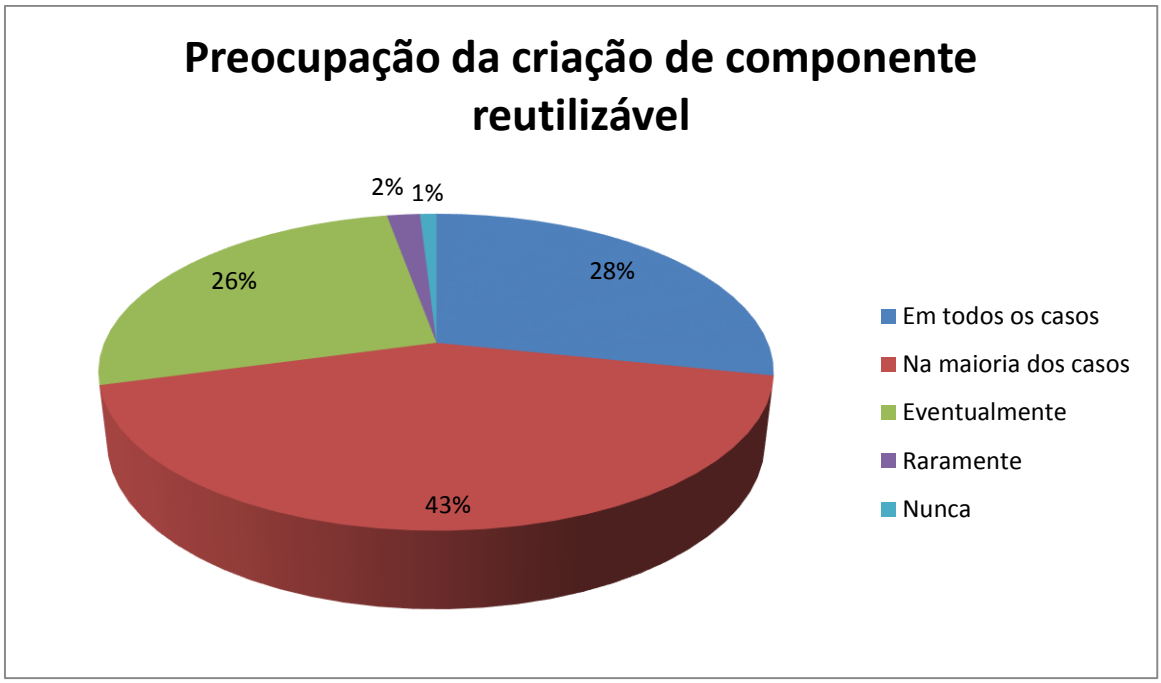

Gráfico 11 - Preocupação da criação de componente reutilizável

Grande parte dos entrevistados, aproximadamente 28% declarou preocupar-se com reutilização dos componentes criados em todos os casos. A maioria declarou preocupar-se na maioria dos casos, cerca de 43%. Outro grande grupo de respondentes, 26% assinalou preocupar-se eventualmente. Apenas 2% e 1% afirmaram preocupar-se raramente e nunca respectivamente. Uma pessoa não respondeu essa questão.

Por intermédio das respostas desta questão, é possível inferir que grande parte dos entrevistados preocupa-se em criar componentes que possam ser reutilizados posteriormente. Juntas as respostas em que os entrevistados se preocupam em todos os casos e na maioria dos casos somam 71% de todas as respostas.

A criação de componentes reutilizáveis é um processo complexo, o interesse em fazê-lo de forma cuidadosa tende a elevar a qualidade dos componentes criados.

O alto índice de profissionais com esta preocupação aponta um nível de maturidade elevado dos profissionais entrevistados no que refere-se a criação de componentes reutilizáveis com qualidade. Profissionais desinteressados na reutilização posterior de seus códigos-fonte, tendem naturalmente a serem displicentes quanto a criar componentes que sejam mais facilmente reutilizáveis.

4.1.2.5. Questão 10: De maneira geral, você julga mais fácil implementar uma funcionalidade do que procurar uma solução reutilizável?

Esta questão especificamente, objetiva compreender a opinião dos entrevistados em relação à facilidade de aplicar o reúso diariamente ou criar um componente sem reutilizar um componente criado anteriormente.

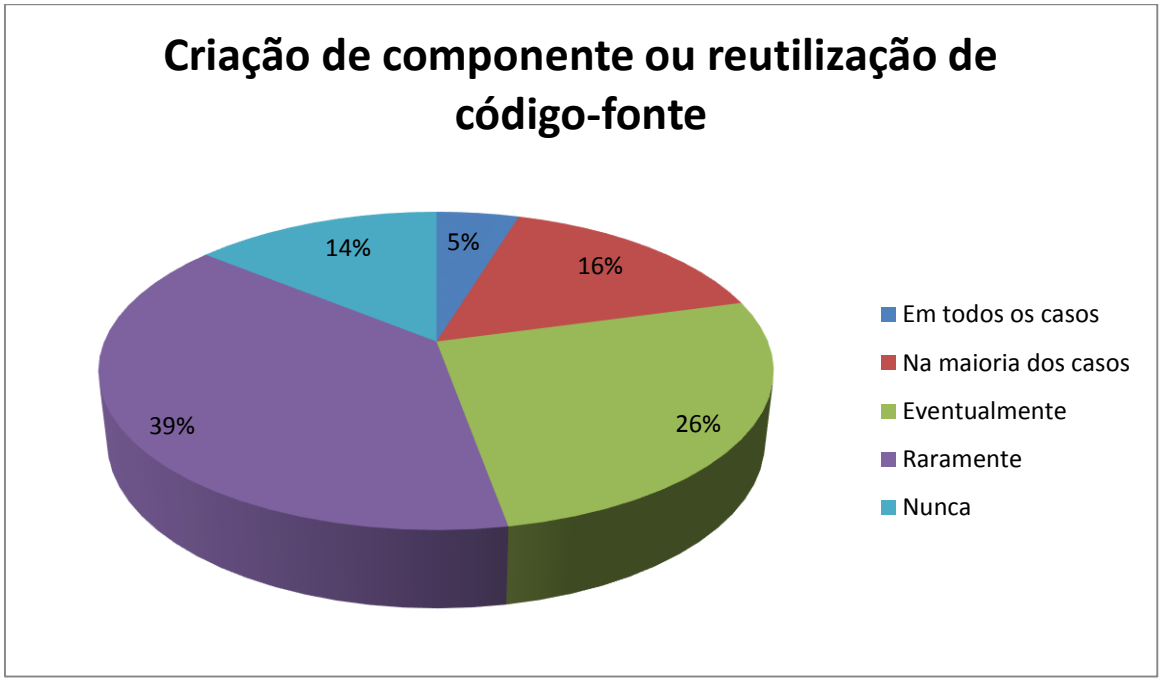

Gráfico 12 - Criação de componente ou reutilização de código-fonte

Apenas 5% dos entrevistados afirmaram que em sua opinião é mais fácil criar um componente do que reutilizar um componente já criado em todos os casos. Outros entrevistados, aproximadamente 16% responderam que isso é mais fácil na maioria dos casos. Seguido de 26%, 39% e 14% em Eventualmente, Raramente e Nunca respectivamente.

Embora as respostas dessa questão tenham sido mais acirradas, é possível inferir que a maioria dos entrevistados entende que é mais fácil reutilizar um

componente criado anteriormente para a criação de um novo do que criar um novo componente sem o reúso de código-fonte. Isso pode ser inferido através das respostas de Raramente e Nunca que somam 53% das respostas.

O fato de grande parte dos entrevistados assinalarem que tendem a procurar uma solução reutilizável ao invés de criar uma nova, comprova o alto índice de aceitabilidade ao reúso. Embora Sommerville (2007) tenha citado a necessidade de compreensão e adaptação de código-fonte reutilizável como um fator dificultador da reusabilidade, aparentemente isso não tem sido empecilho para os profissionais entrevistados. Não obstante à essa necessidade, os entrevistados ainda preferem utilizar soluções prontas à criar novas.

4.1.2.6. Questão 11: Entre as dificuldades de reutilizar código-fonte criado por outra pessoa, na sua opinião, quais mais atrapalham?

Do ponto de vista prático, por intermédio desta questão foi possível determinar quais elementos dificultam mais a aplicação do reúso. Foi possível ao entrevistado selecionar mais de um elemento prejudicial à reusabilidade em sua opinião.

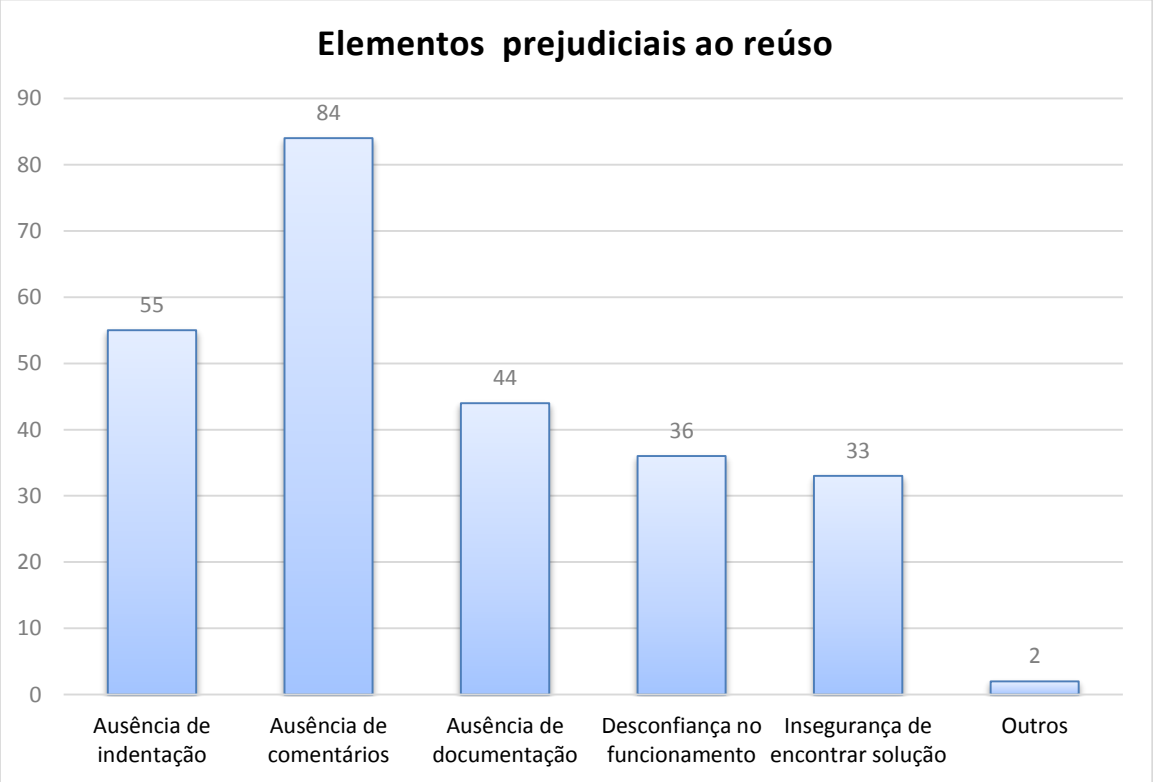

Gráfico 13 - Elementos prejudiciais ao reúso

A maior parte dos entrevistados (84 pessoas) declarou que a ausência de comentários no código-fonte é o principal elemento a dificultar a aplicação do reúso. O segundo elemento, escolhido por 54 pessoas é a ausência de indentação. Seguido pela ausência de documentação assinalada por 44 pessoas, desconfiança do funcionamento selecionado por 36 pessoas e por 33 pessoas a insegurança em encontrar uma solução reutilizável eficiente.

