Revista INTERFACES
SAÚDE, HUMANAS E TECNOLOGIA

Vol. 4(13), pp. 243-253, 01 de Dezembro, 2017

DOI: 10.16891/2317-434X.v4.e13.a2017.pp243-253
ISSN 2317-434X

http://www.interfaces.leaosampaio.edu.br

UNILEÃO

DIFERENÇAS ENTRE ENGENHARIA REVERSA E REENGENHARIA NOS SISTEMAS DE INFORMAÇÃO

DIFFERENCES BETWEEN REVERSE ENGINEERING AND REENGINEERING IN INFORMATION SYSTEMS

BARBOSA[1], Pedro Luis Saraiva
Faculdade de Análise e Desenvolvimento de Sistemas Leão Sampaio-FLS/Juazeiro do Norte-CE, Brasil

CANDIDO, Adriano Lima
Faculdade de Análise e Desenvolvimento de Sistemas Vale do Salgado-FVS/Icó Ceará - CE, Brasil

Recebido em: 09/08/2017; Aceito: 20/11/2017; Publicado: 01/12/2017

Resumo: As tecnologias e técnicas de criação de sistemas estão em constante evolução, os sistemas legados inflexíveis devem acompanhar esse avanço, ou podem tornar-se obsoletos ou até extintos no mercado. Sendo assim, é necessário a compreensão de algumas formas de acompanhar essa evolução, podendo fazer uso das técnicas de engenharia reversa ou reengenharia. Em razão da semelhança entre as técnicas de engenharia reversa e reengenharia pode surgir a dúvida entre qual melhor se adequa ao problema proposto, e a escolha da técnica errada poderá ocasionar custos extras e desnecessários. O objetivo desta pesquisa é esclarecer os conceitos sobre as técnicas de engenharia reversa e reengenharia através de uma revisão sistemática da literatura, definindo suas diferenças no que diz respeito a recriação de sistemas ou documentação de sistemas legados. Como o tema tratado neste artigo necessita que as informações sejam catalogadas, organizadas, criticadas e transformadas em conhecimento, e a intenção é dissertar sobre diferentes perspectivas, será utilizado como metodologia uma revisão sistemática (RS), a qual permite buscar diversas pesquisas, seja pesquisas bibliográficas, experimentais ou outras. A ideia é ter uma visão ampla do assunto e estabelecer novos pensamentos. Para a inclusão dos artigos, foram selecionados aqueles que apresentaram texto coerente e pesquisas na área da engenharia de software voltadas para a engenharia reversa e reengenharia.
Palavras Chaves: Engenharia Reversa, Reengenharia, Sistemas Legados.

Abstract: The technologies and techniques for creating systems are constantly evolving, inflexible legacy systems must accompany this advance, or can become obsolete or even extinct in the market. Therefore, it is necessary to understand some forms of this development and can make use of reverse engineering techniques or reengineering. Because of the similarity between the reverse engineering techniques and reengineering may arise doubt among which best suits the proposed problem, and choosing the wrong technique can cause additional and unnecessary costs. The objective of this research is to clarify the concepts on reverse engineering techniques and reengineering through a systematic literature review, defining their differences regarding the recreation systems or documentation of legacy systems. As the subject covered in this article requires that the information is cataloged, organized, critiqued and transformed into knowledge, and the intention is to speak about different perspectives, will be used as a systematic review methodology (RS), which allows searching various research, is research bibliographical, experimental or otherwise. The idea is to have a broad view of the subject and establish new thoughts. For the inclusion of articles, we selected those with coherent text and research in software engineering area aimed at reverse engineering and reengineering.
Keyword: Reverse Engineering, Reengineering, Legacy systems.

[1] Av. Monsenhor Frota, 609. Icó-CE. Email: pedroluis@leaosampaio.edu.br

INTRODUÇÃO

Ao passar dos dias, pode-se perceber que cada vez mais os softwares estão tomando conta do meio em que vivemos, para Pfleeger (2007), nos dias atuais os softwares estão presentes explicitamente ou mesmo sem se fazer notar, em todos os aspectos de nossas vidas, inclusive nos sistemas críticos e complexos que podem afetar nossa saúde e bem-estar, por essa razão, a engenharia de software tornou-se mais importante do que nunca, ressalta ainda que boas práticas de engenharia de software deve assegurar que o software tenha uma boa, grande e positiva contribuição para as vidas dos cidadãos.

A engenharia de software surgiu da necessidade de criar sistemas com qualidade e de forma mais ágil, visando atender as exigências do mercado. Para Falbo (2005), a engenharia de software trata de aspectos relacionados ao estabelecimento de processos, métodos, técnicas, ferramentas e ambientes de suporte ao desenvolvimento de software. Pressman (2011), complementa dizendo que softwares, em todas as suas formas e em todos os seus campos de aplicação, deve passar pelos processos de engenharia de software, ainda ressalta que para que a engenharia de software funcione, deve ser definido processos, e que a partir destes será possível desenvolver um sistema de forma clara e dentro do prazo.

