REUSO DE SOFTWARE: SUAS VANTAGENS, TÉCNICAS E PRÁTICAS

Hiran N. M. Ferreira[1], **Thiago F. Naves**[1]

[1]Faculdade de Computação – Universidade Federal de Uberlândia (UFU)
Uberlândia – MG – Brasil

{hirannonato,tf.naves}@gmail.com

Abstract. *The Software reuse is a practice that is growing and becoming increasingly present in projects involving computing and technology. Even before the Software Engineering consolidate itself, this approach was already widespread in companies and their contributions were clear in the projects. In recent times, with advances and research on software reuse is important to observe more and better the techniques, approaches and advantages of this concept that has become one of the most important within the Software Engineering. This article attempts to show an overview of the concept of software reuse, depicting its history, as well as its main techniques and approaches.*

Resumo. *O Reuso de software é uma prática que vem crescendo e tornando-se cada vez mais presente em projetos que envolvem computação e tecnologia. Antes mesmo da Engenharia de Software consolidar-se, essa abordagem já era difundida em empresas e suas contribuições ficavam claras nos projetos. Em tempos atuais, com os avanços e pesquisas sobre o reuso de software é importante observamos mais e melhor as técnicas, abordagens e vantagens desse conceito que tornou-se um dos mais importantes dentro da Engenharia de Software. Esse artigo procura mostrar uma visão geral do conceito de reuso de software, retratando sua história, bem como, suas principais técnicas e abordagens.*

1. Introdução

Com o crescimento da demanda por software, surge a necessidade de desenvolvimento de sistemas cada vez mais complexos e com qualidade. Com isso, surgiram diversas abordagens para tornar o processo de desenvolvimento mais rápido, como: reengenharia, geradores de código fonte, ferramentas *CASE*, entre outras.

Pesquisadores identificaram que uma grande parte do desenvolvimento de software pode ser reutilizada para construção de outros softwares; surge então, a ideia de reuso de software. Reuso de software é uma abordagem que procura reusar partes do processo de construção de um software que já foi realizado, não somente o código, mas todas as abordagens utilizadas no processo de criação de um software.

Reusar software tem grandes vantagens, entre as principais pode-se citar: maior produtividade no processo de desenvolvimento, aumento da qualidade do software e diminuição do tempo/prazo de entrega do software. É mostrada neste artigo uma visão geral de reuso de software, suas principais características, contribuições e obstáculos. São apresentadas também algumas técnicas que são utilizadas para alcançar um reuso amplo,

bem como, uma visão deste conceito na prática, onde são mostrados alguns casos de uso desta abordagem.

Este artigo está estruturado da seguinte forma: a Seção 2 mostra uma visão geral de reuso de software; a Seção 3 mostra a sua evolução com o passar dos anos; a Seção 4 mostra os principais benefícios e obstáculos da utilização desta abordagem; a Seção 5 mostra as principais técnicas abordadas para utilização de reuso de software; a Seção 6 mostra alguns fatos e falácias; a Seção 7 mostra alguns casos práticos da sua utilização e por fim, a Seção 8 traz uma conclusão acerca deste tema.

2. Visão geral do reuso de software

A utilização de reuso de software é considerada por muitos pesquisadores o maior objetivo da Engenharia de Software. Reusar componentes de software significa reduzir custos e tempo no processo de desenvolvimento, aumentar a qualidade, entre outras diversas vantagens que tornam esse conceito praticável.

Cada vez mais os softwares estão se tornando maiores e mais complexos, aumentando assim a exigência pela qualidade, pois estes sistemas são responsáveis pela integridade de grandes massas de informações, junto a isso surge a necessidade de desenvolvimento rápido e com qualidade. Alguns avanços na Engenharia de Software têm contribuído para aumentar a produtividade no desenvolvimento de software, como: programação orientada a objetos, desenvolvimento baseado em componentes, engenharia de domínio, entre outros. Todos esses avanços são formas de alcançar o reuso de software [Alvaro et al. 2011].