Dois entrevistados selecionaram a opção Outros, tendo especificado a ausência de padrão utilizado e a infrequência de atualização do código-fonte nas bibliotecas de componentes, como elementos capazes de prejudicar a aplicação do reúso diariamente.

É visível por meio do gráfico que a inexistência de comentários e indentação no código-fonte dificultam muito sua reutilização. Em contrapartida, a criação de componentes de maneira melhor organizada tende a tornar o reúso menos complexo. Uma vez que grande parte dos entrevistados tenham afirmado preocupar-se em criar componentes reutilizáveis, ainda é necessário que mudanças sejam adotadas a fim de possibilitar a aplicação da reusabilidade em maior escala.

4.1.2.7. Questão 12: De maneira geral você acha possível reutilizar um componente criado primordialmente com uma finalidade muito diferente da atual?

Apesar de que a respostas da maior parte dos entrevistados tenham se dividido em três alternativas, foi possível inferir que segundo a opinião dos mesmos, é possível Reutilizar um componente criado com uma finalidade diferente da atual na maioria dos casos ou em grande parte deles.

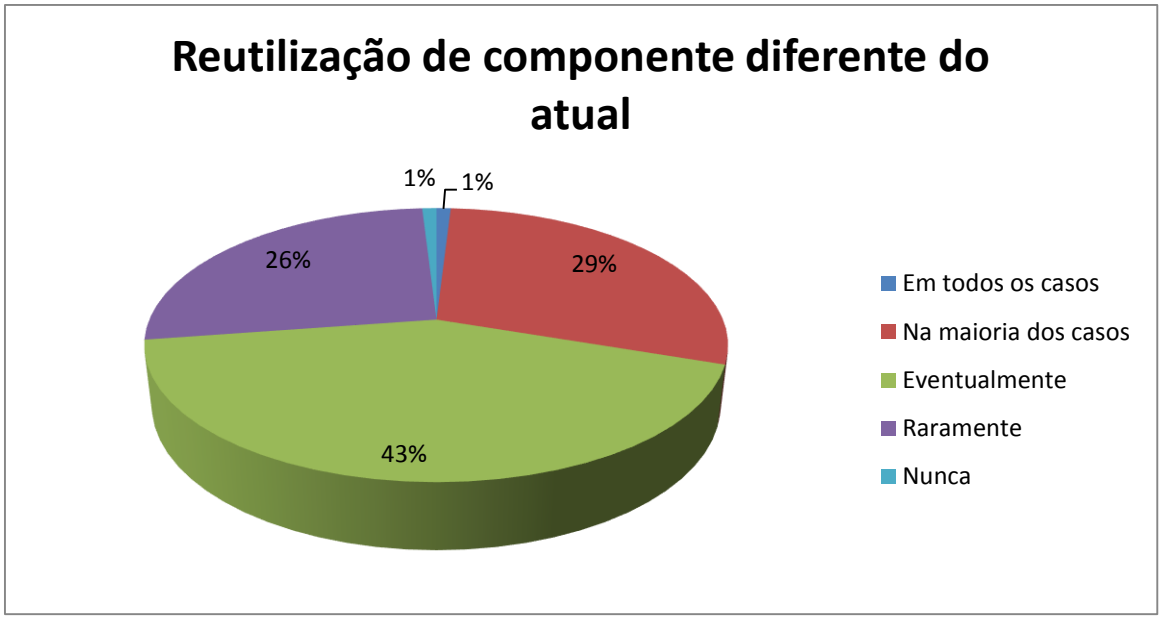

Gráfico 14 - Reutilização de componente diferente do atual

Embora apenas 1% dos entrevistados tenha declarado que é possível reutilizar componentes criados com finalidade muito diferente da atual em todos os casos, a maior parte dos entrevistados afirma que é possível reutilizar na maioria dos casos ou eventualmente, somadas essas duas alternativas resultam em 72% de todas as respostas.

As respostas obtidas nesta questão comprovam um alto índice de aceitação em relação ao reúso. Todavia tenha afirmado Sommerville (2007) que a reusabilidade costuma ser possível apenas na criação de componentes com finalidades parecidas com os componentes reutilizáveis, os entrevistados em parte discordaram dessa afirmação, afirmando que nessas condições a reusabilidade ainda é possível.

Outros 26% dos entrevistados responderam que essa aplicação é raramente possível e apenas 1% afirma que é impossível reutilizar um componente nessa condição. Um entrevistado não respondeu a questão.

4.1.2.8. Questão 13: O que geralmente ajuda você a reutilizar componentes?

Essa questão é fundamental ao presente estudo. Através dela, espera-se conhecer do ponto de vista prático elementos capazes de simplificar o reúso de código-fonte.

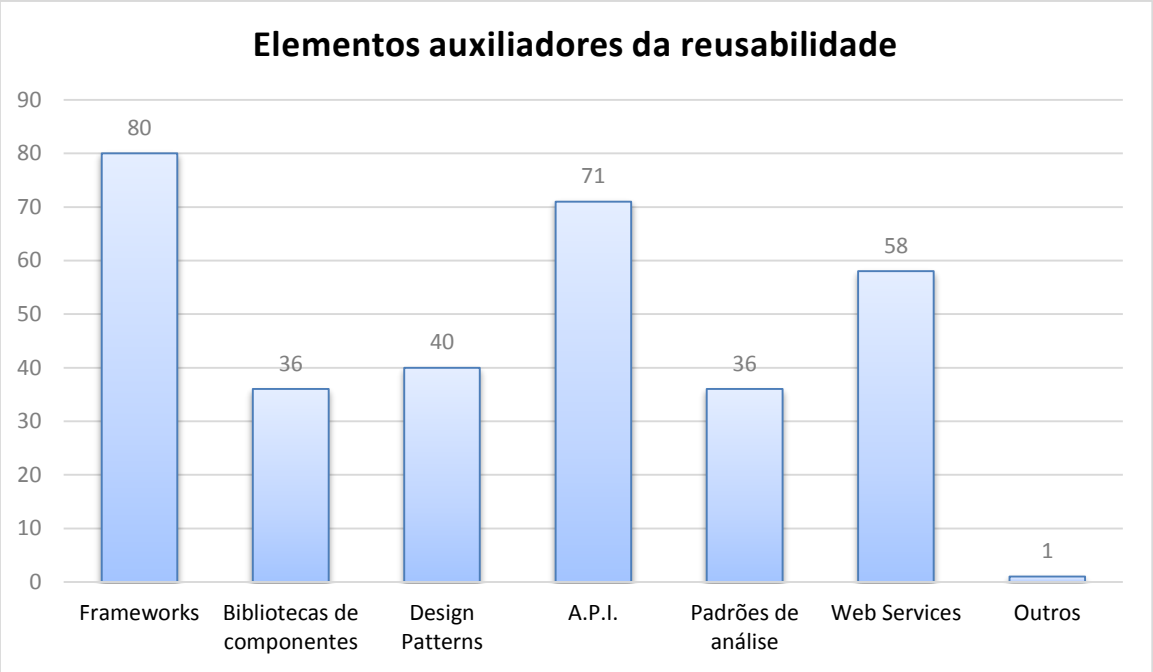

Gráfico 15 - Elementos auxiliadores da reusabilidade

Com a análise das respostas dessa questão foi possível concluir que todos os elementos estudados anteriormente são capazes de auxiliar profissionais desejosos de construir *softwares* utilizando o reúso de código-fonte.

Entretanto, elementos como *Frameworks* (assinalado por 80 pessoas), API's (assinalado por 71 pessoas) e *Web Services* (assinalado por 58 pessoas), são os principais elementos a serem considerados para essa aplicação. Os demais elementos, *Design Patterns*, Bibliotecas de componentes e Padrões de análise, foram assinalados por 40, 36 e 36 pessoas respectivamente. Apenas um entrevistado selecionou a alternativa Outros. Este especificou "*User Controls"* como ferramenta capaz de auxiliá-lo a reutilizar código-fonte diariamente.

De acordo com o gráfico acima, nota-se que os *Frameworks* e as API's são ferramentas muito eficientes para a criação de *softwares* com reúso. Logo, eles são capazes de simplificar o reúso. Isso comprova que é desejável que profissionais interessados em construir componentes reutilizáveis ou aplicar a reusabilidade na produção de sistemas, conheça diversos *Frameworks* e API's.

4.1.2.9. Questão 14: Ao criar um componente, depois de compreender um problema e pensar na solução. Na maioria dos casos qual seu próximo passo?

As respostas dessa questão tem o objetivo de compreender a propensão do entrevistado a reutilizar código-fonte.

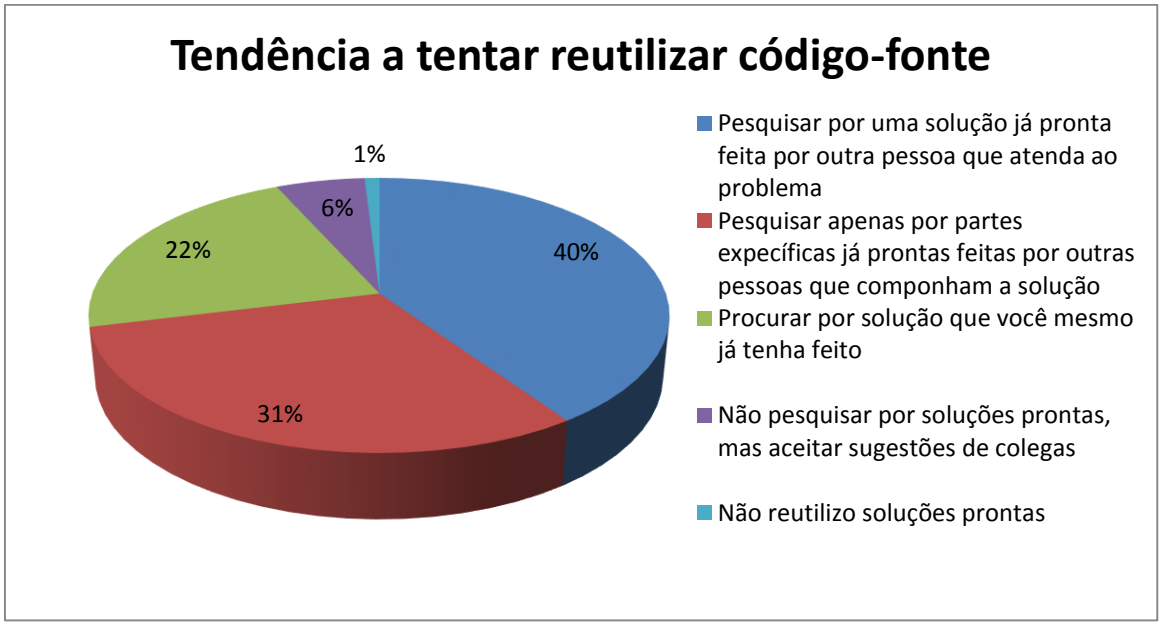

Gráfico 16 - Tendência a tentar reutilizar código-fonte

Foi possível notar que a maioria dos entrevistados tende a reutilizar código-fonte o máximo possível. Isso é visível pela alternativa “Pesquisar por uma solução já pronta feita por outra pessoa que atenda ao problema”, “Pesquisar apenas por partes específicas já prontas feitas por outras pessoas que componham a solução” e “Procurar por solução que você mesmo já tenha feito” que correspondem juntas a 93% das respostas, ou 40%, 31% e 22% respectivamente.

Apenas 6% dos entrevistados declarou não pesquisar por soluções prontas embora aceite sugestões de colegas. Somente 1% dos entrevistados afirmou não reutilizar soluções prontas. Três entrevistados não responderam esta questão.