Para manter um software útil, durável e que se adeque a necessidade do cliente, pode-se fazer necessário o uso de algumas ramificações da engenharia de software, como a manutenção de sistemas. Para Sommerville (2011), o desenvolvimento de software não acaba quando o sistema é implantando, mas continua por toda a vida útil do sistema, argumenta ainda que depois que o sistema é implantado, para que ele se mantenha usável, é inevitável que ocorram mudanças, uma vez que ocorrem alterações nos negócios e expectativas dos usuários constantemente, e consequentemente essas modificações geram novos requisitos para o software.

Pode-se afirmar que um sistema bem sucedido não necessariamente tem que ser um sistema criado em uma linguagem de programação robusta, a prova disso é que vários softwares que fazem sucesso hoje no mercado, foram desenvolvidos em linguagens de programação que atualmente são obsoletas, usando métodos antigos e técnicas fracas e que dificilmente possuem um projeto.

Para Pinto e Braga (2004, p. 1), sistemas legados são sistemas críticos e que estão em uso a um longo período, foram desenvolvidos em tecnologias possivelmente ultrapassadas, porém são peças de suma importância para a organização, complementa dizendo que em razão da constante evolução da tecnologia estes sistemas não conseguem acompanhar esse processo, visto que estão internamente programados de uma forma tão antiga que compromete a agilidade nesse processo de readaptação, impactando diretamente o tempo necessário para que seja realizada a modificação para novas tecnologias, diz também que boa parte destes, não possuem ao menos um projeto que condiz com a realidade atual do software, dificultando e atrasando ainda mais esse processo. Na intenção de aumentar o desempenho e diminuir o tempo gasto na manutenção, são necessárias algumas técnicas para análise de código fonte, para ser possível o entendimento do funcionamento do sistema e a partir disso, criar um projeto de software, que será usado para desenvolvimento de um sistema semelhante, que deverá atender as mesmas regras e procedimentos, esses processos são denominados engenharia reversa e reengenharia.

Os conceitos de engenharia reversa e reengenharia apesar de parecidos, resultam em

artefatos completamente distintos, essas técnicas são muito utilizadas quando é necessário manter a utilização de um sistema legado, pois, apesar de antigos, atendem a necessidade da organização, contudo, dificulta o processo de manutenibilidade e evolução impactando diretamente no custo. Sendo assim, pode-se fazer necessário a redocumentação e/ou recriação do sistema para tecnologias atualizadas que podem diminuir o tempo de manutenção e/ou permitir um maior desempenho e performance, aumentando o tempo de vida do sistema.

Quais as diferenças entre reengenharia de sistemas e engenharia reversa que são técnicas da área de manutenibilidade de software, e se confundem pela semelhança em suas palavras?

O objetivo desta pesquisa é esclarecer os conceitos sobre as técnicas de engenharia reversa e reengenharia através de uma revisão sistemática da literatura, para que através de suas diferenças, possa auxiliar na decisão de qual das técnicas usar em um determinado sistema legado, onde faz-se necessário analisar o software para criar uma documentação e reprojetar o sistema para melhorar a qualidade, desempenho e manutenção.

Justifica-se esta pesquisa ao fato da compreensão entre os conceitos das técnicas de engenharia reversa e reengenharia e pela semelhança entre estas palavras, pois como resultam em objetivos totalmente diferentes, a escolha de uma técnica errada, resultaria em um artefato inadequado, ocasionando perca de tempo e consequentemente gerando um custo maior, dado que a despesa de um processo de engenharia reversa é bem menor que o de uma reengenharia, e como ambas são independentes, a escolha entre qual usar pode ser um fator importante para o sucesso da manutenção e evolução.

DESENVOLVIMENTO

Engenharia Reversa

Dentro do contexto de criação de sistemas, manutenção e evolução são itens presentes em boa parte dos sistemas bem sucedidos. Ao passar dos dias pode-se perceber que o mercado passa por modificações constantes e é necessário que esses sistemas se adaptem a estes novos cenários e visões de mercado.

Porém, na prática, a manutenção e evolução pode se tornar algo complicado e caro de se realizar, Pfleeger (2007), entende que os custos e tempo investido em manutenções e evolução tende a crescer conforme o período que um determinado sistema está no mercado.

A manutenção e evolução pode se tornar financeiramente inviável quando não se tem a documentação do sistema, pois qualquer modificação poderá impactar em outras funcionalidades do software ou até mesmo no tempo, pois poderá ser maior para modificação de uma determinada função. Neste caso podese fazer uso da engenharia reversa para documentar um sistema que já está em funcionamento, no intuito de criar uma documentação para entender seu funcionamento e facilitar as alterações.