Segundo Ezran [Ezran et al. 2002], alguns estudos na área de reuso têm mostrado que 40% a 60% de código é reutilizável de uma aplicação para outra, 60% do projeto e do código são reutilizáveis em aplicações de negócio, 75% das funções são comuns em mais de um programa, e somente 15% do código de um programa é único. Pode-se observar claramente com os dados mostrados o potencial do reuso de software. Isso tem impacto significante no processo de desenvolvimento de um software.

De acordo com Ezran [Ezran et al. 2002], existem três características fundamentais no reuso de software:

- Reuso é uma prática sistemática de desenvolvimento de software.
  Reuso deve acontecer de forma sistemática, isso significa que para sua implantação, uma série de estratégias técnicas e gerenciais devem ser definidas.
- Reuso explora semelhanças nos requisitos/arquitetura entre aplicações.
  O reuso de uma aplicação para originar outra deve possuir requisitos ou arquiteturas semelhantes.
- Reuso oferece benefícios na produtividade, qualidade e desempenho dos negócios.
  Reuso de software é uma técnica empregada para resolver a necessidade por melhora na qualidade e eficiência do desenvolvimento do software [Krueger 1992]. Qualidade e produtividade podem ser melhoradas reutilizando todas as formas de experiência comprovada, incluindo produtos e processos, bem como, modelo de qualidade e produtividade. Produtividade pode ser aumentada usando experiência existente, ao invés de criar tudo do início [Alvaro et al. 2011].

Com as informações já apresentadas, podemos definir reuso de software como o uso de conhecimento, artefatos e demais componentes de um software já existente e

desenvolvido em um padrão reutilizável para a geração de um novo, onde vantagens como produção acelerada, redução nos custos e qualidade no desenvolvimento são alcançadas.

1774 em 7/3/23

3. Reuso de software - evolução

2018 citações

A ideia de reuso de software surgiu em 1968 em uma conferência de Engenharia de Software com um artigo de McIlroy [McIlroy 1968]. A ideia inicial era utilizar o reuso de pequenas partes do software com o intuito de superar a crise no desenvolvimento de soluções que acontecera naquela época. Depois da ideia proposta por McIlroy, diversas pesquisas continuaram surgindo nos anos seguintes. Cada vez mais procuravam reusar partes maiores dos sistemas, sempre tomando como base indústrias de diversas áreas, como por exemplo, a automobilística [Alvaro et al. 2011].

Ainda baseado no pensamento de McIlroy, pesquisadores começaram a unir a capacidade de reuso com a utilização de objetos. Surgiu então o conceito de reuso de objetos. Ainda buscando aumentar esse conceito, cada vez mais eram reusadas partes maiores dos sistemas, na década de 1990 essa abordagem estendeu-se para os componentes, que estão em um nível de abstração mais alto que os objetos.

Com o crescimento dos sistemas distribuídos, tornava-se necessário uma maior comunicação entre eles. Surgiu assim o conceito de serviços, o qual viria a facilitar a troca de informação entre esses sistemas. Sempre com a busca de reusar sempre mais, surgiu na década de 2000 o reuso de serviços [SEI 2011]. Após o reuso de serviços, surgiu um novo conceito conhecido como famílias de softwares ou linhas de produto de software, que serão explicados com mais detalhe na sessão 5.

4. Vantagens e Obstáculos

O reuso traz inúmeras vantagens as diversas áreas em que pode ser utilizado, tal como as diversas engenharias, com composição de componentes como um padrão para um desenvolvimento mais robusto e consistente [Alvaro et al. 2011]. Ná área da computação o grande avanço que a Engenharia de Software vem tendo em relação ao reuso demonstra que os projetos tendem a reaproveitar todos os tipos de componentes, funções e subsistemas que forem possíveis. Para ter um melhor entendimento das vantajens que esse reaproveitamento traz segue abaixo uma descrição dos principais benefícios do reuso de *software*.