É notório o alto índice de aceitabilidade dos entrevistados em relação à reusabilidade, reafirmando sua eficácia como ferramenta capaz de beneficiar a produção de *software*. A tendência dos entrevistados a buscar por soluções reutilizáveis, comprova que segundo suas experiências, a reusabilidade tem melhorado a produção de *software* e que existe a tendência de permanecer dessa forma.

A maior parte dos entrevistados demonstrou estar familiarizado com a reusabilidade. Foi perceptível o interesse dos entrevistados em reutilizar código-fonte sempre que possível e a preocupação em criar componentes que sejam facilmente reutilizados. Além da propensão em reutilizar componentes preferencialmente em relação a criar componentes sem reúso. Assim sendo, é possível inferir que a reusabilidade é uma preocupação constante na produção de *softwares*, o que contribui significativamente a produção de componentes reutilizáveis com qualidade. Não apenas isso, foi visível que a falta de comentários em códigos-fonte são extremamente prejudiciais à reusabilidade contrapondo o fato de que *Frameworks* e API's são ótimos aliados para a reutilização.

4.1.3. Terceira seção: Sobre como você enxerga a reusabilidade

Nesta seção serão dispostas as resposta obtidas através da terceira seção do questionário. Estas tem o objetivo de conhecer a opinião dos entrevistados sobre a reusabilidade.

Cada uma das subseções seguintes é referente a uma questão específica da terceira seção do questionário. As questões aqui aparecerão na ordem que aparecem no questionário.

4.1.3.1. Questão 15: Você concorda que a reutilização de código-fonte reduz o tempo necessário para a criação de uma solução:

Das respostas dessa questão esperava-se conhecer a opinião dos entrevistados sobre a vantagem de reutilizar código-fonte no que se refere ao tempo de fabricação do *software.*

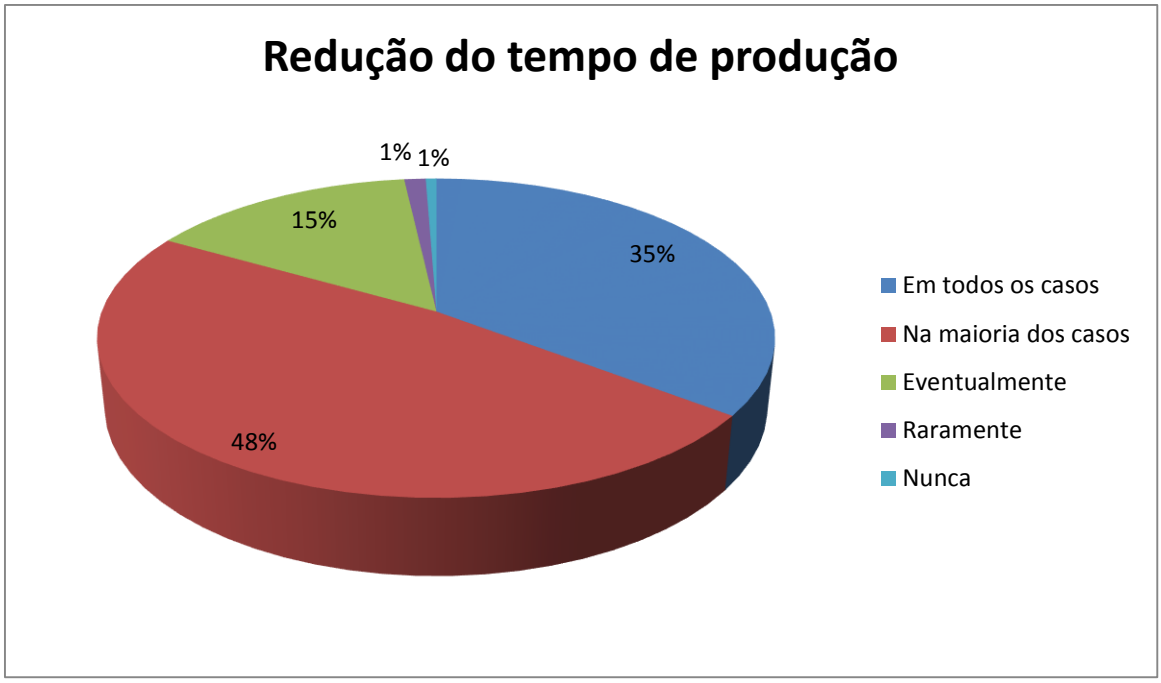

Gráfico 17 - Redução do tempo de fabricação

Muitos entrevistados (35%) responderam que a reusabilidade de código-fonte é capaz de reduzir o tempo de produção em todos os casos. A maioria dos entrevistados (48%) afirmou que essa redução acontece na maioria dos casos. Aproximadamente 15% respondeu que a redução do tempo de fabricação acontece eventualmente. Apenas 1% dos respondentes entende que a redução do tempo ocorre raramente ou nunca. Dois entrevistados não responderam essa questão.

Segundo Mazzola (2010) o processo de produção de softwares demanda um período de tempo extenso. É possível afirmar que segundo a opinião dos entrevistados a reusabilidade é capaz de reduzir o tempo de fabricação de *softwares* na maioria dos casos. Desse modo, é perceptível a vantagem do reúso sobre esse aspecto. Ou seja, o reúso de código-fonte é boa metodologia de redução do tempo de produção.

4.1.3.2. Questão 16: Na sua opinião, qual impacto a reusabilidade exerce sobre o software produzido?

Com esta questão esperava-se conhecer diretamente a opinião dos entrevistados sobre a influência da reusabilidade sobre o *software* produzido.

Aos entrevistados foi possível selecionar alternativas de respostas entre 1 e 10, onde 1 significava que o impacto era negativo e 10 significava que o impacto era muito positivo.

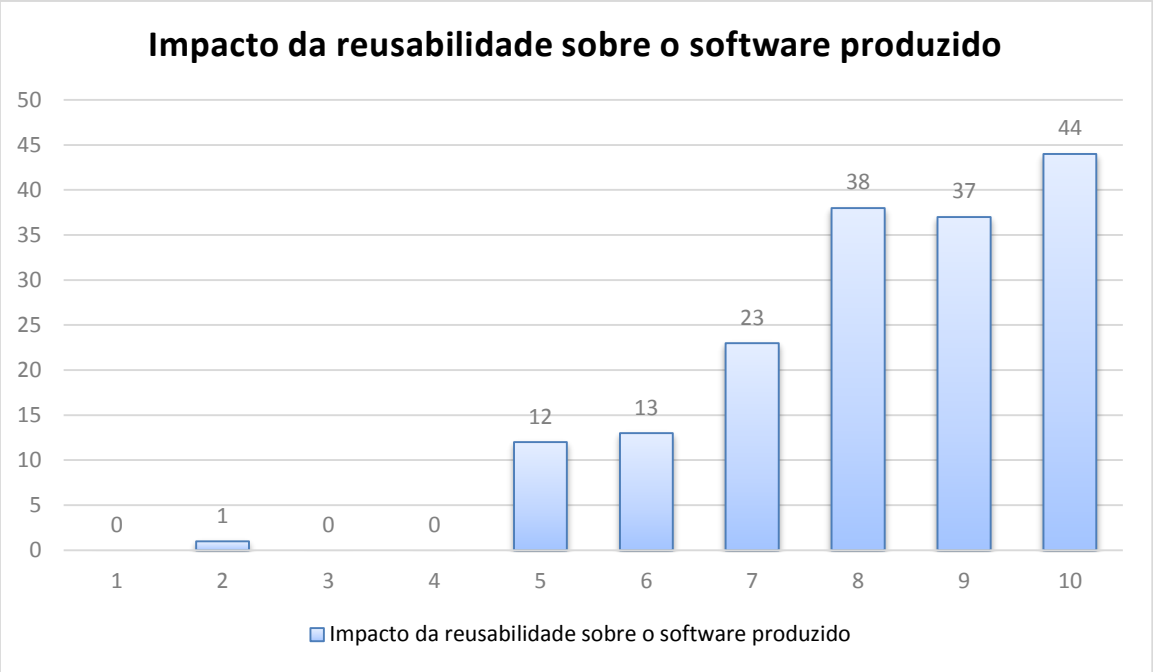

Gráfico 18 - Impacto da reusabilidade sobre o software produzido

A maior parte dos entrevistados (44 pessoas) selecionou a opção 10. Outras 37 pessoas selecionaram a opção 9. A opção 8 foi selecionada por 38 entrevistados. Em sequência as opções 7, 6, 5 e 2 foram assinaladas por 23, 13, 12 e 1 pessoa respectivamente. As demais opções (1, 3 e 4) não foram selecionadas por nenhum entrevistado. Apenas um entrevistado não respondeu a questão.

De acordo com Rezende (2005), os principais objetivos da reusabilidade são a qualidade, produtividade e a efetividade na produção de *softwares.* A opinião dos entrevistados convergiu com sua obra, reafirmando as vantagens de se reutilizar *softwares.* Tendo as opções 8, 9 e 10 compreendido a opinião da maior parte dos entrevistados, é possível afirmar que segundo suas opiniões, a reusabilidade exerce um impacto muito positivo no *software* produzido.

4.1.3.3. Questão 17: Você concorda que reutilizar um componente é uma forma de readquirir lucro sobre o investimento de criação do software?

Esta questão destinava-se a conhecer a opinião dos entrevistados sobre a eficácia da reusabilidade para aumentar a lucratividade do *software* produzido.

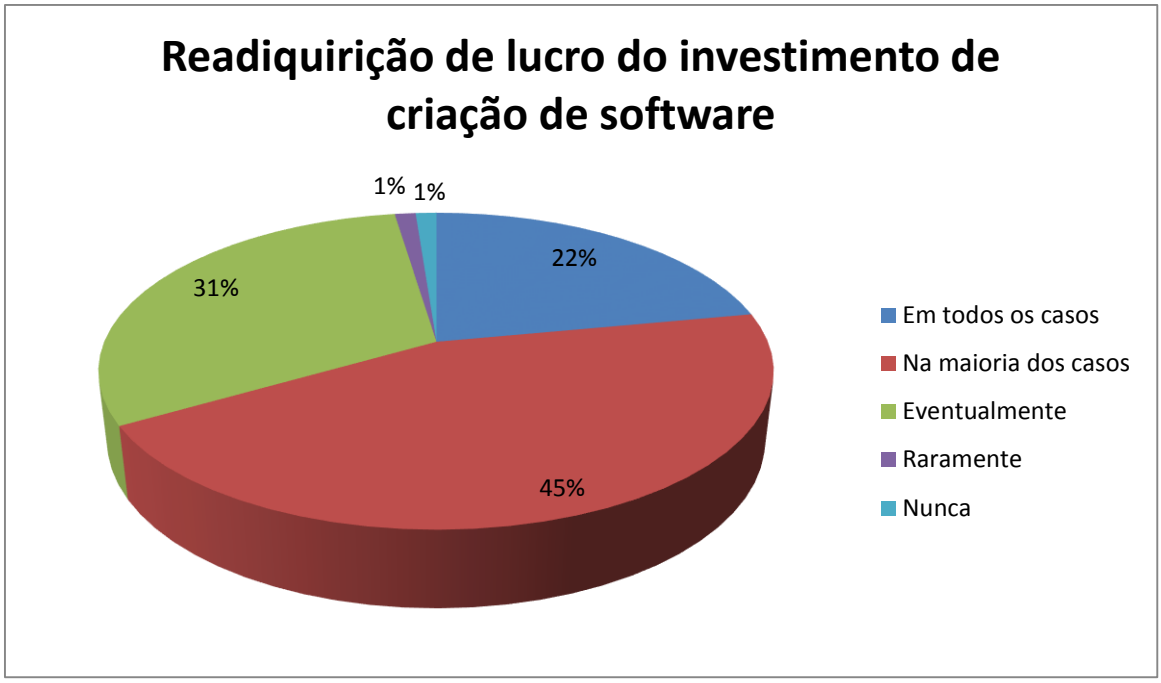

Gráfico 19 - Readiquirição de lucro do investimento de criação de *software*

A maior parte dos entrevistados (45%) afirmou que através da reusabilidade é possível reobter lucro sobre o investimento de produção de *software* na maioria dos casos. Aproximadamente 22% dos entrevistados disseram que isso é possível em todos os casos. Outros 31% dos entrevistados afirmam que essa readquirição é possível eventualmente. Apenas 1% dos entrevistados afirma que essa ocorrência é rara ou inexistente. Uma pessoa não respondeu a essa questão.

