Tutoriais do XI Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais

Organizadores
Plinio Thomaz Aquino Junior
Simone Diniz Junqueira Barbosa

SBC
Sociedade Brasileira de Computação

Organizadores
Plinio Thomaz Aquino Junior
Simone Diniz Junqueira Barbosa

Tutoriais do XI Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais

Porto Alegre

Sociedade Brasileira de Computação –SBC

2012

Dados Internacionais de Catalogação na Publicação (CIP)

T966 Tutoriais do Simpósio Brasileiro sobre Fatores Humanos (11. : 05 – 09 novembro 2012 : Cuiabá)
Tutoriais do IHC 2012 [recurso eletrônico] / organização: Plinio Thomaz Aquino Junior ; Simone Diniz Junqueira Barbosa. Dados eletrônicos. – Porto Alegre : Sociedade Brasileira de Computação, 2023.
126 p. : il. : PDF ; 24.7MB

Modo de acesso: World Wide Web. Inclui bibliografia
ISBN 978-85-7669-544-8 (e-book)

1. Computação – Brasil – Evento. 2. Fatores humanos. 3. Sistemas computacionais. I. Aquino Junior, Plinio Thomaz. II. Barbosa, Simone Diniz Junqueira. III. Sociedade Brasileira de Computação. IV. Título.

CDU 004(063)

Ficha catalográfica elaborada por Annie Casali – CRB-10/2339

Biblioteca Digital da SBC – SBC OpenLib

Índices para catálogo sistemático:

1. Ciência e tecnologia dos computadores : Informática – Publicação de conferências, congressos e simpósios etc. ... 004(063)

Cuiabá - MT

Tutoriais do XI Simpósio Brasileiro sobre Fatores Humanos em SistemasComputacionais

Tutorials of the 11th Brazilian Symposium on Human Factors in Computing Systems

05 a 09 de Novembro de 2012
Cuiabá – Mato Grosso - Brasil

Organizadores
Plinio Thomaz Aquino Junior (FEI)
Simone Diniz Junqueira Barbosa (PUC-Rio)

Organizadores do IHC 2012
Cristiano Maciel (UFMT)
Patricia Cristiane de Souza (UFMT)
Júnia Coutinho Anacleto (USP)
Vânia Paula de Almeida Neris (UFSCar)

Realizado pela **Sociedade Brasileira de Computação**

Adicionalmente patrocinado por:

Ministério da **Educação**

Sumário

Prefácio II

Capítulo 1 1

Elizabete Munzlinger, Fabricio B. Narcizo e José Eustáquio R. de Queiroz. **Sistematização de revisões bibliográficas em pesquisas da área de IHC**

Capítulo 2 21

Alexandre Maciel, Rodrigo Lins, Pablo Barros e Edson Carvalho. **Desenvolvendo Soluções com Interface Baseada em Voz**

Capítulo 3 42

Andrey Araujo Masiero, Leonardo Anjoletto Ferreira e Plinio Thomaz Aquino Junior. **Algoritmos de Clusterização e Python Científico apoiando Modelagem de Usuário**

Capítulo 4..... 82

Patricia Cristiane de Souza, Cristiano Maciel, Luciana A. O. de Moraes. **Projetando Sistemas Web com o uso de Técnicas de Interação Humano-Computador**

Prefácio

Temos a satisfação de apresentar, pela primeira vez, um livro com tutoriais ministrados no Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais. Na 11ª edição do evento, em 2012, foram quatro os tutoriais aceitos, cobrindo uma vasta gama de assuntos de interesse da comunidade.

O primeiro tutorial apresenta uma metodologia de pesquisa fundamentada em Revisão Sistemática, visando a investigações científicas mais robustas quanto ao tratamento e integração das evidências advindas da pesquisa bibliográfica. Dentre as vantagens da Revisão Sistemática, destaca-se o aumento da qualidade da pesquisa realizada e das publicações resultantes.

No segundo tutorial, é apresentada uma abordagem de desenvolvimento de aplicações com interfaces de voz. São discutidas algumas diretrizes para o desenvolvimento deste tipo de interface, de modo a torná-las funcionais e úteis. E, para apoiar o desenvolvimento de tais aplicações, foi apresentado o ambiente de desenvolvimento FIVE (*Framework for an Integrated Voice Environment*), cuja utilização é demonstrada passo a passo para facilitar o seu aprendizado.