- Aumento da confiança na produção: Software reusado já foi usado e testado em ambientes e sistemas de trabalho e deve ser mais confiável que software novo, pois seus defeitos de projeto e implementação já foram encontrados, corrigidos e documentados [Alvaro et al. 2011].
- Gerenciamento de processo reduzido: O custo do software existente que vai ser reusado já é conhecido. Mesmo que avaliar os custos de desenvolvimento represente um desafio na avaliação do gerenciamento de projetos, esse conceito é importante, pois diminui a margem de erro na estimativa do custo de produção. Essa vantagem é ainda maior quando são reusados grandes subsistemas ou sistemas completos.
- Conformidade com padrões: Algumas funcionalidades e características presentes em projetos de interface ou ferramentas de trabalho para o usuário pode ser implementadas e reusadas como um conjunto pré-definido de padrões

[Ezran et al. 2002]. Com isso, os programas apresentam o mesmo formato de interface para o usuário, o que melhora a confiança do produto e o interesse do próprio.

Para conseguir alcançar as vantagens do reuso de *software* é preciso ter o processo de desenvolvimento voltado para esse conceito, senão sua utilização pode ter um efeito contrário ao esperado. Conseguir alcançar o correto processo de desenvolvimento é um dos obstáculos na utilização do reuso e os demais vão surgindo a medida que o desenvolvimento avança. Abaixo segue uma decrição dos princípais obstaculos na adaptação ao reuso.

- Custos de manutenção aumentados: Se o código fonte de um componente ou de um sistema reusável não estiver disponível, então os custos de manutenção podem ser aumentados, pois os elementos reusados do software podem tornar-se cada vez mais incompatíveis com as mudanças necessárias para adaptação ao novo sistema [Alvaro et al. 2011].
- Não produzido aqui: Os desenvolvedores tendem a reescrever componentes, pois acreditam que podem melhorá-los. Essa premissa é vista como uma responsabilidade que esses consideram de sua natureza, junto com o fato de considerarem a escrita original mais desafiadora e obrigação de um bom profissional.
- Criação e manutenção de uma biblioteca de componentes: Criar uma biblioteca de componentes reusáveis e assegurar aos desenvolvedores que podem usar essa biblioteca pode ser muito caro. A própria Engenharia de Software considera que as atuais técnicas de classificação, catalogação e recuperação de componentes de software são imaturas para garantir sucesso em tal iniciativa [Ezran et al. 2002].

Com a descrição das principais vantagens e dificuldades na utilização do reuso é possível ter uma visão mais detalhada sobre as técnicas e abordagens existentes sobre esse conceito, essas serão apresentadas na próxima seção.

5. Principais técnicas e abordagens para reuso de software

Existem várias técnicas e abordagens que conduzem e auxiliam na tarefa de reuso de software. Essas técnicas demonstram muitas vezes quais ações e estratégias a se tomar para que o desenvolvimento de um projeto seja feito em conformidade com os padrões de reuso, também como utilizar os conteúdos já existentes evitando o retrabalho. A seguir, serão apresentadas duas técnicas muito utilizadas para aplicar o reuso de software: Linhas de produto e engenharia reversa.

5.1. Linhas de produto de software

Quando McIlroy falou pela primeira vez em reutilizar componentes para construção de software, ele tinha a ideia de reusar pequenos artefatos de um sistema. Com o passar dos anos, pesquisadores perceberam que seria possível a reutilização não somente de pequenas partes, mas sim partes essenciais para o software. Com os avanços posteriores, chegaram à conclusão de que seria possível reusar a funcionalidade principal, ou seja, somente seriam desenvolvidos os componentes realmente específicos de cada domínio; chegaram então ao conceito de linha de produto de software.

Segundo [Alvaro et al. 2011] o processo de reuso de software iniciou por meio de reutilização de sub-rotinas na década de 1960, após isso, surgiram os desenvolvimentos

reutilizando módulos na década de 1970, os objetos na década de 1980, e componentes na década de 1990. Hoje o conceito de reuso extrapola as barreiras de componentes e passa para uma nova fase, as linhas de produto de software. A Fig. 1 ilustra esse progresso.