A maior parte dos entrevistados respondeu que é possível reobter lucro sob o investimento de criação de *software* com a reusabilidade em todos os casos ou na maioria deles. Isso comprova que segundo a opinião dos entrevistados, o investimento de criação de componentes reutilizáveis pode ser reavido na maioria dos casos.

Inicialmente esperava-se obter um padrão de respostas diferente para esta questão em relação a experiência e a função desempenhada pelos entrevistados. Contudo, não obstante a essa possibilidade na prática, não foi possível notar um padrão de respostas diferenciado para esta questão. Assim sendo, não foi possível perceber que a experiência e a função desempenhada pelos profissionais da área exerçam influência em suas opiniões sobre reaquisição do investimento de criação de *softwares.* Um estudo específico sobre esse assunto deve ser capaz de determinar essa influência.

Embora Sommerville (2007) tenha afirmado que *softwares* construídos com reúso tenham o custo de manutenção aumentado, de acordo com o gráfico, esse investimento poderá ser reavido em circunstâncias normais. Assim sendo, apesar do aumento do custo de manutenção, a reusabilidade permanece vantajosa.

4.1.3.4. Questão 18: Você concorda que é desejável que um software seja constituído de partes novas e partes reutilizadas funcionando em conjunto?

Esperava-se compreender a aceitabilidade dos entrevistados em relação a construção de *softwares* constituídos de partes novas e reutilizadas.

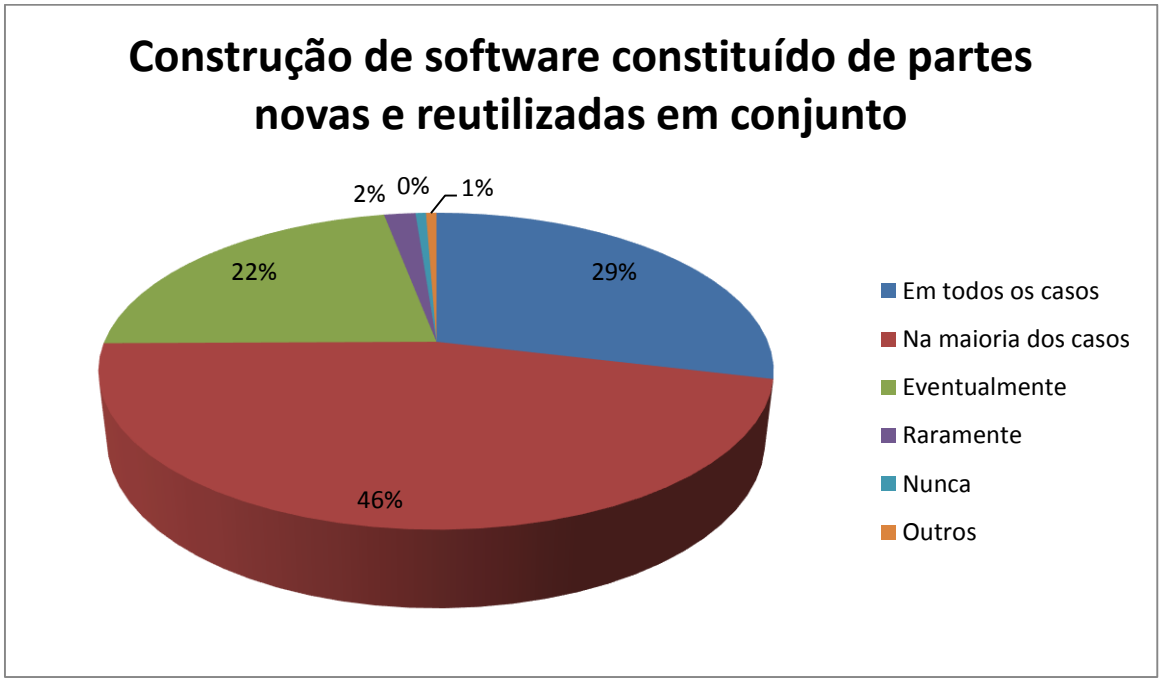

Gráfico 20 - Software constituído de partes novas e reutilizadas em conjunto

A maior parte dos entrevistados (46%) afirma que um *software* deve ser constituído de partes novas e partes reutilizadas funcionando em conjunto na maioria dos casos. Outra grande parte dos entrevistados (29%) afirma que isso é desejável em todos os casos. É então possível inferir que segundo a opinião dos entrevistados, isso é quase sempre desejável.

Os demais entrevistados assinalaram que essa forma de constituição de *software* é desejável eventualmente, raramente e nunca. Tendo sido escolhidas por 22%, 2% e aproximadamente 0% dos entrevistados. Uma pessoa selecionou a alternativa “Outros”, esse especificou que essa constituição é desejável apenas quando necessário. Duas pessoas não responderam essa questão.

Evidentemente, a construção de um *software* implica na produção de código-fonte embora não necessariamente que solucione todas as demandas. Segundo Mazzola (2010) a criação de um *software* deve ser feita com a produção mínima possível de código-fonte e reutilizando ao máximo componentes já prontos. Essa afirmação foi confirmada pela opinião da maior parte dos entrevistados

4.1.3.5. Questão 19: Na sua opinião, qual impacto a reusabilidade exerce sobre a produtividade nos processos de produção de software?

De acordo com a opinião da maior parte dos entrevistados, foi possível concluir que a reusabilidade gera um impacto muito desejável na produtividade das equipes de desenvolvimento de *software.*

Os respondentes deveriam assinalar alternativas em uma escala entre 1 e 10. As alternativas de menor valor implicam em afirmar que o impacto é indesejável, as de maior de valor correspondem a um impacto muito desejável segundo a opinião do entrevistado.

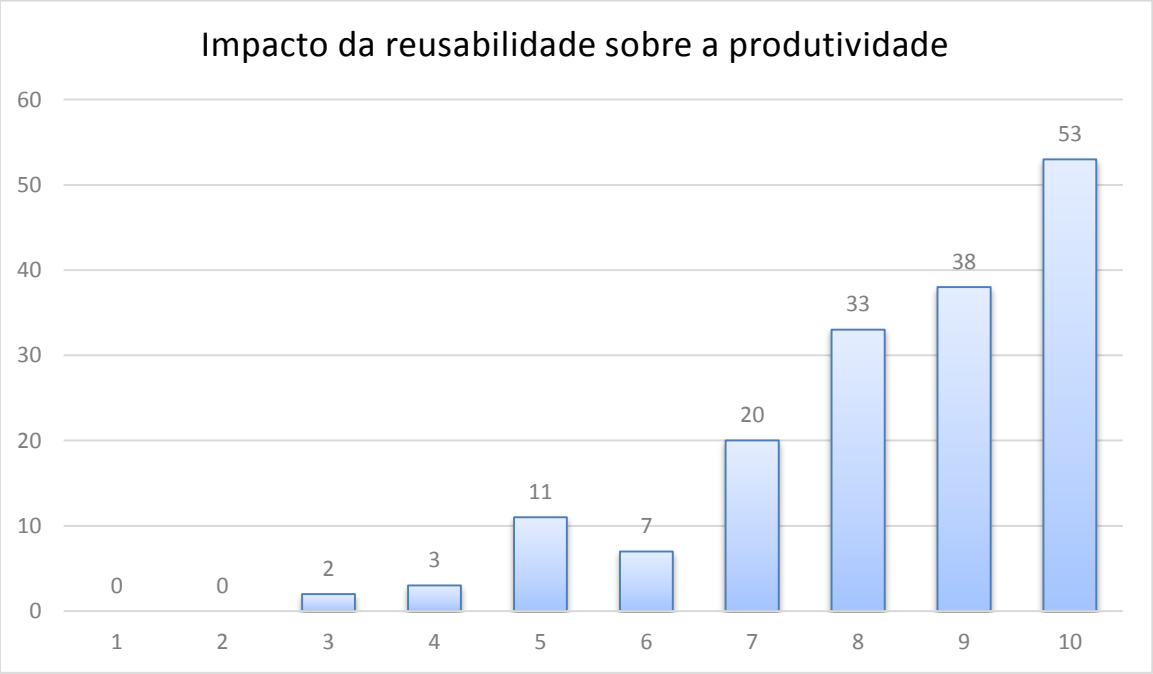

Gráfico 21 - Impacto da reusabilidade sobre a produtividade

A maioria dos entrevistados (53 pessoas) assinalou a opção 10 afirmando que a reusabilidade exerce um impacto muito positivo quanto à produtividade durante a produção de *softwares.* As próximas alternativas mais votadas foram 9 (38 pessoas) e 8 (33 pessoas). Assim sendo, é visível que a maioria dos entrevistados concordam que a reusabilidade é capaz de auxiliar na produtividade nos processos de produção de *software.*

As alternativas 3, 4, 5, 6 e 7 foram assinaladas por 2, 3, 11, 7 e 20 pessoas respectivamente. As demais alternativas não foram escolhidas por nenhum entrevistado. Dois entrevistados não responderam essa questão.

Segundo Rezende (2005) a produtividade é um dos principais objetivos da reusabilidade. Ainda segundo Sommerville (2007), a reusabilidade beneficia a produtividade na produção de *softwares*. Essas opiniões assemelham-se as opiniões dos entrevistados, demonstrando que a reusabilidade é capaz de melhorar a produtividade nos processos de produção de *softwares.*

4.1.3.6. Questão 20: Você acha que embora seja necessário compreender e adaptar um código-fonte reutilizável, a reusabilidade é capaz de reduzir o custo de fabricação?

De acordo com a opinião da maioria dos entrevistados, embora o processo de compreensão e adaptação do código-fonte reutilizável, a reusabilidade ainda é capaz de reduzir o custo de fabricação de *software* na maioria dos casos.