De acordo com Pressman (2011, p. 699)

> O termo engenharia reversa tem suas origens no mundo do hardware. Uma empresa desmontar um produto de hardware competitivo na tentativa de conhecer os “segredos” de projeto e fabricação do concorrente. Os segredos poderiam ser facilmente entendidos se fosse possível obter as especificações de projeto e fabricação do concorrente. Mas esses documentos são de propriedade privada e não estão disponíveis para a empresa que está fazendo a engenharia reversa. Essencialmente, uma engenharia reversa bem-sucedida resulta em uma especificação de projeto e fabricação para

> um produto pelo exame de amostras atuais do produto.

Já a engenharia reversa na área de sistemas, surgiu da necessidade de compreender o funcionamento e técnicas usadas na criação de um determinado sistema a partir do estudo de seu código fonte. Onde o resultado deste estudo poderá resultar em uma documentação com uma visão abstrata dos comportamentos e ações do mesmo, podendo diminuir o impacto da manutenção e evolução, já que é bastante arriscado alterar um sistema sem saber o que essa modificação poderá provocar.

Para Pressman (2011), a engenharia reversa na área de sistemas é bem similar ao hardware, reforça dizendo que ao contrário do hardware, é mais comum se deparar com a engenharia reversa sendo realizada no sistema da própria empresa, ainda ressalta que há segredos a serem entendidos, visto que nenhuma 3 especificação foi desenvolvida, em seguida, afirma que a engenharia de software é um processo usado para analisar um programa na tentativa de criar uma representação gráfica do sistema em um nível mais alto de abstração do código fonte em questão.

Sommerville (2011), menciona que a engenharia reversa serve para analisar um programa onde a partir dessa análise se possa abstrair e extrair informações que serão usadas para produzir a documentação da organização e funcionalidades de um determinado sistema.

Cognin (1999), repassa uma visão de negócio bastante ampla, pois ela define que a engenharia reversa foi criada para amenizar os problemas enfrentados com a substituição de um sistema legado, pois a troca seria financeiramente e operacionalmente inviável, manter o software atual seria o mais indicado, por que todos já estavam acostumados com ele, e a troca, poderia ocasionar lentidão na organização, já que seria necessário um treinamento e adaptação dos usuários ao novo sistema, com a engenharia reversa seria possível evoluir e manter o sistema através da alteração do próprio sistema, pois com o artefato gerado pela engenharia reversa o sistema poderia ser alterado sem gerar grandes impactos em sua estrutura, eliminando assim o custo que seria necessário com a troca do mesmo.

Pfleeger (2007), possui uma visão um pouco diferente, ele define que a engenharia reversa é semelhante a redocumentação, pois fornece informações da especificação e do projeto de um determinado sistema a partir de seu próprio código fonte, fortalece dizendo que a engenharia reversa vai além disso, pois ela busca recuperar informações e dados de engenharia com base em métodos da especificação de um projeto de sistema, lembra também que é necessário que essas informações sejam organizadas de uma forma que seja possível consultá-las e manipulá- las em outro momento, ressalta alertando que as informações extraídas não são necessariamente completa, pois muitos componentes originais estão geralmente associados a um ou mais componentes de projeto, e que por essa razão, um sistema criado a partir de engenharia reversa, pode na verdade ter menos informações que o sistema original.

Pode-se afirmar que para realizar uma engenharia reversa é necessário um alto nível de abstração e entendimento de projetos e códigos de sistemas, pois segundo Pfleeger (2007, p. 408), “A chave para a engenharia reversa é a habilidade de abstrair especificações a partir da implementação detalhada do código-fonte”.

Ainda segundo Pfleeger (2007), pode-se definir uma engenharia reversa bem sucedida quando as expectativas são pequenas. Ou seja, quando as ferramentas são capazes de determinar todos os

elementos de dados e as chamadas relacionadas a um componente específico. Onde podem exibir a estrutura de um sistema complexo e pode reconhecer inconsistência e violações de padrões de projetos.

Reengenharia

Ao passar dos anos é notável a evolução do mercado e as constantes modificações em regras, legislações e até mesmo surgimento de novas visões de mercado das empresas. Os sistemas devem evoluir de acordo com essas alterações, porém nem sempre é possível, pode chegar um ponto que o software atinge seu limite e qualquer alteração por mais simples que seja, se torna inviável. Pois uma simples implementação de uma determinada funcionalidade poderá ocasionar problemas no sistema inteiro, prejudicando as atividades da empresa ou até impossibilitando realiza-las.

Daí então vem a necessidade da readaptação do software a uma nova linguagem ou plataforma, podendo fazer uso da sua própria documentação, ou seja, recriar um sistema em uma linguagem que tenha facilidade de adaptação a novas tecnologias, permitindo um maior tempo de vida do sistema a partir do atual. Este processo é denominado reengenharia de software.

Para Pressman (2011), não é novidade que um software chegue a um ponto que não consiga mais ser mantido e que cada vez mais é dada uma maior ênfase sobre a reengenharia de software em razão dos problemas de manutenção e evolução criados por mais de quatro décadas.