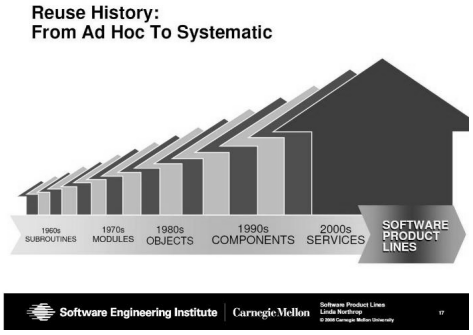

Figura 1. Evolução do reuso de software [SEI 2011]

O desenvolvimento por meio de linhas de produto de software é uma abordagem utilizada para alcançar o reuso de forma bastante abrangente (*reuse-in-the-large*). O uso dessa abordagem não possui a premissa de reusar apenas código, mas sim um conjunto de *assets* (características) do sistema. Segundo [Alvaro et al. 2011] são exemplos de *assets*:

- Requisitos e análise de requisitos;
- Modelo do domínio;
- Arquitetura e design do software;
- Documentação;
- Testes;
- Componentes.

Por ser um conceito bastante novo, muitas pessoas interpretam erroneamente a verdadeira proposta de linha de produto de software. Muitos pensam que é apenas um desenvolvimento baseado em componentes ou serviços, outros ainda podem achar que são apenas versões de um único produto. Mas, essa técnica é muito mais que isso, é utilizar de uma arquitetura principal (*core*) para criar novos produtos.

No desenvolvimento com linhas de produto, fica clara a relação entre as vantagens obtidas uma vez que os obstáculos do reuso são evitados, como é mencionado na seção 4. Um exemplo disso é o desenvolvimento de sistemas operacionais, tal como o Ubuntu ou Debian, ambos produzidos a partir de um core comum (Kernel Linux), outro exemplo é a família de produtos Adobe que conta com uma série de softwares desenvolvidos a partir de um modelo comum de arquitetura. Esses exemplos de produção salietam como a qualidade e desenvolvimento rápido podem ser alcançados em conformidade, uma vez que bibliotecas de reuso, repositórios e a documentação dos componentes do software estão bem estabelecidos.

5.2. Engenharia reversa e reengenharia

Quando se pensa em um processo de construção de software, três fases deste podem ser consideradas padrões: levantamento de requisitos onde se especifica o problema a ser resolvido junto com os objetivos, as restrições e regras de negociação; o projeto onde são

feitas todas as especificações da solução; e a implementação que contém a codificação os testes e a adaptação ao ambiente operacional.

O avanço progressivo através das fases citadas anteriormente, com alta abstração na fase inicial e baixa na fase final, constitui a tradicional técnica de engenharia progressiva, que tem por objetivo obter o sistema implementado do começo ao fim. Para obter informações de um software, componentes e todo o seu processo de construção é preciso seguir a ordem inversa entre as fases de construção mencionadas, partindo de um baixo até um alto nível de abstração, esse processo é chamado de engenharia reversa.

A engenharia reversa é uma área da reengenharia que pode ser considerada a reconstrução de algo do mundo real com o objetivo de entender a sua construção, buscando o entendimento necessário para melhorá-la [Vitharana 2003]. Dessa forma, ela torna-se uma importante técnica para auxiliar no reuso de software, uma vez que é possível obter componentes, funções e abstrações, que às vezes são mais altas que o próprio código fonte de um programa, o que permite obter diferentes visões a partir dos níveis de abstração de um sistema. Entre essas possíveis visualizações do software as principais são: Visão em nível implementacional, que abstrai características da linguagem de programação e características específicas da implementação e visão em nível de domínio, que abstrai o contexto em que o sistema está operando, ou seja, o porquê do sistema a ser desenvolvido.

Observando as etapas descritas anteriormente, o processo de engenharia reversa pode gerar códigos, artefatos e componentes que muitas vezes podem ser diferentes do sistema original, mas proporcionam ao desenvolvedor um conhecimento adicional sobre o problema gerando melhorias que não foram feitas no processo progressivo e que dão mais ênfase e qualidade ao reuso. Com isso, as técnicas de engenharia são proveitosas também em completar a documentação dos novos sistemas desenvolvidos, o que contribui para aumentar a qualidade do reuso diminuindo alguns dos obstáculos, tais como falta de documetação e informações pertinentes a lógica do sistema.

