UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

JADSON FRASSETTO

ELABORAÇÃO DE UMA PLATAFORMA PARA METANÁLISE DIAGNÓSTICA UTILIZANDO CONCEITOS E ANÁLISE DE REQUISITOS DA ENGENHARIA DE SOFTWARE

CRICIÚMA

2015

JADSON FRASSETTO

ELABORAÇÃO DE UMA PLATAFORMA PARA METANÁLISE DIAGNÓSTICA UTILIZANDO CONCEITOS E ANÁLISE DE REQUISITOS DA ENGENHARIA DE SOFTWARE

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientadora: Profa. MSc. Ana Cláudia Garcia Barbosa

Coorientador: Prof. MSc. Kristian Madeira

CRICIÚMA

2015

JADSON FRASSETTO

ELABORAÇÃO DE UMA PLATAFORMA PARA METANÁLISE DIAGNÓSTICA UTILIZANDO CONCEITOS E ANÁLISE DE REQUISITOS DA ENGENHARIA DE SOFTWARE

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel, no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Engenharia de Software.

Criciúma, 24 de junho de 2015.

BANCA EXAMINADORA

Profª. Ana Cláudia Garcia Barbosa – MSc. - (UNESC) – Orientador

Prof. Kristian Madeira – MSc. - (UNESC) – Coorientador

Prof. Gustavo Bisognin – MSc. - (UNESC)

Profª. Merisandra Côrtes de Mattos Garcia – Dra. - (UNESC)

Dedico este estudo aos meus pais que me apoiaram, e a todos os meus familiares.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por permitir que eu pudesse alcançar este tão sonhado objetivo. Posteriormente, aos meus pais pela educação que me proporcionaram, por mostrar como trilhar o caminho correto, ser digno em tudo que eu faço e acima de tudo por toda a força, os conselhos e as expectativas que depositaram em mim. Em terceiro, agradeço aos meus orientadores que me auxiliaram durante a elaboração deste estudo, não medindo esforços para que pudessem responder aos questionamentos que lhe eram feitos, aos professores que lecionaram durante toda minha estadia como acadêmico do curso, e por último a todos aqueles que contribuíram direta ou indiretamente na minha formação. Àqueles que não mencionei aqui, peço-lhes que não se sintam magoados, porque recordar de todas as pessoas nessa jornada não é fácil, mas quero-lhes prestar meus sinceros agradecimentos por participarem desta minha trajetória.

.

“Desconfie do destino e acredite em você. Gaste mais horas realizando que sonhando, fazendo que planejando, vivendo que esperando porque, embora quem quase morre esteja vivo, quem quase vive já morreu.”

Sarah Westphal

RESUMO

Com a grande e crescente quantidade de artigos, estudos e trabalhos publicados no meio científico tornou-se difícil fazer uma revisão qualificada da literatura. Um dos desafios para os pesquisadores da área biomédica é sintetizar os resultados de estudos independentes, dado que eles apresentam variações decorrentes de diferentes amostragens, populações, culturas, metodologias e outras variáveis que os envolvem. Desta forma, surgiu a metanálise, que tem por objetivo determinar um resultado com maior evidência por meio de uma análise estatística rigorosa de estudos primários, reunidos pela revisão sistemática. Entretanto, para a completa realização da metanálise diagnóstica é preciso seguir algumas etapas que visam determinar a qualidade dos estudos incluídos e aplicar alguns testes estatísticos importantes. Tais etapas, hoje, são realizadas com o auxílio de softwares. Porém, não se tem uma única plataforma que auxilie os pesquisadores da área biomédica na aplicação de todos os testes e etapas necessárias na condução da metanálise, obrigando os pesquisadores a utilizarem diferentes softwares. Sendo assim, o objetivo deste trabalho é elaborar um novo software, capaz de conciliar todas as etapas em uma única plataforma, para que assim auxilie os pesquisadores a realizarem metanálises de maneira mais efetiva. Com o uso das práticas e conceitos da engenharia de software foi possível avaliar a melhor forma de realizar o estudo. A metodologia empregada pelo trabalho consistiu em pesquisar e estudar os conceitos da engenharia de software, da engenharia de requisitos e da arquitetura de software, como também em avaliar a viabilidade do projeto, compreender a metanálise, avaliar os softwares que são utilizados pelos pesquisadores na realização de metanálises diagnósticas, levantar, classificar e documentar os requisitos do sistema, prototipá-los e validá-los. Foi verificado por meio do estudo da viabilidade que, são utilizados três softwares distintos para realização da metanálise constando-se, portanto, a necessidade da elaboração de uma nova ferramenta. Os requisitos para essa ferramenta foram levantados com base em entrevistas e avaliações dos softwares existentes. Com o desenvolvimento do protótipo foi possível validá-lo junto às partes interessadas verificando-se cada um dos requisitos elicitados por meio de um questionário aplicado com pesquisadores experientes no desenvolvimento de metanálises diagnósticas. Ficou caracterizado a importância do ambiente WEB na condução colaborativa de metanálises, como também os requisitos relativos aos módulos do sistema, a internacionalização e os métodos de análises utilizados. Conclui-se que com estudos mais aprofundados sobre engenharia de software é possível conciliar teoria com prática, e fornecer ao mercado ferramentas que visem maior compromisso com a qualidade de softwares, sobretudo fornecendo softwares para as mais diversas áreas do conhecimento, auxiliando outros ramos de atividades a exercerem seu papel na sociedade de forma organizada, ágil e simplificada.

Palavras-chave: Metanálise diagnóstica. Engenharia de software. Engenharia de requisitos. Arquitetura de software.

ABSTRACT

With the large and growing number of articles, studies and papers published in scientific circles it became more difficult to make a qualified review of the literature. One of the challenges for biomedical researchers is to summarize the results of independent studies as they present variations on different samples, populations, cultures, methodologies and other surrounding variables. Thus came the meta-analysis, which aims to determine a more clearly result through a rigorous statistical analysis of primary studies, gathered by systematic review. However, for the full realization of diagnostic meta-analysis we need to follow some steps which seek to determine the quality of the included studies and apply some important statistical tests. Such steps today are performed with the assistance of software. However it does not exist a single platform which would help biomedical researchers in the application of all the tests and steps required in conducting the meta-analysis, forcing them to use different software. Therefore the objective of this work is to develop a new software, capable of combine all the steps on a single platform to assist researchers to perform meta-analyses in a more effective way. With the use of practices and concepts of software engineering it was possible to evaluate the best way of conducting the study. The methodology used for the work consisted of researching and studying software engineering concepts, requirements engineering and software architecture, as well as to measure the viability of the project, understand the meta-analysis, evaluate the software used by researchers in diagnostic meta-analyses, to gather, classify and document the system requirements and prototype and validate them. It was verified by a viability study that are used three different software to perform the meta-analysis and it was noticed the need to create a new tool. The requirements for this tool were gathered based on interviews and evaluations of the existing software. With the development of the prototype it was possible to validate it with the stakeholders and verify each of the requirements elicited using a questionnaire with experienced researchers in the development of diagnostic meta-analyses. It was characterized the importance of WEB environment in the collaborative conduction of meta-analyses, as well as the requirements for system modules, internationalization and methods of analysis that were used. It was concluded that with more exhaustive studies of software engineering it is possible to combine theory with practice and provide to the market tools that aim to a greater commitment concerning quality software, particularly providing software for the most diverse areas of knowledge, helping other business segments to prosecute their role in society in an organized, agile and simplified manner.

Key words: Diagnostic meta-analysis. Software engineering. Requirements engineering. Software architecture.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CMMI	Capability Maturity Model Integration
CDR	Chemical Data Report
DOR	Diagnostic Odds Ratio
NHS	National Institute for Health Research
DNS	Domain Name Service
EJB	Enterprise Java Beans
GDT	Guideline Development Tool
HTTP	Hypertext Transfer Protocol
HRA	Hospital Regional de Araranguá
IEC	International Electrotechnical Commission
IP	Internet Protocol
ISO	International Organization for Standardization
JEE	Java Enterprise Edition
JPA	Java Persistence API
JSF	Java Server Faces
JSP	Java Server Pages
MVC	Model-View-Controller
PBE	Prática Baseada em Evidências
RMM	Relationship Management Methodology
ROC	Receiver Operation Characteristic
SMADP	Shell Metanalysis Diagnostic Pearson
SBIS/CFM	Sociedade Brasileira de Informática em Saúde e do Conselho Federal de Medicina
UNESC	Universidade do Extremo Sul Catarinense
UTI	Unidade de Terapia Intensiva

SUMÁRIO

1 INTRODUÇÃO

A revisão sistemática é um dos métodos de pesquisa de maior importância para a obtenção de evidências, visto que juntamente com a metanálise ocupa o topo da pirâmide de evidência epidemiológica (COOK et al., 1995). Trata-se de um método crítico e rigoroso de reunião de estudos primários sobre determinado tema a ser pesquisado, visando o conhecimento sobre novos tratamentos, medicamentos e/ou diagnósticos.

Por meio da metanálise, sendo esta a análise estatística empregada após a revisão sistemática, é possível determinar de maneira imparcial os impactos e efeitos dos tratamentos em pacientes.

Porém, os pesquisadores envolvidos em revisões sistemáticas com metanálises possuem certa dificuldade no processo de análise dos dados coletados, pois as ferramentas atualmente disponíveis no mercado cumprem apenas parcialmente com as análises necessárias, visto que não se tem uma plataforma única e centralizada, ao qual se possa fazer todo o processo de metanálise.

De certa forma, é necessário que os profissionais envolvidos utilizem os softwares já existentes, de maneira isolada, para cada etapa da metanálise, tornando o processo mais demorado e também mais suscetível a erros. Pois é necessário que se transcreva o banco de dados de um software para outro. Além do que muitos dos softwares existentes são pagos ou possuem funcionalidades limitadas.

Com isso, mediante estudos mais aprofundados sobre requisitos e engenharia de software, torna-se possível conciliar os conceitos da computação, aplicada à obtenção de software, com a necessidade de se desenvolver uma ferramenta mais completa para auxiliar de uma forma mais independente aos pesquisadores que atuam na área biomédica.

1.1 OBJETIVO GERAL

Aplicar a engenharia de software para estruturar uma plataforma de metanálise diagnóstica por meio de elicitação e modelo de requisitos para auxiliar aos profissionais da área biomédica.

1.2 OBJETIVOS ESPECÍFICOS

O trabalho tem como objetivos específicos:

a) compreender a engenharia de software;
b) estudar técnicas para levantamento de requisitos;
c) entender os objetivos da metanálise diagnóstica e como são realizadas;
d) aplicar a engenharia de requisitos para modelar a plataforma WEB;
e) aplicar a engenharia de software para definir a arquitetura;
f) desenvolver um protótipo para apresentação, que constituirá a base para a implementação futura da plataforma WEB.

1.3 JUSTIFICATIVA

Nas últimas décadas a produção científica aumentou consideravelmente, trazendo consigo maior dificuldade para obtenção de informações concretas, como evidencia Lovatto (2007) ao afirmar que muitas dessas produções contribuíram para uma análise menos qualificada da literatura.

Desta forma, a revisão sistemática é necessária, pois permite sintetizar resultados de vários estudos por meio de um método rigoroso e crítico e de uma abordagem sistemática (SILVA, 2003).

A metanálise complementa a revisão sistemática, pois trata-se de uma etapa em que se empregam modelos matemáticos e análises estatísticas para combinar resultados de estudos independentes a fim de se obter um resultado final conciso (PETITI, 2000).

Mediante importância da metanálise e revisão sistemática para condução de avaliações de pesquisas clínicas, faz-se necessário uma ferramenta para que se possa trabalhar com os estudos primários. Portanto, uma plataforma com amplas funcionalidades, contendo módulos integrados, para que se evite erros de transcrições de banco de dados, facilitará a condução das metanálises por parte dos pesquisadores.

Com isso, dado que vários profissionais estão envolvidos em muitas etapas da revisão sistemática e metanálise, fica evidente que um ambiente colaborativo proporciona maior facilidade para o fluxo do trabalho destes. Desta

forma, o ambiente WEB proporciona um cenário melhor para que se desenvolva as práticas da revisão sistemática e metanálise.

Entretanto, para que se possibilite a estruturação de uma ferramenta completa é necessário que se aplique os conceitos da computação, sobretudo os que envolvem engenharia de software e de requisitos, pois segundo Turine (1996) um software que é mal especificado certamente irá desapontar o usuário e, culminará em maiores problemas para a equipe de desenvolvimento, que terá que readequar o produto.

Sobretudo a responsabilidade de planejar o desenvolvimento de maneira eficiente, produtiva e segura fica a cargo da engenharia de software (PRESSMAN, 2011).

Portanto, para o desenvolvimento de um modelo que abranja todas as questões relativas ao projeto de um software para realização de metanálises é necessário que se empreguem estes princípios e fundamentos.

1.4 ESTRUTURA DO TRABALHO

O trabalho de conclusão de curso é composto por 8 capítulos, sendo o primeiro uma contextualização, introduzindo o leitor a cerca do assunto desenvolvido, seguindo da justificativa do desenvolvimento e os objetivos pretendidos.

O capítulo 2 fundamenta a engenharia de software tratando de sua importância na concepção de novos projetos. Diferencia um projeto de hardware de um projeto de software e apresenta os meios pelos quais projetos empresariais e não empresariais podem garantir a qualidade de software.

No capítulo 3 é abordada a engenharia de requisitos, bem como as fases dela e as técnicas utilizadas para extrair requisitos das partes interessadas.

A importância da arquitetura de software no sucesso do mesmo, os fundamentos sobre os frameworks de desenvolvimento utilizados neste trabalho e a arquitetura do ambiente WEB é fundamentada no capítulo 4.

O capítulo 5 faz uma correlação entre medicina e informática, descrevendo como a computação tem auxiliado outras áreas do conhecimento. Ele também apresenta as dificuldades de se sintetizar resultados de estudos independentes e as formas de se produzir revisões sistemáticas e metanálises

diagnósticas de maneira qualificada.

No capítulo 6 são apresentados alguns trabalhos correlatos, que dão uma ideia sobre os softwares utilizados pelos pesquisadores para o desenvolvimento da metanálise diagnóstica. Também são apresentados os trabalhos que fomentam a reutilização de requisitos de software e a modelagem com foco na qualidade do software.

O sétimo capítulo expõe as metodologias aplicadas para obtenção dos requisitos e modelagem da plataforma, demonstrando os resultados obtidos em cada etapa do processo.

Por fim, o capítulo 8 conclui este trabalho e apresenta sugestões para temas que explorem a engenharia de software e conduzam à implementação da plataforma.

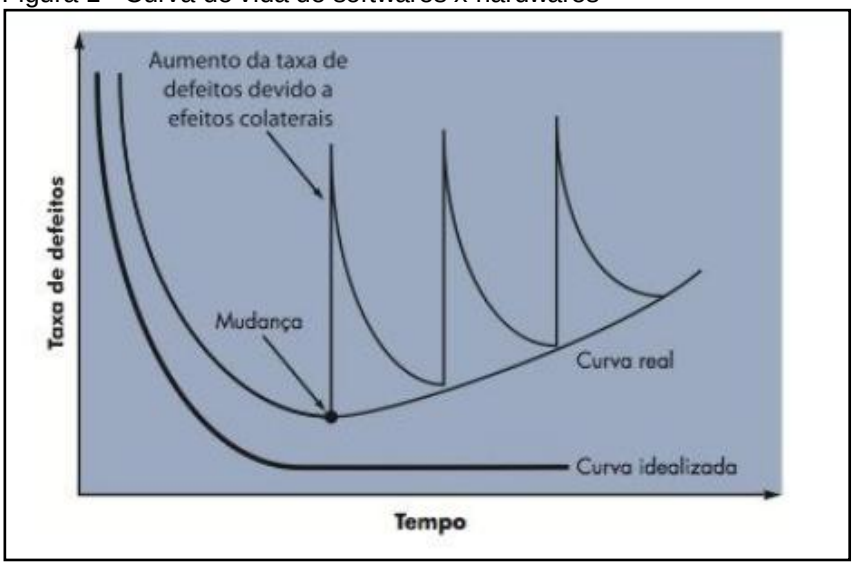
2 ENGENHARIA DE SOFTWARE

É necessário, antes de tudo, definir o que é um software, para que se possa compreender sua engenharia.

Segundo Pressman (2011) software são instruções que quando executadas fornecem os dados necessários para seu usuário. Consiste ainda de uma estrutura de dados que permite a manipulação das informações. Porém, como afirma Pressman, estas definições não são suficientes para determinar o que é o software.

Ao diferenciar um projeto de hardware de um projeto de software, tanto um como outro, é idealizado para uma função específica, porém ao final do desenvolvimento do software, torna-se fácil replicá-lo, sem demais custos adicionais de fabricação, visto que os custos iniciais com o desenvolvimento de um software estão no projeto e na sua programação. Já o hardware, além de ter um projeto bem elaborado, deve ainda garantir que cada novo produto criado não apresente erros na execução do projeto. Dessa forma, podem-se definir a curva de vida de um software e de um hardware.

Figura 1 - Curva de vida de softwares x hardwares



Fonte: Pressman (2011).

A figura 1 apresenta a diferença entre a vida de um software e de um hardware. O cenário ideal na concepção dos dois projetos é o de que ambos ao longo do tempo possam minimizar a apresentação de falhas decorrentes do projeto

inicial. No entanto, a curva real demonstra que essa situação por muitas vezes não ocorre.

Em se tratando do hardware, verifica-se que isso se deve ao fato de que os componentes que o compõe se desgastam ao longo dos anos, sobretudo pelas condições ambientais em que operam e por seu próprio uso. Desta forma, novos erros são introduzidos, fazendo com que a curva dê uma reviravolta e comece a evidenciar novas falhas, distanciando-se da curva idealizada. Essas falhas podem ser corrigidas com o reparo ou reposição dos componentes do hardware.

Já o software, também apresenta um ponto de mudança, e que se verifica por mais de uma vez durante sua vida. É muito comum que intervenções sejam feitas durante a operação dele, algumas vezes para resolver problemas decorrentes da elaboração do projeto inicial, outras para corrigir falhas da implementação, e também desenvolver novas funcionalidades para ele. Cada uma dessas intervenções pode introduzir novos erros no software, e desta forma a curva real se distancia da curva idealizada. Diz-se então que diferentemente do hardware, o software não se desgasta, mas se deteriora, na medida em que novas alterações são propostas quando o software já está em funcionamento (PRESSMAN, 2011).

Quanto melhor planejado for o software, e isso é uma responsabilidade do projeto do software, mais bem preparado ele estará para as mudanças. Como é de conhecimento comum, o software irá apresentar alterações. Um projeto de software não é suficientemente ideal para prever todas as mudanças possíveis no software, porém é de fundamental importância, para que ele possa receber as alterações minimizando o impacto naquilo que já está em execução.

Dessa forma, entende-se por engenharia de software, a responsabilidade de planejar o desenvolvimento de maneira eficiente, produtiva e segura. Onde, segundo Pressman (2011) se estabelece o emprego de sólidos princípios de forma a obter software econômico, confiável e eficiente. Porém também constitui a gerência do desenvolvimento do software sobre aspectos técnicos e a documentação sobre o modelo e os dados, sobretudo garantindo a qualidade e a administração das mudanças de forma apropriada.

Visto a importância da engenharia de software para o desenvolvimento de sistemas, é necessário que se garanta a qualidade no produto final, como em todo o processo de desenvolvimento. Como mencionado anteriormente, é um dos princípios da engenharia de software garantir a qualidade desse desenvolvimento.

Refletindo sobre isso, é necessário pensar em como garantir a qualidade do software.

2.1 QUALIDADE DE SOFTWARE

Uma das medidas para se garantir a qualidade de um software é empregar os modelos de software. Estes têm por finalidade regulamentar o desenvolvimento de software, e visam à melhoria contínua do processo de desenvolvimento. Os quais podem-se destacar os modelos: CMMI, ISO/IEC 12207, ISO/IEC 15504, como também o modelo nacional Melhoria de Processo do Software Brasileiro (MPS-BR).

Segundo Nogueira (2006), no Brasil constatou-se que o uso de boas práticas da engenharia de software são empregadas somente quando são exigidas em avaliações de processos. Acredita-se que o emprego dessas boas práticas possa melhorar a competitividade de uma organização, atuando sobre o custo, prazo, produtividade, qualidade e a satisfação do cliente.

O modelo nacional MPS-BR estabelece níveis de maturidade de processo, que permitem traçar o perfil de desenvolvimento de software de uma organização e definir quais pontos são necessários para que a empresa atue a fim de galgar novos níveis de maturidade, tornando-as mais efetivas em desenvolvimento de software.

O modelo nacional MPS-BR foi baseado no modelo internacional CMMI, já que em muitos pontos, como os níveis de maturidade, se baseia neste. Sendo assim, faz-se necessário conhecer o modelo internacional.

Desta forma, segundo Baker (2010) e Kenett (2010, tradução nossa) o modelo internacional pode ser concebido em cinco níveis, onde:

a) o primeiro nível, ou nível inicial presume que a empresa está em uma fase “caótica”, onde não se tem conhecimento de planos ou cronogramas, nem tão pouco se tem preocupação com custos de software. Sobretudo, requisitos de qualidade são impraticáveis. Logo, o sucesso do software depende somente de profissionais competentes dentro da organização;
b) no segundo nível, também chamado de nível gerenciado, a organização se cerca de tecnologias e infra-estrutura para conduzir o

desenvolvimento de software e para assegurar uma produção efetiva no processo de software, que é relativo de projeto a projeto;

c) no terceiro nível, ou nível definido, existem processos padrões dentro da organização para conduzir projetos de software e dar suporte as demais atividades relacionadas, baseando-se em linhas guias de planejamento adaptado a cada projeto individual. Projetos se tornam pró-ativos ao invés de reativos e contribuem para produtos melhor trabalhados, medições e indicadores de processo;

d) no quarto nível, pode-se afirmar que a empresa está gerenciada quantitativamente. São estabelecidos performances para processos, metas de qualidade de software e objetivos. Projetos são gerenciados quantitativamente para se ter conhecimento de onde eles podem chegar ou estão indo, desta forma estatísticas sobre performance de projetos e qualidade permitem determinar o quanto o projeto está sendo conduzido dentro da meta e quais as causas de sua variação;

e) no quinto e último nível, a organização continua reunindo esforços para incorporar ainda mais controles sobre o projeto, e tem seu foco voltado na redução de custos.

Ao passo em que a empresa toma conhecimento dos níveis da especificação de processos de software, ela se torna mais madura, pois agrega maior noção sobre boas práticas de desenvolvimento, devido a isso, podem ser assim chamados de níveis de maturidade, pois ao passo em que a empresa implementa as regras dos processos aqui definidos, se conquista uma maior prática no desenvolvimento de software. A empresa se torna mais rigorosa quanto à qualidade e controle de seu processo, e isso contribui de forma geral para um software mais bem preparado.

No âmbito empresarial, o modo com que a empresa encara e implementa os processos de software do MPS-BR dita a responsabilidade com a qualidade do software. Entretanto, a garantia de uma maior qualidade tanto para projetos empresariais quanto para projetos que não tem caráter empresarial pode ser concebida por meio da avaliação dos princípios da engenharia de software.

2.2 PRINCÍPIOS DA ENGENHARIA DE SOFTWARE

Segundo McConnell (2004), a maioria dos programadores toma como engenharia de software o conhecimento específico de uma linguagem, porém uma linguagem de programação torna-se obsoleta a medida que novas tecnologias são empregadas. Há um outro tipo de conhecimento, ao qual os programadores deveriam saber, e muito provavelmente os acompanhará por toda sua carreira, ao qual se chama “princípios da engenharia de software”.

Para Pressman (2011), a engenharia de software pode ser compreendida pelo estudo de alguns princípios:

a) **de processo:** norteiam o desenvolvimento do software para se obtê-lo de maneira mais ágil, considerando economizar ações de trabalho na medida em que se propõe a manter o código mais limpo e conciso visando à qualidade em todas as etapas do processo, de maneira que se obtenha na sua totalidade um software mais adaptável;
b) **práticos:** primam pela divisão de trabalho, norteando o desenvolvimento a consquistar um maior grau de abstração na sua implementação, tornando o código mais simples em meio ao ambiente complexo, permitindo um desenvolvimento mais coeso e que auxilie na manutenabilidade do software;
c) **metodológicos:** orientam os engenheiros de software na obtenção de conhecimento a cerca do problema, mostrando-lhes como captar as informações necessárias, anotar e registrar informações e negociar os requisitos dos softwares;
d) **de planejamento:** tentam direcionar a equipe de desenvolvimento, sem que se tomem tempos extras para um planejamento muito detalhado que não represente a atividade fim, ou sem que se pule esta etapa definindo um plano que não contribua como guia para o sucesso do software. Nesta fase deve-se observar o plano, monitorando-o junto ao cronograma, reconhecendo que o processo de desenvolvimento de software é iterativo, e portanto, a medida em que se inicia o desenvolvimento dele, algumas alterações serão necessárias, desta forma, deve-se considerá-los no plano, e determinar quando e como as alterações deverão ser realizadas;

e) **de modelagem:** tratam da representação do produto final, sendo comum realizar protótipos ou espécies de modelos em escalas reduzidas. Pode-se definir em duas linhas fundamentais, uma sobre o aspecto do cliente, onde a modelagem deve representar as características desejadas pelo usuário e o comportamento do software. A outra linha, diz respeito à técnica, e deve prever como a arquitetura e as funções definidas irão possibilitar a transformação do software;

f) **de construção:** orientam o programador na codificação do software, e direciona-os a compreender melhor os problemas, pensando em como manter o código simples e fácil de ser testado. Permite ainda definir a linguagem de programação e as ferramentas de trabalho que facilitem o desenvolvimento;

a) **de disponibilização:** regem a conduta frente à entrega do software ao cliente de maneira a propiciar instruções de como o software deve ser utilizado e como o mesmo será disponibilizado ao seu usuário final. Dentre os quais, segundo o aspecto da engenharia de software deve contemplar as expectativas dos clientes, a entrega do software sem erros, fornecer suporte e manuais de utilização.

Tendo-se observado esses princípios, bem como os modelos de software e fazendo-se as ponderações necessárias para cada projeto, pode-se pensar em produzir software visando à qualidade no resultado.

3 ENGENHARIA DE REQUISITOS

Quando se contrata um profissional para produzir um evento ou uma festa, este indaga sobre quais pontos cruciais serão necessários para fazer a festa segundo as convenções de seu cliente. O mesmo os contempla com opções de cores, opções de pratos, opções das mais variadas possíveis e como a decisão final é do cliente, este quer que tudo saia de acordo com o planejado e sobretudo, que não haja exorbitâncias.

De certa forma, há como fazer uma análise comparativa com um projeto de um software. O cliente necessita resolver um problema, e cabe aos engenheiros de software extrair dele todos os requisitos necessários para que a equipe de desenvolvimento possa implementar a solução que ele espera.

Um requisito de software é uma condição ou capacidade necessária para o usuário resolver um problema ou atingir um objetivo (IEEE, 1998).

Define-se por engenharia de requisitos, as técnicas que levam ao entendimento dos requisitos aos quais traçam uma ponte entre o levantamento de requisitos e a construção do software (PRESSMAN, 2011).

Laplante (2014, tradução nossa) baseado na definição sobre engenharia de software de Pamela Zave, reescreve a definição de engenharia de requisitos como sendo o ramo que estuda os objetivos do mundo real e sua relação com software, buscando determinar as funções e as restrições de sistemas com propósito de determinar o comportamento do mesmo, através de especificações precisas e com base na relação dele com outros sistemas mediante a evolução do tempo.

> A engenharia de requisitos fornece o mecanismo apropriado para entender aquilo que o cliente deseja, analisando as necessidades, avaliando a viabilidade, negociando uma solução razoável, especificando a solução sem ambigüidades, validando a especificação e gerenciando as necessidades à medida que são transformadas em um sistema operacional. (THAYER, 1997 apud PRESSMAN, 2011, p. 127).

Baseando-se nestas três definições, pode-se compreender a engenharia de requisitos como sendo o ramo que estabelece técnicas para extrair as informações necessárias do seu objeto de referência, seja ele um cliente, o ambiente a sua volta ou sendo mais generalista, o mundo real, buscando sobretudo compreender sua relação com software para determinar os requisitos para

construção do mesmo, a medida em que se pensa em obter uma solução sem ambigüidades, viável e que contemple as necessidades do objeto de referência.

3.1 FASES DA ENGENHARIA DE REQUISITOS

De acordo com Pressman (2011), a engenharia de requisitos pode ser compreendida em sete etapas, muitas das quais podem ocorrer simultaneamente, e são elas:

a) **concepção:** esta fase compreende o entendimento básico do problema, quais são as pessoas envolvidas, qual área de atuação da solução e a comunicação entre os interessados e a equipe de desenvolvimento;
b) **levantamento:** consiste em extrair dos clientes os requisitos para o desenvolvimento do programa. Porém, é uma das fases que mais agregam complexidade na definição dos requisitos, pois os clientes acabam por especificar detalhes desnecessários, ou detalhes técnicos que mais confundem do que esclarecem, além de que muitas vezes não sabem direito o que querem, tem dificuldade de transmitir o que querem ou omitem informações que consideram “óbvias”. Por fim, os requisitos mudam com o passar do tempo, e alinhar essas mudanças à definição do projeto se torna complicado;
c) **elaboração:** é a fase que alinha a concepção e levantamento à um modelo, onde são produzidos muitos diagramas para representar os requisitos e se trabalham com cenários para extrair deles as classes necessárias para o entendimento do projeto;
d) **negociação:** compreende o afinamento dos requisitos junto ao cliente, uma vez que é bastante comum o cliente requisitar mais do que pode ser produzido em determinado prazo. Nessa fase define-se prioridades para o desenvolvimento dos requisitos, muitos dos quais podem ser mesclados, cortados do desenvolvimento do projeto ou refinados;
e) **especificação:** consiste na elaboração de documentos, por vezes modelos matemáticos ou gráficos ou até mesmo cenários de uso do produto;
f) **validação:** impõe uma revisão sobre a especificação para determinar

> que a solução seja especificada sem ambigüidades. Procura na especificação pontos em que se tenha omissões, inconsistências e erros.