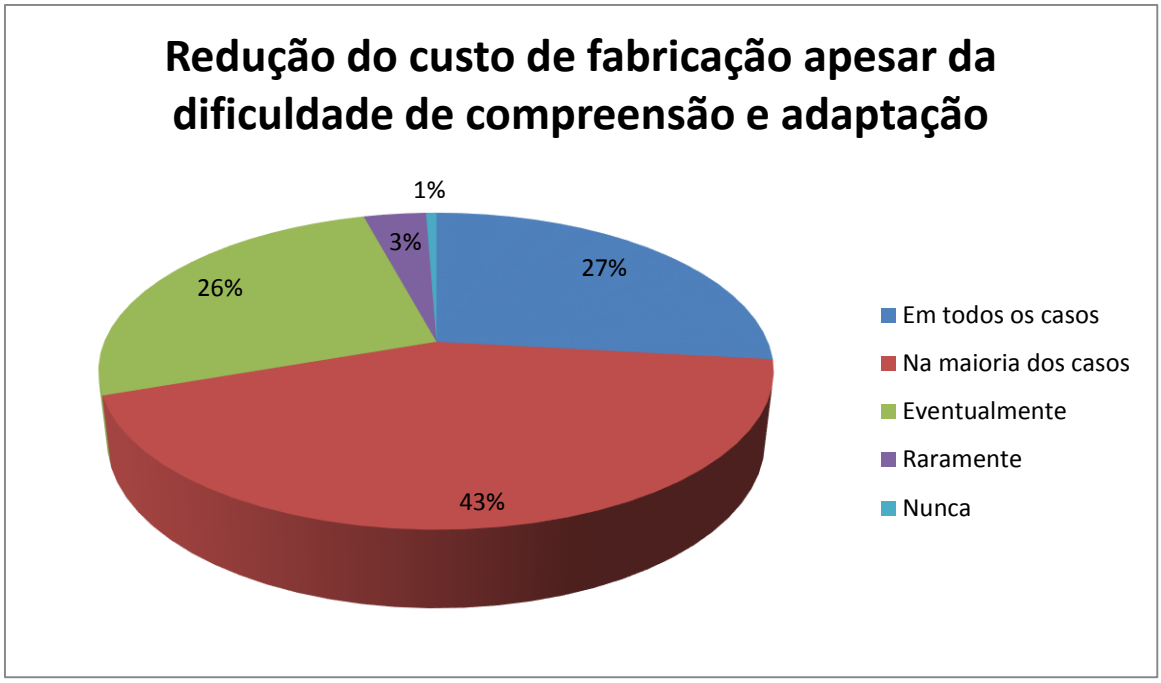

Gráfico 22 - Custo de fabricação e dificuldade de compreensão e adaptação

Grande parte dos entrevistados (27%) assinalou que apesar de ser necessário compreender e adaptar o código-fonte reutilizável, ainda assim a reusabilidade é compensatória em todos os casos. A maioria dos entrevistados (43%), entretanto, respondeu que nessa condição a reusabilidade é compensatória na maioria dos casos. Outro grande grupo de pessoas (26%) afirmou que a reusabilidade é compensatória nessa condição apenas eventualmente.

Apenas 3% dos entrevistados afirmaram que nessa condição a reusabilidade é raramente compensatória e somente 1% respondeu que a reusabilidade não é compensatória nessa condição. Uma pessoa não respondeu essa questão.

Sommerville (2007) relata a necessidade de compreender e adaptar um componente reutilizável. Contudo, Davis (1994) e Rezende (2005) afirmam que o reúso é capaz de reduzir o custo de fabricação de *softwares.* Segundo a opinião dos entrevistados, a influência da compreensão e adaptação do código-fonte não exerce um impacto preponderante quanto ao custo de fabricação de *softwares.*

4.1.3.7. Questão 21: Criar um componente reutilizável implica em testá-lo e documentá-lo de maneira detalhada. Você acha que o tempo investido nesses processos mantém a reusabilidade compensatória?

Esta questão especificamente objetiva conhecer a opinião dos entrevistados sobre a criação de componente reutilizável não obstante a necessidade de testá-lo e documenta-lo detalhadamente.

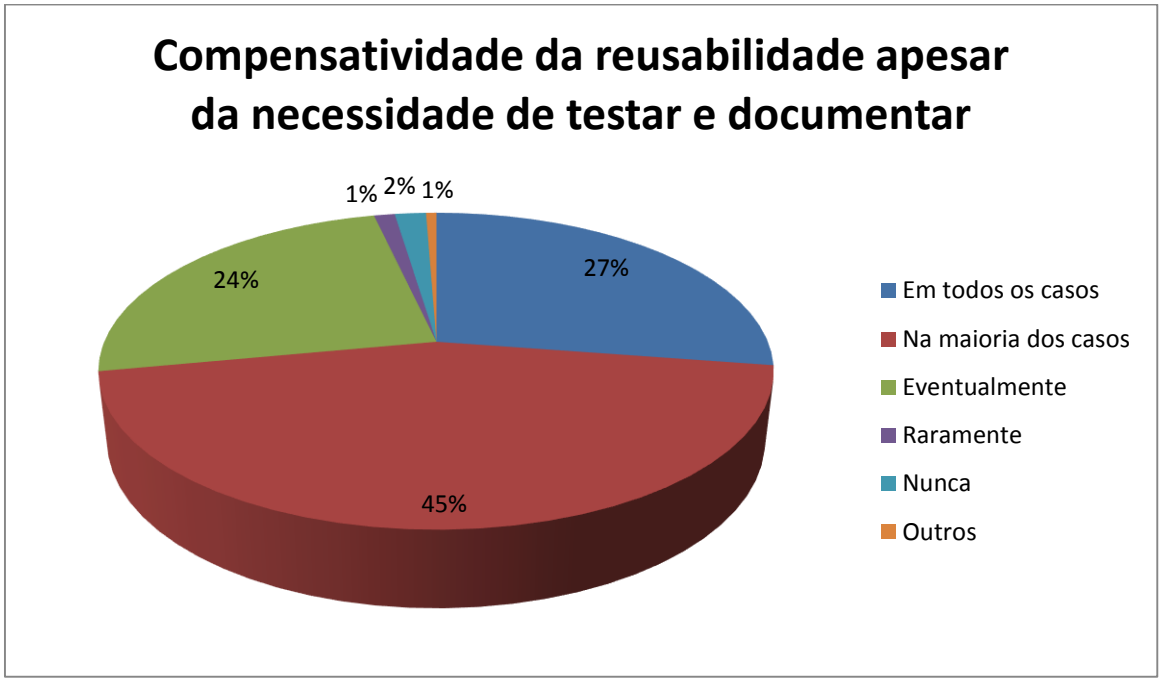

Gráfico 23 - Reusabilidade e a necessidade de testar e documentar

Embora a necessidade de testar e documentar um componente reutilizável, a maior parte dos entrevistados (45%) afirmou que ainda assim a reusabilidade é compensatória na maioria dos casos. Aproximadamente 27% dos entrevistados afirmaram que mesmo nessa circunstância a reusabilidade é compensatória em todos os casos. Ainda 24% dos entrevistados assinalaram que a reusabilidade é benéfica eventualmente.

Seguramente, na opinião dos entrevistados a reusabilidade é capaz de aprimorar a produção de *softwares* embora seja necessário testar e documentar detalhadamente os componentes reutilizáveis criados.

Apenas 1% dos entrevistados respondeu que a criação de componentes reutilizáveis embora a necessidade de testá-lo e documenta-lo é raramente viável. Apenas 2% responderam que a reusabilidade nessas condições é inviável. Um entrevistado assinalou a opção “Outros” e justificou afirmando não saber responder.

4.1.3.8. Questão 22: Você acha mais fácil estimar o custo de reutilizar um componente do que estimar o custo de fabricar um novo?

Estimar o custo de fabricação de *software* pode ser um processo complexo. Contudo, estimar o custo de fabricar um *software* utilizando componentes reutilizáveis pode tornar o cálculo de estimativa mais simples.

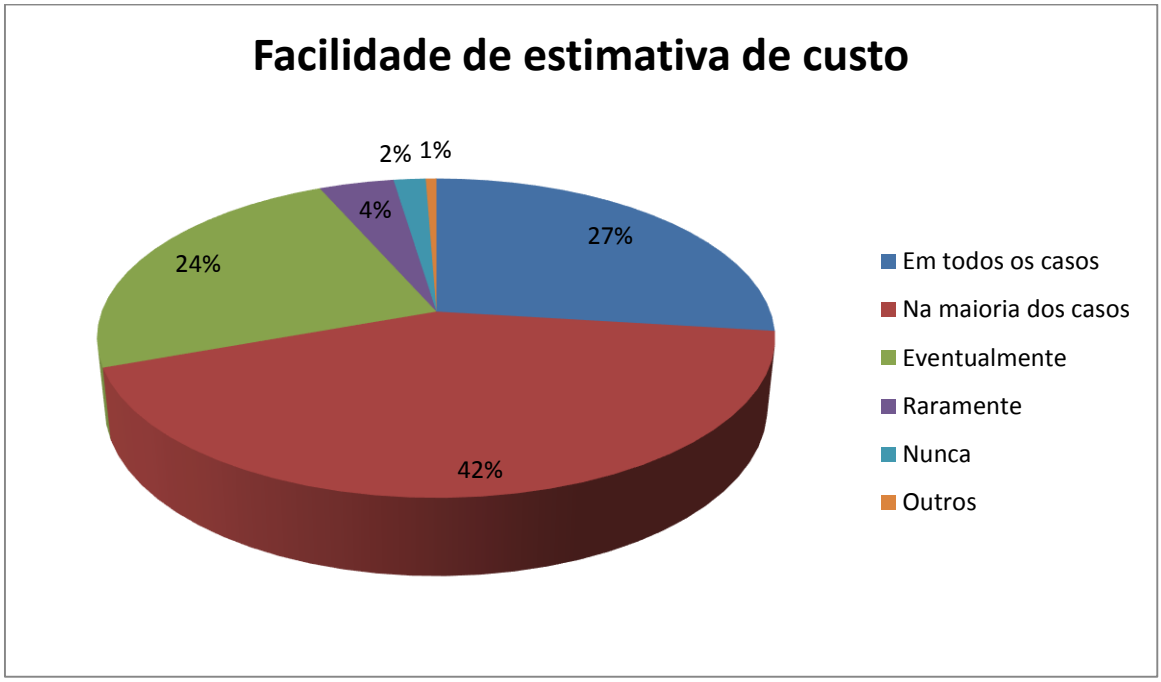

Gráfico 24 - Facilidade de estimativa de custo

De acordo com a opinião geral dos entrevistados é possível inferir que é mais simples estimar o custo de reutilização do que o custo de criação de um componente reutilizável em todos os casos ou na maioria deles. Somadas estas opções resultam em 69% de todas as respostas, ou 27% e 42% respectivamente.

Outro grande grupo de entrevistados (24%) afirmou que a afirmação acima é eventualmente verdadeira. Apenas 4% afirmaram que a afirmação é raramente correta, 2% afirmaram que a afirmação é falsa. Dois entrevistados não responderam a questão. Um respondente após selecionar a opção “Outros”, alegou não saber responder a questão.

Segundo Sommerville (2007), o custo de um componente reutilizável já é conhecido, contudo o custo de fabricação de um novo componente requer uma nova estimativa, estando sujeito a falhas no processo de estimar o custo de fabricação. Tendo sido confrontada nesta questão, a opinião dos entrevistados assemelhou-se com sua opinião, reafirmando a simplicidade de estimar o custo de reutilização de uma solução pronta.

4.1.3.9. Questão 23: Como você julga o impacto da reusabilidade sobre o custo de fabricação de software?

Aos respondentes solicitou-se que selecionasse alternativas entre 1 e 10. As alternativas de menor valor correspondem a um impacto indesejável ou um impacto desejável às alternativas de maior valor.