6. Reuso de software - opiniões e fatos dos que utilizam

Frakes realizou uma pesquisa com 113 pessoas de 28 organizações europeias com o intuito de verificar atitudes, crenças e práticas de reutilização de código. A Fig. 2 resume as dezesseis questões sobre reuso com base no levantamento.

O estudo citado anteriormente, resultou em um livro sobre Engenharia de Software, onde o autor relata 55 fatos que têm grande importância no processo de desenvolvimento de software. Dentre esses fatos, destacam-se 3 que estão associados com o reuso de software, como seguem [Alvaro et al. 2011]:

- Fato 1: *Reuse-in-the-small* (bibliotecas e rotinas) começou há quase 50 anos e é um problema bem resolvido.
- Fato 2: *Reuse-in-the-large* (componentes) continua sendo um problema não resolvido, embora todos concordem que é importante e desejável.
- Fato 3: Existem duas regras no processo de reuso. Primeira: É três vezes mais difícil construir componentes reusáveis do que construir componentes não reusáveis. Segunda: Um componente reusável deve ser testado em três aplicações diferentes antes de ser considerado uma biblioteca de reuso.

Questões	*Respostas*	*Obs.*
1. Quais propriedades/vantagens do reuso são muito utilizadas?	Várias	Algumas (ex., Ferramentas Unix) são amplamente reusados, outros (ex., Cosmic) não são.
2. Linguagens de programação afetam o reuso?	Não	
3. Ferramentas *CASE* promovem reuso?	Não	
4. Desenvolvedores preferem construir do zero ou reusar?	Preferem reusar	
5. Existe percepção de viabilidade econômica ?	Sim	
6. Educação sobre reuso influencia em sua prática?	Sim	Treinamentos corporativos são essenciais.
7. Experiência em Engenharia de Software influencia no reuso?	Sim	
8. Recompensas e reconhecimento aumentam o reuso?	Não	
9. Software comum promove reuso?	Provavelmente	Entrevistados disseram que não, mas reuso sugere níveis que ajudam no processo.
10. Problemas jurídicos dificultam o reuso?	Não	Pode alterar no futuro.
11. Repositório de reuso melhora o reuso de código?	Não	
12. Reuso é mais comum em certas indústrias?	Sim	Mais comum em telecomunicação e menos em aeroespacial.
13. Companhias, divisões ou tamanho de projetos são significado de reuso dentro das organizações?	Não	
14. Preocupações com qualidade inibem reuso?	Não	Podem alterar no futuro.
15. Organizações medem reuso, qualidade e produtividade?	Geralmente não	
16. Medida (Métricas) de reuso influencia no reuso?	Não	Medidas geralmente não são usadas.

Figura 2. Levantamento da utilização de reuso de software [Alvaro et al. 2011]

7. Reuso de software na prática

Para demonstrar um pouco das vantagens do reuso de software na prática, serão descritos os relatos de duas grandes empresas de tecnologia com suas histórias, motivações e ganhos com a incorporação do conceito de reuso de software em suas aplicações.

A IBM (International Business Machines), uma das maiores empresas em desenvolvimento de software, é uma das pioneiras na produção voltada para o reuso de software [Alvaro et al. 2011]. Em 1981 a empresa criou o Reusable Parts Technology Center, um grupo que tinha o objetivo de desenvolver tecnologias para o reuso. Entre as primeiras dificuldades encontradas a principal foi a falta de um design de linguagem comum, que colocaria o trabalho de todos sobre um padrão único; com as pesquisas eles identificarão os padrões existentes entre as estruturas de dados, funções e módulos que eram desenvolvidos.

O primeiro projeto da empresa totalmente voltado para o reuso foi desenvolvido explorando o uso de abstração de dados. Dessa forma, eles construíram sete tipos de funções abstratas chamadas *building blocks* que representavam 1/3 da quantidade de linhas do software [Alvaro et al. 2011]. O bom aproveitamento da abstração de dados fez com que o grupo sumarizasse um modelo reusável baseado nesse conceito. Com isso, desenvolveram também ferramentas de modelagem que davam suporte ao reuso lançando-as em forma de componentes e bibliotecas para o C++.