Ainda segundo Pressman (2011), reengenharia toma muito tempo, e possui um custo financeiro bastante alto e acaba absorvendo recursos que poderiam ser usados em necessidades mais imediatas. Por estas razões a reengenharia pode não ser realizada em alguns meses ou anos. E completa afirmando que todas as organizações precisam de uma estratégia pragmática para reengenharia de software.

Já para Sommerville (2011), sistemas mais velhos, são difíceis de serem compreendidos e mudados, pois ao longo dos anos, com as modificações, o sistema sofreu bastante alteração e que a estrutura inicial pode ter sido danificada em razão dessa série de mudanças.

Apesar de ser altamente recomendável a troca do sistema, nem sempre é viável para a organização. Financeiramente o custo é bastante elevado e a demora com à familiarização com o novo sistema, poderia acarretar um atraso na execução das tarefas da empresa. Pois os usuários deveriam receber treinamentos de como se portar diante da nova aplicação, neste caso, poderia ser aplicado técnicas de reengenharia no próprio sistema, descartando a possibilidade de criação de um novo software.

Para Sommerville (2011), fazer com que os sistemas legados sejam mais simples e fáceis de serem mantidos, é necessário fazer uso da reengenharia nesses sistemas, pois através dela, é possível melhorar a sua estrutura e inteligibilidade. Reforça dizendo que a reengenharia pode envolver a redocumentação do sistema que é um artefato gerado da engenharia reversa, cita também que seria ideal a refatoração da arquitetura e mudança da linguagem de programação para uma linguagem moderna, que permita modificações e atualizações da estrutura e dos dados do sistema. Alerta também que a funcionalidade do software não pode ser alterada, e que deve ser evitado grandes mudanças na arquitetura do sistema.

Já Jesus, Fukuda e Prado (2000), afirmam que na maioria dos casos, os sistemas legados não possuem

documentação, e mesmo que possuam, estas podem estar desatualizadas, e não condizente com a realidade atual do sistema. Onde o problema poderia ter sido ocasionado pelas modificações do sistema, e que não foram adicionados à documentação. Isso dificulta o processo de manutenção do sistema. Ainda sugere que seriam necessárias técnicas de reengenharia de software para reconstruir partes do sistema baseado no código fonte.

Pode-se perceber que o processo de reengenharia requer bastante tempo, dedicação e acima de tudo conhecimento do sistema proposto e em questão. As regras e restrições das funcionalidades devem ser atendidas à risca, procurando reduzir ao máximo o impacto destas alterações. Pois a reestruturação de uma determinada parte do código, poderia impactar em diversas outras funcionalidades que dependiam da que foi modificada. Por esta razão é recomendável estudar cautelosamente a documentação, seus relacionamentos e dependências, para evitar ao máximo tal problema.

Pfleeger (2007), justifica o uso da reengenharia pelo fato de tornar o código do software mais fácil de ser entendido e modificado. Completa dizendo que são necessárias algumas regras de transformação para simplificar a representação interna, reitera afirmando que pode-se fazer necessário o uso de ferramentas para determinar a manutenibilidade do código para avaliar os efeitos que serão causados após a reestruturação, afim de provar que após a mesma, houve diminuição na complexidade

METODOLOGIA

Para Lakatos (2005 apud ANDER-EGG, 1978, p. 28), a pesquisa caracteriza-se como um "procedimento reflexivo sistemático, controlado e crítico, que permite descobrir novos fatos ou dados, relações ou leis, em qualquer campo do conhecimento". A pesquisa é um procedimento formal que leva a caminhos para conhecer a realidade ou para descobrir verdades parciais

Para Lakatos (2005), a pesquisa compreende seis passos:

1- Seleção do tópico ou problema para a investigação.

2- Definição e diferenciação do problema.

3- Levantamento de hipóteses de trabalho.

4- Coleta, sistematização e classificação dos dados.

5- Análise e interpretação dos dados.

6- Relatório do resultado da pesquisa.

Como o tema tratado neste artigo necessita que as informações sejam catalogadas, organizadas, criticadas e transformadas em conhecimento, e a intenção é dissertar sobre diferentes perspectivas, será utilizado como metodologia uma revisão sistemática (RS), que segundo Sampaio e Macini (2007), é uma pesquisa baseada na literatura sobre determinado tema.

Para Bertozolli et. al (2010), RS é uma metodologia rigorosa, que serve para encontrar vários estudos sobre um determinado tema, através de buscas explicítas, sempre verificando a qualidade e validade dos estudos encontrados nas bases de dados escolhidas.

Pode-se resumir RS como uma sintetização de vários estudos, onde este método permite buscar diversas pesquisas, seja pesquisas bibliográficas, experimentais ou outras. A ideia é ter uma visão ampla do assunto e estabelecer novos pensamentos.