O terceiro tutorial apresenta e discute algoritmos de clusterização para apoio à modelagem de usuário, implementados sobre bibliotecas em Python com foco em pesquisas científicas. Essas bibliotecas apoiam o rápido desenvolvimento de aplicações que podem ser usadas por profissionais de IHC na coleta de informações sobre perfil de usuário. O tutorial deixa clara a relação entre as bibliotecas, algoritmos de clusterização e técnicas de modelagem de usuários.

Por fim, o quarto tutorial descreve uma proposta de incorporação de técnicas de projeto e avaliação de IHC no desenvolvimento de aplicações Web. Para isso, baseia-se em um arcabouço ágil proposto dentro da Engenharia Web, que engloba o projeto da interface, estético, de conteúdo, navegacional, funcional, arquitetural e de componente, assim como a própria construção da aplicação.

Somos testemunhas do empenho dos autores dos tutoriais na confecção de cada capítulo. Essa pequena amostra da diversidade de objetos de estudo e prática da área de IHC visa a promover a troca de ideias e conhecimentos

aplicados na academia e na indústria. Desejamos que todos aproveitem este material em suas pesquisas científicas e projetos tecnológicos, contribuindo ainda mais para o aumento da qualidade da interação de usuários com e através de sistemas computacionais.

Plinio Thomaz Aquino Junior
Simone Diniz Junqueira Barbosa

Capítulo 1

Sistematização de revisões bibliográficas em pesquisas da área de IHC

Elizabete Munzlinger, Fabricio B. Narcizo e José Eustáquio R. de Queiroz

Abstract

During the development of a scientific research in the field of Human–Computer Interaction (HCI) it is necessary to identify, read, analyze and interpret relevant publications about a given theme or subject. Such a task usually results in large sets of bibliographical data. As a consequence, they must be put together in a single document, with the aim of making easy to assess the strength of the evidence of interest. Nonetheless, this is not an easy task to accomplish. Systematic reviews (SR) come to fulfill this need, by providing comprehensive, reliable, and unbiased summaries of a research on a single topic. Additionally, SR bring together large numbers of individual studies in a single report, provides the best evidence for decision making, and also help determine future research needs.

Resumo

Durante o desenvolvimento de uma pesquisa científica na área de Interação Humano-Computador (IHC) é necessário identificar, ler, analisar e interpretar publicações relevantes sobre determinado tema ou assunto. Tal tarefa resulta usualmente em extensos volumes de dados bibliográficos, os quais devem ser integrados em um único documento, com o objetivo de facilitar a avaliação da robustez da evidência em questão. Não obstante, isto não constitui uma tarefa fácil de realizar. As revisões sistemáticas têm o papel de preencher tal lacuna, provendo sínteses abrangentes, confiáveis e imparciais de uma pesquisa sobre um tema específico, integrando em um único relatório grandes volumes de estudos individuais, proporcionando a melhor evidência para a tomada de decisões e também auxiliando a identificar novos nichos de pesquisa.

1.1 Revisão Sistemática

1.1.1 Introdução e contexto

Estudos primários, que incluem, dentre outras metodologias, revisões bibliográficas, estudos experimentais e estudos de casos, usualmente combinados, agregam evidências

de pesquisas que guiam as atividades das comunidades científica e industrial da área de Interação Humano-Computador (IHC).

As revisões bibliográficas são frequentemente efetuadas sob a forma de estudos primários, com pouca ou nenhuma sistematização, o que as torna passíveis de produzir resultados enviesados, duplicados e/ou com pouco valor científico [Mafra e Travassos 2006]. Resultados dessa natureza podem tornar oneroso o processo de revisão, tanto para os pesquisadores, uma vez que toda e qualquer pesquisa necessita de investimento de tempo e esforço, quanto para as instituições, tendo em vista que fornecem os recursos para o desenvolvimento da pesquisa.

Tais problemas podem facilmente ocorrer com os resultados das pesquisas, uma vez que as publicações científicas consideradas na revisão bibliográfica podem ser selecionadas, por vezes, apenas intuitivamente ou seguindo cascatas de citações/palavras-chave, por vezes definidas de modo pouco criterioso ou sem adotar critério algum. Contudo, estudos primários não representam a única forma de conduzir revisões bibliográficas e, como forma de estudo secundário, a *Revisão Sistemática* (*RS*) pode ser incorporada ao estudo, complementado-o.