A empresa HP (Hewlett-Packard) começou a implantar a política de reuso com o objetivo inicial de aumentar a produtividade e a consistência dos softwares desenvolvidos, com isso iniciou uma corporação de reuso que implantaria um programa para difundir o conceito. O programa de reuso corporativo mirava a criação de um pacote de melhores práticas para desenvolvimento e um guia para evitar as falhas mais comuns nesse processo. À medida que o conceito foi se espalhando pela empresa eles conseguiram uma mudança organizacional e criação de uma mentalidade voltada para o reuso.

Os avanços conseguidos pela HP com a implantação da política de reuso de software fizeram com que ela divulgasse um estudo para outras empresas, onde eram demonstradas as providências necessárias para auxiliar cada setor a incorporar o conceito [Alvaro et al. 2011]. Por exemplo, no setor de desenvolvimento foi demonstrado que a produção de componentes envolve diversas tarefas como análise de domínio e testes,

com isso, era necessária uma organização que padronizasse tais componentes. Atualmente a HP conta com um modelo para desenvolvimento com reuso que é usado dentro da empresa, mas para atingir uma maturidade suficiente foi preciso investimento sobre a estrutura e treinamento dos desenvolvedores.

8. Conclusão

O conceito de reuso de software vem crescendo com o passar dos anos e tornando-se um padrão que acelera e qualifica o desenvolvimento. Todo esse crescimento vem sendo apoiado nas diversas técnicas e abordagens que estão cada vez mais preparadas para serem implementadas nos diferentes tipos de construção de software. Sua evolução é marcada pelas necessidades que os desenvolvedores começaram a observar em relação ao reaproveitamento de trabalho e as atualizações que surgiam em relação às novas linguagens de programação e padrões de projeto.

Abordagens como linhas de produto e engenharia reversa demonstram a efetividade e evolução do reuso de software ao longo dos anos. Devido a muitas inverdades que foram sendo criadas sobre esse conceito, ainda existe um atraso em relação ao seu uso por muitos profissionais que relutam em aderi-lo, mas com os resultados práticos, junto ao uso e incentivo cada vez maior por parte das grandes empresas tecnológicas, o reuso de software vem conseguindo superar essas dificuldades e marcar o seu território na computação em geral.

Como qualquer abordagem, o reuso de software é uma doutrina que leva certo tempo e necessita de dedicação e empenho por parte dos desenvolvedores para ser corretamente incorporado e ter um bom retorno. Como já foi mencionado nos relatos de grandes empresas como a IBM e HP, essa adaptação conta com processos de produção bem detalhados, mudança em relação à mentalidade do desenvolvimento com ênfase na utilização de componentes e subsistemas reusáveis, além do tempo de retorno que no início é mais longo, mas que traz inúmeras vantagens a qualquer projeto junto com boas práticas na sua construção.

Referências

Alvaro, A., Lucredio, D., de Almeida, E. S., Macena, J. C. C. P., Nascimento, L. M., Buregio, A., Garcia, V. C. e Meira, S. L. (2011). ”CRUISE - Component Reuse In Software Engineering”. Disponível em http://cruise.cesar.org.br/index.html. Acessado em 24/03/2011.

Ezran, M., Morisio, M. e Tully, C. (2002). ”Practical software reuse”. *Springer, Inglaterra*.

Krueger, C. (1992). ”Software reuse”. *ACM Computing Serveys*, V. 24, N. 02 p.131-183.

McIlroy, M. D. (1968). ”Mass produced software components”. *NATO Software Engineering Conference Report*, p. 79-85.

SEI (2011). Software Engineering Institute. Disponível em http://www.sei.cmu.edu/. Acessado em 19/03/2011.

Vitharana, P. (2003). ”Risks and challenges of component-based software”. *ICSE - International Conference on Software Engineering*.