Para a inclusão dos artigos, foram selecionados aqueles que apresentaram texto coerente e

pesquisas na área da engenharia de software voltadas para a engenharia reversa e reengenharia. Foram utilizadas bases de dados como: Revista de Informática Aplicada[2], RITA- Revista de Informática Teórica e Aplicada[3], Revista de Computação e Tecnologia[4], IEEE[5] e SBC Journal on Interactive Systems[6].

Os artigos que foram selecionados possuem datas de publicação dos últimos 20 anos, são artigos completos e publicados em língua portuguesa, como critério de exclusão foram estabelecidos: artigos incompletos. Os descritores para as buscas foram: engenharia reversa, ingeniería inversa, reverse engineering, reengenhaira, reingeniería e reengineering.

ANÁLISE DOS RESULTADOS

Foram analisados 6 artigos publicados em língua portuguesa obedecendo aos critérios de inclusão e exclusão escrito na seção anterior, que trata do método de pesquisa. Para análise dos artigos foi realizado uma investigação sobre dois assuntos: engenharia reversa e reengenharia. Abaixo é apresentado um quadro (Quadro 1), onde é exposto as pesquisas encontradas. Os artigos estão identificados como A1, A2, A3, etc. Logo após, é exibida a referência do artigo, em outra coluna o ano de publicação, na seguinte a metodologia utilizada, e por último uma síntese deste.

O QUE DIZ A REVISÃO? UM OLHAR SOBRE A REENGENHARIA E A ENGENHARIA REVE A1

Através de uma revisão dos principais artigos da área de engenharia de software, procuram fundamentar os conceitos de engenharia reversa e reengenharia através de uma pesquisa bibliográfica, explica também o que, como, e quando usar essas técnicas em sistemas legados, e ainda destinam a pesquisa ao apoio pedagógico. Ao decorrer da pesquisa, os mesmos evidenciam que os objetivos das técnicas de engenharia reversa e reengenharia são completamente distintos. Completa dizendo que o objetivo da engenharia reversa é criar o projeto e especificação de um sistema, partindo-se de seu código fonte. Já o objetivo da reengenharia é produzir um sistema novo a partir do que existe ou melhorá-lo, possibilitando maior facilidade de manutenção. Reforça dizendo que o processo de engenharia reversa pode ser usado como parte do processo de reengenharia, pois fornece um entendimento maior do sistema a ser melhorado ou construído.

A2 concordam com A1, pois, através de um estudo de caso aplicando a reengenharia em um software científico de cálculo de capacidade de carga e adaptando-o para a linguagem de programação Java, conseguiram concluir que o objetivo do processo de reengenharia é a ação de analisar e modificar um sistema para recriá-lo e reimplementá-lo com uma nova estrutura, já o objetivo da engenharia reversa, é analisar um sistema procurando identificar os componentes destes, para criar uma representação do software em um nível mais alto de abstração, a partir de seu código fonte.

[2] http://seer.uscs.edu.br/index.php/revista_informatica_aplicada

[3] http://seer.ufrgs.br/rita/index

[4] http://revistas.pucsp.br/index.php/ReCET/about

[5] http://www.ewh.ieee.org/reg/9/etrans/esp/index.html

[6] http://seer.ufrgs.br/jis/index

Quadro 1 – Artigos Pesquisados

Id	Referência do Artigo	Ano	Metodologia	Síntese
A1	PIEKARSKI, A. E. T; QUINÁIA, M. A. Reengenharia de Software: o que, por quê e como. Departamento de Informática, Guarapuava-PR.	2000	Pesquisa Bibliográfica	Este artigo tem como objetivo fornecer um embasamento sobre o processo de manutenção de software, mais especificamente sobre reengenharia e engenharia reversa, sendo destinado basicamente para apoio didático a essa parte da Engenharia de Software. Devido a isso, são apresentadas as definições - o "o que" são essas modalidades de manutenção, os casos em que se aplicam − o "por que" utilizá-las e a forma de adotá-las − o "como" realizá-las.
A2	LIRA, A. M.; PERES, V. C.; MORAIS, E. C.; COUTO, W. O. Reengenharia de Software Científico para Cálculo de Capacidade de Carga, Amazonas.	2012	Estudo de Caso	Este artigo apresenta um estudo de caso utilizando a Reengenharia aplicado a um software cientifico de Cálculo de Capacidade de Carga, sendo apresentados as etapas da reengenharia e as vantagens de utiliza-la na melhoria de softwares científicos.
A3	PENTEADO, R. A. D.; GERMANO, F. S. R.; MASIERO, P. C. Engenharia Reversa Orientada a Objetos do Ambiente StatSim: método utilizado e resultados obtidos, São Carlos.	1995	Estudo de Caso	Este artigo tem como objetivo apresentar uma abordagem concreta de engenharia reversa utilizando o método Fusion, a partir de um sistema não orientado a objetos, para obter seu documento de projeto, e através disso, fazer uso da reengenharia para transformá-lo em um sistema orientado a objetos utilizando o ambiente Statsim.
A4	JESUS, E. S.; FUKUDA, A. P.; PRADO, A. F. Reengenharia de Software para Plataformas Distribuídas Orientadas a Objetos, São Carlos.	2000	Estudo de Caso	Este artigo apresenta um estudo de caso sobre uma estratégia para a reengenharia de software que reconstrói sistemas legados tornando-os operacionais em outras plataformas de hardware e software, e que a partir do código fonte o mesmo é reorganizado com os conceitos de Orientação a Objeto.
A5	AMORIM, T. A.; BRUNETTO, M. A. O.; KASTER, D. S.; FERRACIOLI, F. Remodelagem do Software SACAR-Web Usando Técnicas de Engenharia Reversa e Reengeharia de Software, Londrina.	2006	Estudo de Caso	O presente artigo tem como objetivo fazer a remodelagem da aplicação SACAR-Web (Software para Avaliação Cardiorrespiratória para Web) usando técnicas de engenharia reversa e reengenharia de software para que a manutenção da mesma se tornasse uma tarefa mais simples.
A6	NOVAIS, E. R. A; PRADO, A. F. Reengenharia de Software Orientada a Componentes Distribuídos, São Carlos.	2000	Estudo de Caso	Este artigo apresenta um estudo de caso o qual aborda uma estratégia de Reengenharia de Software Orientada a Componentes para Softwares Legados, sendo tal estratégia dividida em 5 passos concluindo estes com as especificações em Catalysis as quais devem ser transformadas para uma linguagem de programação orientada a objetos, resultando na implementação final do sistema.