3.2 COMPREENSÃO E CLASSIFICAÇÃO DE REQUISITOS

Requisitos muitas vezes não são bem compreendidos, e ultimamente tem se evidenciado alguns fatores que levam à má compreensão dos mesmos, tais como problemas com a linguagem natural (imprecisões e ambigüidades), complexidade demasiada alta (soluções muito completas e ricas em detalhes), dificuldades em entender comportamentos do sistema, omissões de requisitos, sobrecarga (elicitações de requisitos sem fundamentos, que contribuem para distorcer a realidade da solução), inconsistências, erros, entre outros (LAPLANTE, 2014, tradução nossa).

O desafio para o engenheiro de software é reconhecer que clientes oferecem requisitos confusos e algumas vezes os próprios engenheiros os confundem (LAPLANTE, 2014, tradução nossa). Sendo assim, a engenharia de software fornece as ferramentas adequadas para uma abordagem ao cliente que resulte em requisitos mais claros e bem compreendidos.

O objetivo da gestão de requisitos é obter uma, e somente uma interpretação consistente dos requisitos de todas as partes envolvidas (MILLER, 2009, tradução nossa).

Sommervile (2005) sugere que requisitos sejam organizados em três níveis: requisitos de usuários, requisitos de sistemas e especificações de projetos.

Desta forma, pode-se compreender os requisitos de usuários como informações e restrições na forma de linguagem natural que podem vir acompanhadas de diagramas e especificam que serviços os usuários esperam que os sistemas lhes proporcionem. Por outro lado, requisitos de sistemas podem ser entendidos como descrições detalhadas de serviços e restrições de sistema e muitas vezes referem-se à especificações funcionais ou técnicas. Esses requisitos provém de análises dos requisitos de usuários e devem ser estruturados e precisos. Por fim, as especificações de projetos são o compilado final para que os desenvolvedores implementem os requisitos de usuários e de sistemas. Eles são essenciais e derivam da análise direta dos requisitos de sistemas (LAPLANTE,

2014, tradução nossa).

Sommerville (2005) também sugere que os requisitos possam ser divididos em: funcionais, não-funcionais e de domínio.

A elaboração de requisitos funcionais tem foco nos serviços que devem ser providenciados ao usuário, e como eles devem reagir à entrada de dados. Eles podem ser definidos de maneira geral, a exemplo dos requisitos de usuários, ou de forma detalhada provendo entradas, saídas, exceções, etc (LAPLANTE, 2014, tradução nossa). Descrevem funcionalmente os modelos de negócio que devem ser construídos para permitir que o usuário execute seus objetivos e tarefas (MILLER, 2009, tradução nossa).

Os requisitos não funcionais prevêem como o sistema irá operar e detalham restrições do negócio, propriedades, qualidades, linguagem de programação, etc. Por exemplo, eles podem tratar de restrições da legislação aplicada, da usabilidade, performance, ou seja, estão envolvidos com o propósito do engenheiro de software e dos clientes (LAPLANTE, 2014, tradução nossa). Descrevem uma visão do sistema ou suas características, de maneira que abordam as restrições impostas pelo usuário e na construção do sistema, como também definem as qualidades e atributos necessários para permitir o crescimento e a troca de informações com o usuário (MILLER, 2009, tradução nossa).

Requisitos de domínio estão relacionados com o domínio da aplicação e consistem em novos requisitos funcionais ou restrições. Muitas vezes estão relacionados com computadores em particular, a maneira que operam, o ambiente em que estão situados. Por exemplo, na WEB deve ser considerado o modelo cliente-servidor, que prevê que as conexões não mantém estado, enquanto que no ambiente desktop essa preocupação não é necessária (LAPLANTE, 2014, tradução nossa).

Conhecer e estudar a engenharia de requisitos é contribuir para o sucesso do software, pois segundo Turine (1996) o resultado de se especificar indevidamente um software contribui para manutenções indesejadas por parte da equipe de desenvolvimento. Uma vez que se conheça bem os requisitos do software, se projeta também melhor esse software. Sendo assim, menores as chances de o software ter de ser alterado por inadequações à finalidade do cliente.

3.3 TÉCNICAS PARA LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos exige do engenheiro de software alguns conhecimentos básicos de como proceder para a obtenção dos requisitos do software. Um dos meios para obtenção de requisitos é entender as técnicas existentes e praticá-las.

As técnicas para levantamento de requisitos buscam orientar os engenheiros na obtenção dos requisitos do software, dando-lhes as ferramentas necessárias para extrair das partes interessadas as características do software almejado.

3.3.1 Entrevistas

Entrevistas é a técnica mais freqüentemente utilizada pelos engenheiros de software, e consiste em uma conversa direcionada, onde uma pessoa fala a outra as necessidades as quais enfrenta na sua rotina diária (COURAGE, 2005, tradução nossa).

É um método sistemático, objetivo e de rápida coleta, podendo ser uma conversa formal ou informal. O propósito primário é obter as necessidades mais evidentes, as restrições e suposições (HOSSENLOPP, 2008, tradução nossa).

Nem todas as técnicas fornecem ao engenheiro de software a possibilidade de debater os requisitos com usuários individuais. Sendo assim, a técnica de entrevista é sempre aconselhada quando se quer obter informações específicas, por exemplo, de um usuário que exerce determinado papel dentro de uma empresa, tal como um operador em uma fábrica, ou um fiscal em um ofício (COURAGE, 2005, tradução nossa).

Alguns tipos de abordagens podem ser utilizadas no que se refere ao levantamento de requisitos utilizando-se da técnica de entrevistas.

Uma das abordagens, diz respeito à entrevista pessoal, que pode ser mais formal, registrando e documentando cada resposta, ou pré-preparando perguntas as quais abordem funcionalidades do software. Porém, a entrevista pode ser conduzida de forma mais informal, sem a necessidade de questões pré-formuladas ou um rigoroso registro das informações. Tanto uma quanto outra abordagem serve de estímulo para identificar novos requisitos, mudar os requisitos

já existentes ou identificar restrições e regras de negócio (HOSSENLOPP, 2008, tradução nossa).

Outras abordagens baseiam-se na observação das tarefas executadas, as quais levantam questões baseadas no acompanhamento de um funcionário na execução de suas tarefas na jornada de trabalho dele (HOSSENLOPP, 2008, tradução nossa).

Segundo Hossenlopp (2008, tradução nossa) podem-se definir algumas vantagens e desvantagens do uso da técnica de entrevistas, dentre os benefícios destacam-se:

a) permite que o entrevistador e o entrevistado discutam pontos e os registrem sobre as mesmas condições trazendo como resultado uma interpretação consistente dos requisitos;
b) promove discussões interativas para explorar os detalhes das informações;
c) encoraja e constrói relações harmoniosas entre as partes interessadas;
d) permite observar comportamentos não verbais.

As desvantagens podem ser definidas como:

a) requer acesso e compromisso das partes interessadas;
b) requer treino e preparação para conduzir entrevistas bem sucedidas;
c) transcrever e analisar as entrevistas podem se tornar complexas e exaustivas;
d) a documentação resultante é o resultado subjetivo das interpretações do entrevistador perante o entrevistado.

3.3.2 Inspeção

Inspeções são um meio rápido de obter informações das partes interessadas, podendo ser uma ótima maneira para obter requisitos para novos softwares ou para softwares existentes.

Segundo Courage (2005, tradução nossa), nos casos em que se pensa em novos softwares, a inspeção pode:

a) ajudar a identificar o potencial de uso do software;
b) encontrar as necessidades dos usuários no produto proposto.

No caso de softwares existentes, a inspeção pode:

a) ajudar a compreender as pessoas que utilizam o software e suas características;
b) levantar os pontos bons e ruins de produtos existentes segundo a visão do usuário;
c) ajudar no aprendizado de como os usuários utilizam os softwares existentes.

Existem dois tipos de questões abordadas pelo método de inspeção: abertas e fechadas (HOSSENLOPP, 2008, tradução nossa).

Questões abertas não têm uma resposta correta, mas permitem que os entrevistados possam respondê-las com suas próprias palavras. Mas, é claro, torna a interpretação muito mais difícil (HOSSENLOPP, 2008, tradução nossa).

Questões fechadas propõem uma única resposta correta, sendo de mais fácil escolha, porém de complexa formulação (HOSSENLOPP, 2008, tradução nossa).

De acordo com Courage (2005, tradução nossa), a preparação do método consiste em:

a) identificar os objetivos do estudo;
b) levantar e se aprimorar nas atividades envolvidas;
c) formular as questões;
d) determinar como serão analisados os resultados;
e) criar a inspeção;
f) avaliar as condições e fazer as considerações quando expor o método para uma inspeção distribuída;
g) distribuir a inspeção via web, e-mail ou cartas;
h) testar a inspeção.

Conforme Hossenlopp (2008, tradução nossa) os benefícios da utilização deste método incluem:

a) requer um tempo limitado das partes interessadas;
b) eficaz em alcançar as partes interessadas geograficamente dispersas;
c) escalável para largas audiências;
d) relativamente rápido e fácil para os administradores;
e) fornece mais opiniões subjetivas através de entrevistas com as partes interessadas.

Dentre as desvantagens, podem se citar, segundo Hossenlopp (2008, tradução nossa):

a) relativamente baixa taxa de respostas;
b) incentivos para respostas podem ser exaustivos;
c) uso de questões abertas requer maior análise do especialista;
d) questões mal formuladas podem fornecer respostas insatisfatórias.

3.3.3 Observação

Observações são a maneira pela qual o engenheiro de software coleta informações para entender o contexto dos usuários, tarefas e metas e pode ser utilizada para obter informações de como construir protótipos para auxiliar na coleta de requisitos (SHARP, 2007, tradução nossa).

Elas fornecem o mecanismo necessário para observar indivíduos em seu ambiente e como eles executam seus trabalhos, tarefas e processos (PROJECT MANAGEMENT INSTITUTE, 2013).

Entre os benefícios da observação, destacam-se segundo Alexander (2009, tradução nossa):

a) obtenção da sensação do ambiente de trabalho;
b) descoberta de questões práticas;
c) estimula o diálogo entre as partes interessadas;
d) percepção de que dada abordagem não irá funcionar.

3.3.4 Workshops

Workshops possibilitam reunir as partes interessadas, e é considerada a chave primária para definir os requisitos funcionais do software, pelo fato de que permite uma conciliação entre as partes interessadas (PROJECT MANAGEMENT INSTITUTE, 2013, tradução nossa).

Segundo Alexander (2009, tradução nossa), workshop é um encontro especializado e estruturado para permitir com que as partes interessadas trabalhem juntas seguindo um planejamento de atividades.

3.3.5 Prototipação

Construir protótipos e demonstrá-los às partes interessadas em entrevistas ou workshops é uma maneira efetiva de descobrir ou validar requisitos (ALEXANDER, 2009, tradução nossa).

Prototipação é um método para obtenção de respostas rápidas das partes interessadas por providenciar um modelo de trabalho parecido com o esperado para o produto final, sem que haja a necessidade de começar a construí-lo (PROJECT MANAGEMENT INSTITUTE, 2013, tradução nossa).

Segundo Alexander (2009, tradução nossa), protótipos são desenvolvidos para muitos propósitos, incluindo avaliação tecnológica e arquitetural. Porém, prototipação de requisitos existe para um único propósito: levantar requisitos. A prototipação permite:

a) estimular as pessoas a melhorarem seus requisitos;
b) tornar a obtenção dos requisitos subjetiva, de forma a proporcionar um ambiente aberto para descobrir os requisitos.

Prototipação oferece o conceito de elaboração progressiva em ciclos interativos de criação de maquetes, experimentação de usuários, geração de respostas, e revisão. Quando os ciclos interativos tem fim, os requisitos extraídos possibilitam avançar o projeto para a fase de construção (PROJECT MANAGEMENT INSTITUTE, 2013, tradução nossa).

3.3.6 Reuso de requisitos

De acordo com Alexander (2009, tradução nossa) o reuso de requisitos serve para responder as seguintes questões:

a) posso conservar tempo ou dinheiro reusando alguma coisa?
b) já possuo algo que faça XYZ?
c) o que esse produto possui que nós podemos reusar?

Ainda, na concepção de Alexander (2009, tradução nossa) reusar requisitos não é uma tarefa fácil, pois o reuso de requisitos não tem sentido se não tiver imerso a um contexto.

O reuso de requisitos consiste em buscar semelhanças entre os softwares já existentes no mercado com o escopo do projeto atual. Pode ainda incluir regulamentações, padrões ou melhores práticas da indústria (ALEXANDER, 2009).

3.4 ENGENHARIA WEB

Para Ginige (2001) e Murugesan (2001) muitas pessoas confundem engenharia WEB como uma cópia da engenharia de software. Embora ela adote muitos dos princípios da engenharia de software, o seu foco principal está no desenvolvimento e manutenção de sistemas WEB de qualidade. Que para tanto incorpora novas técnicas e linhas guias para atingir os objetivos únicos do ambiente WEB.

Baseado nas definições propostas por Ginige e Murugesan (2001), consolida-se como um marco na área de engenharia, uma vez que trata com maior relevância questões antes não pensadas no âmbito dos sistemas tradicionais, tais como: interfaces gráficas com o usuário, aplicações voltadas para conteúdo, desenvolvimento de hipermídia, indexação de informação para recuperação ágil de conteúdo e concorrência (acesso simultâneo a dados).

Devido à crescente ascensão da internet evidenciou-se uma dificuldade na construção de páginas, visto que a simplicidade tornou-se obsoleta, uma vez que a internet foi aprimorando, aperfeiçoando e incluindo novas formas de conteúdo. Dessa forma, a engenharia de software veio acompanhando essa evolução e dispondo de outros instrumentos para auxiliar os engenheiros de software. O modelo Relationship Management Methodology (RMM) proposto por Isakowitzet al (1995) contempla sólidos princípios e está dividido em sete fases fundamentais:

a) **modelagem de entidade-relacionamento:** contempla a informação de domínio da entidade e suas relações com outras entidades;
b) **modelagem de fatias:** trata de como informações isoladas devem ser apresentadas;
c) **modelagem de navegação:** contempla a forma de acesso a informação por parte dos usuários;
d) **modelagem de interface:** trata a apresentação de informações ao usuário;
e) **protocolo de conversão:** verifica como transformar informações

abstratas em construções físicas;

f) **comportamento de execução:** trata como a aplicação se comporta frente a dados dinâmicos;

g) **construção e testes:** contempla a maneira pela qual softwares são desenvolvidos e testados.

Rodriguez et al (2002) segundo sua visão definem que o processo de desenvolvimento para WEB pode ser retratado em função de dois sub-processos: autoria e desenvolvimento de infra-estrutura.

A figura 2 proposta por Rodriguez et al (2002) retrata o fluxo de trabalho em um modelo cascata. Onde se tem conhecimento:

a) dos requisitos de autoria (*Reqauth*) aos quais são obtidos através dos requisitos gerais;

b) dos domínios de autoria (*Domauth*) demonstrando os atributos das entidades envolvidas no processo, bem como o relacionamento com as demais entidades da aplicação;

c) dos domínios de navegação (*Domnav*) demonstrando a inter-relação entre páginas;

d) da interface com o usuário (UI) apresentando como o conteúdo deverá ser visualizado pelo usuário;

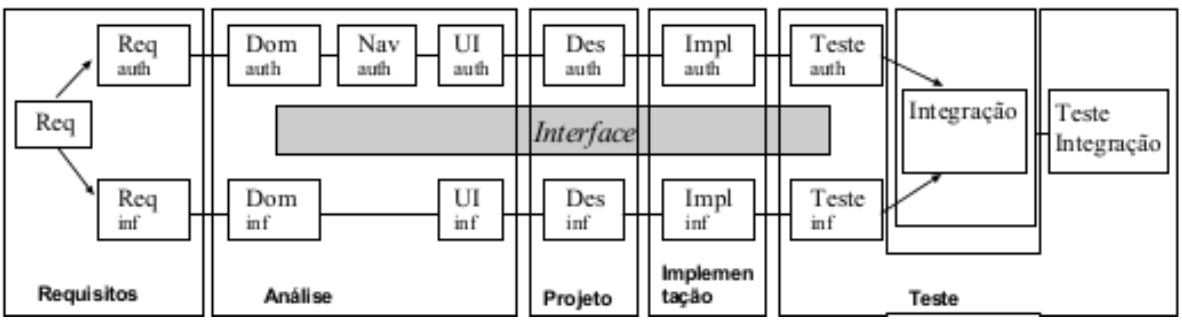
e) desenvolvimento de autoria onde cria-se o diagrama de classes completo do projeto;

f) implementação de autoria, que cuida dos demais conteúdos da página, como links e hipermídias;

g) teste de autoria para validação do sistema.

Embora os autores tenham mencionado os elementos de infra-estrutura, não foi aprofundado quanto aos detalhes do mesmo.

Figura 2 - Fluxo de trabalho do modelo cascata na engenharia WEB



Fonte: Rodriguez et al (2002).

Com o uso do conhecimento obtido por meio dos estudos de engenharia de software e das técnicas para levantamento de requisitos torna-se possível obter os requisitos de forma mais clara e objetiva, e os modelar de acordo com a arquitetura que melhor represente as propostas do projeto.

Sobretudo, com o avanço da tecnologia houve também uma adequação da engenharia de software e da engenharia de requisitos, cercando os engenheiros de novas instrumentações para realizarem seu trabalho idealizando plataformas mais bem estruturadas.

4 ARQUITETURA DE SOFTWARE

Conceitua-se arquitetura de software, baseando-se nas primeiras definições propostas por Perry (1992) e Wolf (1992, tradução nossa) como sendo um conjunto composto por elementos arquiteturais, sua forma de uso e a escolha dessas formas e elementos em um projeto.

Bass, Clements e Kazzman (2013, tradução nossa) definem arquitetura de software como o conjunto de estruturas necessárias para realizar um sistema, compreendendo elementos, relações e propriedades entre eles.

Baseado nestas duas definições compreende-se que a arquitetura de software é a peça fundamental para o desenvolvimento de uma aplicação, visto que estabelece os elementos que farão parte do projeto, bem como o relacionamento deles.

As decisões tomadas durante a criação da arquitetura são fundamentais, pois estabelecem um estágio para outras decisões que virão a ser tomadas no futuro. Alguns sistemas, que não empregam o uso de uma arquitetura refinada são difíceis de construir e manter e podem não atender a necessidade dos consumidores. Empregar arquitetura de software é dar um passo a mais para o sucesso do software (HANMER, 2012).

Para se ter noção da importância da arquitetura de software, Dashofy, Medvidovic e Taylor (2010) fazem uma analogia com os antigos castelos medievais, onde o emprego de paredes grossas, janelas altas e estreitas ou inexistentes contribuíram para a característica primordial do castelo: defender-se de ataques de espadas e flechas. Essa característica da construção, ou arquitetura do castelo determinou uma de suas propriedades, e é assim que se evidencia a prática da construção de software, as características dos mesmos dependem fundamentalmente de sua arquitetura.

A arquitetura de software cobre alguns aspectos dos sistemas, como o propósito do mesmo, quem o usará, de quais outros sistemas ele é dependente, como irá funcionar nos computadores do usuário final, os subsistemas e componentes, sua interface com o usuário, entre outras (HANMER, 2012, tradução nossa).

Embora existam vários modelos de arquitetura de software, um modelo ganhou notório destaque, segundo Schildt (2011, tradução nossa) este modelo

prevê a separação de componentes baseado em três aspectos fundamentais:

a) como o componente é visualizado na tela;

b) como o componente reage as interações com o usuário;

c) o estado das informações associadas ao componente.

Baseado nessa última característica podem-se diferenciar explicitamente uma arquitetura WEB de uma desktop. Fundamentalmente, toda arquitetura desktop mantém estado, em virtude da execução do software acontecer em uma única máquina e ser dependente do ambiente em que executa, ao passo em que na WEB, toda conexão via protocolo Hypertext Transfer Protocol (HTTP) é livre de estado. Porém, isso não impede que componentes na WEB armazenem estado, devido a isso, no ambiente WEB constata-se arquiteturas de software *stateful* (mantém estado) e *stateless* (não mantém estado). Para se compreender melhor as arquiteturas no ambiente WEB, é necessário conhecer o protocolo HTTP, para que se tenha noção de como intimamente se dá o seu funcionamento.

4.1 HTTP

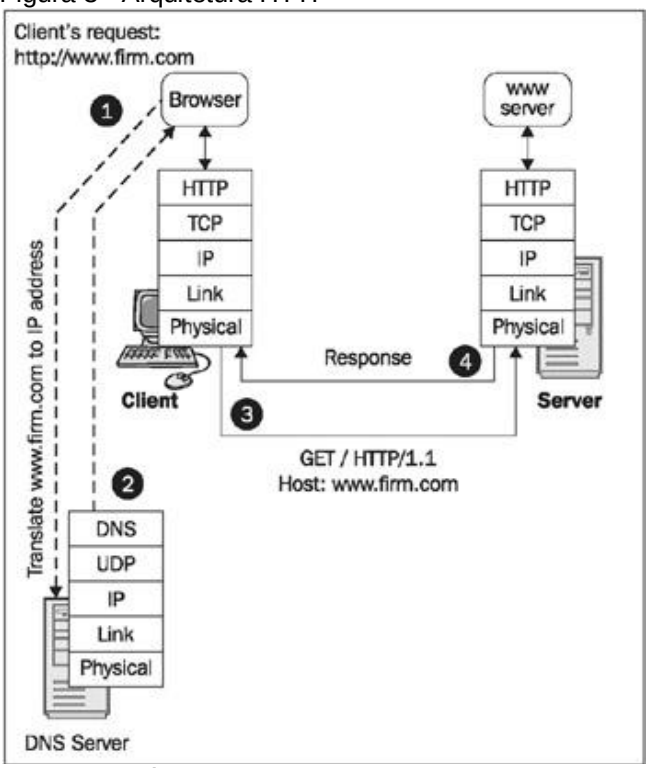
A WEB dispõe aos seus usuários um ambiente dinâmico, pois permite a troca de informações e a interação com um meio flexível, demonstrando os mais variados tipos de conteúdos, desde puros textos até conteúdos de hipermídia complexos, tais como vídeos. Muito dessa dinâmica deve-se ao protocolo HTTP, que estabelece as formas dessa interação por meio de uma arquitetura completa.

A arquitetura do protocolo HTTP permite que uma máquina se conecte a uma rede de computadores trocando mensagens com outros elementos integrantes desta rede, sejam eles outras pessoas fazendo o uso de seus computadores pessoais ou servidores de páginas.

A troca dessas mensagens deve-se as regras impostas pelo modelo do protocolo, que determinam como uma máquina deve interagir com outra na rede para que possam se entender.

A figura 3 demonstra a relação entre todos os elementos envolvidos durante a comunicação de uma máquina com um servidor de páginas, relacionando as etapas da solicitação iniciada pela máquina pedinte aos elementos da arquitetura HTTP, com o propósito de demonstrar a função de cada elemento dela.

Figura 3 - Arquitetura HTTP



Fonte: Dostálek (2006).

Fundamentalmente a arquitetura HTTP é composta por dois elementos que interagem, de um lado o cliente e de outro o servidor de páginas.

Ao solicitar uma página, primeiramente o *browser* resolve o endereço através de um pedido à um servidor Domain Name Service (DNS), o qual será responsável por entregar o Internet Protocol (IP) do servidor de páginas solicitado, como pode ser observado na figura 3, etapas 1 e 2 (DOSTÁLEK, 2006, tradução nossa).

Através do endereçamento, a requisição é entregue ao servidor HTTP, na etapa 3, o qual é prontamente interpretado pelo servidor que lhe fornece uma resposta, etapa 4 (DOSTÁLEK, 2006, tradução nossa).

Com isso, pode-se perceber que o cliente fica responsável por demonstrar a página e solicitar pedidos de navegações, enquanto o servidor deve fazer todo o processamento dos dados para construção da página e decidir para quais páginas encaminhar o cliente.

Desta forma, o cliente não mantém nenhum estado sobre os dados quais está trafegando, sendo responsabilidade do servidor identificar qual cliente está interagindo consigo e fornecer os dados apropriados.

Isto diverge da arquitetura desktop, uma vez que neste tipo de arquitetura

não há distinção entre cliente e servidor, ocorrendo o processo por inteiro na máquina do cliente.

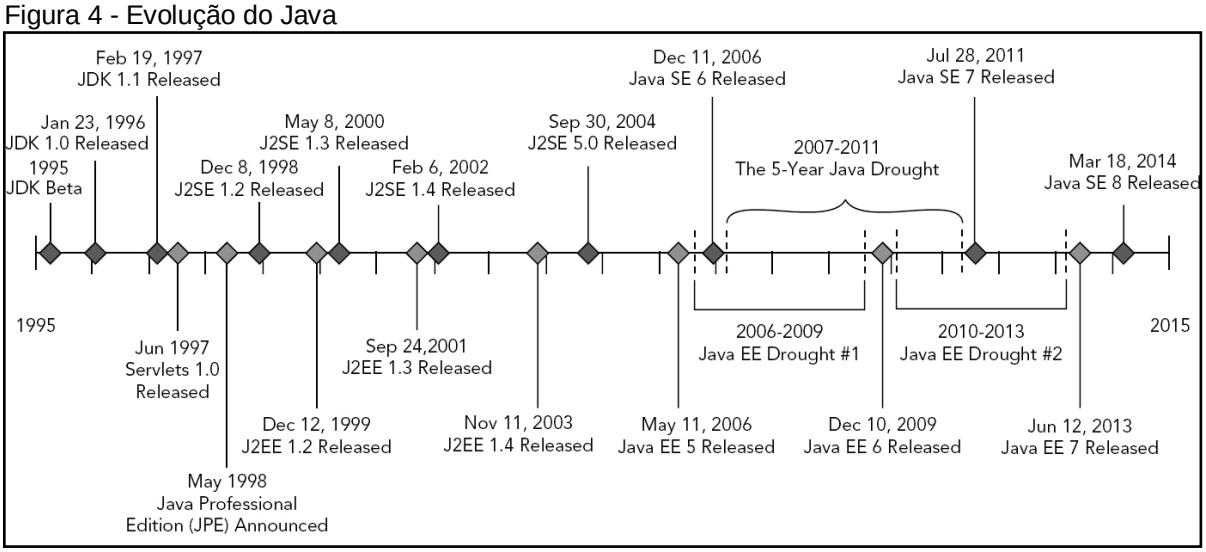
Conhecendo a base da WEB e o funcionamento do protocolo HTTP fica mais fácil entender o porquê do modelo Model-View-Controller (MVC), ou modelo-visão-controlador, ser um sucesso no desenvolvimento de software. Isso se dá pelo fato de que cada peça de seu modelo corresponde a uma das três partes propostas por Schildt. O modelo corresponde à informação de estado da arquitetura associada a cada componente. A visão determina como o componente é demonstrado na tela, e é dependente das informações do modelo para determinar se uma ou outra informação deve deixar de aparecer na tela. O controlador é o gerenciador dos estados do modelo e de qual visualização deve ser demonstrada ao usuário, ele é a peça que reage as ações do usuário e determina o fluxo da aplicação (SCHILDT, 2011, tradução nossa).

Dado a importância dessa divisão entre camadas, alguns dos *frameworks* mais conhecidos em desenvolvimento para WEB utilizam essa divisão para compor sua arquitetura.

4.2 JAVA PARA WEB E FRAMEWORKS DE DESENVOLVIMENTO

Um dos motivos que encorajaram a escolha do Java como plataforma de desenvolvimento para esse trabalho de conclusão de curso, está na sua história e recursos incorporados a seu kit de desenvolvimento desde sua origem. A figura 4 demonstra a história do Java como plataforma de desenvolvimento.

Figura 4 - Evolução do Java



Fonte: Schildt (2011).

Alguns pontos cruciais que fazem da plataforma Java uma das mais completas ferramentas de desenvolvimento para WEB podem ser descritos, segundo Williams (2014) como:

a) linguagem familiar fortemente tipada e orientada a objetos;
b) adaptabilidade em diferentes plataformas, uma vez que roda sobre uma máquina virtual que interpreta os códigos Java para a máquina real;
c) kit de desenvolvimento dispõe de compilador, gerador de documentação, ferramentas para se trabalhar com código nativo e depurador de erros.

Além das características da plataforma de desenvolvimento, podem-se ressaltar a arquitetura de desenvolvimento proposta pela versão *enterprise*, que prevê a separação da aplicação em camadas, e conta com uma rica e detalhada especificação que determina como a estrutura deve ser implementada.