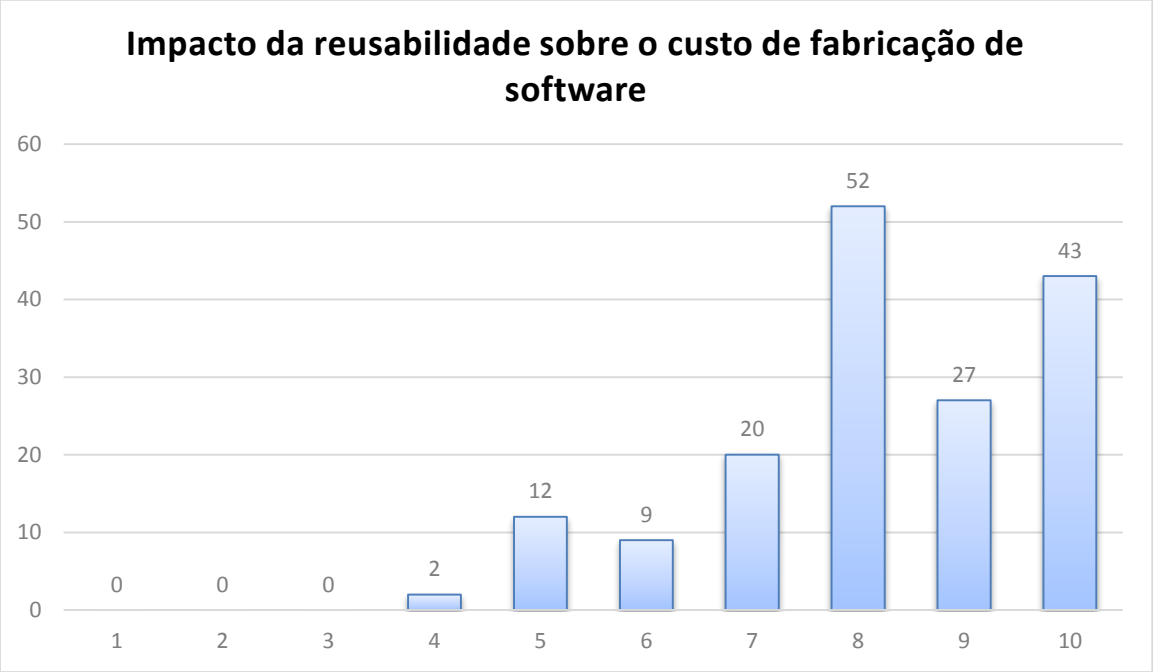

Gráfico 25 - Impacto da reusabilidade sobre o custo de fabricação de software

A maioria dos entrevistados (52 pessoas) assinalou a opção 8, outros 43 entrevistados assinalaram a opção 10 e 27 pessoas a opção 9. As opções 7, 6, 5, 4 foram assinaladas respectivamente por 20, 9, 12 e 12 pessoas. 4 entrevistados não responderam a questão.

Segundo, Davis (1994) e Rezende (2005), a reusabilidade é capaz de reduzir o custo de fabricação de *softwares.* A opção dos entrevistados pelas opções 8, 9 e 10 corresponde a 74% de todas as respostas. É possível concluir que segundo a opinião geral a reusabilidade é capaz de melhorar o custo de fabricação de *softwares.*

4.1.3.10. Questão 24: Reutilizar um componente, naturalmente fará com que o novo sistema se pareça com o reutilizado. Na sua opinião, a semelhança entre o sistema novo e o reutilizado é um fator positivo?

A semelhança visual ou não entre o *software* de onde foram reutilizados componentes e o novo *software* criado é recorrente quando se fala em reusabilidade de *software*. Espera-se conhecer a aceitação dos entrevistados quanto a essa semelhança.

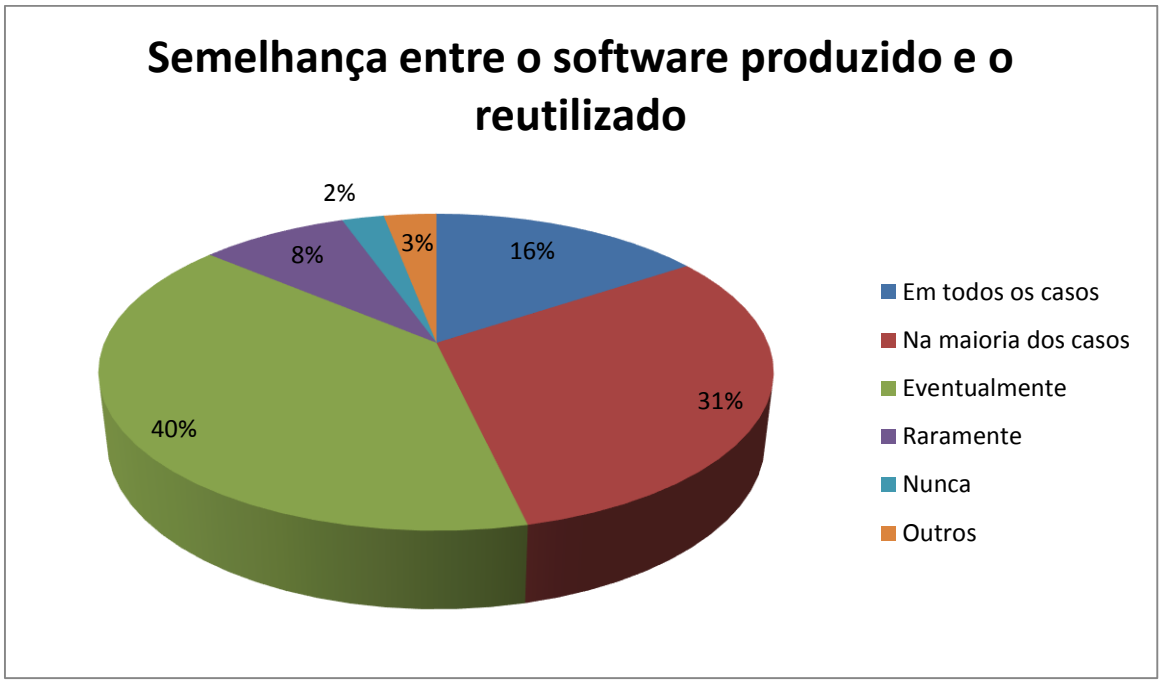

Gráfico 26 - Semelhança entre o software produzido e o reutilizado

A maior parte dos entrevistados (40%) afirmou que a semelhança entre o *software* criado apenas eventualmente é um elemento positivo. A segunda maior parte dos entrevistados (31%) respondeu que na maioria dos casos a semelhança é vantajosa. Aproximadamente 16% dos respondentes afirmaram que a semelhança entre o *software* criado e o reutilizado é vantajosa em todos os casos.

Os demais entrevistados responderam respectivamente “raramente” e “nunca” com 8% e 2%. Outros 5 entrevistados após selecionar a opção “Outros” justificaram afirmando que é um fator que não influencia negativamente ou positivamente e que essa semelhança pode ser positiva ou negativa em circunstâncias específicas. Três entrevistados não responderam a questão.

É perceptível que segundo os entrevistados a semelhança citada acima é vantajosa apenas em casos específicos. Ou seja, a semelhança pode ser positiva em determinados casos. Assim sendo, provavelmente seja necessária uma análise detalhada de cada caso para determinar as vantagens dessa semelhança. Contudo, segundo Sommerville (2007), um usuário estará menos propenso a cometer falhas caso utilize uma interface com a qual já esteja familiarizado.

4.1.3.11. Questão 25: Em sua opinião qual impacto a reusabilidade exerce sobre o tempo de fabricação de software?

Solicitou-se dos entrevistados que atribuíssem valores correspondentes a sua opinião sobre o impacto exercido pela reusabilidade sobre o tempo de fabricação de *softwares.* As opções de valores menores equivalem a um impacto indesejável enquanto as opções de valores maiores equivalem a um impacto mais desejável.

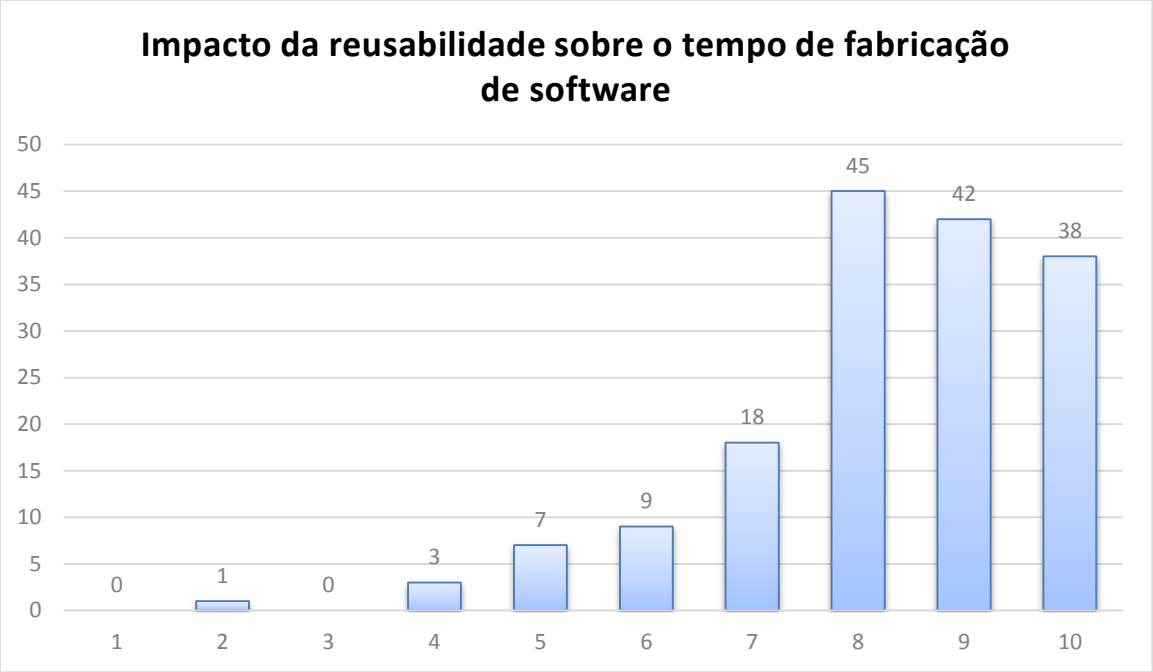

Gráfico 27 - Impacto da reusabilidade sobre o tempo de fabricação de software

Tendo 45, 42 e 38 pessoas selecionados as opções 8,9 e 10 respectivamente, elas resultam em um total de 77% de todas as respostas. As opções 7, 6, 5, 4 e 2 tiveram respectivamente 18, 9, 7, 3 e 1 voto cada uma. Outros 6 candidatos não responderam a questão.

Segundo Davis (1994) e Rezende (2005), a reusabilidade reduz o tempo de produção de um componente, por agregar qualidade e produtividade; A opinião geral nos leva a conclusão de que o reúso de código-fonte é uma boa metodologia para melhorar o tempo de produção de *softwares.*

4.1.3.12. Questão 26: Você concorda que se um software for constituído de partes já testadas é menos provável que o mesmo apresente falhas?

A opinião dos entrevistados nessa questão tem o objetivo de determinar a eficácia da reusabilidade sobre a existência de falhas nos *softwares* produzidos com reúso.

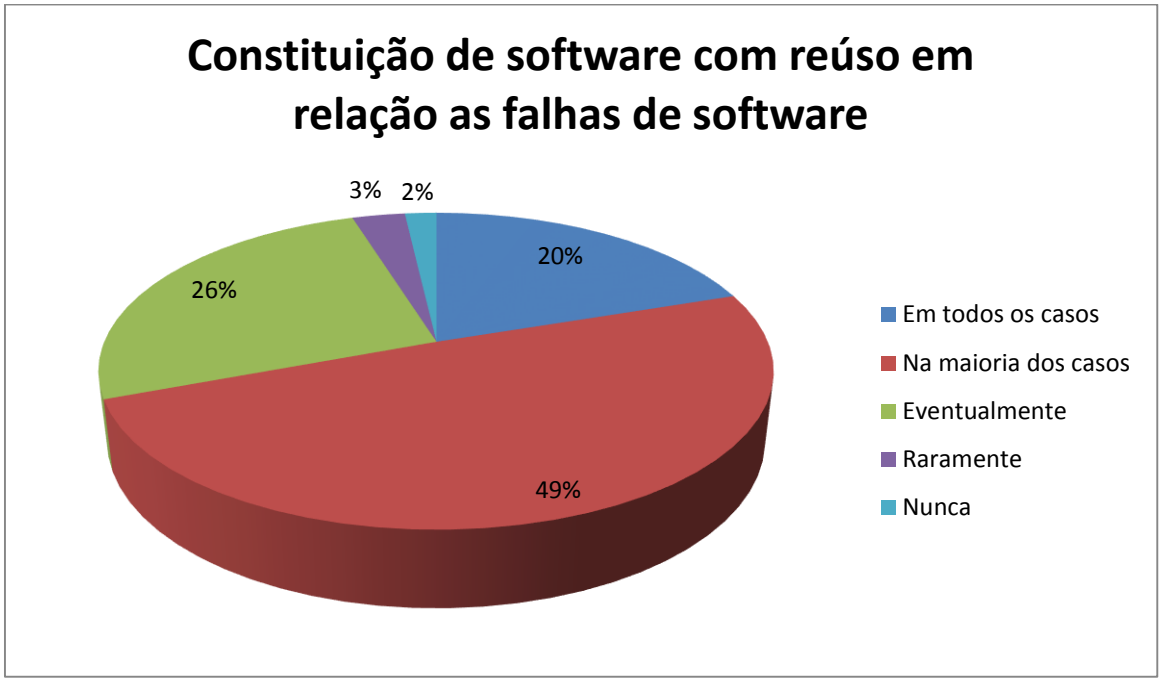

Gráfico 28 - Constituição de software com reúso em relação as falhas de software

A maioria dos entrevistados (49%) afirmou que um *software* constituído de partes novas e partes já testadas está menos propício a apresentar falhas na maioria dos casos. Aproximadamente 26% dos entrevistados responderam que essa propensão é eventualmente real. Outros 20% dos entrevistados creem que nessa condição a possibilidade de ocorrência de falhas é menor em todos os casos. Apenas 3% e 2% dos entrevistados afirmaram que a possibilidade de ocorrência de erros é menor nessa condição em casos raros e nunca respectivamente. Três pessoas não responderam a questão.