A3 concordam com conceitos abordados por A1 e A2, pois através de um estudo de caso aplicado a um sistema não orientado a objetos, que passou a ser orientado a objetos com o uso das técnicas de engenharia reversa e reengenharia e fazendo uso do ambiente de desenvolvimento StartSim, afirmam que o termo engenharia reversa refere-se à recuperação da arquitetura e do modelo de análise de um sistema existente, baseada em observações sobre o comportamento do software, fazendo uso do código fonte e eventualmente documentos disponíveis. Já a reengenharia, existe quando a partir do modelo produzido por uma

atividade de engenharia reversa o sistema possa ser construído com um processo normal de engenharia.

Já os autores A4 discordam da abordagem do conceito de manutenção na reengenharia citados por A1, A2 e A3, pois através de um estudo de caso aplicado a um sistema para oficina auto elétrica e mecânica, fazendo uso da abordagem Fusion/RE, efetuaram a conversão do código fonte do sistema que era até então orientado a procedimentos, definiram que o objetivo da reengenharia de software é facilitar sua evolução disciplinada, desde o estado corrente até o novo estado desejado, descartando a ideia de que a reengenharia pode ser usada para melhorar a manutenibilidade do sistema, mas concordam com a abordagem do conceito de engenharia reversa é agir nos diferentes níveis de abstração do ciclo de vida do software, onde tem início no código fonte do sistema legado, recuperando ou recriando seu projeto, compreendendo os requisitos que foram implementados.

Já A5 concordam com todos os conceitos abordados por A1, A2 e A3, discordando apenas dos conceitos de reengenharia de A4, quando dizem que a reengenharia não pode ser utilizada para melhorar a manutenibilidade do sistema, pois através de um estudo de caso aplicado a remodelagem do sistema Sacar-Web usando técnicas de engenharia reversa e reengenharia de software na intensão de facilitar o processo de manutenção do software, pode-se identificar que a reengenharia de software pode ser utilizada com o objetivo de remodelagem de um sistema existente ou para criação de um novo, podendo usar técnicas de engenharia reversa para análise de código fonte, pois através desta, seria possível um documento que condiz com a realidade do sistema, complementando a reengenharia de determinadas funcionalidades deste software, na intensão de facilitar sua manutenção.

A6 concordam com os conceitos apresentados por A1, A2, A3 e A5, e assim como os demais, discorda da abordagem de A4 quando diz que reengenharia não pode ajudar na manutenibilidade do sistema, concordando apenas com o conceito de engenharia reversa por este apresentado, pois através de um estudo de caso aplicado a um sistema legado, apresentaram uma estratégia de conversão de sistemas orientados a procedimentos, para sistemas orientados a componentes distribuídos, com o objetivo de reconstruir sistemas legados escritos em linguagem procedural, para serem executados em plataformas mais modernas, facilitando a manutenção através do reuso de componentes distribuídos, utilizando o método Fusion/RE assim como A4 acrescentando o método Catalysis, com isso, conseguiram definir que a reengenharia de software pode ser uma forma de reuso a qual permite adquirir o entendimento da aplicação, sendo possível recuperar informações de etapas como análise de projeto e reorganizar de forma coerente para que possa ser reutilizável. Afirma ainda que a engenharia reversa pode estar sendo utilizada no ato de entender o código e restituir sua estrutura para uma reconstrução do sistema reutilizando das funcionalidades do software legado.