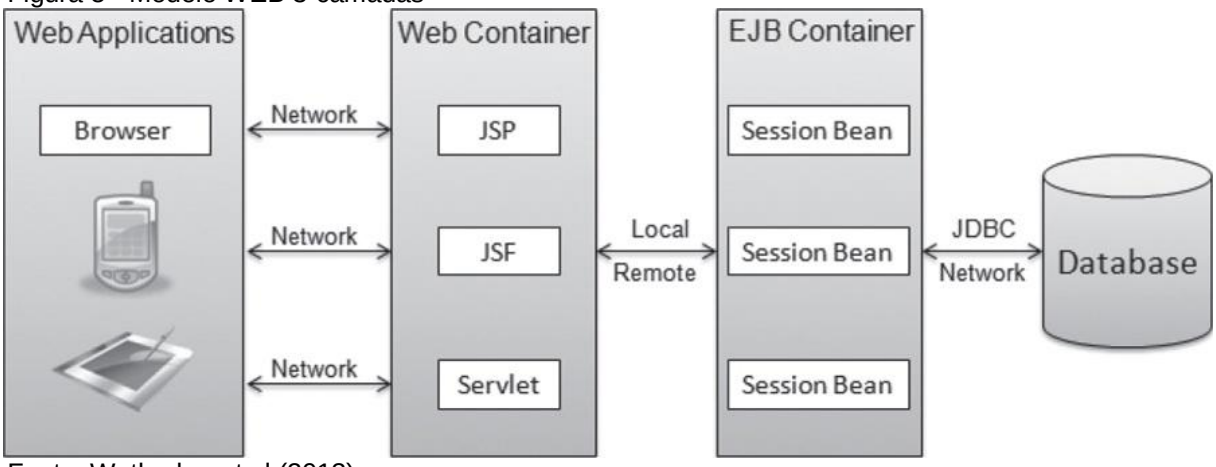
Pode-se ainda, com relação ao tradicional modelo MVC, fazer uma comparação com o modelo de arquitetura Java e verificar que várias tecnologias implementam as características do modelo MVC no mundo Java. A primeira delas, trata-se da Enterprise Java Beans (EJB), e diz respeito a camada de modelo, onde se tem a gerência das transações e regras do negócio que envolvem os modelos de dados da aplicação. A segunda tecnologia, diz respeito as camadas de visão e controle. Eis que nesse âmbito, alguns frameworks de alto desempenho estão presentes, alguns dos mais famosos são: Google Web Toolkit, Spring MVC e Java Server Faces.

Os servidores de aplicação que implementam o modelo EJB, dispõe de um contêiner responsável por administrar os objetos que implementam as regras de negócio da aplicação. Esses objetos são mais conhecidos como *Session Beans*.

Session Beans são componentes que residem em um contêiner EJB e são responsáveis por implementar uma tarefa ou um caso de uso mantendo um estado de conversação com a aplicação cliente. O contêiner providencia recursos para os *Session Beans* em formas de serviços, que são indicados pelos próprios *Session Beans* através de um recurso Java, conhecido como anotações (WETHERBEE et al, 2013).

A figura 5 retrata o modelo WEB em três camadas usando a arquitetura Enterprise Java Beans.

Figura 5 - Modelo WEB 3-camadas



Fonte: Wetherbeeet al (2013).

Este modelo mostra a relação entre as camadas. Mediante uma solicitação ou requisição por parte de uma aplicação WEB rodando em um *browser*, o contêiner do servidor é responsável por direcionar a requisição para uma das tecnologias presentes em seu ambiente, e através da comunicação com os EJBs realiza as operações necessárias às regras de negócio da aplicação.

Como se pode constatar, muitos dos EJBs podem comunicar-se com unidades de persistências, tais como banco de dados, sendo assim, uma outra tecnologia muito difundida no mundo Java, diz respeito a persistência de dados, e é conhecida como Java Persistence API (JPA).

Um dos mais famosos e reconhecidos frameworks responsável por implementar as especificações da JPA é o Hibernate. Ele trata da comunicação entre a aplicação e o banco de dados através de um *Session Bean*. Esta tecnologia é capaz de fazer um mapeamento objeto-relacional entre as entidades da aplicação e as tabelas de persistência do banco.

Os modelos de objetos e as tabelas do banco de dados tem muita coisa em comum, deveria haver uma maneira de se converter um modelo em outro. Mediante esta ideia surgiu o mapeamento objeto-relacional, que tem como fundamento fazer uma relação entre a tabela do banco de dados com o modelo da aplicação, tornando possível e mais fácil trabalhar com códigos puramente Java, ao invés da linguagem de banco (KEITH, 2009; SCHINCARIOL, 2009).

Por fim, a camada de apresentação dispõe de ferramentas para demonstrar conteúdo aos usuários. O resultado do processamento dos dados nas camadas de modelo e controle deve de alguma forma chegar ao usuário final e esta

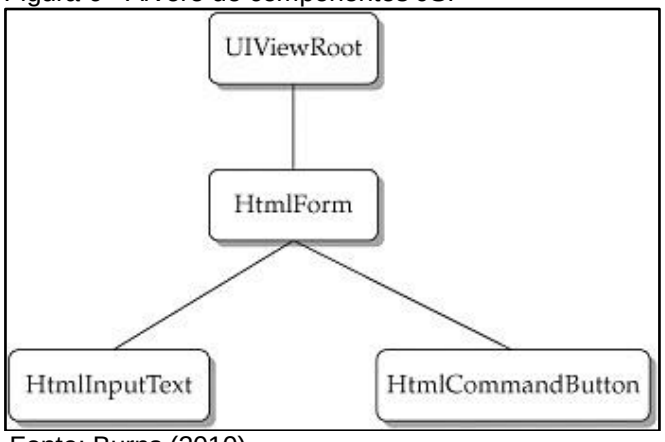
é a função das tecnologias de apresentação.

Entre as tecnologias mais empregadas no mundo Java pode-se destacar o Java Server Pages (JSP) e o Java Server Faces (JSF). Esta última, é uma implementação repaginada e melhorada da primeira tecnologia, e torna a implementação da tela ou visualização menos verbosa através do uso de *tags*.

Java Server Faces (JSF) é um *framework* para desenvolvimento de páginas, ou mais precisamente, desenvolvimento da camada de visão da Java Enterprise Edition (JEE). Ele simplifica a maneira pela qual o desenvolvedor constrói a interface gráfica da aplicação WEB através do uso de componentes ricos e que mantém estado durante as enésimas requisições de iteração com a aplicação. Fornece ainda um ambiente amigável e uma série de padrões de arquitetura para aplicações WEB (BURNS, 2010, tradução nossa).

O JSF permite aos desenvolvedores construir a aplicação utilizando várias visualizações, como se fosse um quebra-cabeça, onde em cada lugar uma peça deve ser encaixada. Algumas dessas peças fundamentais são os componentes, aos quais são dispostos na visualização em forma de *tags*, de forma estruturada, e muitas vezes um englobando o outro, através do recurso de abre e fecha *tags*. Essas *tags* possuem atributos que informam como o componente irá ser construído, ou até mesmo seu comportamento. Ao passo em que, o desenvolvimento por parte do programador está em escrever *tags*, o JSF se encarrega de interpretá-las e executar as rotinas desenvolvidas em código Java para escrever a saída final ao usuário (JUNEAU, 2013, tradução nossa).

Figura 6 - Árvore de componentes JSF



Fonte: Burns (2010)

As *tags* escritas pelo programador são interpretadas e estruturadas no servidor em forma de uma árvore de componentes, como pode ser obervado na

figura 6, a qual mantém o estado dos componentes entre as requisições (BURNS, 2010, tradução nossa).

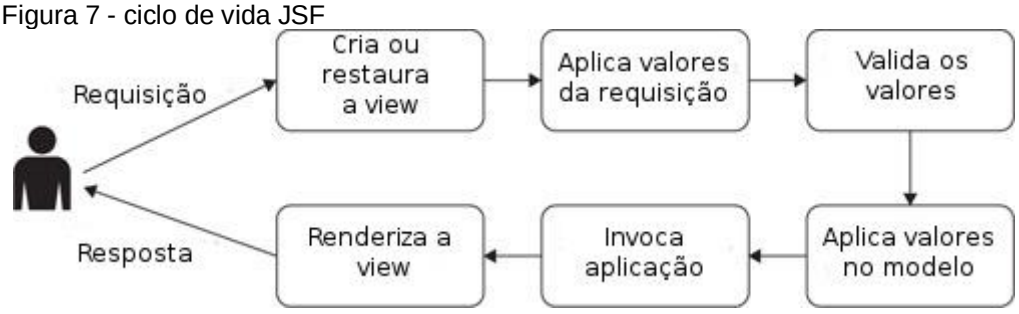
Através do uso de *facelets*, é possível trabalhar com *template* de páginas, as quais supõem a construção das páginas através da composição de outras páginas (VOHRA, 2014, tradução nossa).

Outro recurso de notório destaque é a possibilidade de se trabalhar de forma nativa com *ajax*.

Ajax possibilita transferências de dados de forma assíncrona entre o cliente e o servidor. JSF permite que requisões *ajax* sejam atendidas e promovam um processamento parcial da página e de um grupo de componentes (VOHRA, 2014).

O funcionamento do *framework* pode ser demonstrado na figura 7.

Figura 7 - ciclo de vida JSF



Fonte: Adaptado de Burns (2010).

Segundo Burns (2010) quando o usuário requisita pela primeira vez uma página no servidor, uma *view* (visualização) contendo os componentes da página é criada e então retornada ao usuário, prontamente renderizando-a, sem que se execute as demais fases do ciclo de vida.

Quando sucessivas requisições são realizadas ao servidor buscando a mesma página, então o JSF se encarrega de restaurar a *view*, aplicar os valores de entradas provenientes do usuário em campos de formulário, processar a validação dessas entradas, podendo ocorrer até duas rotas no ciclo de vida dependendo se os valores da requisição são válidos ou não. Caso não-válidos, renderizam a *view* e informam o usuário das entradas inválidas. Caso válidos, aplicam os valores no modelo, invocam a aplicação para tratamento de eventos específicos e regras do negócio e posteriormente renderizam a visualização como resposta final ao usuário (BURNS, 2010).

5 MEDICINA E INFORMÁTICA

A utilização de softwares na área da saúde, sobretudo, os que realizam estudos de metanálise, são de fundamental importância para refinar os conhecimentos sobre quais métodos de diagnósticos, medicamentos e /ou tratamentos são mais efetivos para a maior parcela da população.

A área de informática médica dispõe de ferramentas e instrumentos que exercem um papel de apoio em uma consulta médica, desde a captura da informação, seu armazenamento e o respectivo processamento das informações do paciente, que contribuirão para um diagnóstico efetivo e uma orientação terapêutica correta, como também contribuirá na melhoria do próprio conhecimento médico, disponibilizando-o aonde ele for necessário, para que haja uma tomada de decisão apropriada (WECHSLER et al, 2003).

Porém, como afirma Cluter e McClellan (2001), o crescente uso de tecnologias e inovações tecnológicas não só contribuiu para a queda de mortalidade e melhoria de vida, como também desencadeou um aumento vertiginoso nos gastos com assistência médica.

Devido a este aumento, a partir da década de 80, ocorreu um maior investimento por parte de políticas governamentais de diversos países em atividades de avaliação tecnológica em saúde, as quais incluem a revisão sistemática, metanálise, a análise de decisão e a análise de custo-efetividade (SILVA, 2003).

5.1 REVISÃO SISTEMÁTICA

De acordo com Rosemberg (1995), nem todos os diagnósticos, prognósticos e tratamentos terapêuticos são obtidos com total certeza, muitos são resultados de uma análise baseada em evidências.

A constatação de que evidências geradas por pesquisadores em todo o mundo não chegavam de forma adequada aos médicos contribuiu para o desenvolvimento das Práticas Baseadas em Evidência (PBE). Ela tem por objetivo determinar a potencialidade de uso e validade de uma evidência por meio da aplicação de conhecimentos da epidemiologia e bioestatística (SACKETT, 2003).

Aplicar esse paradigma exige uma questão clínica que gere dúvidas, então, tem-se a realização de uma revisão sistemática da literatura médica sobre o

tema envolvido nesta dúvida. O uso da revisão sistemática tem por objetivo auxiliar os médicos na implementação de condutas validadas por meio da análise crítica de estudos científicos (MEDEIROS, 2001).

A revisão sistemática trata-se de uma forma de estudo que visa, a partir de uma abordagem sistêmica, sintetizar resultados de vários estudos sobre o impacto e efeito de determinadas tecnologias (SILVA, 2003).

Trata-se de uma revisão planejada segundo Castro (2001), onde se pretende responder uma pergunta específica utilizando métodos sistemáticos para identificar, selecionar e avaliar de modo crítico os estudos incluídos na revisão.

O estudo é realizado de modo que se determine o tamanho do efeito e para que pacientes ou pessoas o efeito ocorreu, sem que se distorça o tamanho do efeito estudado. Sobretudo, gerando uma base de conhecimento para a tomada de decisão de clínicos, gerentes e planejadores (SILVA, 2003).

Segundo Dickersin (1994 apud SILVA, 2003), uma revisão sistemática consiste em abranger o máximo de ensaios clínicos, estudos, artigos e outros materiais relevantes, sejam eles, publicados ou não, utilizando-se de uma busca eficiente, através de uma estratégia já validada.

Para tanto, tem-se como base para um planejamento adequado de uma revisão sistemática duas abordagens. A primeira baseada na publicação da colaboração Cochrane em seu livro Cochrane Handbook. A segunda baseada na publicação do CDR Report produzido pelo National Institute for Health Research (NHS) Centre for Reviews and Dissemination (CASTRO, 2001).

5.1.1 Revisão sistemática segundo Cochrane

Segundo Castro (2001) a colaboração Cochrane detalha a revisão sistemática bem elaborada em sete passos fundamentais:

a) **formular a pergunta:** a pergunta deve abranger os pacientes ou doenças e a intervenção. Deve ser concisa e clara, pois uma questão mal elaborada contribui para uma revisão de má qualidade;
b) **localizar e selecionar assuntos:** trata das fontes dos dados e a preocupação em avaliar as referências bibliográficas dos estudos;

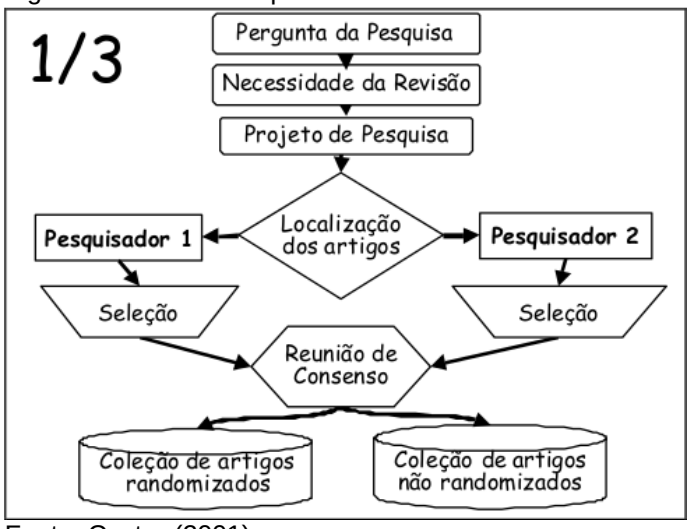
c) **avaliação crítica dos estudos:** determina os estudos válidos a serem utilizados na revisão sistemática por meio de critérios e através da probabilidade de suas conclusões se basearem em dados viciados;
d) **coletar dados:** nesta etapa avalia-se as variáveis do estudo, como também as características do método, dos participantes e dos desfechos clínicos, a fim de se obter uma medida em que se possa comparar, se possível for, os estudos selecionados;
e) **analisar e apresentar:** agrupa-se os estudos com base em suas semelhanças a fim de que se possa aplicar a metanálise;
f) **interpretar os dados:** determina-se a força da evidência encontrada em cada estudo, como também outras informações, tais como: aplicabilidade dos resultados e custos;
g) **aprimorar e atualizar a revisão:** caracteriza-se pela pós-publicação, onde tem-se em mente que ela será alvo de críticas que contribuirão para aperfeiçoá-la. De modo que, toda revisão deve ser atualizada para se manter viva, sendo que, a necessidade da revisão dependerá da quantidade de publicações relacionadas ao tema publicadas recentemente na área. Quanto mais publicações fomentando o assunto, maior a necessidade de revisá-la.

5.1.2 Revisão sistemática segundo NHS/York

Assim como na revisão segundo Cochrane, deve-se iniciar com uma pergunta a ser respondida, e então definir a necessidade de se fazer a revisão. Com base nisso deve-se selecionar os artigos. Etapa pela qual pode ser desenvolvida por vários pesquisadores (CASTRO, 2001).

A figura 8 demonstra a primeira parte da revisão sistemática.

Figura 8 - Primeira etapa da revisão sistemática



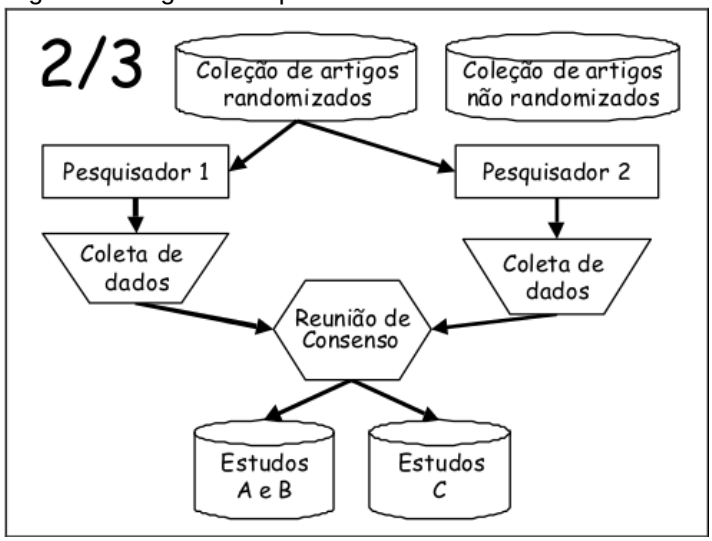
Fonte: Castro (2001).

Como pode ser observado, após a realização da localização dos artigos por parte dos pesquisadores, há uma reunião para consentir sobre os documentos pesquisados e separá-los sobre o ponto de vista randomizado ou não (CASTRO, 2001).

Na segunda etapa, segundo Castro (2001) os estudos randomizados são distribuídos novamente aos pesquisadores dispondo de formulários padronizados a fim de nessa etapa obter-se a coleta de dados de quatro componentes: geral e sigilo da alocação, qualidade do estudo, resumo do estudo e resultado das variáveis estudadas. Ao final dessa etapa, uma nova reunião de consenso é esperada, onde se dividirá o resultado do trabalho dos pesquisadores com base no sigilo da alocação. Onde haverá estudos com descrição adequada (classe A), descrição inadequada (classe B) e os estudos onde o sigilo de alocação foi feita de forma errada (classe C).

A figura 9 resume a segunda etapa da revisão sistemática.

Figura 9 - Segunda etapa da revisão sistemática

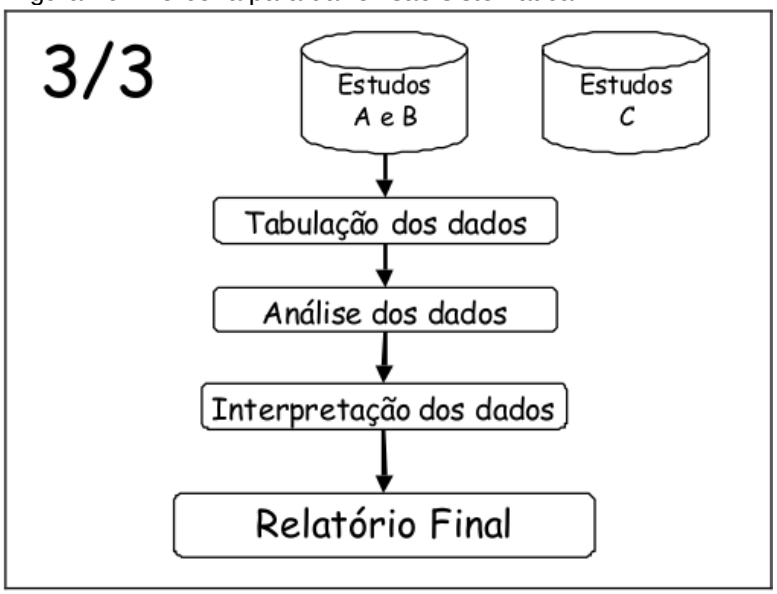


Fonte: Castro (2001).

Devido ao fato de se refinar tanto os materiais de consulta, agrupando-os por qualidade, participante e intervenção, normalmente têm-se um número menor de estudos que farão parte da metanálise.

No último passo da revisão sistemática, segundo Castro (2001) tem-se a tabulação e análise dos dados com posterior redação do manuscrito a ser encaminhado para publicação, assim como demonstra a figura 10.

Figura 10 - Terceira para da revisão sistemática



Fonte: Castro (2001).

Segundo Leeflang (2013, tradução nossa) revisões sistemáticas, ao passo em que sintetizam resultados de diferentes testes de precisão, também ajudam os pesquisadores a investigar porque os resultados variam entre estudos,

compararam as alternativas de testes de diferentes estudos e colocam a evidência em um contexto clínico.

5.2 METANÁLISE

A metanálise é o processo final de uma revisão sistemática onde se emprega análises estatísticas e modelos matemáticos que combinarão os resultados de estudos primários independentes sobre uma mesma tecnologia, buscando extrair dela uma medida concisa do efeito analisado (PETITTI, 2000).

Segundo Mulrow (1996), Oxman (1996) e Peto (1987 apud SILVA, 2003) a metanálise permite:

a) providenciar maior capacidade estatística para eliminar dúvidas;
b) mensurar melhor o tamanho do efeito em pacientes em geral e em subgrupos através das estatísticas;
c) resolver incertezas em estudos discordantes.

Ela combina resultados de vários estudos, a fim de se obter com maior precisão uma síntese que caracterize os efeitos e tamanho dos efeitos analisados em cada estudo, aumentando a potência estatística da pesquisa e dos efeitos do tratamento (BOISSEL, 1994).

É interessante pensar na metanálise como uma forma de auxiliar na sintetização de resultados de vários estudos, pois segundo Lovatto (2007), nas últimas décadas a produção científica aumentou consideravelmente, e portanto, o elevado número de publicações se tornou um problema para a análise qualificada da literatura, uma vez que com muitas publicações pode-se haver dificuldades em contextualizar um problema, ou ainda contribuir para uma análise ou interpretação errada.

Leeflang (2013, tradução nossa) afirma que muitos estudos de diagnósticos se focam em testes de precisão, porém mesmo que um teste tenha sido extremamente cauteloso, ele não significa necessariamente em uma melhora do paciente.

Por isso, durante muito tempo, várias publicações sugeriram métodos diferentes para se realizar a metanálise. Sobretudo, dependendo da área de interesse da pesquisa pode-se utilizar uma ou outra abordagem de metanálise para resolução do problema.

Lovatto (2007) afirma que o método proposto por Mantel e Haenszel em 1959 se tornou um dos principais para resolver a problemática de resultados obtidos por estudos independentes. Já Leeflang (2013, tradução nossa) sugere a utilização do método proposto por Moses e Littenberg para estudos que requerem um método mais rigoroso no tratamento das sensibilidades e especificidades, valores preditivos positivos e negativos e razões de verossimilhança, como também heterogeneidades inexplicadas entre os estudos.

A metanálise teve partida nas ciências sociais, na educação, na medicina e pouco depois na agricultura. Em todas essas áreas a maior problemática era a combinação dos resultados de estudos independentes. Os primeiros trabalhos que se interessaram por essa problemática foram realizados por Cochran (LOVATTO, 2007).

Segundo Leeflang (2013, tradução nossa) as primeiras publicações sobre testes de precisão utilizando metanálise foram feitas entre as décadas de 80 e 90 e tinham enfoque sobre como recuperar e selecionar estudos, assegurar a qualidade deles, sintetizar os resultados, investigar heterogeneidades e desenhar conclusões sobre os mesmos.

Lovatto (2007) reitera que para se obter um conhecimento que seja utilizável, um único experimento não pode ser conclusivo para uma inferência, dado que os seus resultados refletem as condições do experimento. Logo, a metanálise possibilita uma estimativa imparcial diferentemente de métodos tradicionais de revisão que muitas vezes não se preocupam com a análise das estatísticas.

Ainda segundo Lovatto (2007) a metanálise permite:

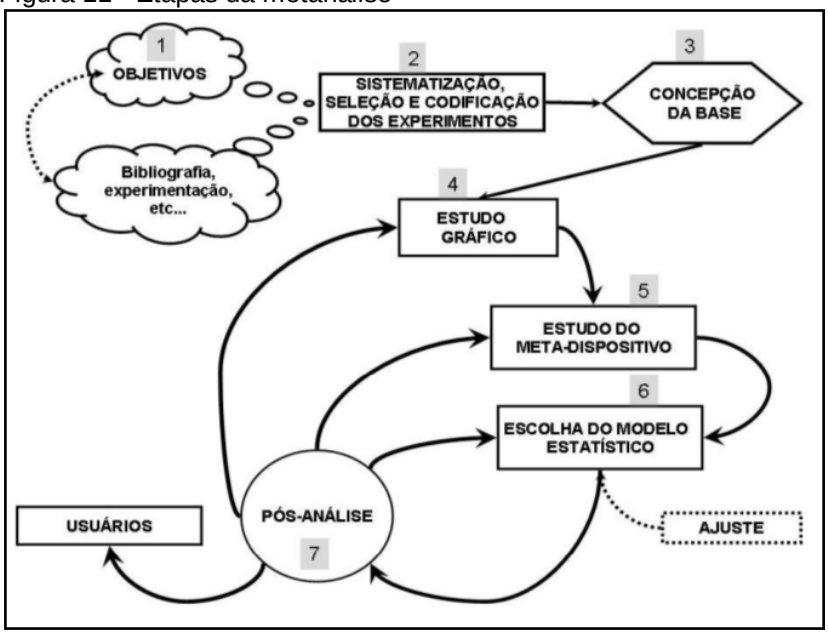
a) obter novos resultados, por melhorar a potência analítica de um modelo e possibilitar a evidência de diferença entre tratamentos permitindo que se obtenha uma conclusão, antes não possível;
b) sintetizar resultados contraditórios, analisando os resultados de estudos conclusivos e não-conclusivos de baixa ou alta potência analítica;
c) aumentar a precisão analítica, uma vez que a essência da metanálise está em selecionar uma maior quantidade de pesquisas para serem analisadas permitindo que se tenha uma fundamentação maior para definir o tamanho do efeito do tratamento;

d) melhorar a representatividade, dado que ao reagrupar resultados de diversos estudos se trabalha com uma população maior, o que permite corrigir a média geral dos estudos e possibilita o aumento da aplicabilidade por se trabalhar com diferentes situações;
e) gerar novas hipóteses, pois pode ser aplicada em qualquer fase da definição do problema. Levando em consideração que um tratamento produz um efeito, os resultados facilitam o cálculo de repetições do experimento e podem identificar novas hipóteses a serem estudadas.

Naturalmente, estudos resgatados pela metanálise diferenciam-se, e isto se chama heterogeneidade. A hetorogeneidade pode ocorrer por diferenças clínicas (população ou no tratamento), diferenças na metodologia do estudo (formulação do estudo e risco) e diferenças de estatísticas (população, amostragem e resultados) (MELSEN, 2014).

Com isso, para formulação de uma metanálise é necessário que se observem boas práticas contribuindo para uma metodologia mais rigorosa possível.

Figura 11 - Etapas da metanálise



Fonte: Lovatto (2007)

Na figura 11 estão representadas as principais fases da metanálise, que segundo Lovatto (2007) consistem em definir os objetivos, sistematizar as informações, codificar os dados, filtrá-los, analisá-los, escolher o modelo estatístico e pós-análisar.

5.2.1 Definição do objetivo

A definição do objetivo é parte fundamental de uma metanálise, pois é ele quem determina todo o processo meta-analítico.

Consiste em determinar qual é a problemática, quem serão os envolvidos e quem será o responsável por definir as seguintes fases, em particular, a codificação dos dados, a filtragem e ponderação e por fim o modelo estatístico (LOVATTO, 2007).

5.2.2 Sistematização das informações

Segundo Lovatto (2007), toda metanálise é baseada em sistematizar dados, que são obtidos principalmente da literatura científica, podendo ser obtidos de experimentos não-publicados.

A sistematização de dados exige que se limite a pesquisa no tempo e espaço, ou seja, em um período exclusivo de publicações de artigo e numa região geográfica específica sem que haja qualquer exclusão de artigos (LOVATTO, 2007).

É importante ainda observar que resultados significativos são publicados mais que os não significativos, por isso a importância de se utilizar também artigos não-publicados quando possível (LOVATTO, 2007).

5.2.3 Codificação dos dados

Trata-se da fase onde se atribui códigos as publicações e experimentos analisados. Esses códigos são fundamentais para interpretar os dados considerando os vários objetivos dos experimentos (LOVATTO, 2007).

5.2.4 Filtragem dos dados

A filtragem dos dados garante a qualidade da metanálise, pois assegura que uma publicação entrará no estudo somente se tiver coerência com os objetivos do mesmo (LOVATTO, 2007).

Conforme Lovatto (2007) as publicações analisadas devem ser exploradas exaustivamente através de uma leitura crítica com posterior transcrição

para a base de dados de estudo, com minuciosa precaução para se evitar erros de transcrição.

5.2.5 Análise dos dados

A análise permite identificar informações e relações importantes, esclarecendo hipóteses ou clareando pontos-chave para a escolha de um modelo estatístico. Permite ainda identificar condições particulares ou fora do real para indivíduos ou experimentos (LOVATTO, 2007).

5.2.6 Escolha do modelo estatístico

A escolha do modelo estatístico é baseado na variável de resposta do estudo, e geralmente se aplicam dois modelos, um para o caso da variável de resposta ser dependente de um fator qualitativo e outro para dependência de um fator quantitativo (LOVATTO, 2007).

Quando a variável de resposta é dependente de um fator qualitativo o modelo estatístico mais comum utilizado é a análise de variância, demonstrada na equação 1.

$$Y_{ij} = \beta_i + \alpha_j + (\beta\alpha)_{ij} + e_{ij} \quad (1)$$

Sendo que Y_{ij} é a variável de resposta, β_i é o efeito do tratamento, α_j é o valor experimental. O efeito da interação entre tratamento e estudo é dado por $(\beta\alpha)_{ij}$, e a variação residual é dada por e_{ij}.