1.1.2 Metodologia de pesquisa baseada em Revisão Sistemática

Revisão Sistemática (*RS*) é uma metodologia de estudo secundário que visa estabelecer um levantamento formal do estado da arte, de forma robusta e consistente, a partir de um planejamento e execução criteriosos [Biolchini *et al.* 2005; Pai *et al.* 2004; Alderson e Higgins 2004; Kan *et al.* 2001]. As RS são planejadas para serem metódicas, explícitas e passíveis de reprodução [Sampaio e Mancini 2007]. Para tanto, o processo de pesquisa é conduzido segundo uma sequência metodologicamente bem definida de etapas, de acordo com um protocolo de estudo previamente planejado.

Com origem na Medicina e fortemente aprimorada por grupos tais como o *Cochrane Collaboration*, a metodologia de pesquisa relativa à RS foi concebida com o propósito de apoiar a prática da clínica médica fundamentada em evidências científicas, bem como de promover a atualização de grupos médicos sobre intervenções, a partir da interpretação de diversos estudos já publicados sobre determinado problema de saúde. Tal estratégia representou a materialização da expressão "*Medicina baseada em evidência*" da década de 1980, como forma de aprendizagem fundamentada em problemas [Sampaio e Mancini 2007] tendo-se difundido, desde então, para outras áreas de conhecimento. Na Ciência da Computação, teve consideráveis adaptações de pesquisadores da área de Engenharia de Software como Mafra e Travassos (2006) e Biolchini *et al.* (2005).

Diversas são as vantagens advindas da RS na área de IHC, dentre as quais se podem destacar:

- Revisão do estado da arte de temas relativos à área, incorporando, de modo abrangente, um espectro maior de resultados relevantes, não limitando as conclusões à leitura de um número reduzido de artigos, obtidos a partir de processos de busca *ad hoc*;
- Realização de estudos e pesquisas científicas não enviesadas e com pouco valor científico;
- Auxílio à identificação de problemas atuais não esgotados na área;

- Subsídios para a identificação de soluções para problemas encontrados no foco da pesquisa de interesse;
- Produção de conteúdo relevante para publicações científicas;
- Rápida produção textual de relatórios, trabalhos de conclusão de curso, dissertações e teses;
- Desenvolvimento do hábito da pesquisa científica sistematizada por uma sequência de etapas e fundamentada em procedimentos bem definidos a partir de um protocolo de estudo;
- Possibilidade de interação mais intensa entre orientador e orientando no desenvolvimento gradual de pesquisas caracterizadas mediante constante geração/atualização da documentação produzida;
- Rápida atualização do estado da arte sobre o tema ou área da pesquisa, graças à possibilidade de repetição da busca, a partir dos mesmos critérios previamente adotados na RS, o que garante a inclusão de novas referências em função do período de meses/anos remanescentes; e
- Rápida incorporação e nivelamento de novos membros no âmbito dos grupos de pesquisa das instituições, mediante o acesso a relatórios de atividades de RS de outros pesquisadores.

Neste capítulo, descrevem-se as etapas fundamentais que constituem o processo de elaboração de uma RS adaptadas de Narcizo (2012). Para melhor compreensão, os procedimentos de cada etapa apresentados neste documento serão ilustrados com base em uma RS conduzida na área de IHC por Munzlinger (2012). Para muitos procedimentos, serão apresentados sob a forma de dicas alguns cuidados a serem tomados, de modo atingir melhores resultados.

1.2 Etapas e procedimentos da Revisão Sistemática

O desenvolvimento da RS conta fundamentalmente com 3 etapas (vide Figura 1.1), as quais são naturalmente realizadas de modo sequencial, embora permitam e, às vezes, necessitem ser realizadas de modo iterativo. A razão para tanto é que, ao final de todo o processo, o *Protocolo de Estudo* (*PE*) deverá, muito além de conter especificações do que se deve ser executado, expressar fielmente o que, de fato, foi executado. Assim sendo, o PE deve incluir todos os ajustes que possam ter sido necessários no decorrer do processo da RS.

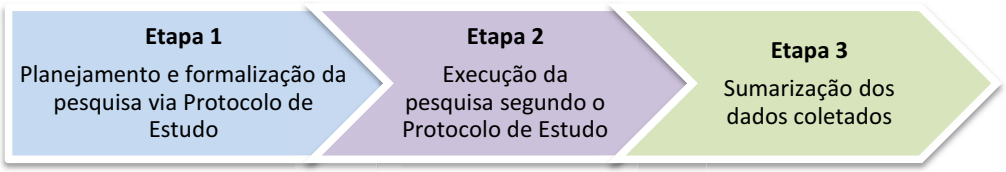


Figura 1.1 As três etapas de uma RS

Uma RS deve ser conduzida de forma metodológica, seguindo uma série de procedimentos ligados às metatarefas e distribuídos entre as três etapas. A Figura 1.2 apresenta uma macro-visão do processo completo de uma RS. Oferece uma visão geral das metatarefas, como estão distribuídas entre as três etapas bem como toda a

documentação gerada no decorrer do processo.