Braga (2013, p. 6-24), afirma que a engenharia reversa é de grande importância para a manutenção de sistema, e pode possibilitar uma futura reengenharia, deixa claro também que ambas são independentes, porém podem se complementar, podendo resultar em um produto com mais qualidade e sem grandes alterações, entende ainda que o objetivo da engenharia reversa é extrair informações do sistema atual em um nível mais alto de abstração do que o código fonte, também explica que há diferença entre as duas técnicas, pois define que a finalidade da reengenharia é examinar e alterar um sistema existe, no intuito de reconstruí-lo de uma nova forma e depois implementá-lo, complementa afirmando que o objetivo principal da reengenharia é melhorar a qualidade global do software, mantendo as funções do sistema existente.

Já Pfleeger (2007), afirma que a reengenharia é uma extensão da engenharia reversa. Pois ao passo que a engenharia reversa abstrai informações e dados, a reengenharia produz um novo código fonte, sem modificar a função do software como um todo, mas deixa claro que ambas podem ser independentes pois nem sempre é necessário criar um novo projeto usando engenharia reversa, pois o mesmo já pode existir, o mesmo vale para a reengenharia já que nem sempre é necessário aplica-la para reestruturar ou recriar um sistema.

Através do conceito da técnica de engenharia reversa, pode-se identificar que esse processo necessita a análise do código fonte do sistema, na tentativa de compreender seu funcionamento, técnicas e relacionamentos que o implementam, para que através dessa análise, se possa produzir um documento de projeto com qualidade e com uma visão ampla, abstrata e atualizada do sistema em questão. Recomenda-se a aplicação quando não há um documento de projeto atualizado ou quando este não existe. Para Pressman (2011, p. 670), a engenharia reversa de software é um processo de recuperação de dados do projeto do sistema, pois através deste processo, é possível a extração de informações do projeto de dados e arquitetura, para a partir disso, criar um documento de projeto.

Diferente da engenharia reversa, a reengenharia não depende obrigatoriamente da análise do código fonte, pois para o sucesso da reengenharia é necessário que a documentação esteja atualizada e condizente com a realidade atual do software, já que a reengenharia resulta em uma reestruturação do código ou em um novo sistema, completamente baseado no projeto do antigo sistema, sendo obrigatório atender a todas as especificações e procurando manter o mesmo padrão, evitando grandes mudanças em suas funcionalidades e usabilidade. Segundo Sommerville (1995 apud Piekarski e Quináia, 2000, p. 8), pode-se descrever o processo de reengenharia como "a reorganização e modificação de sistemas de software existentes, parcial ou totalmente, para torná-los mais manuteníveis."

Abaixo é apresentado outro quadro (Quadro 2), onde é exibido o que é produzido na engenharia reversa e reengenharia, a partir de cada etapa do ciclo do processo proposto, com base na análise dos artigos A1, A2, A3, etc..

Quadro 2 - Diferenças encontradas

Ciclo do processo	Engenharia Reversa	Reengenharia
Início	Análise do código fonte para obter uma visão mais abstrata do sistema.	Análise do documento de projeto para planejar a reestruturação do sistema.
Desenvolvimento	Documentar o sistema a partir dos resultados obtidos com a análise do código fonte.	Reestruturação do código antigo ou criação de um novo.
Conclusão	Documento de projeto atualizado e condizente com a realidade do sistema	Novo sistema ou partes do código reestruturadas, para tornar o sistema mais fácil de ser mantido e evoluído.

CONSIDERAÇÕES FINAIS

Através da revisão sistemática da literatura, pode-se concluir que a engenharia reversa é o processo de analisar o código fonte do sistema, no intuito de desenvolver um documento de projeto, já que através desta, será possível analisar toda a estrutura interna do software, possibilitando extrair suas especificações em um nível mais alto de abstração. Ainda com essa pesquisa, pode-se concluir que a reengenharia de software tem como objetivo construir ou reconstruir a estrutura de um determinado sistema, usando como base seu documento de projeto. Também pôde-se identificar que ambas podem complementar-se, pois o artefato gerado pela engenharia reversa é um documento de projeto, e o material necessário para o sucesso da reengenharia é um documento de projeto atualizado, sendo assim, uma pode sim depender da outra, porém tal dependência não é obrigatória, e pelo motivo de resultar em artefatos totalmente distintos, pode-se concluir que são diferentes.

Esta pesquisa auxilia na tomada de decisão entre qual das técnicas usar em um determinado sistema legado, onde faz-se necessário a criação/atualização de documentos de projeto de software, fazendo uso de seu código fonte, ou até mesmo, uma reestruturação/criação de um código mais limpo e fácil de ser mantido.

Como trabalhos futuros, fica a ideia de criação de um processo de desenvolvimento genérico para aplicar a reengenharia e engenharia reversa em um determinado

sistema legado, já que é evidente a necessidade destas técnicas para auxílio de um sistema que precisa ser mantido e evoluído constantemente, pois o mercado sofre constantes modificações, e se estes não estiverem preparados para se adaptar a essas novas visões, poderão ser extintos do mercado.