Já quando a variável de resposta é dependente de um fator quantitativo então o modelo mais comum é a análise de variância-covariância, representada pela equação 2.

$$Y_{ij} = B_0 + s_i + B_1 * X_{ij} + b_i * X_{ij} + e_{ij} \quad (2)$$

Em análise à fórmula da variância-covariância apresentada, Y_{ij} refere-se à variável de resposta, B_0 refere-se a um termo constante, considerado geralmente como um efeito fixo. s_i é a ordenada à origem aleatória dos experimentos de i. B_1 é o coeficiente de regressão geral de Y sobre X. X_{ij} é a variável explicativa quantitativa, b_i o efeito aleatório do experimento i sobre o coeficiente de regressão de Y sobre X e e_{ij} é o erro aleatório.

5.2.7 Pós-análise

Esta etapa permite avaliar os limites da análise feita até então e definir se serão necessárias mais algumas análises complementares (LOVATTO, 2007).

É importante que se estude a distribuição dos resíduos (eij), se aplique outros testes como os de Qui-quadrado, Shapiro-Wilks e construam-se gráficos. Nesta fase o teste de t de *student* permite analisar se os resultados obtidos são aberrantes, porém toda a exclusão necessária deve ser feita com cautela, pois testes auxiliares podem contribuir para a completa exclusão da metanálise (LOVATTO, 2007).

6 TRABALHOS CORRELATOS

Este capítulo permite relacionar os trabalhos que abordam temas e aplicações similares a fundamentação teórica deste trabalho na obtenção de resultados que explorem a reutilização de requisitos de software, modelagem com foco na qualidade, como também a utilização de softwares no desenvolvimento de metanálises diagnósticas.

6.1 META-DISC: A SOFTWARE FOR META-ANALYSIS OF TEST ACCURACY DATA

Desenvolvido por Zamora et al (2006) e publicado na National Center for Biotechnology Information (NCBI) traz uma abordagem da metanálise através do uso de um software aberto denominado Meta-DiSc. Nesta publicação é abordado o uso do software, como deve ser tratado os resultados de estudos individuais, a heterogeneidade dos estudos, meta-regressão e a obtenção da conclusão da metanálise através do uso do software.

Verifica-se por meio do estudo que o software é uma ferramenta completa na condução de metanálises porque:

a) permite a exploração da heterogeneidade com uma variedade de cálculos estatísticos, tais como: Qui-quadrado, I^2 e testes de correlação de Spearman;
b) implementa técnicas de meta-regressão que exploram relações entre estudos;
c) possui testes estatísticos de acurácia, tais como, sensibilidade e especificidade, razão de verosimilhança e Diagnostic Odds Ratio (DOR) utilizando modelos de efeito fixos ou variados;
d) produz gráficos de alta qualidade, incluindo gráficos de floresta e curva Receiver Operating Characteristic (ROC) que podem ser utilizados nas publicações em revistas científicas.

6.2 A REUTILIZAÇÃO DE REQUISITOS NO DESENVOLVIMENTO E ADAPTAÇÃO DE PRODUTOS DE SOFTWARE

Desenvolvido por Silveira (2006), embora tenha foco na reutilização de requisitos para projetos de Enterprise Resource Plan (ERP) e Consumer Relationship Management (CRM), traz um texto claro e bem explicativo sobre a utilização de requisitos de software, sua importância, diferenças entre requisitos, técnicas e processos para seu levantamento, como também sobre processos de maturidade de software, com o qual este trabalho também está alinhado.

O objetivo do trabalho era desenvolver uma abordagem de engenharia de requisitos que pudesse resolver os problemas quanto a:

a) identificação e variabilidade dos requisitos na configuração e customização de produtos;
b) produção de documentos e modelos integrais na fase de implementação do software;
c) disponibilidade das informações de requisitos e configurações para suportar a evolução do software.

Com o estudo das abordagens já existentes e a elaboração de uma nova abordagem, proposta por este trabalho, constatou-se que:

a) a variabilidade não era modelada de forma explícita, e que agora foi permitido pela inclusão delas em modelos de casos de usos e diagramas de entidades parametrizados;
b) a parametrização era obtida somente ao final do desenvolvimento, e com o desenvolvimento dessa abordagem pôde-se incluir as descrições dos requisitos em cada etapa da implementação;
c) a resolução de problemas e erros foi antecipada pela inclusão prévia nos documentos de requisitos.

6.3 MODELAGEM DE UM PROCESSO PARA O GERENCIAMENTO E DESENVOLVIMENTO DE REQUISITOS BASEADO NO MODELO BRASILEIRO DE QUALIDADE DE PROCESSO MPS.BR

Desenvolvido por Silva (2010) na Universidade do Extremo Sul Catarinense (UNESC), tem foco nos modelos de processo, que contribuem de

maneira significativa para obtenção de software com maior qualidade, visto que norteia o ciclo de desenvolvimento de software através de níveis de maturidade.

O trabalho também contextualiza a importância da engenharia de software e da engenharia de requisitos, tendo como objetivo a elaboração de métodos, técnicas e padrões para o desenvolvimento e gerenciamento de requisitos de software.

Ele apresenta um dos níveis de maturidade de software do modelo MPS-BR que trata da gerência de projetos e de requisitos. O trabalho tem enfoque na gerência de requisitos e com os resultados do estudo concluísse que o emprego dos modelos de software visa à melhoria dos processos de software de uma empresa através do emprego das melhores práticas da engenharia de software.

6.4 UTILIZAÇÃO DOS REQUISITOS MANDATÓRIOS DE CONTEÚDO DA SOCIEDADE BRASILEIRA DE INFORMÁTICA EM SAÚDE E CONSELHO FEDERAL DE MEDICINA NA MODELAGEM DE UM SISTEMA DE REGISTRO ELETRÔNICO EM SAÚDE

O trabalho foi desenvolvido por Diego Garcia em 2009 para obtenção de grau de Bacharel em Ciência da Computação na Universidade do Extremo Sul Catarinense (UNESC).

O objetivo proposto era modelar um software para utilização no ambulatório da UNESC por meio do estudo e aplicação das normas e padrões dos requisitos mandatórios de conteúdo do processo de certificação SBIS/CFM.

Para tanto, a metodologia empregada se deu pelos estudos bibliográficos referentes ao tema, por levantamento de requisitos baseado em entrevistas com profissionais da área e pela confecção de diagramas de casos de uso, atividades e classes.

Teve como resultado os diagramas e protótipo de interface utilizando 21 dos requisitos mandatórios estabelecidos.

6.5 UTILIZAÇÃO DOS REQUISITOS RECOMENDADOS DO MANUAL VERSÃO 3.0 DE CERTIFICAÇÃO PARA SISTEMAS DE REGISTRO ELETRÔNICO EM SAÚDE DA SOCIEDADE BRASILEIRA DE INFORMÁTICA EM SAÚDE E DO CONSELHO FEDERAL DE MEDICINA

Realizado na Universidade do Extremo Sul Catarinense (UNESC) em 2008 por Diego Biz Freitas para obtenção do grau de Bacharelado em Ciência da Computação, pretendia verificar se o Registro Eletrônico de Saúde (RES) do sistema UTInfo 2.0 atende ao requisitos do manual versão 3.0 da certificação SBSI/CFM.

A metodologia teve início com a escolha do RES, a qual foi optado pelo UTInfo, que é um software para acompanhamento de pacientes da Unidade de Terapia Itensiva (UTI) do Hospital Regional de Araranguá (HRA).

Posteriormente, foi analisado tal RES, identificando primeiramente suas funcionalidades e em seguida foi realizado o estudo com base nas linhas guias do manual de requisitos.

O manual contempla os requisitos sobre a abordagem de:

a) **segurança:** inclui controle de versionamento de código, autorização e controle de acesso, segurança de dados (integridade, criptografia, etc);
b) **estrutura e conteúdo:** trata dos requisitos que contemplam a independência, recuperação, estrutura dos dados, e dados relacionados com o aspecto clínico, além da documentação do código fonte, instalação, configuração e operação do sistema;
c) **funcionalidade:** trata de alertas, lembretes, restrições e obrigatoriedades. Também relaciona os requisitos de exportação e importação de dados, trocas de registro e interoperabilidade;

Como resultado do trabalho foi avaliado que o software não atende os requisitos de documentação, e os demais requisitos são demonstrados na tabela 1. Nela foram avaliados 49 sub-requisitos.

Tabela 1 - Resultados obtidos na análise dos requisitos recomendados

Grupo de requisitos recomendados	Atendem	Não atendem	Atendem parcialmente	Total
Requisitos de segurança	1	12	2	15
Requisitos de estrutura e conteúdo	6	7	7	20
Requisitos de funcionalidade	0	9	5	14
Subtotais	7	28	14	49

Fonte: Freitas (2008)

7 ELICITAÇÃO DOS REQUISITOS E APLICAÇÃO DA ENGENHARIA DE SOFTWARE

A engenharia de software fornece os mecanismos e ferramentas necessárias para que se possam entender os requisitos que os clientes fornecem com intuito de se produzir software de qualidade.

Entretanto, é necessário que se conheça o objeto de estudo, para que assim seja possível conciliar as práticas de engenharia de software com os requisitos dos clientes.

Neste capítulo são apresentadas as características elicitadas e referentes ao processo de desenvolvimento da metanálise, expondo suas etapas e os perfis de usuários relacionados confrontando-os com os requisitos necessários para a elaboração do software. Também são demonstrados os métodos pelos quais foi possível obter os requisitos das partes interessadas e como se deu o processo de elaboração dos modelos, a definição da arquitetura de software e a prototipação e validação dos requisitos juntamente aos pesquisadores da área biomédica experientes no desenvolvimento de metanálises diagnósticas.

7.1 METODOLOGIA

A metodologia empregada neste trabalho compreendeu oito etapas, tendo início com o estudo das bibliografias e materiais de apoio sobre engenharia de software e de requisitos, como também sobre metanálise. Posteriormente, foram elencadas as atividades a serem realizadas, as quais são abordadas nas próximas seções.

7.1.1 Compreensão do domínio da aplicação

A compreensão do domínio da aplicação é parte fundamental do trabalho, pois perante as informações obtidas nesta etapa é possível entender e formular os próximos passos para extração dos requisitos e contemplação das partes interessadas.

Como visto na fundamentação teórica, a metanálise diagnóstica é uma forma de avaliação de trabalhos e estudos científicos reunidos pela revisão

sistemática. Nela são empregados testes estatísticos rigorosos desenvolvidos por instituições que regulamentam como a metanálise deve ser desenvolvida para que seja adequada, tais como a Cochrane. A aplicação de tais testes visam determinar uma conclusão sobre os resultados, dado que os estudos reunidos pela revisão sistemática e incluídos na metanálise podem ser discordantes.

Identificou-se que as etapas envolvidas com a elaboração da metanálise diagnósticas são:

a) entrada dos dados dos estudos;
b) avaliação da qualidade dos estudos incluídos;
c) análise tabular dos estudos;
d) análise gráfica dos estudos;
e) análise de viés de publicação.

A base de dados de uma metanálise diagnóstica consiste dos estudos incluídos nela por um processo rigoroso de busca e reunião, também conhecido como revisão sistemática. Na base de dados são incluídas as características dos estudos, tendo como entradas:

a) nome do estudo;
b) autor do estudo;
c) número de verdadeiros positivos;
d) número de falsos positivos;
e) número de verdadeiros negativos;
f) número de falsos negativos.

Essas características são as primordiais para se realizar os processos de cálculos estatísticos. Sendo que cada estudo deve ser composto destas informações. Porém, também é necessário que a base de dados possa permitir a inclusão de outras informações, a fim de se determinar um resultado mais preciso para a metanálise. Por exemplo, às vezes faz-se necessário analisar se o número de pessoas amostradas em cada estudo está influenciando no resultado da metanálise, ou se as condições em que o estudo foi realizado estão de alguma forma alterando o resultado final da metanálise diagnóstica. Existem fatores que podem contribuir para a conclusividade de uma metanálise, tais como:

a) cultura e ambiente, dado que os estudos reunidos podem ter sido feitos com pessoas de nacionalidades diferentes;
b) metodologia aplicada na realização do estudo;

c) quantidade de amostras;
d) método de diagnóstico;
e) idade;
f) sexo;
g) etc.

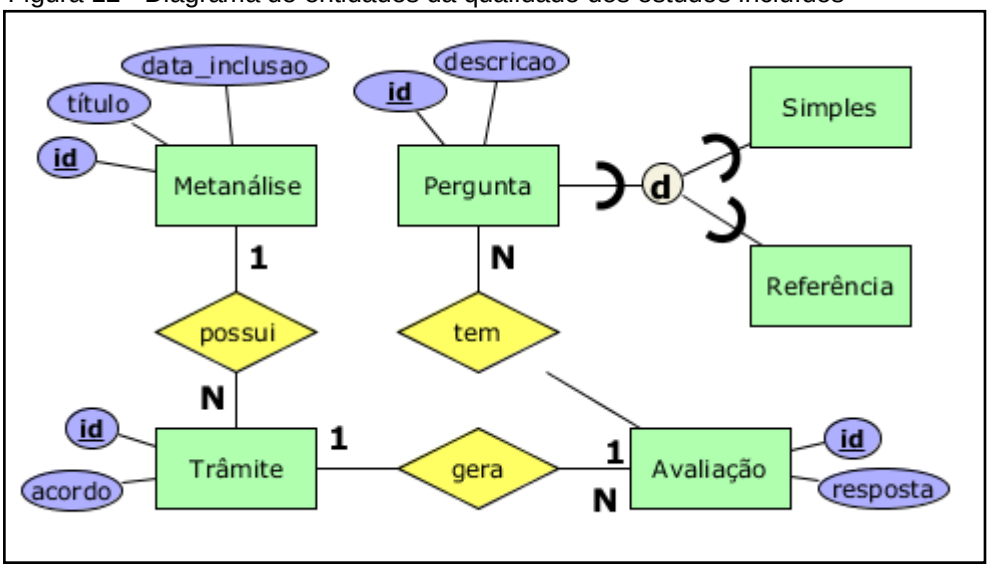
Com o estudo e identificação de tais etapas foi constatado que a base de dados deve estar apta para suportar entradas extras para cada estudo, para os casos em que se devam analisá-las.

Quanto à etapa de avaliação da qualidade dos estudos incluídos, com ela se pretende determinar a qualidade dos estudos que foram resgatados pela revisão sistemática. Ela busca responder algumas questões chaves, elaboradas por instituições que determinam como a revisão sistemática e metanálise diagnóstica devem ser realizadas, para então determinar se os estudos são confiáveis e válidos para fornecer uma resposta conclusiva sobre determinado tema. Porém, como avaliar a qualidade de estudos é muito subjetivo porque depende da opinião do pesquisador, foi constatado que mais de uma avaliação, por parte de outros pesquisadores é necessária.

Uma segunda opinião determina a necessidade de um sistema colaborativo, onde um pesquisador poderá ajudar outro na realização da metanálise. Entretanto, como as opiniões sobre a qualidade dos estudos que foram incluídos podem ser muito divergentes, estas avaliações se darão por meio de trâmites. Os trâmites só deixarão de ser registrados quando os dois pesquisadores que estiverem fazendo a avaliação entrarem em um consenso. Assim, a opinião dos dois contribuirá para uma melhor definição da qualidade dos estudos, diferentemente das ferramentas atuais, que por não serem colaborativas dependem da opinião somente do pesquisador que está desenvolvendo a metanálise. E então, nessas questões que envolvem subjetividade acaba-se prejudicando, por mais que o avaliador tenha experiência, sua metanálise.

O diagrama de entidades da figura 12 demonstra a relação da metanálise com as avaliações dos pesquisadores no que diz respeito à etapa de análise da qualidade dos estudos incluídos.

Figura 12 - Diagrama de entidades da qualidade dos estudos incluídos



Fonte: Do autor.

Como demonstrado no diagrama, verifica-se que:

a) a metanálise possui trâmites;
b) cada trâmite gera uma avaliação;
c) a avaliação é constituída de respostas para as perguntas;
d) as perguntas são relativas a qualidade do estudo incluído;
e) as perguntas podem ser simples e de referência;
f) perguntas simples são aquelas que não são demonstradas nos gráficos de avaliação do estudos, elas simplesmente abordam questões que levam aos pesquisadores a ter maior consciência para responder as perguntas de referência;
g) perguntas de referência são as que farão parte da avaliação dos estudos incluídos;
h) trâmites deixarão de gerar avaliações quando os pesquisadores que responderam as perguntas entraram em um acordo sobre as respostas, visto que a possibilidade de avaliação das perguntas por mais de um pesquisador é um meio para garantir uma melhor avaliação dos estudos incluídos, dado que envolve mais de uma opinião sobre os estudos.

Outra avaliação realizada como parte de uma metanálise, caracteriza-se pela etapa de análise tabular dos dados. Nela são empregados testes estatísticos para avaliar o conjunto dos dados dos estudos incluídos, tais como:

a) razão de verossimilhança;
b) diagnostic odds ratio (DOR)
c) valor preditivo positivo e negativo;
d) sensibilidade e especificidade;
e) I^2;
f) qui-quadrado.

A etapa de análise gráfica auxilia os pesquisadores a avaliar melhor a relação entre a sensibilidade e especificidade, por meio da curva Receiver Operation Characteristic (ROC). Outros gráficos, tais como o de funil, permitem avaliar os vieses de publicação.

A análise de viés de publicação consiste na avaliação de quais estudos foram incluídos na metanálise. E pretende responder as questões relativas a não-inclusão de estudos, ou seja, procura determinar se existem mais trabalhos satisfatórios publicados em relação aos que não apresentaram os resultados esperados. Essa parte verifica se existem publicações que foram intencionalmente deixadas de serem publicadas por não se ter um resultado conclusivo, ou uma metodologia adequada, ou um resultado duvidoso.

Tal compreensão do domínio da aplicação decorreu de entrevistas realizadas com as partes interessadas. Além das etapas da metanálise mencionadas acima, o domínio da aplicação também compreende os usuários e as configurações do sistema.

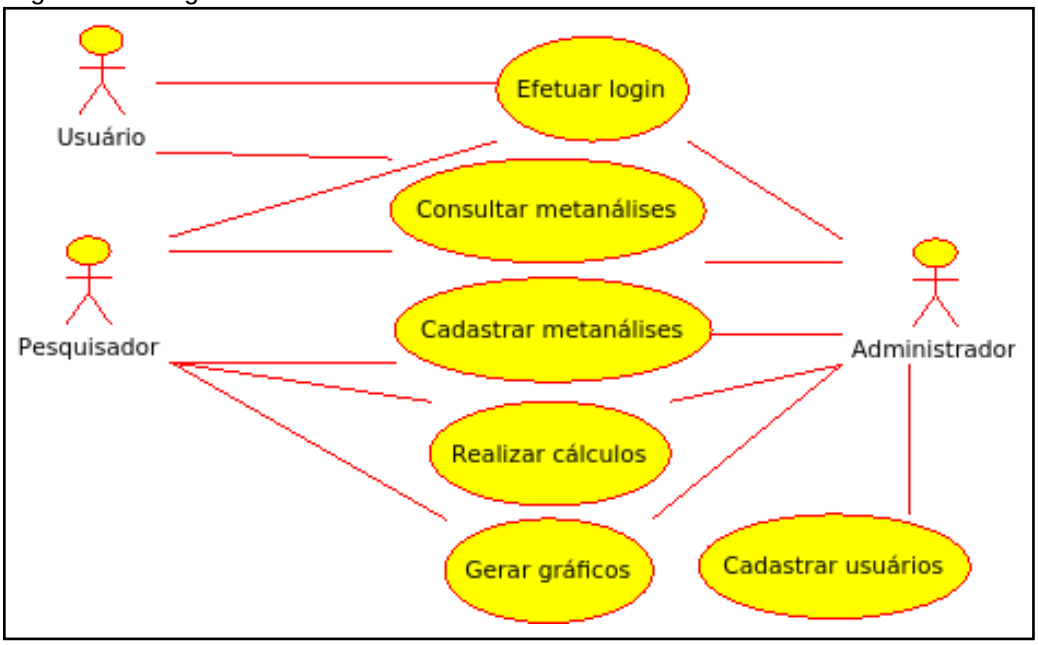
Os usuários podem ser compreendidos em três distintas categorias. O usuário de mais baixo nível, denominado “Consultor”, estará autorizado no sistema para somente consultar as metanálises diagnósticas finalizadas.

O usuário de nível intermediário, assim chamado “Pesquisador” estará autorizado a incluir novas metanálises e participar de metanálises existentes criadas por outros usuários, além de consultar metanálises existentes.

O usuário de mais alto nível, também conhecido como “Administrador” estará apto a realizar todas as tarefas, como também incluir novos usuários no sistema.

O diagrama de casos de uso da figura 13 demonstra a relação dos usuários com o sistema.

Figura 13 - Diagrama de casos de uso de usuários

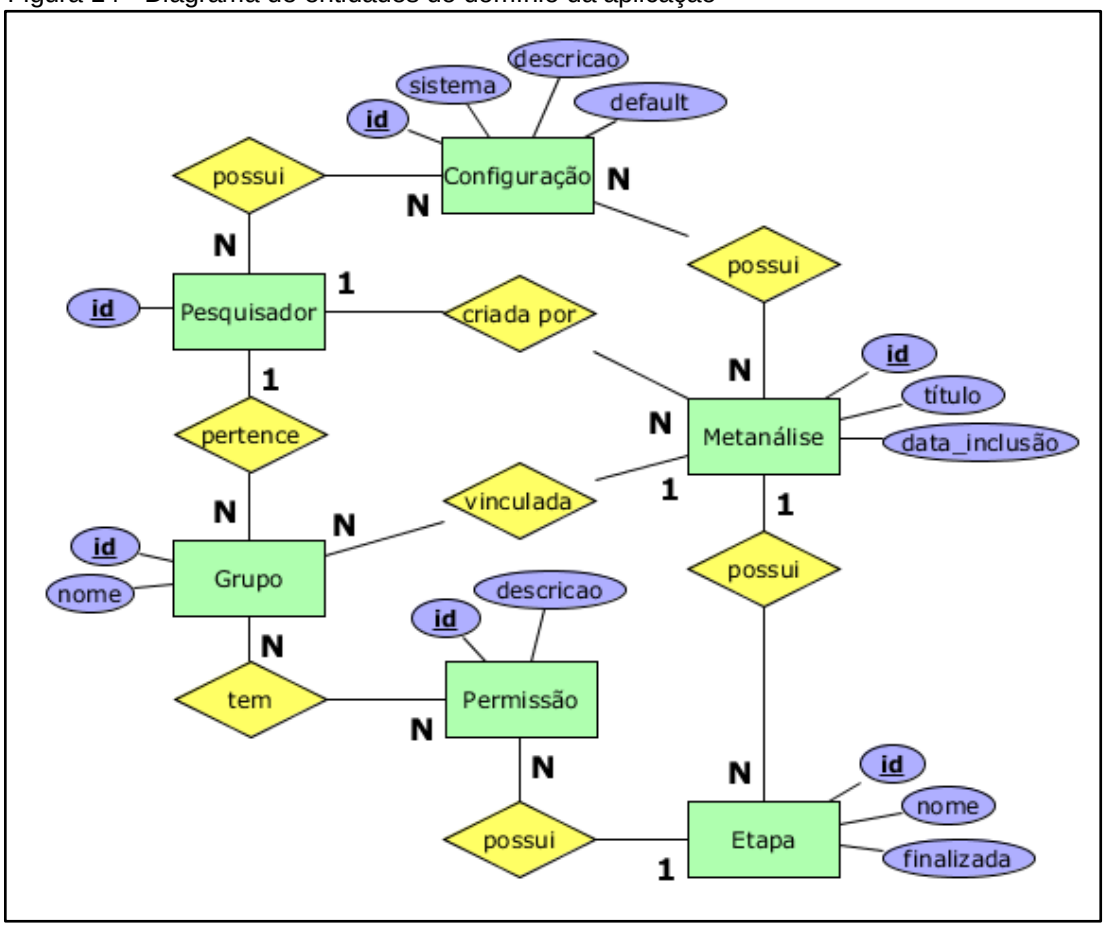


Fonte: Do autor.

Sobre o domínio da aplicação no que se refere a configurações, verificou-se que deve ser possível determinar os aspectos do ambiente como um todo, mas também de cada metanálise cadastrada no sistema.

Complementa-se a visão do domínio da aplicação com a demonstração da relação entre as entidades existentes por meio do diagrama de entidade-relacionamento visto na figura 14.

Figura 14 - Diagrama de entidades do domínio da aplicação



Fonte: Do autor.

Na figura 14 é apresentado o diagrama de entidades. Neste diagrama está incluída a visão do domínio da aplicação segundo a operação de um usuário “Pesquisador”. Entende-se que parte do diagrama exposto serve também para usuários do tipo “Administrador”, uma vez que um administrador é também um pesquisador. Porém, não é válido para usuários do tipo “Consultor”.

Desta forma, podemos entender o sistema, segundo o diagrama como:

a) a metanálise é constituída de um título e uma data de inclusão;
b) a metanálise terá sido criada por um pesquisador;
c) o pesquisador possuirá configurações próprias, tais como idioma de uso;
d) as configurações possuem um nome, um valor padrão e a determinação se são de sistemas ou específicas de metanálise;
e) as metanálises também possuem configurações, as quais regem sua própria características, tais como métodos de cálculo, intervalo de confiança, etc;

f) metanálises são vinculadas a um grupo de permissões;
g) pesquisadores fazem parte de grupos de permissões;
h) metanálises são divididas em etapas. Estas possuem um nome, como base de dados, qualidade dos estudos incluídos, etc. Também possuem a indicação de finalização, indicando se uma etapa foi concluída para a metanálise em que está vinculada;
i) metanálises também estão vinculadas a permissões;
j) os pesquisadores que tiverem incluído a metanálise ou que pertencerem a grupos de permissões que possuam permissão de editar etapas da metanálise incluída poderão alterá-la.

7.1.2 Avaliação da viabilidade

A avaliação da viabilidade se faz necessária como um dos princípios éticos da elaboração de software. Ela pretende verificar se é necessário o esforço para elaboração de uma nova plataforma, ou seja, pretende avaliar se atualmente já não existe algum software semelhante que possa ser indicado ao cliente. Isso se deve ao fato de que, a construção de um novo software demanda muito tempo de desenvolvimento, visto que ele inclui além da programação e codificação, também a captura dos requisitos, elaboração dos modelos, testes e a distribuição do software.

Como parte da avaliação da viabilidade, é preciso identificar os softwares presentes no mercado atualmente e que cumprem com as tarefas pretendidas ao usuário. Desta forma, foi relacionado as etapas envolvidas na metanálise diagnóstica e avaliado como cada software cumpre com as tarefas relacionadas a cada etapa.

Em entrevistas com as partes interessadas e com base na análise dos softwares já existentes no mercado foi possível definir algumas particularidades que demonstram a necessidade de uma nova ferramenta, por não se ter um software completo e que cumpra com todas as etapas da metanálise diagnóstica.

Os softwares identificados e avaliados foram:

a) Meta-DiSc 1.4;
b) StataMP 13 Trial;
c) Review Manager 5.3 (RevMan).

Isso se deve ao fato de que cada um dos softwares listados acima cumpre com parte da metanálise diagnóstica, e não com todas as etapas dela. Além disso, outras características primordiais deixam muitas vezes de fazer parte das soluções.

Em se tratando das entradas de estudos na base, é importante que os softwares possam garantir que todos os estudos incluídos façam parte do cálculo estatístico e isso deve ser transparente ao usuário final. Em decorrência do método de cálculo, os estudos que apresentam valores de verdadeiros e falsos positivos e negativos nulos podem implicar em uma restrição matemática, ou seja, não é possível pela fórmula de cálculo realizar uma divisão por um valor nulo, desta forma, os cientistas responsáveis pela idealização dos cálculos sugerem que nesses casos, para que o estudo não seja descartado da base, se considere os valores nulos como 0,5 pessoas, atendendo uma das regras elaboradas por eles, que baseiam-se nas seguintes condições:

a) se existir apenas uma célula com valor zero adiciona-se 0,5 em todas as células do estudo;
b) se existir duas ou mais células com valor zero exclui-se este estudo da análise.

Desta forma, somente o Meta-Disc possui essa opção de configuração de forma clara. Já o Stata exclui os estudos com valores negativos sem transparência ao pesquisador que está realizando a metanálise, e o RevMan realiza os cálculos utilizando o valor, porém não deixa claro.

Outra consideração a ser realizada com respeito aos cálculos é a possibilidade de configuração do intervalo de confiança. No Meta-Disc é possível configurar os intervalos de 90%, 95% e 99%, tal como no RevMan. Entretanto, o ideal seria fornecer a possibilidade de realizar os cálculos para qualquer intervalo de confiança configurado pelo usuário, como é o caso do Stata.

Quanto a qualidade dos estudos incluídos na metanálise diagnóstica faz-se necessário utilizar o RevMan. Somente ele fornece os mecanismos para avaliar os estudos quanto aos seus aspectos metodológicos, amostragem, apresentação dos resultados, conclusividade e outras características relevantes. Porém, a avaliação da qualidade é feita somente pelo pesquisador que a está desenvolvendo. Sendo assim, seria interessante acrescentar a possibilidade de outro avaliador

contribuir com a análise, gerando uma avaliação mais rigorosa por meio do consenso entre os pesquisadores que responderam as questões sobre a qualidade dos estudos.

As etapas de análises tabulares e gráficas são melhores desenvolvidas utilizando-se o Meta-Disc. Entretanto, a análise de vieses de publicação é feita com o uso do Stata, demonstrando a necessidade de uma outra ferramenta, pois as disponíveis no mercado cumprem com uma parte da metanálise diagnóstica, e não com as etapas como um todo.