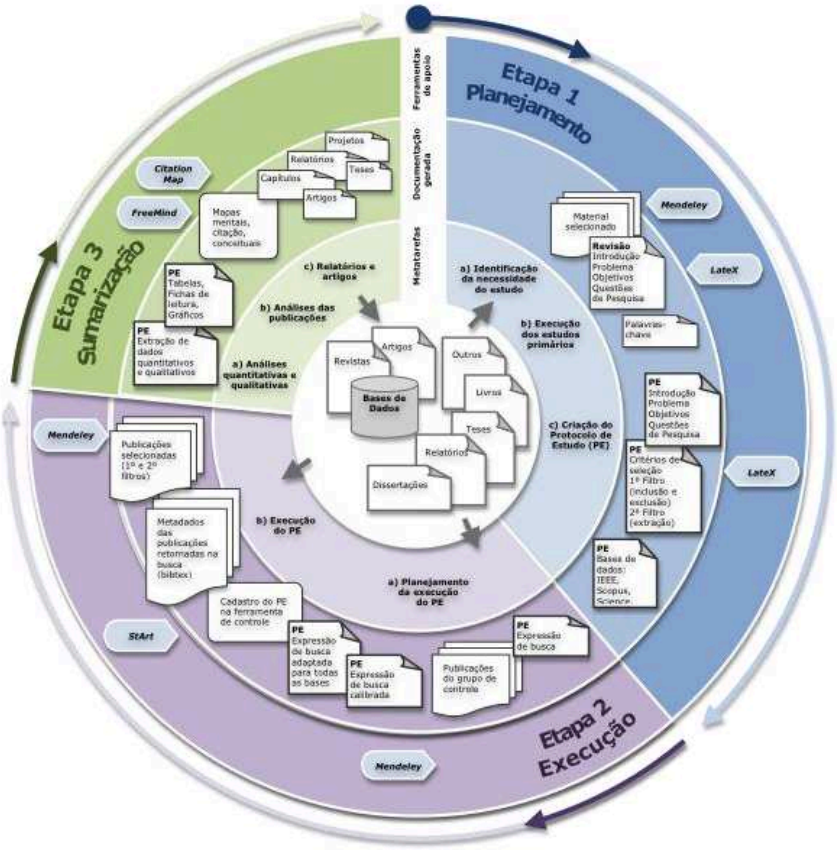


Figura 1.2 Macro-visão do processo de RS.

A primeira etapa compreende o *planejamento e a formalização da pesquisa via PE* a qual representa uma fase fundamental da pesquisa, uma vez que nela se identifica a necessidade do estudo e se delineiam critérios a serem adotados na pesquisa. A segunda etapa refere-se à *execução da pesquisa segundo o PE*. Por fim, na terceira etapa, sumarizam-se os dados coletados a partir da leitura e interpretação das publicações selecionadas para o estudo.

1.2.1 Primeira etapa: Planejamento e formalização da pesquisa via PE

As metatarefas de planejamento incluem a identificação da necessidade do estudo, a execução de estudos primários e a criação do PE (vide Figura 1.3).

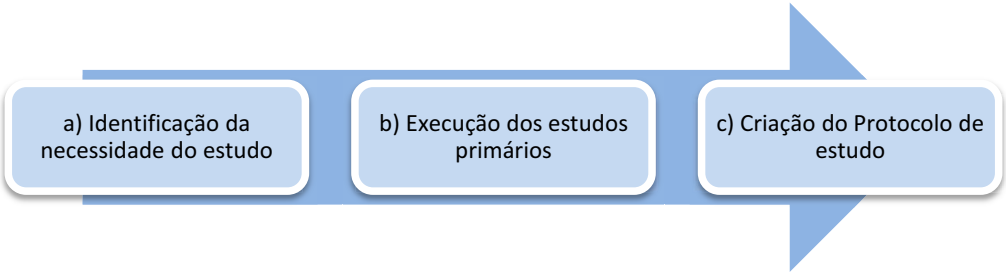


Figura 1.3 Pipeline das metatarefas associadas à Etapa 1

(*Planejamento e formalização da pesquisa via PE*)