Segundo Mazzola (2010), um *software* constituído de partes já testadas está menos propenso a apresentar falhas. Isso se dá devido a correção anterior das mesmas. É visível que segundo os entrevistados a possibilidade de ocorrência de falhas em *softwares* constituídos por partes novas e partes já testadas funcionando em conjunto é menor na maioria dos casos. Assim sendo o aumento da confiabilidade em relação ao *software* produzido tende a aumentar.

4.1.3.13. Questão 27: Na sua opinião, qual impacto a reusabilidade exerce sobre a confiabilidade do novo sistema produzido?

Foi possível aos respondentes estabelecer valores para refletir sua opinião sobre a influência gerada pela reusabilidade na confiabilidade do *software* produzido. As opções de valores menores equivalem a um impacto indesejável enquanto as opções de valores maiores equivalem a um impacto mais desejável.

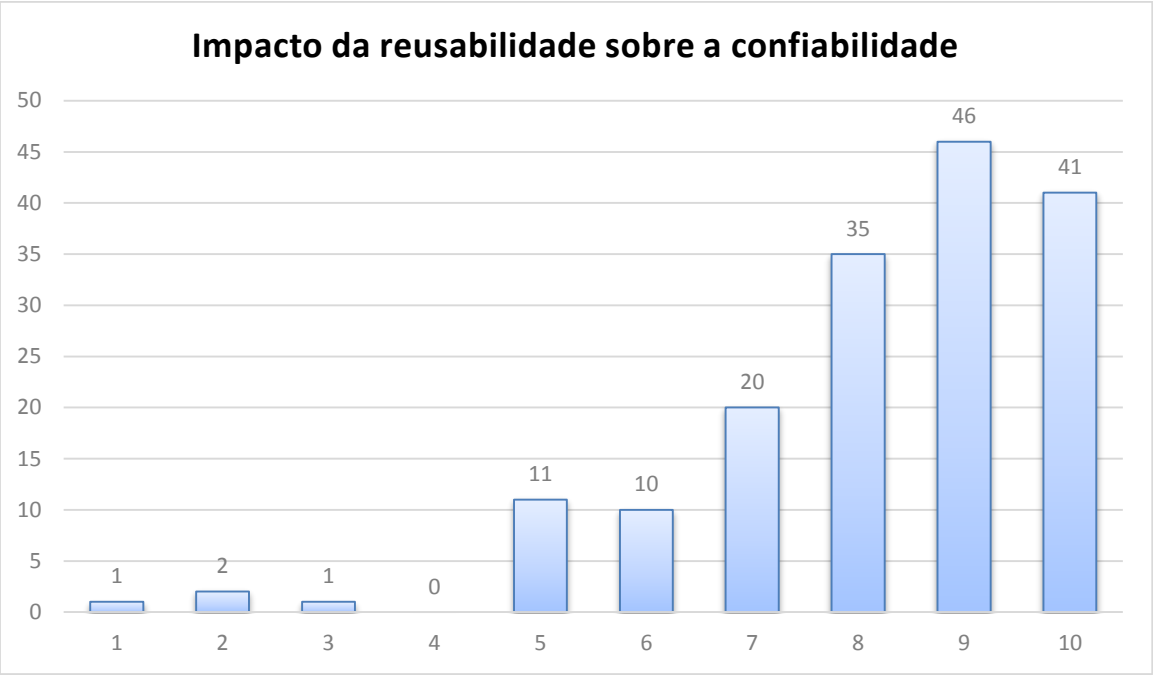

Gráfico 29 - Impacto da reusabilidade sobre a confiabilidade

A maior parte dos entrevistados (46 pessoas) atribuíram valor 9 para o impacto da reusabilidade sobre a confiabilidade do *software* produzido. Outros 41 entrevistados atribuíram valor 10. Um total de 35 pessoas assinalaram a opção 8. As alternativas 7, 6, 5, 3, 2 e 1 foram assinaladas respectivamente por 20, 10, 11, 2 e 1 pessoa cada. Duas pessoas não responderam a questão.

É possível inferir que segundo a opinião geral dos entrevistados a reusabilidade exerce uma influência bastante positiva quanto à qualidade do *software* produzido com reúso. Isso se dá pelo fato de que as respostas das alternativas 8, 9 e 10 somam 74% de todas as respostas. A opinião geral dos entrevistados, converge com a afirmação de Mazzola (2010) sobre o fato de que a reusabilidade agrega confiabilidade e qualidade ao *software* produzido com reúso.

Esta última seção do questionário evidenciou as diversas vantagens de se reutilizar *softwares*. Dentre os principais benefícios da reusabilidade, destacaram-se a redução do tempo de produção, o impacto positivo no *software* produzido, readiquirição do investimento de criação de componente reutilizável, o aumento da produtividade, redução do custo de fabricação, facilidade de estimativa de custo,

semelhança entre o *software* produzido e reutilizado, redução das falhas de *softwares* e aumento da confiabilidade.

Todos os benefícios da reusabilidade reafirmados por meio das respostas dos entrevistados asseguram a vantagem de reutilizar código-fonte, um alto nível de interesse e maturidade em relação à produção de sistemas com reusabilidade. Assim sendo, é seguro afirmar que a reusabilidade pode ser vista como uma metodologia capaz de aprimorar a produção de *softwares* em diversos aspectos.

Após análise minuciosa das respostas obtidas através do questionário como citado acima, foi possível visualizar melhor elementos práticos da reusabilidade e confrontá-los com a opinião de autores.

5. CONCLUSÃO

Com base nos estudos realizados e nas respostas coletadas através do questionário, foi possível concluir que de maneira geral a reusabilidade de *software* é uma metodologia vantajosa, sendo capaz de melhorar os resultados de produção de sistemas, agregando qualidade, confiabilidade, reduzindo os custos e o tempo de fabricação.

Por intermédio das respostas dos entrevistados, concluiu-se que a reusabilidade é aceita pela maioria dos profissionais relacionados à produção de *software* e que estes se preocupam em criar componentes que possam ser reutilizados posteriormente, além de admitirem preferir reutilizar componentes prontos a construir outros sem reúso e tenderem a procurar por soluções prontas na maioria das circunstâncias.

Várias inferências puderam ser feitas a partir dos resultados obtidos neste estudo. Tendo em vista que a reusabilidade possui entre os entrevistados um alto nível de aceitabilidade, como os entrevistados em ampla maioria utilizam a programação orientada a objetos, possivelmente esse é um paradigma que propicia aos interessados a reusabilidade de maneira satisfatória. Contudo, somente um estudo destinado a esse assunto poderá definir isso com mais segurança.

O fato de a maioria dos entrevistados afirmarem reutilizar código-fonte sempre que possível e serem cuidadosos ao criar componentes que sejam facilmente reutilizados é um dado importante. Naturalmente, profissionais desejosos de reutilizar *softwares* de maneira aprimorada, tendem a melhorar sua capacidade de reutilizar código-fonte além de criarem cada vez componentes reutilizáveis com maior qualidade. Assim, nesses casos, a reusabilidade tende a tornar-se cada vez mais vantajosa.

Tendo grande parte dos entrevistados afirmado trabalharem com a produção de *softwares* de modo bastante prático, informações coletadas de suas respostas tornaram-se indicadores seguros sobre como a reusabilidade afeta o dia-a-dia da produção de sistemas de maneira geral. Desse modo, os muitos efeitos positivos da reusabilidade sobre a fabricação de *softwares* tornaram-se evidentes. Os impactos da reusabilidade sobre a produtividade, tempo e custo de produção, confiabilidade e

qualidade foram os mais significativos, demonstrando a reação positiva gerada pela reusabilidade na produção de *softwares.*

Tendo sido realizadas comparações entre a opinião de autores sobre o assunto e as respostas obtidas através do questionário, concluiu-se que de modo geral a opinião dos entrevistados raramente divergiu da opinião dos estudiosos do assunto. Assim sendo, é seguro afirmar que a reusabilidade é capaz de reduzir o tempo e custo de fabricação de *softwares* aumentando a lucratividade. Além de exercer um impacto muito positivo no *software* produzido no que se refere à produtividade, confiabilidade e qualidade. Assim sendo, inferiu-se que a reusabilidade deve ser usada na construção de *softwares* sempre que possível.

Do ponto de vista técnico, foi perceptível que a falta de comentários e indentação do código-fonte são elementos que dificultam muito o reúso de código-fonte. Em contrapartida, os *Frameworks*, API's e *WebServices* auxiliam bastante a aplicação da reusabilidade na prática.

6. TRABALHOS FUTUROS

Uma possível continuação desse estudo poderia ser a realização de testes de produção de *software* com reúso de código-fonte e sem a reusabilidade, com a finalidade de evidenciar através de valores reais os benefícios financeiros da produção de *software* com reúso.

A criação e documentação de uma metodologia de fabricação de *software* totalmente fundamentada na reusabilidade poderiam servir como um manual para projetistas de *softwares.* Assim, já conhecendo fatores inerentes ao reúso de código fonte, seria mais fácil criar iniciativas capazes de minimizar elementos dificultadores da reusabilidade.

A realização de um estudo sobre organização de código-fonte visando levantar características ou metodologias capazes de tornar o código-fonte mais legível para facilitar sua compreensão e adaptação. Esse estudo é extremamente pertinente pelo fato de que a falta de indentação, comentários e documentação são fatores dificultadores da aplicação da reusabilidade.

De modo geral constatou-se que *Frameworks* são extremamente eficientes para a aplicação da reusabilidade. Visualizando esse cenário, um estudo poderia ser realizado visando descrever *Frameworks* mais indicados para essa finalidade.

7. REFERÊNCIAS

BOOCH, Grady. **Object-Oriented Analysis Ad Design With Applications**. 2ed. Massachusett: Addison Wesley Longman Limited,1998.

BRAUDE, Eric J. **Projeto De Software: Da Programação À Arquitetura: Uma Abordagem Baseada Em Java**. Porto Alegre, RS: Bookman, 2005.

COELHO, Paulo R S. **Uma Arquitetura Orientada a Serviços Para Laboratórios Remotos**. Universidade Estadual de Campinas. Campinas, 2006.

COSTA, Luís Filipe Medeiro. **Criação de uma *framework* para desenvolvimento colaborativo**. Instituto Universitário de Lisboa, 2010.

CZARNECKI Krzysztof, EISENECKER, Ulrich. **Generative Programming: Methods, Tools, and Applications**. Addison-Wesley, 2000.