A figura 15 demonstra a relação das atividades com os programas envolvidos, onde as atividades da categoria (a) dizem respeito aos cálculos estatísticos. As que compõem a categoria (b) estão relacionadas com a etapa de análise de vieses de publicação, e as da categoria (c) fazem parte da qualidade dos estudos incluídos.

Figura 15 - Estudo da viabilidade

	Atividades	M	S	R
a)	DOR	🙂	😐	☹
	Razão de verossimilhança	🙂	😐	☹
	Valor preditivo	🙂	😐	☹
	Sensibilidade	🙂	😐	☹
	Especificidade	🙂	😐	☹
	Tabelas prontas	🙂	😐	😐
	I^2	🙂	😐	🙂
	Qui-quadrado	🙂	😐	🙂
	Opções (0.5, efeito fixo ...)	🙂	☹	😐
	Gráfico de floresta	😐	🙂	😐
	Curva ROC	☹	🙂	☹
b)	Análise de vieses	☹	🙂	☹
	Regressão de Egger	☹	🙂	☹
	Teste de Begg's	☹	🙂	☹
	Gráfico de funil	☹	🙂	☹
c)	Qualidade dos estudos	☹	☹	🙂

LEGENDA
M = Meta-Disc
S = Stata
R = RevMan
🙂 = Bem avaliado
😐 = Mal avaliado
☹ = Não possui

Fonte: Do autor.

Além da avaliação das etapas da metanálise diagnóstica em cada software foi constatado alguns problemas existentes atualmente e que dificultam o

trabalho dos pesquisadores, como interface complexa (principalmente no RevMan), e necessidade de uso da linha de comando no Stata.

7.1.3 Levantamento de requisitos

O levantamento de requisitos teve inicio com o estudo dos modelos de engenharia de software para elicitação de requisitos com posterior escolha do modelo e definição das etapas e abordagens necessárias para aplicação do modelo escolhido. Foram estudados os modelos de entrevistas, inspeções, observações, workshop, prototipação e reuso de requisitos.

Foi definido, por questões relativas ao projeto, que alguns dos modelos mencionados acima seriam adotados, de forma parcial. Cada qual, completando a característica do outro.

O levantamento de requisitos foi concebido em sua maior parte através de entrevistas com as partes interessadas, buscando compreender quais as pessoas e tarefas estariam envolvidas com o processo de metanálise diagnóstica. Com base nas entrevistas, foi possível entender as necessidades e principais funcionalidades que o software deveria apresentar para que pudesse atuar de forma significativa na realização da metanálise diagnóstica. Foi possível entender as atuais dificuldades e as expectativas para uma nova plataforma de desenvolvimento de metanálises.

Com a utilização do reuso de requisitos, foi possível, conjuntamente com as partes interessadas elencar as principais funcionalidades já existentes nos softwares atualmente, e extrair delas os requisitos que deveriam fazer parte da nova plataforma, tais como as configurações disponíveis e os testes estatísticos.

Perante o uso da prototipação foi possível demonstrar os requisitos levantados de forma mais transparentes às partes interessadas, e com isso, ter uma ideia da organização dos requisitos e das telas do software.

Todos os requisitos elicitados foram documentados e estão vinculados a este trabalho, por meio de um documento de especificação de requisitos de software, o qual pode ser encontrado nos apêndices. Dentre os quais destacam-se como requisitos funcionais:

a) configurações do método de agrupamento;
b) configurações do modelo de regressão;
c) inclusão de colunas na base de dados;

d) atalhos, tais como desfazer alterações;
e) possibilidade de inserção de perguntas extras na ferramenta de análise de qualidade dos estudos incluídos;
f) possibilidade da inclusão de outro avaliador na etapa da qualidade dos estudos incluídos;
g) divisão em módulos;
h) configuração dos estudos com valores nulos;
i) configuração do intervalo de confiança.

Dentre os requisitos não-funcionais, destacam-se:

a) adaptação a várias resoluções;
b) relatórios para as análises tabulares devem poder ser gerados em pdf;
c) tempo de resposta do sistema as iterações do usuário deve ser menor que dez segundos;
d) interface amigável;
e) a aplicação deve possuir um módulo de ajuda.

7.1.4 Classificação dos requisitos

A etapa de classificação compreendeu em enquadrar os requisitos sobre seus aspectos, ou seja, de qual parte do sistema tratam (localização), se são requisitos funcionais ou não-funcionais (tipo), quais atores envolvidos e a prioridade no seu desenvolvimento.

Quanto ao tipo foram divididos em:

a) funcionais;
b) não-funcionais.

Quanto a localização foram divididos em módulos, sendo os módulos:

a) base de dados;
b) cálculos (análise tabular e gráfica);
c) exportação e importação;
d) ajuda;
e) autores;
f) configurações;
g) gerais do sistema;
h) permissões.

Quanto aos atores envolvidos foram divididos em:

a) administradores;
b) pesquisadores;
c) usuários (Consultores).

Quanto a prioridade de desenvolvimento foram divididos como:

a) essenciais, sendo o requisito primordial para liberação do sistema. É o requisito de prioridade máxima;
b) importantes, sendo que não impedem a liberação do sistema, mas a não implementação de alguns destes culmina na insatisfação do usuário;
c) desejáveis, sendo os requisitos que não tem impacto significativo na funcionalidade do sistema, passíveis de implementações futuras.

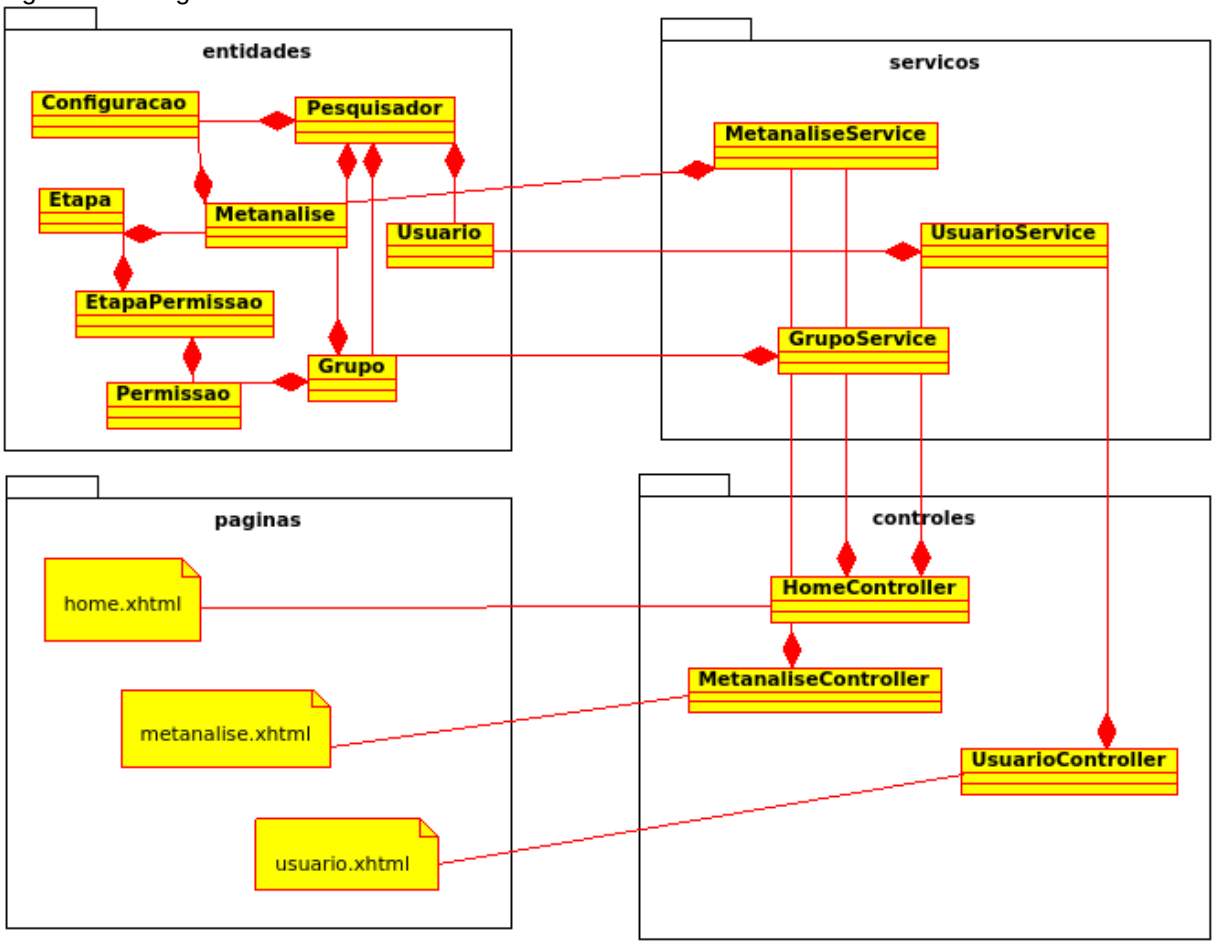
7.1.5 Arquitetura de software

A arquitetura foi definida com base no estudo da especificação do Java Enterprise Edition (Java EE) versão 6. Além da experiência do autor, os motivos que impulsionaram a escolha de tal arquitetura são:

a) controle de transações com o banco gerenciado pelo contêiner do servidor;
b) implementação com Java Server Faces (JSF) força o desenvolvedor a seguir um padrão de divisão de camadas mais rígido, compondo o software das camadas de visão, controle e modelo dos dados;
c) servidor de aplicações conta uma interface de aplicação para segurança das aplicações na WEB;
d) divisão entre camadas da aplicação e camadas do negócio;
e) possibilidade de trabalhar com clusterização, distribuição e acesso remoto;
f) serviço de mensagens, como serviços de emails;
g) injeção de dependência;
h) espaço de nomes para objetos.

O diagrama de pacotes da figura 16 demonstra a divisão da arquitetura.

Figura 16 - Diagrama de entidades



Fonte: Do autor.

As entidades fazem parte da camada de modelo, assim como os serviços. Elas representam os objetos da aplicação e tem forte relação com a persistência das informações no banco de dados. Já os serviços, são os mecanismos pelos quais se associa tarefas às entidades. Tais serviços serão responsáveis por disponibilizar métodos que façam a persistência das informações e o resgate delas do banco de dados. Estes são os Enterprise Java Beans (EJBs) da arquitetura, responsáveis por controlar as transações com o banco de dados.

Os controles fazem a intermediação entre as páginas e os serviços, decidindo quando os serviços serão chamados e para quais páginas o usuário deverá ser direcionado em caso de sucesso ou falha na tarefa que um serviço tenha desempenhado.

As páginas são a janela de visualização de conteúdo do usuário, por meio delas o usuário recebe as informações que deseja, e interage com a aplicação.

7.1.5.1 Ferramentas e frameworks de desenvolvimento

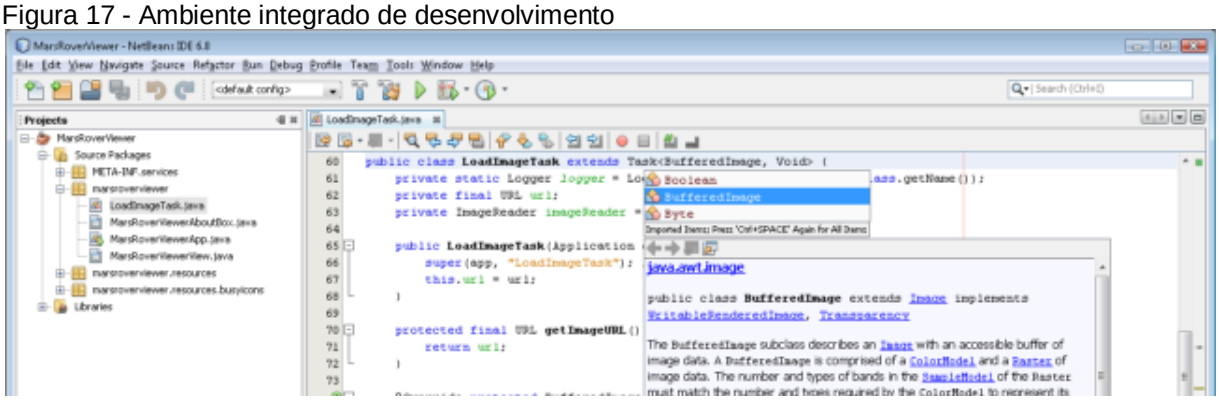
As ferramentas e frameworks de desenvolvimento auxiliam os programadores na implementação do software. Eles fornecem atalhos e facilitadores para o desenvolvimento do software, tais como um ambiente familiar e reuso de códigos já validados por outros desenvolvedores.

Como este trabalho de conclusão de curso não trata da implementação do software final, a escolha de tais ferramentas deve ser realizadas posteriormente, na fase de desenvolvimento. Entretanto, algumas sugestões estudadas durante a etapa da escolha da arquitetura são apresentadas abaixo. Tais ferramentas foram utilizadas para a implementação do protótipo de aplicação e podem servir de base para a condução do desenvolvimento do software na etapa de implementação.

As ferramentas sugeridas visam prover aos desenvolvedores maiores condições para que realizem o trabalho de implementação de forma a propiciar um desenvolvimento menos trabalhoso. Assim, como a utilização de códigos de terceiros, tais como componentes, tem por objetivo diminuir o escopo de implementação do projeto, e como conseqüência, o seu tempo de desenvolvimento.

O ambiente de desenvolvimento integrado, do inglês Integrated Development Enviroment (IDE) indicado foi o Net Beans 8.0. Este ambiente fornece rico suporte a tecnologia Java empregada por este projeto, além de providenciar *plugins* para estender as funcionalidades de tal ferramenta.

Figura 17 - Ambiente integrado de desenvolvimento



Fonte: Do autor.

Indica-se para a construção e implantação do software a ferramenta Apache Maven 2.3.2. Este programa possibilita compilar todos os módulos do software com facilidade, dispondo ainda de ferramentas auxiliares para aplicar testes

de unidades, integrar o software diretamente com o servidor de aplicação e gerenciar as dependências do projeto.

Como servidor de aplicação foi indicado o Wildfly 8.1 pela simplicidade de uso, ser totalmente configurável, dispor de uma interface de gerenciamento descomplicada, possuir um contêiner que dá suporte a Java Transaction API (JTA) e operar com múltiplas requisições HTTP performaticamente.

Outras questões relativas às ferramentas sugeridas e utilizadas na criação do protótipo encontram-se no documento de arquitetura de software, disponível nos apêndices.

7.1.6 Prototipação

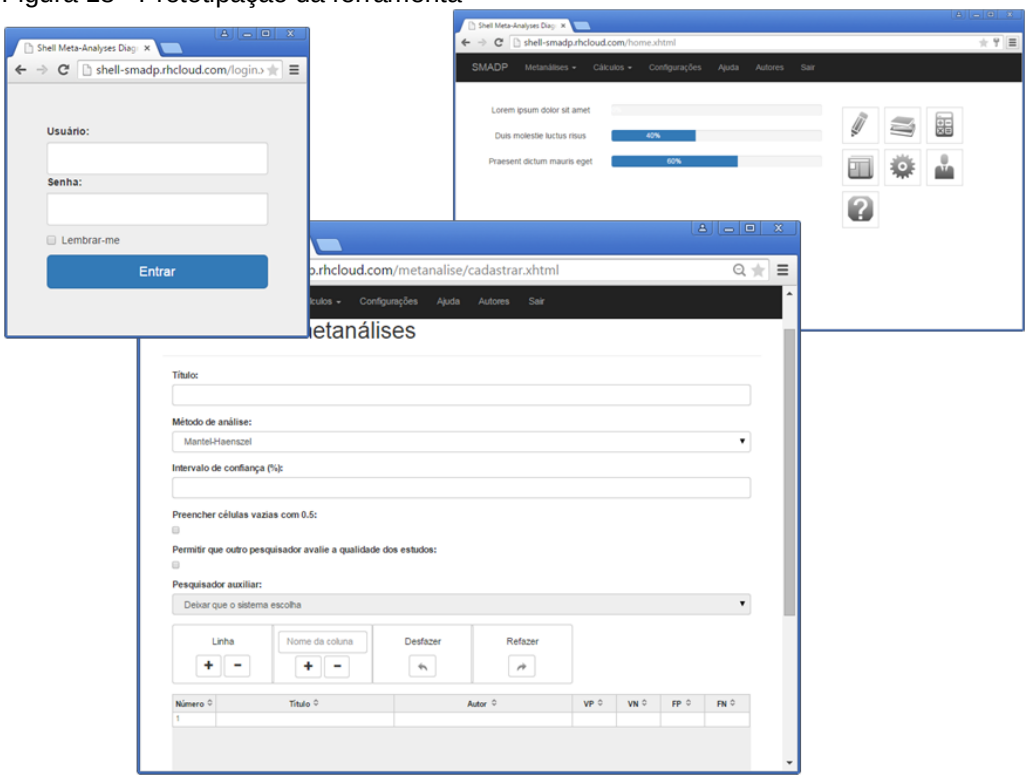
Esta fase do projeto teve como objetivo garantir a validade dos requisitos levantados, apresentando-os de forma visual as partes interessadas.

O protótipo foi concebido utilizando-se da arquitetura, *frameworks* e ferramentas estudadas na etapa de arquitetura de software, e pode ser utilizado como base para a implementação futura do software.

A apresentação do protótipo para as partes interessadas, sendo elas pesquisadores com experiência na realização de metanálises diagnósticas, foi feita com base na sua implantação em um servidor gratuito na WEB. O servidor utilizado foi o Open Shift. Cada um dos pesquisadores que compunha a parte interessada recebeu um usuário e senha tendo um perfil dentro do software.

A figura 18 apresenta a prototipação da interface, demonstrando a tela inicial e a tela de cadastro da metanálise.

Figura 18 - Prototipação da ferramenta



Fonte: Do autor.

Também foi disponibilizado um questionário contendo perguntas relativas aos requisitos, as quais questionavam-os acerca dos seguintes pontos:

a) perfis de usuários no sistema;
b) disposição dos módulos do sistema;
c) métodos de análise;
d) possibilidade da inclusão de outro pesquisador para auxiliar na avaliação da qualidade dos estudos incluídos;
e) internacionalização do sistema.

Além disso, foi pedido para que deixassem opiniões sobre outros requisitos.

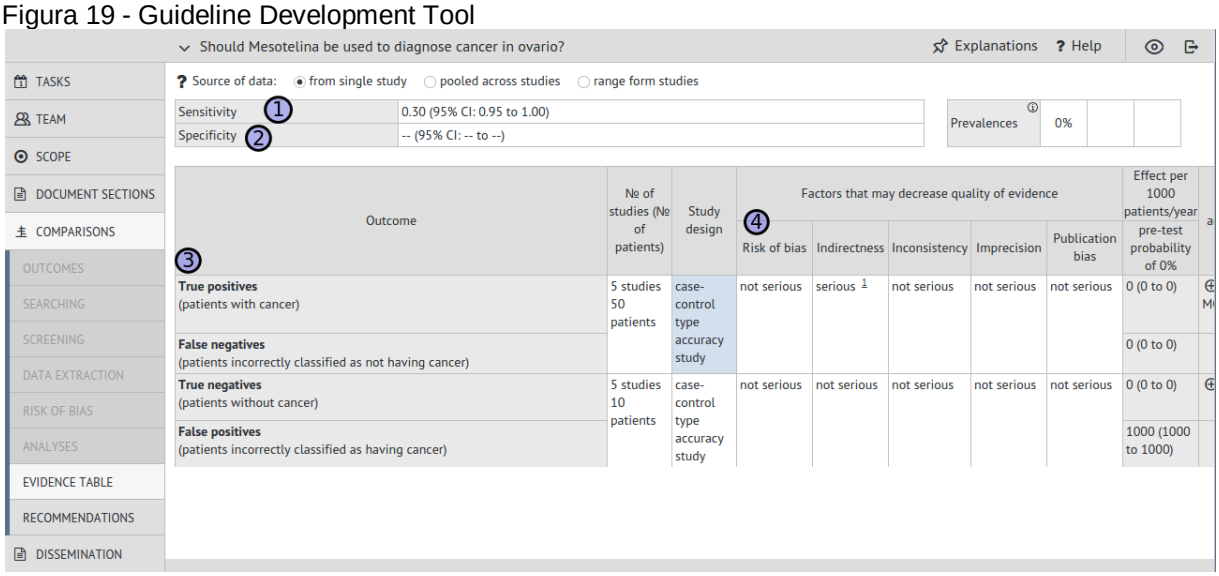
O questionário foi elaborado utilizando a ferramenta para construção de questionários da Google (Google Forms) e aplicado com pesquisadores que já possuem experiência na realização de metanálises diagnósticas. Com isso, foi possível validar os requisitos, dado que os pesquisadores foram de acordo com as propostas demonstradas. Entretanto, foi avaliado a possibilidade da não-inclusão do perfil “Consultor” de usuário. Ainda que sugerido não haver esse perfil, foi optado por

mantê-lo, pelo menos até o início do desenvolvimento da plataforma. Futuramente pode-se pensar em não incluir esse perfil, caso realmente não for necessário tê-lo.

Um dos pesquisadores sugeriu analisar uma outra ferrramenta: Guideline Development Tool (GDT). Essa ferramenta encontra-se disponível de forma gratuita na internet. Trata-se de uma plataforma para realização de metanálise, tanto terapêuticas como diagnósticas.

A figura 18 demonstra uma das telas relacionadas com a metanálise diagnóstica, que diz respeito à tabela de evidências, ou seja, a tabela de entrada dos estudos.

Figura 19 - Guideline Development Tool



Fonte: Do autor.

Foram identificados alguns pontos em comum com a ferramenta desse trabalho, as quais pode-se fazer as seguintes constatações e diferenciações:

a) permite a entrada da sensibilidade (1) e especificidade (2), porém não foi encontrada uma forma de calculá-las;
b) possibilita a entrada dos pacientes, separando-os por verdadeiros e falsos positivos e negativos (3);
c) considera o risco de viés, inconsistência, imprecisão e viés de publicação (3) para avaliar a qualidade dos estudos, porém não apresenta perguntas simples para ajudar aos pesquisadores responder as perguntas de referência;

Outra consideração sobre a plataforma GDT é a de que é uma ótima ferramenta para avaliação da qualidade dos estudos, assim como o RevMan, possibilitando também importar trabalhos já realizados no RevMan.

A possibilidade de escolha de como a metanálise será publicada também em um requisito a ser considerado. Pode ser interessante, na implementação da plataforma proposta por esse trabalho, considerar a possibilidade da escolha de como a metanálise será publicada, sendo assim, não haveria mais a necessidade de um perfil de usuário “Consultor”.

7.2 RESULTADOS OBTIDOS E DISCUSSÕES

Mediante os estudos bibliográficos e perante as etapas desenvolvidas, constatou-se que:

a) não há um único software capaz de realizar todas as etapas da metanálise diagnóstica;
b) os softwares disponíveis possuem limitações. Alguns pela complexidade de uso, outros pela condição de licença, tais como o Stata, que possui licença paga;
c) muitos pesquisadores podem estar envolvidos em uma mesma metanálise, por isso faz-se necessário que o software esteja disponível na WEB, dado que proporciona um ambiente colaborativo;
d) a engenharia de software contribuiu para o planejamento mais adequado do software, por meio dos estudos de técnicas para levantamento de requisitos, princípios de construção de software e análise de arquiteturas para WEB;
e) a análise da viabilidade para determinar a necessidade de uma nova plataforma contribuiu ainda para que fossem levantadas algumas das funcionalidades dos softwares existentes;
f) a prototipação permitiu validar junto as partes interessadas os requisitos do projeto, com o ganho de dispor uma base para a implementação futura da plataforma.

É interessante a comparação dos resultados obtidos por este trabalho em relação aos estudos efetuados por Freitas (2008) em seu trabalho de conclusão de

curso, onde o mesmo faz uma avaliação do software UTInfo 2.0 sobre os aspectos de requisitos do manual versão 3.0 de certificação para sistemas de registro eletrônico em saúde, pois os requisitos descritos neste manual condizem com muitas questões relativas à engenharia de software. A tabela 2 faz um comparativo entre o protótipo criado neste trabalho (SMADP) com o software analisado por Freitas.

Tabela 2 - Comparação SMADP x UTIInfo

Requisito	Sub-requisito	UTIInfo	SMADP
Segurança	Controle de versão	Não possui	Git Hub
	Segurança dos dados	Definida pela separação de papéis (Gestor de segurança, auditor, administrador e operadores)	Definida pela separação de papéis (administrador, pesquisador, consultor) e grupos de permissão criados pelo administrador e pesquisador
	Integridade dos dados	Garantida pelo banco de dados	Garantida pelo banco de dados
	Criptografia dos dados	SHA1 e Assinatura digital	SHA256 somente no login de usuários
Estrutura dos dados	Documentação do código fonte	Não possui	Java Doc
	Documentação da instalação	Não possui	Instalação não é necessário por parte do usuário final
	Documentação de configurações	Não possui	Configuração do ambiente WEB não é feita pelo usuário final
	Documentação de	Não possui	Não possui

	operação		
	Independência	Desenvolvido com Java, independente de sistema operacional	Plataforma WEB, independente de sistema operacional
	Recuperação dos dados	Não permite exportação de dados	Não avaliado
	Estrutura dos dados	Permite inclusão de comentários	Não avaliado
Funcionalidade	Alertas e lembretes	Não possui	Não possui
	Restrições e obrigatoriedades	Não possui	Possui

Fonte: Do autor.

O comparativo demonstrado relaciona os softwares e os descreve segundo os requisitos que compõem a certificação da Sociedade Brasileira de Informática em Saúde e do Conselho Federal de Medicina (SBIS/CFM) evidenciando que o software elaborado por este trabalho de conclusão de curso atende parte dos requisitos, sobretudo verifica-se que algumas outras questões, como por exemplo, relativas ao controle de versão e a estrutura de dados foi pensada durante o planejamento deste projeto que não havia sido pensada na idealização do software UTIInfo.

Alguns dos requisitos correlacionados não puderam ser avaliados, pois não fazem parte do escopo deste trabalho, como por exemplo a recuperação de dados, que deve ser prevista durante a operação do software.

8 CONCLUSÃO

A engenharia de software vem há muito tempo auxiliando todas as pessoas envolvidas com projetos de software, fornecendo subsídios para entender e formular software de maneira simples e clara. Com o avanço das tecnologias de desenvolvimento houve uma readequação da engenharia de software, tendo início a engenharia WEB.

Entender as ferramentas que a engenharia de software e engenharia WEB dispõe é o primeiro passo para construir softwares de qualidade. Outras práticas abordadas nesse trabalho de conclusão de curso mostram como empresas vem adequando seus processos de software para atingir esse objetivo, tais como aplicar os modelos de software brasileiro (MPS-BR) e os internacionais (CMMI). Entretanto, projetos que não tem caráter empresarial podem também ser concebidos visando à qualidade no desenvolvimento, desde que se compreenda o domínio da aplicação, para quem ele será aplicado, e se entenda os mecanismos que a engenharia de software oferece para o levantamento de requisitos.

O estudo da engenharia de requisitos possibilitou uma maior compreensão sobre a necessidade de se aprimorar e se apropriar de conhecimentos relativos à área da computação, como também relativos à área alvo da elaboração do software, visto que os clientes, como também os próprios engenheiros de software acabam muitas vezes por confundir requisitos. Sendo assim, verificou-se que a engenharia de requisitos dispõe de técnicas para elicitação de requisitos, tais como entrevistas, inspeções, reuso de requisitos e prototipação, que contribuem por auxiliar os engenheiros de software na obtenção de requisitos mais claros para a construção do software.

A compreensão do domínio foi uma das dificuldades enfrentadas no desenvolvimento deste trabalho, visto que trata-se de uma área, a primeira vista, distante da computação. Entretanto, a computação vem auxiliando outras áreas do conhecimento e fornecendo softwares que minimizam o trabalho de pesquisadores, sobretudo de pesquisadores da área biomédica.

Verificou-se a importância da metanálise diagnóstica para geração de novas evidências e sintetização de resultados de estudos independentes na tomada de decisão de profissionais da área biomédica.

O estudo permitiu a elaboração de uma nova plataforma de metanálises diagnósticas, por meio do entendimento das etapas e importância delas na condução de novas metanálises. Com a análise da viabilidade foi identificado que os softwares existentes no mercado não dão conta de desenvolver todas as etapas da metanálise diagnóstica, e por meio de um maior aprofundamento foi possível levantar os requisitos e modelar uma solução que atendesse de forma mais efetiva na condução dessas práticas.

Para tanto, a validação dessa plataforma proposta deu-se por meio da apresentação de um protótipo às partes envolvidas, e as devidas readequações foram feitas aos requisitos do software.

Além da contribuição científica deste trabalho para a computação, demonstrando a importância da engenharia de software e utilizando seus conceitos para elaboração de uma nova ferramenta, possibilitou-se também, por meio deste, a construção da base do projeto com a elaboração do protótipo, para que se possa dar continuidade a implementação da ferramenta. Com isso, sugere-se como trabalhos posteriores implementar a solução Shell Metanalysis Diagnostic Pearson (SMADP), tendo em vista que serão necessários implementar pelo menos os seguintes módulos:

a) DOR de efeito-fixo;
b) DOR de efeito-randômico;
c) sensibilidade, especificidade e acurácia;
d) razão de verossimilhança positiva e razão de verossimilhança negativa;
e) valor preditivo positivo e valor preditivo negativo;
f) regressão de Egger;
g) teste de Beggs;
h) metaregressão;
i) gráfico de floresta;
j) curva SROC;
k) gráfico de funil.

Sugere-se também avaliar a implementação da plataforma segundo a certificação SBSI/CFM que regulamenta o desenvolvimento de software para a área da saúde.

Contudo, pode-se afirmar que o estudo da engenharia de software, a aplicação de suas práticas e a elaboração de uma arquitetura que contemple o projeto são práticas que devem ser observadas e que podem determinar o sucesso de um software.