Muitos pesquisadores poderão verificar, neste ponto, se suas pesquisas em andamento já se encontram em consonância às metatarefas a) ou b) da Figura 1.2 e, em caso afirmativo, poderão realizar a metatarefa subsequente, dando continuidade ao trabalho até então desenvolvido, visando a aprofundar a revisão do estado da arte de seu tema de pesquisa.

a) Identificação da necessidade do estudo

A necessidade de um estudo focalizando um tema ou a resolução de um problema de interesse pode surgir naturalmente durante a realização de um trabalho de conclusão de curso, um projeto de pesquisa, uma dissertação de mestrado ou uma tese de doutorado, bem como da necessidade da redação de projetos para fins diversos. Como atividade ligada a esta metatarefa tem-se a *escolha do tema e do problema*.

Comumente, o pesquisador possui algum conhecimento, experiência ou vivência sobre um tema de interesse e, por essa razão, tem ciência de possíveis problemas que podem ser estudados. Um tema ou problema de interesse também podem ser identificados no contato com pesquisadores e suas publicações em congressos e eventos que aliás, oferecem a oportunidade de atualização sobre os tópicos mais recentes pesquisados pelos membros da comunidade da área. Todavia, também é perfeitamente possível que o tema esteja relacionado a algo totalmente novo ao pesquisador. Para todos os casos supramencionados, a RS poderá auxiliar o pesquisador na atualização de seu conhecimento sobre o estado da arte do tema e do problema em foco.

Exemplo – Tema: "Projeto e avaliação de interfaces Web sob perspectivas estéticas"

b) Execução de estudos primários

Entende-se por estudo primário a condução de metodologias de estudo que agreguem evidências de pesquisas para a investigação de um tema, um fenômeno, um problema, uma tese ou uma teoria de interesse, no âmbito de um contexto específico [Khan *et al.* 2001]. Por sua vez, entende-se por estudos secundários a condução de estudos de maior amplitude, com o intuito de identificar, interpretar e avaliar o máximo de resultados relevantes a um determinado tópico de pesquisa [Kitchenham 2004].

As formas mais comuns de estudos primários realizados na área de IHC incluem as revisões bibliográficas, os estudos experimentais, os estudos de casos e os ensaios, usualmente combinados. Ao desenvolver uma RS, estudos primários se fazem necessários como uma etapa preliminar da revisão, exercendo um papel fundamental, como fonte de informação, no processo de amadurecimento do tema e, auxiliando na contextualização do problema, bem como na definição dos objetivos e questões de pesquisa. Estudos primários, também podem fornecer ao pesquisador meios para a identificação das palavras-chave relacionadas ao tema/problema. Esta pré-etapa é importante, uma vez que auxilia os pesquisadores a adequar a pergunta norteadora da revisão com base na informação disponível sobre o tema de interesse.

A revisão bibliográfica, metodologia de pesquisa de estudo primário, é classicamente utilizada pelos pesquisadores no desenvolvimento de seus trabalhos, sendo também a forma de estudo primário mais indicada como etapa preliminar da RS. Os estudos primários em forma de revisão bibliográfica são feitos com auxílio de

materiais diversos relacionados ao tema (e.g. livros, revistas científicas, artigos publicados, relatórios, dissertações, teses). A escolha do material bibliográfico nesta etapa pode ser feita a partir de critérios próprios de seleção e/ou, podem ser também fornecidos pelo orientador ou grupo de pesquisa, quando for o caso.

Assim como qualquer outra investigação científica, uma RS requer uma pergunta norteadora e questões claras e bem formuladas sobre a necessidade do estudo. Os tópicos da revisão bibliográfica farão parte do PE e podem ser:

- **Introdução**: Contextualização do tema e do problema, com revisão prévia do estado da arte fundamentada no material selecionado. Trata-se de uma breve revisão do estado da arte.

- **Definição do problema**: Identificação e definição formal do problema de pesquisa, em potencial, dentro do tema de interesse.

> Exemplo – Problema:
> "Diferentemente da usabilidade, que se encontra fortemente integrada ao processo de desenvolvimento de produtos há algum tempo, a estética ainda não está, juntamente com contribuições de projeto e de avaliação, satisfatoriamente incorporada ao desenvolvimento de interfaces Web, sendo uma ponte que, ainda precisa ser atravessada, com mais informação, estudos de caso e métodos de avaliação adequados. Assim, existe uma carência de instrumentos teóricos e/ou práticos complementares para o apoio ao projeto e à avaliação de interfaces Web, com vistas a aspectos estéticos de projeto visual."
>
> *Dica: É perfeitamente possível que, com a leitura das publicações selecionadas, o pesquisador identifique novos problemas relacionados ao tema.*

- **Definição do objetivo**: definição formal do objetivo da RS.