DAVIS, William S.. **Análise e Projeto de Sistemas: Uma Abordagem Estruturada**. Rio de Janeiro: LTC Livros Técnicos e Científicos S.A., 1994.

GAMMA, Erich, et al. **Padrões de Projeto: Soluções Reutilizáveis de Software Orientado a Objetos**. Porto Alegre: Bookman, 2002. 364 p.

HEINEMAN, G. T. COUNCILL, W. T. **Component - based Software Engineering. Putting the Pieces Together**. Addison – Wesley, 2001.

JOHNSON, Ralph E., **Frameworks = (Components + Patterns), Communications of the ACM**, Vol.40, 1997.

KNUTH, D.E. **The Art Of Programming: Fundamental Algorithms Massachusett**: Addsion – Wesley,1971.

MADEIRA, Charles Andryê Galvão. FORGE V8: **Um framework para o desenvolvimento de jogos de computador e aplicações multimídia.** Universidade Federal de Pernambuco. Recife, 2001.

MAZZOLA, Vitório B.. **Engenharia de Software**. Universidade Federal de Santa Catarina, 2010.

MOORE, Dana. BUDD, Raymond. BENSON, Edward. **Professional Rich Internet Applications: AJAX and Beyond**; Wrok; 2007.

NGOLO, Márcio A F. **Arquitectura Orientada a Serviços REST para Laboratórios Remotos**. Universidade Nova de Lisboa. Lisboa, 2009.

OLIVEIRA, Felipe C. PAULA, Leonardo Lopes de. **Engenharia de Software baseada em componentes: Uma abordagem prática em ambientes Web.** Universidade de Brasília, 2009.

PAULA, Wilson de Pádua F.. **Manual do Engenheiro de Software**. 2000.

PETERS, James F. PEDRYCZ, Withold. **Engenharia de Software: Teoria e Prática**. Rio de Janeiro: Campus, 2001. 602 p.

PIVETA, Eduardo Kessler. Aurélia: **Um modelo de suporte a programação orientada a aspectos.** Universidade Federal de Santa Catarina, 2001.

PRESSMAN, Roger S.. **Engenharia de Software.** 6ª edição. São Paulo: McGraw-Hill/Makron Books do Brasil, 2006. 720 p.

REZENDE, Denis A.. **Engenharia de Software e Sistemas de Informação**. 3ª edição. Rio de Janeiro: BRASPORT Livros e Multimídia LTDA, 2005.

SILVA, Lyrene F. **Uma Estratégia Orientada a Aspectos para Modelagem de Requisitos**. Pontifícia Universidade Católica do Rio de Janeiro, 2006.

SOMMERVILLE, Ian. **Engenharia de Software**. 6ª edição. São Paulo: Pearson Education do Brasil, 2004.

SOMMERVILLE, Ian. **Engenharia de Software**. 8ª edição. São Paulo: Pearson Education do Brasil, 2007.

SOMMERVILLE, Ian. **Engenharia de Software**. 9ª edição. São Paulo: Pearson Education do Brasil, 2011. 568 p.

VALENTE, Carlos. T**ópicos avançados de engenharia de software**. ESAB: Escola Superior Aberta do Brasil, 2008.

8. ANEXOS

8.1. ANEXO 1: QUESTIONÁRIO

Reusabilidade de software

Este questionário tem o intuito de entender como a reutilização contribui durante o processo de produção de software diariamente.

As perguntas estão fundamentadas na opinião de autores sobre a reusabilidade, conforme pesquisa científica realizada entre fevereiro e setembro de 2013.

Dados pessoais não serão analisados nesse questionário.

Autor:
Maicon Ribeiro
8º Período, Ciência da Computação
Rede de Ensino Doctum
maicon.ae@gmail.com

Sobre você:

1) Qual seu nível de formação?

- ☐ Técnico
- ☐ Graduado / Graduando
- ☐ Pós-Graduado / Pós-Graduando
- ☐ Mestre / Mestrando
- ☐ Doutor / Doutorando
- ☐ Outro:

2) Qual seu papel na instituição onde trabalha?

Selecione a opção que mais se assemelha à função desempenhada diariamente.

- ○ Apoio Estratégico (Diretor, CEO, CIO)
- ○ Apoio Gerencial (Gerente de projetos, gerente de qualidade, gerente de teste)

- Apoio Operacional (Desenvolvedor, analista de sistemas, webdesigner, analista de testes)
- Estudante
- Outro:

3) Diariamente, você está envolvido com a produção de softwares de qual natureza?

- Softwares para web
- Softwares de gerenciamento empresarial
- Sistemas para terminais móveis
- Desenvolvimento de sistemas operacionais
- Softwares aplicativos de propósito geral
- Outro:

4) Há quanto tempo aproximadamente você trabalha com produção de softwares?

- Menos de um ano
- Entre um e três anos
- Entre três e cinco anos
- Entre cinco e dez anos
- Mais que dez anos
- Outro:

5) Na instituição onde você trabalha, aproximadamente quantas pessoas estão envolvidas com a produção de softwares?

- ○ Menos de vinte pessoas
- ○ Entre vinte e cinquenta pessoas
- ○ Entre cinquenta e cem pessoas
- ○ Mais de cem pessoas
- ○ Outro: _____

Sobre a reusabilidade no seu dia-a-dia:

6) Quais paradigmas de programação você utiliza diariamente?

- ☐ Programação imperativa
- ☐ Programação estruturada
- ☐ Programação orientada a objetos
- ☐ Programação orientada a aspectos
- ☐ Programação orientada a serviços
- ☐ Programação orientada a eventos
- ☐ Programação funcional
- ☐ Programação lógica
- ☐ Outro: _____

7) Quais linguagens de programação você utiliza diariamente?

- ☐ C
- ☐ Java

- [ ] Objective-C
- [ ] C++
- [ ] PHP
- [ ] C#
- [ ] Visual Basic
- [ ] Python
- [ ] Perl
- [ ] JavaScript
- [ ] Ruby
- [ ] Transact-SQL
- [ ] Visual Basic .NET
- [ ] Lisp
- [ ] Pascal
- [ ] Delphi
- [ ] Assembly
- [ ] Outro:

8) De maneira geral, com qual frequência você reutiliza código-fonte?

- ○ Reutilizo o máximo possível
- ○ Reutilizo com muita frequência
- ○ Habitualmente procuro por soluções reutilizáveis e utilizo-as
- ○ Raramente reutilizo código-fonte
- ○ Jamais reutilizo código-fonte
- ○ Outro:

9) Ao construir um componente de software você se preocupa em fazê-lo de maneira que facilite a reutilização posterior do mesmo?

- ○ Em todos os casos
- ○ Na maioria dos casos
- ○ Eventualmente
- ○ Raramente
- ○ Nunca
- ○ Outro:

10) De maneira geral, você julga mais fácil implementar uma funcionalidade do que procurar uma solução reutilizável?

- ○ Em todos os casos
- ○ Na maioria dos casos
- ○ Eventualmente
- ○ Raramente
- ○ Nunca
- ○ Outro:

11) Entre as dificuldades de reutilizar código-fonte criado por outra pessoa, na sua opinião, quais mais atrapalham?

- ☐ Ausência de indentação (organização do código-fonte)
- ☐ Ausência de comentários (observações de quem escreveu o código-fonte)
- ☐ Ausência de documentação
- ☐ Desconfiança em relação ao funcionamento do componente reutilizável
- ☐ Insegurança em encontrar uma solução eficiente
- ☐ Outro:

12) De maneira geral você acha possível reutilizar um componente criado primordialmente com uma finalidade muito diferente da atual?

- ○ Em todos os casos
- ○ Na maioria dos casos
- ○ Eventualmente
- ○ Raramente
- ○ Nunca
- ○ Outro: ____

13) O que geralmente ajuda você a reutilizar componentes?

- ☐ Frameworks
- ☐ Bibliotecas de componentes
- ☐ Design Patterns
- ☐ A.P.I.
- ☐ Padrões de análise
- ☐ Web Services
- ☐ Outro: ____

14) Ao criar um componente, depois de compreender um problema e pensar na solução. Na maioria dos casos qual seu próximo passo?

- ○ Pesquisar por uma solução já pronta feita por outra pessoa que atenda ao problema
- ○ Pesquisar apenas por partes específicas já prontas feitas por outras pessoas que componham a solução
- ○ Procurar por solução que você mesmo já tenha feito
- ○ Não pesquisar por soluções prontas, mas aceitar sugestões de colegas

- Não reutilizo soluções prontas
- Outro:

Sobre como você enxerga a reusabilidade:

15) Você concorda que a reutilização de código-fonte reduz o tempo necessário para a criação de uma solução:

- Em todos os casos
- Na maioria dos casos
- Eventualmente
- Raramente
- Nunca
- Outro:

16) Na sua opinião, qual impacto a reusabilidade exerce sobre o software produzido?

	1	2	3	4	5	6	7	8	9	10	
Ruim											Bom

17) Você concorda que reutilizar um componente é uma forma de readquirir lucro sobre o investimento de criação do software?

- Em todos os casos
- Na maioria dos casos
- Eventualmente

- Raramente
- Nunca
- Outro:

18) Você concorda que é desejável que um software seja constituído de partes novas e partes reutilizadas funcionando em conjunto?

- Em todos os casos
- Na maioria dos casos
- Eventualmente
- Raramente
- Nunca
- Outro:

19) Na sua opinião, qual impacto a reusabilidade exerce sobre a produtividade nos processos de produção de software?

	1	2	3	4	5	6	7	8	9	10	
Ruim	○	○	○	○	○	○	○	○	○	○	Bom

20) Você acha que embora seja necessário compreender e adaptar um código-fonte reutilizável, a reusabilidade é capaz de reduzir o custo de fabricação?

- Em todos os casos
- Na maioria dos casos
- Eventualmente
- Raramente
- Nunca
- Outro:

21) Criar um componente reutilizável implica em testá-lo e documentá-lo de maneira detalhada. Você acha que o tempo investido nesses processos mantém a reusabilidade compensatória?

- ○ Em todos os casos
- ○ Na maioria dos casos
- ○ Eventualmente
- ○ Raramente
- ○ Nunca
- ○ Outro: ____

22) Você acha mais fácil estimar o custo de reutilizar um componente do que estimar o custo de fabricar um novo?

- ○ Em todos os casos
- ○ Na maioria dos casos
- ○ Eventualmente
- ○ Raramente
- ○ Nunca
- ○ Outro: ____

23) Como você julga o impacto da reusabilidade sobre o custo de fabricação de software?

	1	2	3	4	5	6	7	8	9	10	
Ruim	○	○	○	○	○	○	○	○	○	○	Bom

24) Reutilizar um componente, naturalmente fará com que o novo sistema se pareça com o reutilizado. Na sua opinião, a semelhança entre o sistema novo e o reutilizado é um fator positivo?

- ○ Em todos os casos
- ○ Na maioria dos casos
- ○ Eventualmente
- ○ Raramente
- ○ Nunca
- ○ Outro: _____

25) Em sua opinião qual impacto a reusabilidade exerce sobre o tempo de fabricação de software?

	1	2	3	4	5	6	7	8	9	10	
Ruim	○	○	○	○	○	○	○	○	○	○	Bom

26) Você concorda que se um software for constituído de partes já testadas é menos provável que o mesmo apresente falhas?

- ○ Em todos os casos
- ○ Na maioria dos casos
- ○ Eventualmente
- ○ Raramente
- ○ Nunca
- ○ Outro: _____

27) Na sua opinião, qual impacto a reusabilidade exerce sobre a confiabilidade do novo sistema produzido?

	1	2	3	4	5	6	7	8	9	10	
Ruim	○	○	○	○	○	○	○	○	○	○	Bom