REFERÊNCIAS

ALEXANDER, Ian; BEUS-DUKIC, Ljerka. **Discovering Requirements:** How to specify products and services. Grã-Bretanha: John Wiley and Sons, 2009.

BASS, Len; CLEMENTS, Paul; KAZMAN, Rick.**Software architecture in practice**. 2 ed. Addison-Wesley: Boston, MA, 2003. Disponível em: <http://dl.acm.org/citation.cfm?id=773239>. Acesso em: 12 out. 2014.

BURNS, Ed; SCHALK, Chris; GRIFFIN, Neil.**Java Server Faces 2.0**: The Complete Reference. New York: Osborne/McGraw-Hill, 2010.752p. ISBN:9780071625098.

CASTRO A. A. Revisão sistemática e meta-análise. **Compacta**: temas de cardiologia. v.3, p. 5-9, 2001. Disponível em: <http://metodologia.org/wp-content/uploads/2010/08/meta1.PDF>. Acesso em: 20 out. 2014.

COOK, D. J.; GUYATT, G. H. et al. Clinical recommendations using levels of evidence for antithrombotic agent. **Chest,** v. 108, n. 4 Supplement, p. 227S. 1995. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/7555178?dopt=Abstract&holding=f1000,f1000m,isrctn>. Acesso em: 02 jun. 2015.

COURAGE, Catherine; BAXTER, Kathy. **Understanding Your Users:** A pratical guide to users requirements methods, tools and techniques. New York: Elsevier, 2005.

CUTLER, DM; MCCLELLAN M. **Is technological change in medicine worth it?** Health Affairs, 2001. n. 20 (5), p. 11-29.

DOSTÁLEK, Libor; KABELOVÁ, Alena.**Understanding TCP/IP**: A Clear and Comprehensive Guide to TCP/IP Protocols.Packt Publishing: Birmingham, 2006. 480 p. ISBN: 9781904811718.

FREITAS, Diego Biz. 2008. 132 f. **Utilização dos requisitos recomendados do manual versão 3.0 de certificação para sistemas de registro eletrônico em saúde da sociedade brasileira de informática em saúde e do conselho federal de medicina**. Graduação (Bacharel em Ciência da Computação) – UNESC, Criciúma, 2008.

GINIGE, A., MURUGESAN, S. **Web Engineering**: An Introduction. IEEE MultiMedia, vol. 8, n. 1, Jan.–Mar. 2001, p. 14-18, (2001a).

HANMER, Robert. **Pattern-Oriented Software Architecture For Dummies**. Chichester, UK: John Wiley & Sons, Ltd, 2012. 384p. ISBN-9781119963998.

HOSSENLOPP, Rosemary; HASS, Kathleen B. **Unearthing Business Requirements:** Elicitation tools and techniques. Vienna: Management Concepts, 2008.

IEEE. **IEEE Recommended Practice for Software Requirements Specifications**. ISBN 0-7381-0332-2. 1998. Disponível em: <http://www.math.uaa.alaska.edu/~afkjm/cs401/IEEE830.pdf>. Data de acesso: 13 set. 2014.

ISAKOWITZ, T., STOHR E., BALASUBRAMANIAN P. RMM: A Methodology for Strutured Hypermidia Design. **Communications of ACM**, v. 38, n. 8, ago.1995. Disponível em: <dl.acm.org/citation.cfm/id=208346>. Acesso em: 23 maio 2014.

JUNEAU, Josh. **Java EE 7 Recipes**: A Problem-Solution Approach. New York: Apress, 2013. 748p. ISBN: 9781430244257.

KEITH, Mike; SCHINCARIOL, Merrick.**Pro EJB 3**: Java Persistence API. New York: Apress, 2006. 480p. ISBN-978-1-59059-645-6.

KENETT, Ron S.; BAKER, Emanuel R. **Process Improvement and CMMI for Systems and Software**.Londres: CRC Impress, 2010. 436p.

LAPLANTE, Phillip A. **Requirements Engineering for Software and Systems**. 2.ed. London: CRC Press, 2014. 324p. ISBN-9781466560819.

LEEFLANG, Mariska M. G. et al. Cochrane diagnostic test accuracy reviews. **Systematic reviews**, v. 2, p. 82, 2013. Disponível em: <http://www.systematicreviewsjournal.com/content/2/1/82>. Acesso em: 10 nov. 2014.

LOVATTO, P. A. et al. Meta-análise em pesquisas científicas - enfoque em metodologias. **Revista Brasileira de Zootecnia**, v.36, p.285-294, 2007. Disponível em:<http://www.scielo.br/statjournal.php?lang=pt&issn=1516-3598>. Data de acesso: 10 nov. 2014.

MEDEIROS, Lídia Rosi de Freitas. **Laparoscopia versus Laparotomia nas tumorações ovarianas benignas**. Porto Alegre, 2003. 225 f. Dissertação (Mestrado) - Universidade Federal do Rio Grande do Sul, Curso de Pós-Graduação em Medicina: Epidemiologia.Disponível em: <http://hdl.handle.net/10183/3934 >. Acesso em: 10 nov. 2014.

MELSEN, W. G. The effects of clinical and statistical heterogeneity on the predictive values of results from meta-analyses. **Clinical microbiology and infection**. v. 20, f. 2, p. 123-129, 2014.

MILLER, Roxanne E. **The Quest for Software Requirements**. Greeley St.: Maven Mark Books, 2009. 346p. ISBN-978-1-59598-067-0.

MCCONNELL, Steve. **Professional Software Development**: Shorter Schedules, Better Projects, Superior Products, Enhanced Careers. Boston, MA: Addison-Wesley, 2004. 241p. ISBN: 0321193679.

NOGUEIRA, Mauro Odd. **Qualidade no Setor de Software Brasileiro:** uma avaliação das práticas das organizações. 2006. 324 f. Tese (Doutorado em Engenharia de Sistemas e Computação) – Universidade Federal do Rio de Janeiro, Rio de Janeiro.

PERRY, Dewayne E.; WOLF, Alexander F. **Foundations for the Study of Software Architecture**.1992. 52p. Disponível em: <http://users.ece.utexas.edu/~perry/work/papers/swa-sen.pdf>. Acesso em: 12 out. 2014.

PETITTI DB 2000. Metanalysis**, Decision Analysis, and Cost-Effectiveness Analysis**: Methods for Quantitative Synthesis in Medicine. Oxford University Press. New York. 306 pp.

PRESSMAN, Roger S. **Engenharia de software [recurso eletrônico]**: uma abordagem profissional. Tradução de Ariovaldo Griesi. 7.ed. Porto Alegre: AMGH, 2011. 750p. Editado também como livro impresso em 2011. ISBN-978-85-8055-044-3.

PROJECT MANAGEMENT INSTITUTE. **A Guide to the Project Management Body of Knowledge.** 5 ed. Pensilvania: Project Management Institute, 2013.

RODRIGUEZ, D.; HARRISON, R.; SATPATHY, M. **A Generic Model and Tool Support for Assessing and Improving Web Processes**. Proceedings of the Eighth IEEE Symposium on Software Metrics. IEEE, 2002.

ROSEMBERG, W.; DONALD, A. **Evidence based medicine**: an approach to clinical problem-solving. BMJ., v.310, p.1122-1126, 1995.

SACKETT, David Lawrence et al. **Medicina baseada em evidência.** Porto Alegre: Artmed, 2003. Disponível em: <http://www.scielo.br/pdf/rlae/v13n3/v13m3a17.pdf>. Acesso em: 30 maio 2015.

SCHILDT, Herbert. Java - **The Complete Reference**.8.ed. New York: Oracle Press, 2011. 1152p. ISBN-978-0-07-160630-1.
SHARP, Helen; ROGERS, Yvonne; PREECE, Jenny. **Interaction Design beyond Human-Computer Interaction.** 2 ed. Grã-Bretanha: John Wiley and Sons, 2007.

SILVA, Letícia Krauss. **Metodologias e diretrizes para a incorporação de tecnologias (Revisão do Rol) pela agência nacional de saúde suplementar**

(ANS). ANS, 2003. Disponível em: <http://www.ans.gov.br/portal/upload/biblioteca/TT_AS_Tema2Leticia%20Krauss.pdf >. Acesso em: 19 out. 2014.

SILVA, Ariela Dal Toé**. Modelagem de um processo para o gerenciamento e desenvolvimento de requisitos baseado no modelo brasileiro de qualidade de processo MPS.BR**. 2010. 81 f. Graduação (Ciência da Computação) - Universidade do Extremo Sul Catarinense, Criciúma.

SILVEIRA, Maria Clara dos Santos Pinto. **A reutilização de requisitos no desenvolvimento e adaptação de produtos de software**. 2006. Tese (Doutorado em Engenharia Electrotécnica e de Computadores) - Faculdade de Engenharia, Universidade do Porto, Porto. Disponível em: <http://repositorio-aberto.up.pt/handle/10216/12020>. Acesso em 17 nov. 2014.

SOMMERVILLE, I. **Software Engineering.** 7ed. Boston, MA: Addison-Wesley. 2005.

TAYLOR, Richard N.; MEDVIDOVIC, Nenad; DASHOFY, Eric M. **Software Architecture - Foundations, Theory, and Practice**. New Jersey: John Wiley & Sons, 2010. 712p. ISBN-13 978-0470-16774-8.

THAYER, R.H.; M. Dorfman. 1997. **Software Requirements Engineering**.2ed. IEEE Computer Society Press.

TURINE, Marcelo Augusto Santos; MASIERO, Paulo Cesar. **Especificação de requisitos:** uma introdução. 1996. 26 f. Dissertação (Mestrado) – Usp, São Paulo, 1996.

VOHRA, Deepak. **Java Server Faces 2.0**: EssentialGuide for Developers. Michigan: Cengage Course Technology, 2014. 432p. ISBN: 9781305276185.

WECHSLER, R. et al. **A informática no consultório médico**. Jornal de Pediatria, Rio de Janeiro, v. 79, supl. 1, p. S3-S12, 2003.

WETHERBEE, Jonathan et al. **Beginning EJB 3**: Java EE 7. 7.ed. New York:

Apress, 2013. 452p. ISBN-9781430246923.

WILLIANS, Nick. **Professional Java for Web Applications**. Indianopolis: Wrox Press, 2014. 936p. ISBN-9781118656464.

ZAMORA, Javier et al. Meta-DiSc: a software for meta-analysis of test accuracy data. *BMC Medical Research Methodology*, v. 6, f. 31, jul., 2006. Disponível em: <http://www.biomedcentral.com/1471-2288/6/31>. Acesso em: 17 nov. 2014.

APÊNDICE A – DOCUMENTO DE REQUISITOS

Especificação dos Requisitos do Software e Análise do Projeto Shell Metanalysis Diagnostic Pearson Web

Versão 1.0

Jadson Frassetto

HISTÓRICO DAS REVISÕES

Data	Versão	Descrição	Autor
05/06/2015	1.0	Revisão inicial	Jadson Frassetto

1 INTRODUÇÃO

1.1 OBJETIVOS DESTE DOCUMENTO

Este documento é um complemento ao trabalho de conclusão de curso intitulado como “Elaboração de uma plataforma para metanálise diagnóstica utilizando conceitos e análise de requisitos da engenharia de software”, e tem como objetivo apresentar os requisitos e demais funcionalidades para o projeto de um software WEB, cujo objetivo principal é possibilitar a realização de metanálises diagnósticas.

Público Alvo: Profissionais da área biomédica

1.2 ESCOPO DO PRODUTO

1.2.1 Nome do produto e de seus componentes principais

Shell Metanalysis Diagnostic Pearson:

- Cadastrar metanálises diagnósticas;
- Realizar cálculos estatísticos tabulares;aplicar filtros e refinar os estudos envolvidos na metanálise;
- Desenhar gráficos.

1.2.2 Descrição do produto

Shell Metanalysis Diagnostic Pearson é um projeto para realização de cálculos e estudos relacionados com metanálise diagnóstica, para auxiliar os pesquisadores da área biomédica a definir a relação entre estudos e determinar quão evidentes são os resultados dos mesmos, qual a real aplicabilidade e qual a melhor maneira de se caracterizar a efetividade de um diagnóstico.

1.2.3 Missão do produto

Ser um software completo, colaborativo e usual, para auxiliar os profissionais da área biomédica a realizar metanálises diagnósticas de maneira descomplicada.

1.3 VIABILIDADE DO PROJETO

No que se refere aos softwares existentes no mercado para resolver as questões relativas à metanálises diagnósticas, podemos elencar algumas diferenças entre eles. E confirmar que não há nenhum software completo, que realize todas as etapas para o desenvolvimento de uma metanálise por inteiro. O quadro abaixo demonstra as diferenças entre os softwares, elencando as atividades necessárias para realização da metanálise e os softwares que possuem tais atividades. Para tanto, cada atividade foi ainda assim classificada para demonstrar o quão o software trata bem a atividade. Sendo:

	Possui e foi bem avaliado
	Possui, mas não foi bem avaliado
	Não possui

Atividades	Meta-Disc	Stata	Rev Man
DOR			
Razão de verossimilhança			
Valor preditivo			
Sensibilidade			
Especificidade			

Tabelas prontas			
I^2			
Chi-square			
Opções (+0,5 em estudos, efeito fixo e randômico)			
Gráfico de floresta			
Gráfico (Curva ROC)			
Análise de vieses			
Regressão de Egger			
Testes de Begg’s			
Gráfico de Funil			
Qualidade dos estudos			

Além destas diferenças, foi listado abaixo algumas das opções que dificultam o uso de tais softwares, e algumas das características que fazem falta para realizar uma metanálise com mais refinamento.

Meta-Disc

- Não fornece opções para entrada de um intervalo de confiança qualquer;
- Não fornece intervalo de confiança para I^2.

Stata

- Suporte para metanálise somente através da linha de comando;
- Software não-gratuito;
- Necessário incluir 0,5 pessoas nos estudos que apresentam células vazias para que não haja descarte dos estudos.

Rev Man

- Interface não amigável.

1.4 TÉCNICAS UTILIZADAS PARA LEVANTAMENTO DE REQUISITOS

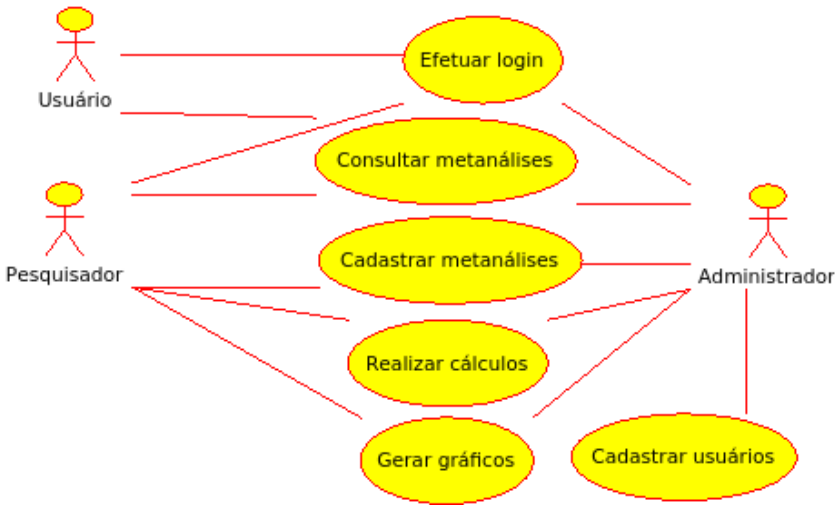
Entrevistas, reuso de requisitos e prototipação.

2 DESCRIÇÃO GERAL DO PRODUTO

2.1 PERSPECTIVA DO PRODUTO

A pretensão do software Shell Metanalysis Diagnostic Pearson, é ser um software colaborativo, onde vários pesquisadores poderão participar da elaboração de uma metanálise diagnóstica, contribuindo nas etapas de sua realização. Entretanto, também é de fundamental importância que as metanálises desenvolvidas no sistemas estejam disponíveis para acessos há outras pessoas registradas nele. De modo que o diagrama abaixo demonstra as interações mais básicas que ocorrerão no sistema.

2.1.1 Diagrama de Contexto



O sistema comportará três tipos de atores, sendo o administrador o ator mais

refinado, o qual possuirá a permissão para alterar ou consultar quaisquer partes do sistema. E o único responsável por incluir usuários e pesquisadores no sistema. Quanto ao pesquisador, este poderá realizar as tarefas relacionadas às metanálises. E o usuário, poderá somente consultar as metanálises que foram realizadas.

2.2 USUÁRIOS E SISTEMAS RELACIONADOS

Número de ordem	Ator	Definição
1	Pesquisador	Qualquer pessoa autorizada a usar o sistema para realizar metanálises.
2	Administrador	Pessoas responsáveis por administrar o sistema, cadastrar novos pesquisadores e usuários.
3	Usuário	Perfil exclusivo para acesso a metanálises finalizadas.

3 REQUISITOS ESPECÍFICOS

3.1 IDENTIFICAÇÃO DOS REQUISITOS

Os requisitos apresentados abaixo seguem um padrão de convenção:

a) Quanto ao tipo

Os requisitos podem ser classificados como funcionais (F) e não funcionais (NF).

b) Quanto ao módulo ou seção do sistema

1xx : Requisitos da base de dados

2xx : Requisitos do módulo de cálculos (análise tabular e gráfica)

3xx : Requisitos do módulo de exportação / importação

4xx : Requisitos do módulo de ajuda

5xx : Requisitos do módulo de autores

6xx : Requisitos do módulo de configurações

7xx : Requisitos gerais do sistema

8xx : Requisitos do módulo de permissões

c) Quanto aos atores envolvidos

Pelo menos um dos três autores especificados acima: Administrador (A), Pesquisador (P) ou Usuário (U).

d) Quanto à prioridade

Podem ser ditos como Essenciais (E), Importantes (I) ou Desejáveis (D). E servem como base para elaboração de um cronograma de implementação, dando prioridade aos requisitos essenciais e os outros consecutivamente.

[Tipo de requisito | Indicador de módulo | Número do requisito | Atores envolvidos | Prioridade]

Exemplo:

RF 105 APU E (Requisito funcional essencial da base de dados, número 5, com os autores Administrador, Pesquisador e Usuário envolvidos).

3.2 PRIORIDADES DOS REQUISITOS

Para estabelecer a prioridade dos requisitos, foram adotadas as denominações: essencial, importante e desejável. Abaixo segue a descrição de significado de cada uma dessas denominações:

Essencial	É o requisito primordial para liberação do sistema. Devem ser todos implementados.
Importante	É o requisito que não impede a liberação do sistema. Mas a não implementação de alguns deles culmina na insatisfação do usuário.
Desejável	É o requisito ao qual não tem impacto significativo na funcionalidade do sistema, sendo que, caso não haja tempo para implementação, podem ser implementados posteriormente em versões futuras.

3.3 DESCRIÇÃO DOS REQUISITOS

[RF 101 AP E] – Cadastro da metanálise

O sistema deve permitir o cadastro de uma nova metanálise, dando origem ao fluxo de desenvolvimento da mesma. A metanálise consiste de uma base de dados de estudos relacionados provenientes da revisão sistemática.

[RF 102 AP E] – Configurações do método de agrupamento

Em função de cada metanálise ser distinta, o sistema deve permitir que as configurações de métodos de agrupamento sejam realizados na etapa de cadastro da metanálise. Os métodos de agrupamento devem ser:

- Mantel-Haenszel;
- Der Simonian-Laird.

[RF 103 AP I] – Configurações do modelo de regressão

Em função de cada metanálise ser distinta, o sistema deve permitir que as configurações de modelo de regressão sejam individuais. Os modelos de regressão devem ser:

- Mínimo dos múltiplos quadrados (Inverso da variância);
- Mínimo dos múltiplos quadrados (Tamanho dos estudos).

[RF 104 AP I] – Permitir inclusão de colunas na base de dados

O sistema deve permitir incluir outras colunas na base de dados para avaliar a influência de outros fatores na metanálise.

[RF 105 AP D] – Importar base de arquivos

O sistema deve permitir importar arquivos dos formatos txt, csv e dat para permitir que uma metanálise anteriormente iniciada através de outros softwares, tais como o Meta-Disc possam ser continuadas nessa nova plataforma. Permitindo assim que outros trabalhos já desenvolvidos possam ser incluídos nessa nova plataforma.

[RF 106 AP D] – Ordenar estudos na base

Permitir ordenação dos estudos com base em qualquer coluna (autor, nome do estudo, ...), inclusive as que forem adicionadas pelo usuário, de forma ascendente e descendente.

[RF 107 AP I] – Permitir desfazer as alterações

Deve ser possível ao usuário desfazer facilmente as alterações realizadas na base. A tecla de atalho CTRL + Z deve realizar esta ação.

[RF 108 AP D] – Permitir impressão da base de dados

O sistema deve permitir a impressão da base de dados, ao qual deve incluir opções de impressão das configurações da metanálise, tais como método de agrupamento.

[RF 109 AP E] – Aplicação de filtros

O sistema deve disponibilizar um cadastro de filtros, a fim de permitir a aplicação de filtros nos estudos envolvidos na metanálise para avaliar a heterogeneidade entre

eles.

[RF 110 AP I] – Vincular metanálise a grupo de permissões

Permitir que além do pesquisador que incluiu a metanálise no sistema, outros usuários vinculados a grupos de permissões sejam capazes de editar a metanálise. Este requisito garante que metanálises possam ser desenvolvidas de forma colaborativa.

[RF 111 AP I] – Configurações do método DOR para computação da curva ROC

Em função de cada metanálise ser distinta, o sistema deve permitir que as configurações para escolha do método DOR para computação da curva ROC seja individual para cada metanálise, devendo ser uma das opções:

- Mantel-Haenszel;
- Der Simonian-Laird;
- Moses' constant of linear model.

[RF 112 AP E] – Permitir inserir outras perguntas além das disponíveis na avaliação da qualidade dos estudos

Disponibilizar o cadastro de novas perguntas para cada metanálise. As perguntas incluem a modalidade de: Perguntas simples, e Perguntas de referência.

As perguntas simples não são demonstradas no gráfico e estão relacionadas com a pergunta de referência. Elas servem de base para fazer com que os pesquisadores pensem um pouco mais na hora de responder a pergunta de referência.

As perguntas de referências são as perguntas chave e as quais são demonstradas no gráfico de comparação dos estudos.

[RF 113 AP I] – Possibilitar incluir outro avaliador para ajudar na etapa de avaliação da qualidade dos estudos

Permitir que outro avaliador possa auxiliar na etapa de avaliação da qualidade dos estudos, respondendo as mesmas questões sobre os estudos incluídos na metanálise. Isso dará início a uma análise de concordância com posteriores tramitações de opniões dos avaliadores até que se chegue em um consenso sobre os estudos que foram incluídos.

[RF 114 AP I] – Possibilitar desmarcar outro avaliador para ajudar na etapa de avaliação da qualidade dos estudos

O objetivo é desvincular o segundo avaliador em caso de possíveis atrasos na avaliação e impedimento da conclusão da metanálise.

[RF 201 AP E] – Subdivir o módulo de cálculos em análise e vieses

O sistema deve fornecer uma separação do módulo de cálculos, afim de avaliar

separadamente os estudos.

[RF 202 AP E] – Submódulo Análise

Deve conter as análises de:

- Sensibilidade e especificidade;
- Razão de verossimilhança;
- DOR.

Deve conter também os gráficos de:

- Sensibilidade;
- Especificidade;
- Positive LR;
- Negative LR;
- DOR;
- Curva ROC.

[RF 203 AP E] – Submódulo Vieses

Deve conter as análises de:

- Regressão de Egger;
- Teste de Beggs.

Deve conter também os gráficos de:

- Funil;
- Egger.

[RF 204 AP E] – Relatório de cálculos

Os cálculos devem ser demonstrados em forma de relatórios.

[RF 205 AP I] – Acesso rápido a filtros

Facilitar a aplicação de filtros sem que haja necessidade de voltar a base de dados.

[RF 206 AP I] – Permitir avaliação da qualidade dos estudos pela análise de mais de um avaliador

Calcular o índice Kappa, para avaliar a concordância das respostas das perguntas do teste de qualidade dos estudos.

[RF 207 AP I] – Trâmites de atividades da qualidade dos estudos

Permitir que a avaliação da qualidade dos estudos por mais de um avaliador, seja dada em trâmites, assim que cada avaliador fizer sua avaliação. Até que se tenha uma concordância final sobre a qualidade dos estudos que foram incluídos na metanálise.

[RF 208 AP E] – Gráficos da qualidade dos estudos

O módulo de gráfico para a análise da qualidade dos estudos deve compilar as informações das respostas dadas as perguntas cadastradas no módulo de base. Deve conter o gráfico geral e o gráfico por estudo.

[RF 501 APU I] – Histórico de autores

O sistema deve possuir um histórico de autores contendo o nome de todas as pessoas envolvidas na elaboração do projeto Shell Metanalysis Diagnostic Pearson, como também as informações dos anos em que participaram e da implementação que estas desenvolveram.

[RF 601 AP E] – Células vazias

O sistema deve dispor uma opção de configuração para determinar qual o comportamento deve ser tomado quando estudos com células vazias são incluídos em uma metanálise. Dentre as opções de configuração deve ser prevista a adição de 0,5 pessoas aos estudos que apresentarem células vazias, ou o descarte dos estudos nos cálculos realizados. O padrão deve ser incluir 0,5 pessoas nos estudos que apresentarem células vazias.

[RF 602 AP E] – Intervalo de confiança

O sistema deve dispor uma opção de configuração para determinar qual o intervalo de confiança adotado para o sistema na realização dos cálculos da metanálise. As opções de intervalo devem ser: 90%, 95% e 99%, tendo como padrão 95%.

[RF 603 AP D] – Intervalo de confiança

É desejável que seja possível configurar outros intervalos de confiança, tais como 92%, 98%.

[RF 604 AP I] – Padrão de simetria da curva ROC

O sistema deve dispor uma opção de configuração para determinar qual o modelo de simetria da curva ROC. Sendo as opções:

- Simétrico;
- Assimétrico.

Deve ter como opção padrão o valor Simétrico.

[RF 605 AP I] – Configurações por usuário

As opções do módulo de configurações devem ser gerais, e não distintas para cada metanálise. Porém, as configurações definidas devem ser armazenadas para cada usuário. Desta forma, cada usuário do sistema vai dispor as configurações da forma a qual melhor se adapta.

[RF 606 APU E] – Configuração dos ícones da qualidade de estudos

Deve ser possível configurar os ícones que serão gerados na forma gráfica da etapa de avaliação da qualidade.

[RF 607 APU E] – Configuração das cores dos ícones da qualidade de estudos

Deve ser possível configurar a cor que deverá ser gerado os gráficos dos estudos incluídos na etapa de avaliação da qualidade. O objetivo é possibilitar gerar gráficos coloridos para apresentações, e em escala de cinza para impressões em artigos e revistas.

[RF 701 APU I] – Internacionalização do sistema

O sistema deverá ser internacionalizado e dispor inicialmente dos seguintes idiomas:

- Espanhol;
- Inglês;
- Português.

[RF 801 AP I] – Dashboard na tela inicial

A página inicial do sistema é a primeira página visualizada pelo usuário após a operação de entrada no sistema (login). Sendo que será uma das páginas de acesso mais constantes. Dessa forma, devem ser dispostas dashboards para melhorar a experiência do usuário com o sistema. Uma das dashboards deve apresentar o andamento das metanálises ainda não finalizadas, com base nas etapas que já foram concluídas. As etapas a serem consideradas são:

- Base de dados;
- Qualidade metodológica dos estudos incluídos;
- Análise tabular;
- Análise gráfica;
- Análise de viés de publicação.

[RF 802 AP I] – Atalhos na tela inicial

Por ser uma das páginas mais acessadas no sistema, o acesso aos outros módulos deve ser facilitado pelo uso de atalhos, ou seja, ícones contendo links que levem o usuário diretamente para um dos módulos do sistema.

[RF 803 AP E] – Histórico de atividades

O sistema deve manter um histórico de atividades relacionadas com a metanálise. E

seus respectivos pesquisadores, os quais estão diretamente envolvidos na metanálise, devem receber as notificações de atividades relacionadas a elas.

[RNF 804 APU I] – Resolução da tela

O sistema deve rodar na resolução mínima de 1024 x 768.

[RNF 805 APU D] – Adaptação

O sistema deve ser responsivo e se adaptar a vários tipos de dispositivos, tais como computadores de mesa, tablets, celulares e outros dispositivos móveis.

[RNF 806 APU I] – Relatórios

Os relatórios gerados pela análise tabular dos dados devem ser possíveis de serem impressos e também exportados como pdf.

[RNF 807 APU D] – Tempo de resposta

Cada interação do usuário com o sistema não deve demorar mais do que dez segundos.

[RNF 808 APU D] – Interface amigável

O sistema deve prover uma interface simples, de fácil recordação e que maximize a experiência do usuário.

[RNF 809 APU D] – Ajuda

Todos os módulos devem possuir um atalho fácil para o módulo de ajuda.