REFERÊNCIAS BIBLIOGRÁFICAS

AMORIM, T. A.; BRUNETTO, M. A. O.; KASTER, D. S.; FERRACIOLI, F. **Remodelagem do Software SACAR-Web Usando Técnicas de Engenharia Reversa e Reengeharia de Software**, Londrina, 2006. Disponível em: <http://www.sbis.org.br/cbis/arquivos/1030.pdf> Acesso em: 01 de Jun. 2015.

BERTOLOZZI, M. R.; TAKAHASI, R. F.; DE-LA-TORRE-UGARTE-GUANILO, M. C. Revisão Sistemática: Noções Gerais. **Rev. Esc Enferm** USP, p. 1260-1266, 2010. Disponível em: http://www.scielo.br/pdf/reeusp/v45n5/v45n5a33.pdf. Acesso em: 01 de Jun. 2015.

BRAGA, R. T. V. **Padrões de Software a partir da Engenharia Reversa de Sistemas Legados**, Espírito Santo, 1998. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-24012001-163455/publico/Dissertacao.pdf> Acesso em: 01 de Jun. 2015.

CAGNIN, M. S. **Avaliação das vantagens quanto à facilidade de manutenção e expansão de sistemas legados sujeitos à engenharia reversa e segmentação**, São Carlos, 1999. Disponível em: <http://www.ufrgs.br/niee/eventos/SBC/2000/pdf/ctd/dissertacao/Lugar02Mestrado.pdf/> Acesso em: 01 de Jun. 2015.

FALBO, R. A. **Engenharia de software**, Espírito Santo, 2005. Disponível em: <http://www.inf.ufes.br/~falbo/download/aulas/es-g/2005-1/NotasDeAula.pdf/> Acesso em: 01 de Jun. 2015.

JESUS, E. S.; FUKUDA, A. P.; PRADO, A. F. **Reengenharia de Software para Plataformas Distribuídas Orientadas a Objetos**, São Carlos, 2000. Disponível em: <http://www.inf.ufsc.br/sbes99/anais/SBES-Completo/26.pdf/> Acesso em: 01 de Jun. 2015.

LIRA, A. M.; PERES, V. C.; MORAIS, E. C.; COUTO, W. O. **Reengenharia de Software Científico para Cálculo de Capacidade de Carga**, Amazônas, 2012. Disponível em: <http://www.inatel.br/ic/component/docman/doc_download/68-reengenharia-de-software-cientifico-para-calculo-de-capacidade-de-carga/> Acesso em: 01 de Jun. 2015.

NOVAIS, E. R. A; PRADO, A. F. **Reengenharia de Software Orientada a Componentes Distribuídos**, São Carlos, 2000. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/sbes/2001/015.pdf> Acesso em: 01 de Jun. 2015.

PFLEEGER, S. L. **Engenharia de Software: teoria e prática**, 2° ed. São Paulo, 2004, PEARSON.

PIEKARSKI, A. E. T.; QUINÁIA, M. C. **Reengenharia de Software: o que, por quê e como,** Guarapuava, 2005. Disponível em: <http://revistas.unicentro.br/index.php/RECEN/article/download/528/697/> Acesso em: 01 de Jun. 2015.

PINTO, H. L. M.; BRAGA, J. L. **Sistemas Legados e as Novas Tecnologias: técnicas de integração e estudo de caso**, v.7, 2004. Disponível em: <http://www.inf.ufsc.br/sbes99/anais/SBES-Completo/26.pdf/> Acesso em: 02 de Jun. 2015.

PRESSMAN, R. S. **Engenharia de Software: uma abordagem profissional**, 7° ed. Porto Alegre, 2011, AMGH.

PENTEADO, R. A. D.; GERMANO, F. S. R.; MASIERO, P. C. **Engenharia Reversa Orientada a Objetos do Ambiente StatSim: método utilizado e resultados obtidos**, São Carlos, 1995. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/sbes/1995/0019.pdf/> Acesso em: 02 de Jun. 2015.

SAMPAIO, R. F.; MANCINI, M. C. Estudos de Revisão Sistemática: Um guia para síntese criteriosa da evidência científica. **Rev. bras. fisioter**., São Carlos, v.11, n.1, p. 83-89, jan/fev, 2007. Disponível em: http://www.scielo.br/pdf/rbfis/v11n1/12.pdf. Acesso em: 01 de Jun. 2015.

SOMMERVILLE, I. **Engenharia de Software**, 9° ed. São Paulo, 2011, PEARSON.

VIDOTTI FILHO, E.; SANTOS, P. L. V. A. C.; VIDOTTI, S. A. B. G. **Reengenharia, Qualidade Total e Unidades de Informação**, Londrina, 1999. Disponível em: <http://www.uel.br/revistas/uel/index.php/informacao/article/download/1635/1389/> Acesso em: 01 de Jun. 2015.