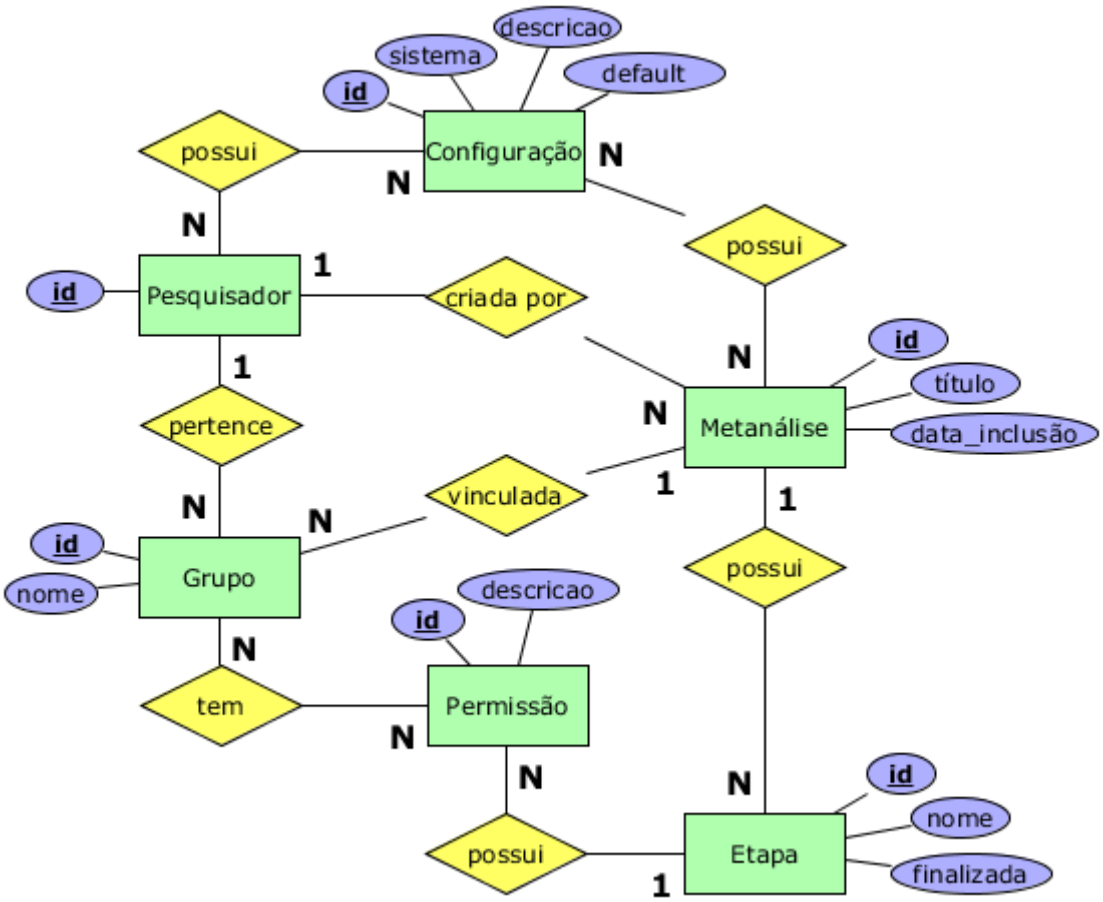
[RNF 810 APU I] – Atalhos

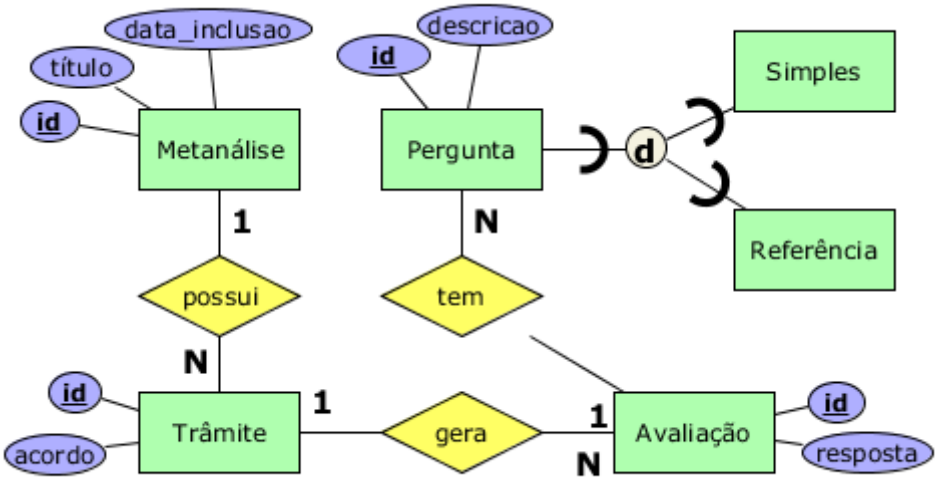
As operações realizadas no sistema devem ser também executadas através de atalhos do teclado.

[RNF 811 APU I] – Auditoria

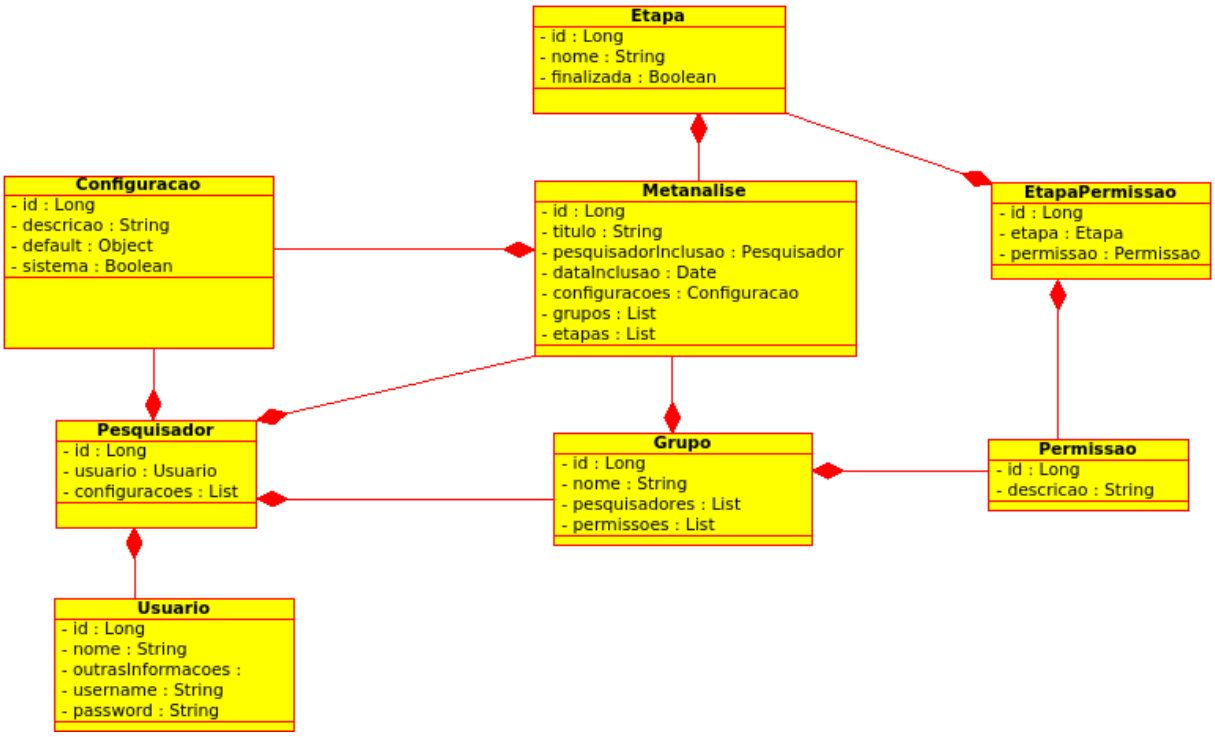
Todas as operações realizadas no sistema devem ser gravadas para posteriores consultas de auditoria. Os registros devem conter a operação realizada, os valores modificados e o usuário que realizou a operação.

4 DIAGRAMAS DE ENTIDADE-RELACIONAMENTO

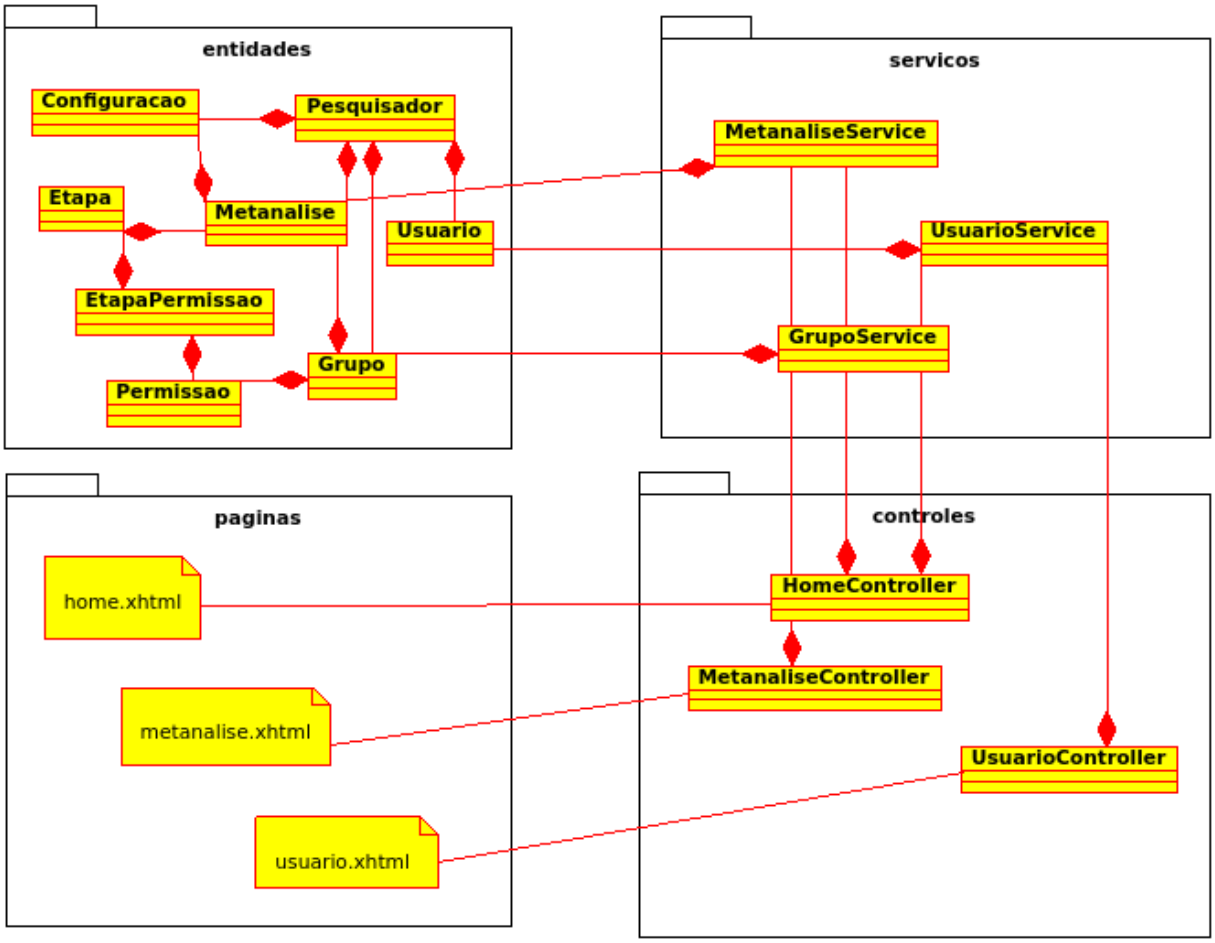




5 DIAGRAMA DE CLASSES



6 DIAGRAMA DE PACOTES



APÊNDICE B – DOCUMENTO DE ARQUITETURA DE SOFTWARE

PROJETO SMADP

DOCUMENTO DE ARQUITETURA DE SOFTWARE DO PROJETO SHELL METANALYSIS DIAGNOSTIC PEARSON

Jadson Frassetto

Criciúma, SC – 2015

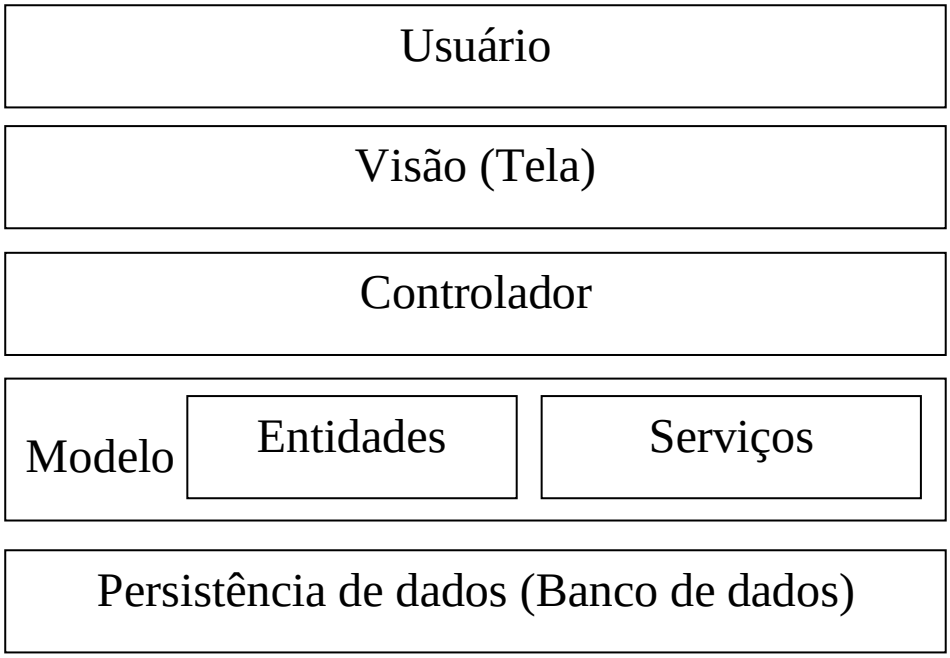
1 HISTÓRICO DE REVISÕES

Data	Versão	Descrição	Autor
03/05/2015	1.0	Versão inicial do documento	Jadson Frassetto

2 INTRODUÇÃO

O presente documento visa esclarecer a arquitetura do projeto Shell Metanalysis Diagnostic Pearson e as ferramentas idealizadas para implementação do projeto.

3 VISÃO GERAL



A arquitetura compreende as camadas de Modelo, Visão e Controlador.

As entidades fazem parte da camada de modelo, assim como os serviços. Elas representam os objetos da aplicação e tem forte relação com a persistência das informações no banco de dados. Já os serviços, são os mecanismos pelos quais se associa tarefas às entidades. Tais serviços serão responsáveis por disponibilizar métodos que façam a persistência das informações e o resgate delas do banco de dados. Estes são os Enterprise Java Beans (EJBs) da arquitetura, responsáveis por controlar as transações com o banco de dados.

Os controles fazem a intermediação entre as páginas e os serviços, decidindo quando os serviços serão chamados e para quais páginas o usuário deverá ser direcionado em caso de sucesso ou falha na tarefa que um serviço tenha desempenhado.

As páginas são a janela de visualização de conteúdo do usuário, por meio delas o

usuário recebe as informações que deseja, e interage com a aplicação.

4 REQUISITOS NÃO-FUNCIONAIS

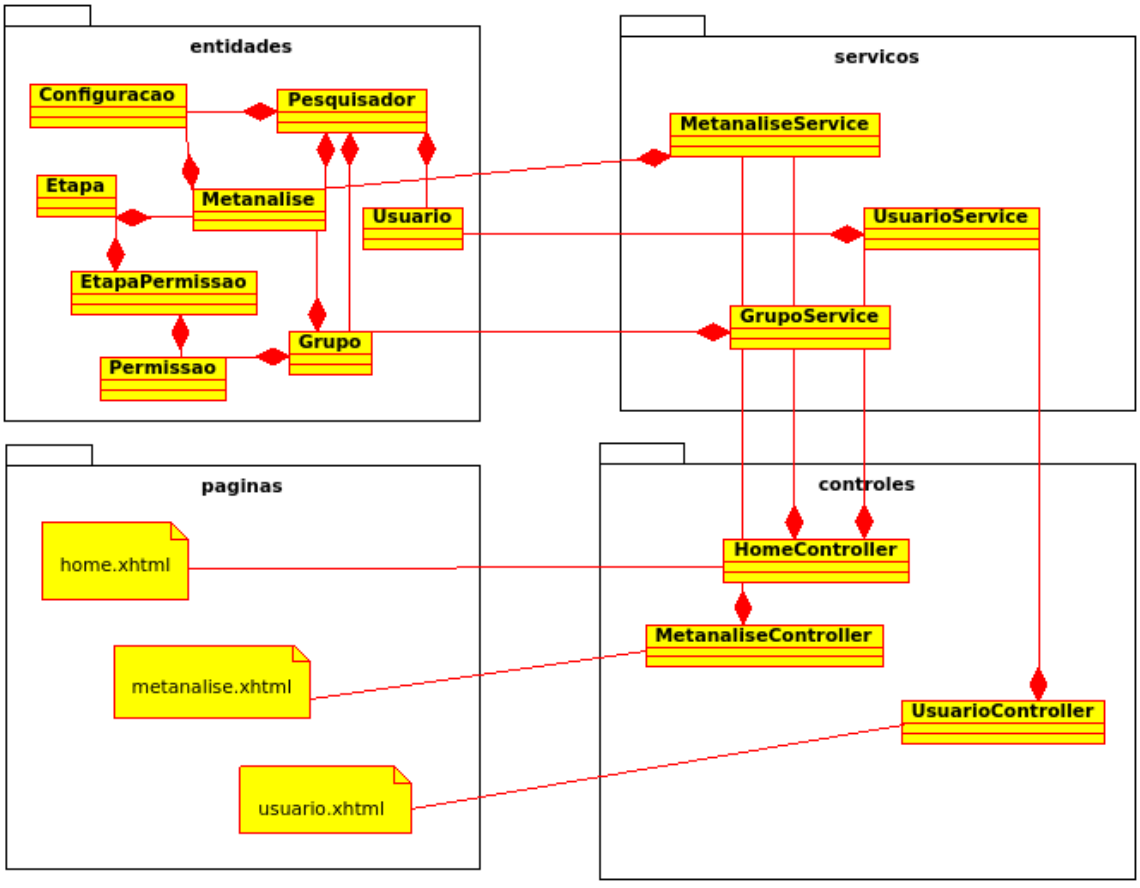
Adaptabilidade	1. O sistema deve rodar na resolução mínima de 1024 x 768. 2. O sistema deve ser responsivo e se adaptar a vários tipos de dispositivos, tais como computadores de mesa, tablets, celulares e outros dispositivos móveis. 3. O sistema deve prover uma interface simples, de fácil recordação e que maximize a experiência do usuário.
Desempenho	1. Cada interação do usuário com o sistema não deve demorar mais do que dez segundos.
Produtividade	1. Todos os módulos devem possuir um atalho fácil para o módulo de ajuda. 2. As operações realizadas no sistema devem ser também executadas através de atalhos do teclado.
Segurança	1. Todas as operações realizadas no sistema devem ser gravadas para posteriores consultas de auditoria. Os registros devem conter a operação realizada, os valores modificados e o usuário que realizou a operação.

5 MECANISMOS ARQUITETURAIS

MECANISMO DE ANÁLISE	MECANISMO DE DESIGN	MECANISMO DE IMPLEMENTAÇÃO
Persistência	Banco de dados relacional	MySQL Server
Log	Recursos de log do componente de persistência	Log4j
Servidor de aplicação	Servidor HTTP	Wildfly Server 8.2

Comunicação com banco	Serviço de persistência	Hibernate ORM
Front-End	Apresentação com usuário	XHTML (Java Server Faces), javascript
Autenticação e Autorização	Segurança da aplicação e controle de acesso	Apache Shiro
Tratamento de requisições	Servlet Java	Faces Servlet (Java Server Faces)
Build	Construção da aplicação	Maven
Deploy	Implantação da aplicação pela IDE	Maven-wildfly-plugin
Componentes Ricos	Suíte de componentes para WEB	PrimeFaces 5.2

6 COMPONENTES



Entidades: São as classes Java que representam as entidades do banco de dados. Cada uma das entidades mapeia seus atributos para as colunas do banco de dados. Na aplicação elas são administradas através de um modelo que utiliza a orientação de objetos, já no banco elas são persistidas em tabelas de forma relacional.

Serviços: Os serviços comandam as atividades com as entidades. Criam novos registros, consultam os registros na base persistente e realizam operações com as entidades da aplicação.

Controles: Os controles decidem quais serviços são necessários para atender as requisições do usuário, estruturam a lógica do software e redirecionam o usuário para a visão correta.

Visão: São as páginas do sistema. Nela o usuário pode ver o conteúdo obtido das demais camadas do software e interagir com eles, através de botões ou links que são tratados pelo controlador da página.

7 AMBIENTE DE DESENVOLVIMENTO

O ambiente de desenvolvimento é o proposto abaixo:

Atividade	**Ferramenta**	**Versão**
Edição de texto	NetBeans IDE	8.0
Compilação e implantação	Apache Maven	3.3
Repositório de códigos	Github	-

APÊNDICE C – QUESTIONÁRIO APLICADO

1) Identificamos três perfis de usuários do sistema:

a. Administradores

b. Pesquisadores

c. Consultores

Os administradores serão responsáveis por incluir pesquisadores, além de possuir todos os acessos liberados ao sistema;

Os pesquisadores poderão incluir metanálises, desenvolvê-las e participar de outras metanálises que tiverem permissão;

Os consultores serão os usuários com o acesso ao sistema somente para fazer download das metanálises finalizadas.

Quanto ao perfil dos usuários:

a) Concordo

b) Consultores não são necessários

c) Pesquisadores não são necessários

d) Administradores não são necessários

e) Gostaria de incluir um outro grupo: _____

2) Quanto à disposição dos módulos (Metanálises, Cálculos, Configurações):

a) Adequada

b) Inadequada, sugestão:

_____
_____
_____
_____
_____

3) Sobre os métodos de análise (agrupamento) disponíveis no cadastro de metanálises:

a) Ok, deve haver o método “Mantel-Haenszel” e “Der Simonian-Laird” como exposto

b) Está faltando métodos. Quais:

_____
_____

4) Sobre a possibilidade de sugerir que outro pesquisador cadastrado no sistema auxilie na avaliação dos estudos incluídos em uma metanálise para se ter uma avaliação mais rigorosa e adequada:

a) A ideia é válida

b) Não é necessário tal funcionalidade, cada pesquisador é o único responsável por sua metanálise, sempre.

5) Quanto a possibilidade do sistema ser internacionalizado:

a) Ótimo

b) Não é necessário, somente em português está bom

c) Gostaria de incluir mais idiomas:

_____
_____

Gostaríamos de outras opiniões, ou ideais para novos requisitos. Talvez algum requisito não foi levantado por esse estudo. Sugestões:

_____
_____
_____
_____
_____
_____
_____
_____
_____

APÊNDICE D – ARTIGO

Elaboração de uma Plataforma para Metanálise Diagnóstica Utilizando Conceitos e Análise de Requisitos da Engenharia de Software

Jadson Frassetto[1]

[1]Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC)
Criúma – SC – Brasil

jadsonfrassetto@gmail.com

Abstract. *With the large number of studies published in scientific circles it became more difficult to make a qualified review of the literature. Biomedical researches have used softwares to summarize the results of these studies by meta-analysis. However, such software don't comply with all steps her. This work enabled the development of a new platform by studies and software engineering enforces, measure the viability, eliciting requirements, modeling, documentation and prototyping. Is is found to possible to combine theory with practice and provide to the market tools that aim to a greater commitment concerning quality software, particularly providing software for the most diverse areas of knowledge, helping them to prosecute their role in society in an organized, agile and simplified manner.*

Resumo. *Com a crescente quantidade de trabalhos publicados no meio científico tornou-se difícil fazer uma revisão qualificada da literatura. Pesquisadores da área biomédica têm utilizado softwares para sintetizar os resultados destes estudos, por meio da metanálise. Entretanto, tais softwares não cumprem com todas as etapas dela. Este trabalho possibilitou a elaboração de uma nova plataforma, por meio de estudos e aplicação da engenharia de software, avaliação de viabilidade, elicitação de requisitos, modelagem, documentação e prototipação. Constatou-se ser possível conciliar teoria com prática e fornecer ao mercado ferramentas que visem maior compromisso com a qualidade de software, sobretudo, fornecendo softwares para outras áreas do conhecimento, auxiliando-as a exercerem seu papel na sociedade de forma organizada, ágil e simplificada.*

1. Introdução

Nas últimas décadas a produção científica aumentou consideravelmente, trazendo consigo maior dificuldade na análise qualificada da literatura (LOVATTO, 2007). A constatação de que evidências geradas por pesquisadores em todo o mundo não chegavam de forma adequada aos médicos contribuiu para o desenvolvimento das Práticas Baseadas em Evidência (PBE) (SACKETT, 2003). Com isso, a maior problemática enfrentada por pesquisadores da área biomédica é sintetizar os resultados de vários estudos independentes para obter um resultado com maior evidência que possa levar a um diagnóstico mais correto e conclusivo.

Ao longo dos anos vários pesquisadores estiveram desenvolvendo métodos para combinar os resultados de estudos independentes, e com o apoio de políticas governamentais houve a inclusão da revisão sistemática, metanálise, análise de decisão e de custo-efetividade na avaliação tecnológica em saúde (SILVA, 2003).

Com o uso de softwares, os pesquisadores puderam desenvolver as práticas relativas à revisão sistemática e a metanálise. Porém, eles possuem certa dificuldade no processo de análise dos dados coletados, uma vez que as ferramentas disponíveis cumprem apenas parcialmente com as análises necessárias.

O objetivo deste estudo consiste em aplicar a engenharia de software para estruturar uma plataforma de metanálise diagnóstica que auxilie os profissionais da área biomédica no desenvolvimento de metanálises diagnósticas, visto que, é responsabilidade da engenharia de software planejar o desenvolvimento de softwares de forma eficiente, produtiva e segura (PRESSMAN, 2011), pois um software que foi mal especificado poderá desapontar o usuário e culminar em maiores problemas para a equipe de desenvolvimento que terá que readequá-lo (TURINE, 1996).

2. Engenharia de Software

A obtenção de software, de forma responsável, visando à qualidade em todas as etapas do desenvolvimento é uma das responsabilidades da engenharia de software (PRESSMAN, 2011).

Softwares são instruções que quando executadas fornecem os dados necessários aos seus usuários (PRESSMAN, 2011). Entretanto, entender quais são esses dados e como devem ser trabalhos para atingir o objetivo é uma das questões que devem ser analisadas pelos engenheiros de software.

Uma das medidas que tem auxiliado os engenheiros de software na obtenção desses dados e contribuído para a construção de software visando à qualidade em todo seu desenvolvimento são os modelos de software.

Os modelos de software têm por objetivo auxiliar as empresas em todas as fases de desenvolvimento, desde o levantamento de requisitos, passando pelos modelos, até a codificação, testes e distribuição de software, agregando maturidade nos seus processos e contribuindo para um software mais bem planejado e de maior qualidade (BAKER, 2010; KENETT, 2010).

Em se tratando de projetos não-empresariais, como é o caso deste trabalho, a engenharia de software dispõe de princípios de projetos que norteiam o desenvolvimento de software em todas as etapas da sua construção, visto que o conhecimento necessário para construção de software não está somente no entendimento de uma linguagem de programação, mas em todo um conjunto de técnicas e etapas de desenvolvimento, denominados de “princípios da engenharia de software” (MCCONNELL, 2004).

A engenharia de software pode ser compreendida pelo estudo de princípios (PRESSMAN, 2011):

a) de processos, norteando o desenvolvimento para obtenção de softwares de forma mais ágil, considerando economizar ações de trabalho, na medida em que se propõe manter o código mais limpo e conciso;

b) práticos, primando pela divisão de trabalho, ao passo em que busca um maior grau de abstração na sua implementação, tornando o código mais simples em meio ao ambiente complexo, permitindo um desenvolvimento mais coeso e que auxilie na manutenibilidade do software;

c) metodológicos, orientando os engenheiros de software na obtenção de conhecimento a cerca do problema, mostrando-lhes como captar as informações necessárias, anotar e registrar informações e negociar os requisitos de software;

d) de planejamento, tentando direcionar a equipe de desenvolvimento, sem que se tomem tempos extras para um planejamento muito detalhado que não represente a atividade fim, ou sem que se pule esta etapa definindo um plano que não contribua como guia para o sucesso do software;

e) de modelagem, norteando o desenvolvimento de protótipos e a representação do produto final, por meio de modelos que apresentam os aspectos dos clientes e a arquitetura de software;

f) de construção, orientando o programador na codificação do software e na compreensão do problema;

g) de disponibilização, regendo a conduta frente à entrega do software ao cliente, contemplando as expectativas dos clientes e fornecendo o software sem erros, com suporte e os manuais de atualização.

Tendo-se observado esses princípios, bem como os modelos de software e fazendo-se as ponderações necessárias para cada projeto, pode-se pensar em produzir software visando à qualidade no resultado.

3. Engenharia de requisitos

Analogamente, pode-se comparar a elicitação de um software com a realização de uma festa. Quando um cliente procura um profissional para produzir um evento este o indaga sobre as características pretendidas pelo cliente para a realização da festa, e o comtempla com opções de pratos, decorações, etc. O cliente, por sua vez, fornecerá os requisitos necessários para a realização do evento. De certa forma, no desenvolvimento de um software é preciso resolver um problema do cliente e extrair dele os requisitos necessários para atender este problema e propor uma solução que os agradem.

Define-se por engenharia de requisitos, as técnicas que levam ao entendimento dos requisitos aos quais traçam uma ponte entre o levantamento de requisitos e a construção do software (PRESSMAN, 2011).

A engenharia de requisitos busca o entendimento dos requisitos, uma vez que os clientes fornecem requisitos confusos e algumas vezes os próprios engenheiros os confundem (LAPLANTE, 2014), e ultimamente tem se evidenciado alguns fatores que levam à má compreensão de requisitos, tais como:

a) problemas com a linguagem natural (imprecisões e ambigüidades);

b) complexidade demasiada alta (soluções muito complexas e ricas em detalhes);

c) dificuldades em entender comportamentos do sistema;

d) omissões de requisitos;

e) sobrecarga (elicitações de requisitos sem fundamentos, que contribuem para distorcer a realidade da solução);

f) inconsistências;

g) erros;

h) entre outros.

O objetivo da gestão de requisitos é obter uma, e somente uma interpretação consistente dos requisitos de todas as partes envolvidas (MILLER, 2009). Para tanto, a engenharia de requisitos dispõe de técnicas que auxiliam o engenheiro no levantamento de requisitos, auxiliando-os a obter requisitos das partes interessadas.

Entre as técnicas mais comumente utilizadas estão entrevistas, inspeção, observação, workshops, prototipação e reuso de requisitos.

Neste projeto optou-se por utilizar as técnicas de entrevistas, dado que permitem obter as necessidades mais evidentes, as restrições e suposições, como também promovem discussões interativas para explorar os detalhes das informações, encoraja e constrói relações harmoniosas entre as partes interessadas e permite observar comportamentos não verbais (HOSSENLOPP, 2008). Outra técnica utilizada foi o reuso de requisitos, pois permite conservar tempo ou dinheiro obtendo requisitos de softwares que já desempenham tarefas similares (ALEXANDER, 2009). Também foi utilizado a prototipação, visto que é um método para obtenção de respostas rápidas (PROJECT MANAGEMENT INSTITUTE, 2013) e permite validar requisitos por meio da demonstração do protótipo as partes interessadas, estimulando as pessoas a melhorarem seus requisitos (ALEXANDER, 2009).

4. Arquitetura de Software

A arquitetura de software cobre alguns aspectos dos sistemas, como o propósito do mesmo, quem o usará, de quais outros sistemas ele é dependente, como irá funcionar nos computadores do usuário final, os subsistemas e componentes, sua interface com o usuário, entre outras (HANMER, 2012).

Entende-se a importância da arquitetura de software fazendo-se uma analogia com os antigos castelos medievais, onde o emprego de paredes grossas, janelas altas e estreitas ou inexistentes contribuíram para a característica primordial do castelo: defender-se de ataques de espadas e flechas. Essa característica da construção, ou arquitetura do castelo determinou uma de suas propriedades, e é assim que se evidencia a prática da construção de software, as características dos mesmos dependem fundamentalmente de sua arquitetura (MEDVIDOVIC, 2010; TAYLOR, 2010).

Em se tratanto de arquiteturas para WEB, uma ganhou notório destaque, pois permitiu com que a comunicação entre computadores fosse possível. A figura 1 demonstra essa arquitetura, conhecida como Hypertext Transfer Protocol (HTTP).

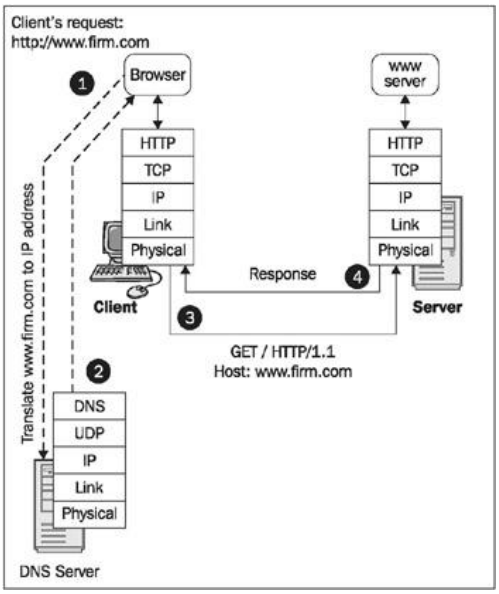


Figure 1. Arquitetura HTTP

Composta por dois elementos que interagem, de um lado o cliente e de outro o servidor de páginas, tem seu funcionamento descrito pelas etapas (DOSTALÉK, 2006):

a) o browser resolve o endereço por meio de um pedido à um servidor Domain Name Service (DNS), que lhe entrega o Internet Protocol (IP), nas etapas 1 e 2;

b) por meio do endereçamento a requisição é entregue ao servidor HTTP na etapa 3;

c) o servidor fornece a resposta ao cliente, na etapa 4.

Com a contribuição de frameworks de desenvolvimento, que abstraem parte do processo da arquitetura de pedido-resposta, torna-se mais viável construir softwares que atendam aos requisitos dos clientes e ainda se preocupe em manter uma arquitetura robusta e flexível. Um dos frameworks de grande utilização no desenvolvimento de softwares para a WEB é o Java Server Faces (JSF). Ele permite construir software por meio da separação de camadas, sendo dividido entre as camadas de visualização, controle e modelo. A visualização pode ser obtida por meio da escrita de tags, que são interpretadas e estruturadas em forma de uma árvore de componentes que mantém estado entre as enésimas requisições ao servidor (BURNS, 2010). A árvore repassa as informações das requisições aos controladores que por sua vez decidem quais serviços executar na camada de modelo e para qual visualização encaminhar os usuários.

5. Revisão Sistemática e Metanálise Diagnóstica

Nem todos os diagnósticos, prognósticos e tratamentos terapêuticos são obtidos com total certeza, muitos são resultados de uma análise baseada em evidências (ROSEMBERG, 1995).

A constatação de que evidências geradas por pesquisadores em todo o mundo não chegavam de forma adequada aos médicos contribuiu para o desenvolvimento das Práticas Baseadas em Evidência (PBE).

Por intermédio da revisão sistemática tornou-se possível sintetizar o resultado de vários estudos sobre impacto e efeito de determinadas tecnologias, visto que ela é uma revisão planejada de estudos publicados ou não, com o objetivo de responder a uma pergunta específica por meio da identificação, seleção e avaliação crítica deles (CASTRO, 2001).

A colaboração Cochrane detalha a revisão sistemática bem elaborada em sete passos: formulação da pergunta, localização e seleção de assuntos na fonte de dados, avaliação crítica dos estudos, coleta de dados, análise e apresentação, interpretação, aprimoramento e atualização (CASTRO, 2001).

A metanálise é o processo final de uma revisão sistemática onde se emprega análises estatísticas e modelos matemáticos que combinarão os resultados de estudos primários independentes sobre uma mesma tecnologia, buscando extrair dela uma medida concisa do efeito analisado (PETITI, 2000).

Teve partida nas ciências sociais, na educação, na medicina e pouco depois na agricultura. Em todas essas áreas a maior problemática era a combinação dos resultados de estudos independentes. Os primeiros trabalhos que se interessaram por essa problemática foram realizados por Cochran (LOVATTO, 2007) e as primeiras publicações sobre testes de precisão utilizando metanálise foram feitas entre as décadas de 80 e 90 tendo enfoque sobre como recuperar e selecionar estudos, assegurar a qualidade deles, sintetizar os resultados, investigar heterogeneidades e desenhar conclusões sobre os mesmos (LEEFLANG, 2013).

Naturalmente, estudos resgatados pela revisão sistemática e incluídos na metanálise diferenciam-se, e isto se chama heterogeneidade. Ela pode ocorrer por diferenças clínicas (população ou no tratamento), diferenças na metodologia do estudo (formulação do estudo e risco) e diferenças estatísticas (população, amostragem e resultados) (MELSEN, 2014).

Com isso, para a formulação de uma metanálise é necessário que se observe boas práticas, que tem se evidenciado pelo uso de softwares que auxiliam na sua realização, contribuindo para uma metodologia mais rigorosa e qualificada.

6. Elicitação dos Requisitos e Aplicação da Engenharia de Software

A necessidade do entendimento do objeto de estudo tem como propósito conciliar as práticas da engenharia de software com sua elaboração visando à qualidade. Como primeiro passo para a produção de software faz-se necessário entender os problemas que levam aos engenheiros propor novos softwares como resposta a eles. Com o entendimento dos problemas precisa-se compreender o domínio da aplicação e sua relação com as partes envolvidas para, posteriormente, extrair das partes interessadas os requisitos necessários para a construção do software. O trabalho é apresentado em seis etapas que demonstram a compreensão do problema, do domínio, dos requisitos e a elaboração dos modelos.

6.1 Compreensão do domínio da aplicação

O problema abordado por este trabalho foi esclarecido na introdução deste artigo, e consiste na elaboração de uma nova plataforma para realização de metanálises diagnósticas, visto que as ferramentas atuais não dispõem de todas as funcionalidades

necessárias para a realização da metanálise ou não englobam todas essas funcionalidades em um único software.

Foi identificado que as etapas envolvidas com a elaboração de metanálises diagnósticas são:

a) entrada dos dados dos estudos na base;

b) avaliação da qualidade dos estudos incluídos pela revisão sistemática;

c) análise tabular dos estudos;

d) análise gráfica dos estudos;

e) análise de viés de publicação.

A base de dados de uma metanálise diagnóstica consiste dos estudos incluídos nela pela revisão sistemática e deve estar apta a aceitar algumas características dos estudos, tais como: nome do estudo, autor do estudo, número de verdadeiros positivos, número de falsos positivos, número de verdadeiros negativos e número de falsos negativos. Mas também deve permitir a entrada de outras características, tais como idade, sexo, método de diagnóstico, etc.

A etapa de avaliação da qualidade dos estudos incluídos pretende verificar que tipos de estudos foram incluídos na metanálise. Conta com o auxílio de questões elaboradas por instituições, tais como a Cochrane, para entender a qualidade dos estudos sobre os aspectos metodológicos e para validar se estes estudos são confiáveis para fornecer uma resposta com maior evidência sobre determinado tema. Porém, como a avaliação da qualidade é muito subjetiva, pois depende da opinião do pesquisador que a avalia, foi constatado que mais de uma avaliação, por parte de outros pesquisadores faz-se necessária.

Uma segunda opinião determina a necessidade de um sistema colaborativo, onde um pesquisador poderá ajudar outro na realização da metanálise. Entretanto, como as opiniões sobre a qualidade dos estudos podem ser muito divergentes, estas avaliações se darão por meio de trâmites, que só deixarão de ser registrados quando os pesquisadores entrarem em um consenso sobre a qualidade dos estudos, contribuindo para uma avaliação mais rigorosa.

Outra avaliação realizada diz respeito à análise tabular dos dados. Nela são empregados testes estatísticos para avaliar o conjunto dos dados dos estudos incluídos, tais como:

a) razão de verossimilhança;

b) diagnostic odds ratio (DOR);

c) valor preditivo positivo e negativo;

d) sensibilidade e especificidade;

e) I^2;

f) qui-quadrado.

A análise gráfica auxilia os pesquisadores a avaliar melhor a relação entre a sensibilidade e especificidade, por meio da curva Receiver Operation Characteristic (ROC). Outros gráficos, tais como o de funil, permitem avaliar os vieses de publicação.

A avaliação de vieses de publicação consiste na análise de quais estudos foram incluídos na metanálise. E pretende responder as questões relativas a não-inclusão de estudos, ou seja, procura determinar se existem mais trabalhos satisfatórios publicados em relação aos que não apresentam os resultados esperados. Essa parte verifica se existem publicações que foram intencionalmente deixadas de serem publicadas por não se ter um resultado conclusivo, ou uma metodologia adequada, ou um resultado duvidoso.

Como parte incluída na compreensão do domínio da aplicação, buscou-se entender os perfis de usuários e sua relação com as atividades no sistema. A figura demonstra essas relações por meio do diagrama de casos de uso.

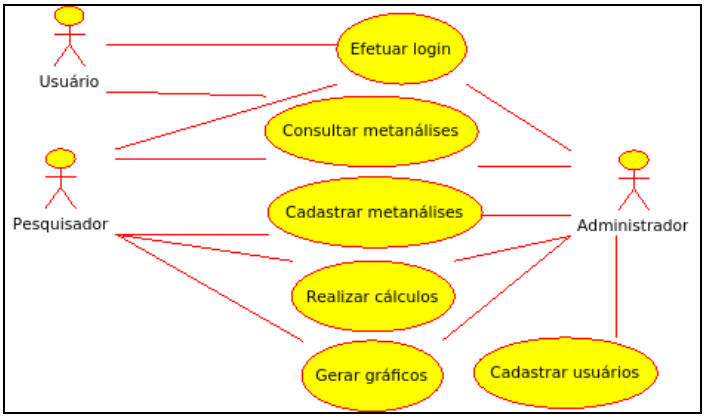


Figure 1. Perfis de usuário

Onde se verifica a existência de três perfis, sendo o primeiro deles (Usuário) o de mais baixo nível dentro da aplicação, o segundo (Pesquisador) de nível intermediário e o último (Administrador), o mais alto nível dentro da aplicação.

6.2 Avaliação da viabilidade

A avaliação da viabilidade se faz necessária como um dos princípios éticos da elaboração de software, e tem como objetivo verificar se já existe algum software capaz de realizar as mesmas atividades de forma semelhante minimizando o esforço para a elaboração de uma nova plataforma.

Neste estudo considerou-se avaliar os softwares mais utilizados por pesquisadores na elaboração de metanálises diagnósticas, sendo eles:

a) Meta-DiSc 1.4;

b) StataMP 13 Trial;

c) Review Manager 5.3 (RevMan).

Analisando as etapas necessárias para a realização da metanálise diagnóstica verifica-se que para a etapa de entrada dos estudos na base é preciso que o software permita a adição de 0,5 pessoas às células do estudo se uma delas estiver nula, ou descartar o estudo da base se existir mais de uma célula vazia. Esta configuração só é possível no Meta-DiSc, e no RevMan. Em se tratando da possibilidade de escolha do intervalo de confiança, somente o Stata permite a escolha de forma livre, já os demais limitam aos valores 90%, 95% e 99%.

Em relação à etapa de avaliação da qualidade dos estudos faz-se necessário o uso do RevMan. Porém, a utilização dele implica na avaliação por parte de somente um pesquisador. Seria interessante acrescentar outro pesquisador na avaliação, tornando a plataforma colaborativa, e contribuindo para uma análise mais qualificada da qualidade.

As etapas de análises gráficas e tabulares são melhores desenvolvidas utilizando-se o Meta-DiSc e a etapa de análise de viés com o uso do Stata. A figura 3 relaciona as etapas da metanálise com os softwares estudados, sendo que na categoria (a) são listadas as atividades relacionadas com a etapa de análise tabular e gráfica, na categoria (b) são demonstradas as atividades relacionadas com a etapa de análise de vieses e na categoria (c) verifica-se as etapas relativas a qualidade dos estudos incluídos.

	Atividades	M	S	R
a)	DOR	😊	😐	☹
	Razão de verossimilhança	😊	😐	☹
	Valor preditivo	😊	😐	☹
	Sensibilidade	😊	😐	☹
	Especificidade	😊	😐	☹
	Tabelas prontas	😊	😐	😐
	I^2	😊	😐	😊
	Qui-quadrado	😊	😐	😊
	Opções (0.5, efeito fixo ...)	😊	☹	😐
	Gráfico de floresta	😐	😊	😐
	Curva ROC	☹	😊	☹
b)	Análise de vieses	☹	😊	☹
	Regressão de Egger	☹	😊	☹
	Teste de Begg's	☹	😊	☹
	Gráfico de funil	☹	😊	☹
c)	Qualidade dos estudos	☹	☹	😊

LEGENDA
M = Meta-Disc
S = Stata
R = RevMan
😊 = Bem avaliado
😐 = Mal avaliado
☹ = Não possui

Figure 1. Avaliação da viabilidade

Além da avaliação das etapas da metanálise diagnóstica em cada software, foi constatado alguns problemas existentes e que dificultam o trabalho dos pesquisadores, tais como interface complexa (principalmente no RevMan) e necessidade de uso da linha de comando no Stata.

6.3 Levantamento e classificação dos requisitos

O levantamento de requisitos teve início com o estudo dos modelos de engenharia de software para elicitação de requisitos com posterior escolha do modelo e definição das etapas e abordagens necessárias para aplicação do modelo escolhido. Foram estudados os modelos de entrevistas, inspeções, observações, workshop, prototipação e reuso de requisitos.

A aplicação dos modelos se deu em maior parte pelo uso de entrevistas, reuso de requisitos e prototipação, tendo como requisitos funcionais principais:

a) configuração do método de agrupamento;

b) configuração do modelo de regressão;

c) inclusão de colunas na base de dados;

d) atalhos, tais como desfazer alterações;

e) possibilidade de adição de perguntas na avaliação de qualidade dos estudos;

f) possibilidade de inclusão de outro pesquisador na avaliação da qualidade dos estudos;

g) divisão em módulos;

h) configuração dos estudos com valores nulos;

i) configuração do intervalo de confiança.

Sendo que, foram classificados quanto ao tipo (funcionais ou não-funcionais), localização (base de dados, cálculos, exportação e importação, ajuda, autores, configurações, gerais e permissões), atores (consultores, pesquisadores e administradores) e prioridade (desejáveis, importantes, essenciais).

6.4 Arquitetura de software

A arquitetura foi definida com base no estudo da especificação do Java Enterprise Edition (Java EE) versão 6. Além da experiência do autor, os motivos que impulsionaram a escolha de tal arquitetura são:

a) controle de transações com o banco gerenciado pelo contêiner do servidor;

b) implementação com Java Server Faces (JSF) força o desenvolvedor a seguir um padrão de divisão de camadas mais rígido, compondo o software das camadas de visão, controle e modelo dos dados;

c) servidor de aplicações conta uma interface de aplicação para segurança das aplicações na WEB;

d) divisão entre camadas da aplicação e camadas do negócio;

e) possibilidade de trabalhar com clusterização, distribuição e acesso remoto;

f) serviço de mensagens, como serviços de emails;

g) injeção de dependência;

h) espaço de nomes para objetos.

O diagrama de pacotes da figura 4 demonstra a arquitetura do projeto.

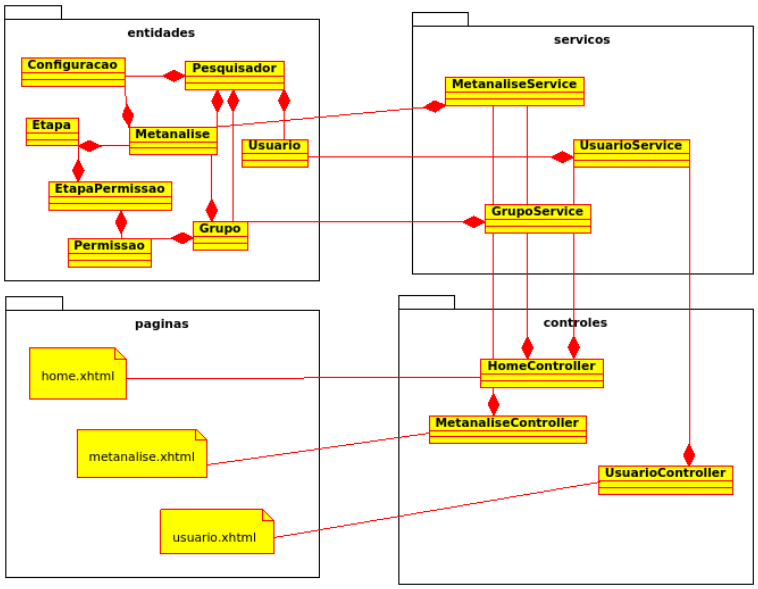


Figure 1. Diagrama de pacotes

As entidades fazem parte da camada de modelo, assim como os serviços. Elas representam os objetos da aplicação e tem forte relação com a persistência das informações no banco de dados. Já os serviços, são os mecanismos pelos quais se associa tarefas às entidades. Tais serviços serão responsáveis por disponibilizar métodos que façam a persistência das informações e o resgate delas do banco de dados. Estes são os Enterprise Java Beans (EJBs) da arquitetura, responsáveis por controlar as transações com o banco de dados.

Os controles fazem a intermediação entre as páginas e os serviços, decidindo quando os serviços serão chamados e para quais páginas o usuário deverá ser direcionado em caso de sucesso ou falha na tarefa que um serviço tenha desempenhado.

As páginas são a janela de visualização de conteúdo do usuário, por meio delas o usuário recebe as informações que deseja, e interage com a aplicação.

6.5 Prototipação

Esta fase do projeto teve como objetivo garantir a validade dos requisitos apresentado-os de forma visual as partes interessadas por meio da implantação do sistema em um servidor WEB gratuito. Também foi disponibilizado o documento de especificação de requisitos e um questionário que abordava os requisitos do sistema quanto ao perfil dos usuários, a disposição dos módulos, os métodos de análise, a possibilidade da inclusão de outro pesquisador na etapa de avaliação da qualidade dos estudos incluídos e a internacionalização do sistema.

A figura 5 demonstra as telas do protótipo.

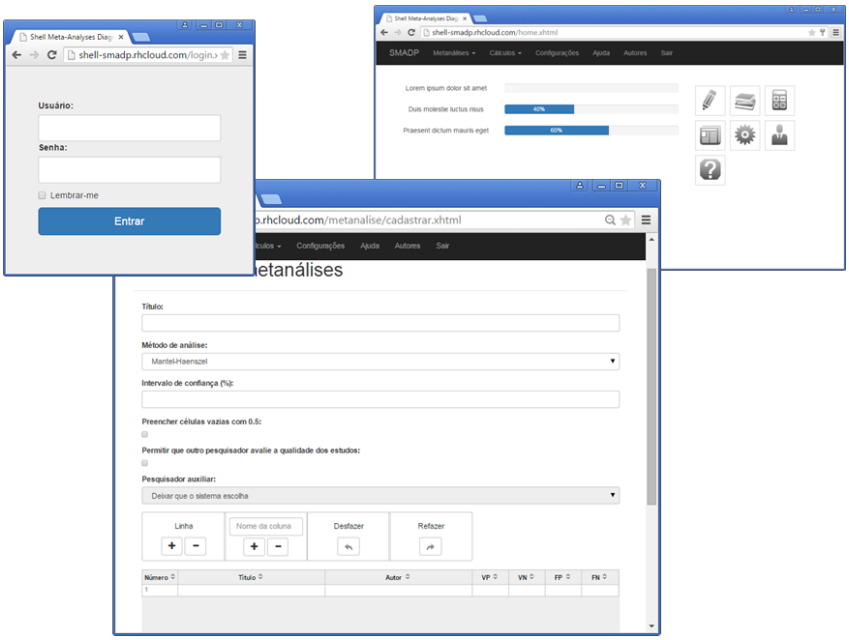


Figure 1. Protótipo

Os requisitos foram validados, visto que as respostas obtidas foram concordantes com as propostas de requisitos demonstradas.

7. Resultados

Com o disposto constatou-se que:

a) não há um único software capaz de realizar todas as etapas da metanálise diagnóstica;

b) os softwares disponíveis possuem limitações. Alguns pela complexidade de uso, outros pela condição de licença, tais como o Stata, que possui licença paga;

c) muitos pesquisadores podem estar envolvidos em uma mesma metanálise, por isso faz-se necessário que o software esteja disponível na WEB, dado que proporciona um ambiente colaborativo;

d) a engenharia de software contribuiu para o planejamento mais adequado do software, por meio dos estudos de técnicas para levantamento de requisitos, princípios de construção de software e análise de arquiteturas para WEB;

e) a análise da viabilidade para determinar a necessidade de uma nova plataforma contribuiu ainda para que fossem levantadas algumas das funcionalidades dos softwares existentes;

f) a prototipação permitiu validar junto as partes interessadas os requisitos do projeto, com o ganho de dispor uma base para a implementação futura da plataforma.

8. Conclusão

A engenharia de software vem há muito tempo auxiliando todas as pessoas envolvidas com projetos de software, fornecendo subsídios para entender e formular software de maneira simples e clara.

O estudo da engenharia de requisitos possibilitou uma maior compreensão sobre a necessidade de se aprimorar e se apropriar de conhecimentos relativos à área da computação, como também relativos à área alvo da elaboração do software, visto que os clientes, como também os próprios engenheiros de software acabam muitas vezes por confundir requisitos. Sendo assim, verificou-se que a engenharia de requisitos dispõe de técnicas para elicitação de requisitos, tais como entrevistas, inspeções, reuso de requisitos e prototipação, que contribuem por auxiliar os engenheiros de software na obtenção de requisitos mais claros para a construção do software.

A compreensão do domínio foi uma das dificuldades enfrentadas no desenvolvimento deste trabalho, visto que trata-se de uma área, a primeira vista, distante da computação. Entretanto, a computação vem auxiliando outras áreas do conhecimento e fornecendo softwares que minimizam o trabalho de pesquisadores, sobretudo de pesquisadores da área biomédica.

Verificou-se a importância da metanálise diagnóstica para geração de novas evidências e sintetização de resultados de estudos independentes na tomada de decisão de profissionais da área biomédica.

O estudo permitiu a elaboração de uma nova plataforma de metanálises diagnósticas, por meio do entendimento das etapas e importância delas na condução de novas metanálises. Com a análise da viabilidade foi identificado que os softwares existentes no mercado não dão conta de desenvolver todas as etapas da metanálise diagnóstica, e por meio de um maior aprofundamento foi possível levantar os requisitos e modelar uma solução que atendesse de forma mais efetiva na condução dessas práticas.

Além da contribuição científica deste trabalho para a computação, demonstrando a importância da engenharia de software e utilizando seus conceitos para elaboração de uma nova ferramenta, possibilitou-se também, por meio deste, a construção da base do projeto com a elaboração do protótipo, para que se possa dar continuidade a implementação da ferramenta.

Contudo, pode-se afirmar que o estudo da engenharia de software, a aplicação de suas práticas e a elaboração de uma arquitetura que contemple o projeto são práticas que devem ser observadas e que podem determinar o sucesso de um software.

References

ALEXANDER, Ian; BEUS-DUKIC, Ljerka. “Discovering Requirements: How to specify products and services”. Grã-Bretanha: John Wiley and Sons, 2009.

BURNS, Ed; SCHALK, Chris; GRIFFIN, Neil. “Java Server Faces 2.0”: The Complete Reference. New York: Osborne/McGraw-Hill, 2010.752p. ISBN:9780071625098.

CASTRO A. A. “Revisão sistemática e meta-análise”. Compacta: temas de cardiologia. v.3, p. 5-9, 2001. Disponível em: <http://metodologia.org/wp-content/uploads/2010/08/meta1.PDF>. Acesso em: 20 out. 2014.

DOSTÁLEK, Libor; KABELOVÁ, Alena.Understanding TCP/IP: “A Clear and Comprehensive Guide to TCP/IP Protocols”. Packt Publishing: Birmingham, 2006. 480 p. ISBN: 9781904811718.

HANMER, Robert. “Pattern-Oriented Software Architecture For Dummies”. Chichester, UK: John Wiley & Sons, Ltd, 2012. 384p. ISBN-9781119963998.

HOSSENLOPP, Rosemary; HASS, Kathleen B. “Unearthing Business Requirements: Elicitation tools and techniques”. Vienna: Management Concepts, 2008.

KENETT, Ron S.; BAKER, Emanuel R. “Process Improvement and CMMI for Systems and Software.Londres”: CRC Impress, 2010. 436p.

LAPLANTE, Phillip A. Requirements “Engineering for Software and Systems”. 2.ed. London: CRC Press, 2014. 324p. ISBN-9781466560819.

LEEFLANG, Mariska M. G. et al. “Cochrane diagnostic test accuracy reviews”. Systematic reviews, v. 2, p. 82, 2013. Disponível em: <http://www.systematicreviewsjournal.com/content/2/1/82>. Acesso em: 10 nov. 2014.

LOVATTO, P. A. et al. “Meta-análise em pesquisas científicas - enfoque em metodologias”. Revista Brasileira de Zootecnia, v.36, p.285-294, 2007. Disponível em:<http://www.scielo.br/statjournal.php?lang=pt&issn=1516-3598>. Data de acesso: 10 nov. 2014.

MCCONNELL, Steve. “Professional Software Development”: Shorter Schedules, Better Projects, Superior Products, Enhanced Careers. Boston, MA: Addison-Wesley, 2004. 241p. ISBN: 0321193679.

MELSEN, W. G. “The effects of clinical and statistical heterogeneity on the predictive values of results from meta-analyses”. Clinical microbiology and infection. v. 20, f. 2, p. 123-129, 2014.

MILLER, Roxanne E. “The Quest for Software Requirements”. Greeley St.: Maven Mark Books, 2009. 346p. ISBN-978-1-59598-067-0.

PETITTI DB 2000. “Metanalysis, Decision Analysis, and Cost-Effectiveness Analysis”: Methods for Quantitative Synthesis in Medicine. Oxford University Press. New York. 306 pp.

PRESSMAN, Roger S. “Engenharia de software [recurso eletrônico]”: uma abordagem profissional. Tradução de Ariovaldo Griesi. 7.ed. Porto Alegre: AMGH, 2011. 750p. Editado também como livro impresso em 2011. ISBN-978-85-8055-044-3.

PROJECT MANAGEMENT INSTITUTE. “A Guide to the Project Management Body of Knowledge”. 5 ed. Pensilvania: Project Management Institute, 2013.

ROSEMBERG, W.; DONALD, A. “Evidence based medicine: an approach to clinical problem-solving”. BMJ., v.310, p.1122-1126, 1995.

SACKETT, David Lawrence et al. “Medicina baseada em evidência”. Porto Alegre: Artmed, 2003. Disponível em: <http://www.scielo.br/pdf/rlae/v13n3/v13m3a17.pdf>. Acesso em: 30 maio 2015.

SILVA, Letícia Krauss. “Metodologias e diretrizes para a incorporação de tecnologias (Revisão do Rol) pela agência nacional de saúde suplementar (ANS)”. ANS, 2003. Disponível em: <http://www.ans.gov.br/portal/upload/biblioteca/TT_AS_Tema2Leticia%20Krauss.pdf>. Acesso em: 19 out. 2014.

TAYLOR, Richard N.; MEDVIDOVIC, Nenad; DASHOFY, Eric M. “Software Architecture - Foundations, Theory, and Practice”. New Jersey: John Wiley & Sons, 2010. 712p. ISBN-13 978-0470-16774-8.

TURINE, Marcelo Augusto Santos; MASIERO, Paulo Cesar. “Especificação de requisitos”: uma introdução. 1996. 26 f. Dissertação (Mestrado) – Usp, São Paulo, 1996.