> Exemplo – Objetivo:
> Revisar e analisar, por meio de publicações científicas, o estado da arte sobre instrumentos teóricos e/ou práticos complementares para o apoio ao projeto e à avaliação de interfaces Web, com vistas a aspectos estéticos do projeto visual e suas influências sobre a usabilidade.
>
> *Dica: Este objetivo não deve estar relacionado com o que se deseja realizar em relação ao problema. O objetivo da RS sempre será uma revisão do estado da arte relacionado ao tema e ao problema. Somente ao final da RS, já com respostas para as questões de pesquisa, é que o pesquisador poderá então, em uma nova proposta de trabalho, apresentar os seus objetivos relacionados à resolução do problema adotado.*

- **Definição das questões de pesquisa**: Questões de pesquisa envolvem os diversos aspectos de interesse que permeiam o tema e o problema. Trata-se de um dos aspectos mais importantes da RS, pois, a partir das respostas obtidas, o pesquisador identificará o nível de abrangência do problema pesquisado, bem como possíveis soluções já propostas por outros pesquisadores para o problema em questão. Ainda, é provável que sejam identificados novos problemas e oportunidades de pesquisa. Assim, ao final da RS, então, o pesquisador, diante de respostas para suas questões de pesquisa, saberá *como* e *onde* atuar em relação ao problema.

> Exemplo – Questões de pesquisa:
> Questão principal: Com foco em aspectos estéticos do projeto visual e suas influências, quais são os recentes instrumentos teóricos ou práticos existentes complementares para o apoio ao projeto e à avaliação de interfaces Web?
> Questões secundárias:

a) Quais destes instrumentos teóricos e/ou práticos têm foco de aplicação na fase de projeto ou de avaliação de interfaces Web?
b) Quais as influências positivas ou negativas destes instrumentos no processo de projeto ou avaliação de interfaces? Como estas influências foram avaliadas ou analisadas?
c) Os instrumentos teóricos e/ou práticos são avaliados? Como?
d) Quais as influências positivas ou negativas destes instrumentos na usabilidade? Como essas influências são mensuradas?

Dica: Não há limite para a quantidade de questões de pesquisa. Contudo é necessária atenção, pois durante a leitura de cada publicação selecionada, as questões de pesquisa devem ser respondidas e documentadas.

- **Identificação de palavras-chave**: Com a execução dos estudos primários, o pesquisador usualmente identifica os termos, siglas e palavras-chave relacionadas ao tema e ao problema de interesse, aos quais guiarão o pesquisador na busca por publicações afins, e passarão por um refinamento até a estruturação de uma expressão de busca que traga como resultado publicações pertinentes e interessantes. O refinamento se dá no processo de calibração da expressão de busca, durante o planejamento da execução da RS, na segunda etapa.

Exemplo – Identificação inicial de palavras-chave:
Relacionadas a interfaces para a web: "web interface", website, etc
Relacionadas à estética: aesthetic, attractive, beauty, etc
Relacionadas à usabilidade: usability, "easy-to-use", etc
Relacionadas à avaliação e projeto: evaluation, measure, assessment, design, project, approach, etc.

Dica: Neste estágio não é necessário identificar todos os sinônimos e variações das palavras uma vez que tal processo ocorrerá durante a calibração da expressão de busca. Assim, é necessário identificar apenas as palavras-chave clássicas e comumente referenciadas nas publicações lidas durante os estudos primários.

c) Criação do PE

A criação do PE é necessária à formalização da pesquisa, uma vez que este norteia a condução do levantamento bibliográfico de forma sistemática e também torna a revisão passível de reprodução por outros pesquisadores. Definido no início do processo de RS, o PE é passível de atualizações durante as fases posteriores. Os tópicos presentes no referido protocolo incluem, além dos itens listados a seguir, a redação textual dos tópicos definidos no estudo primário.

- **Redação do estudo primário**: Transcrição dos itens *Introdução*, *Problema de pesquisa*, *Objetivos do estudo* e *Questões de pesquisa*, todos já definidos em metatarefas anteriores, durante a execução dos estudos primários.

- **Definição dos critérios de seleção**: Os critérios de seleção são denominados *filtros* e se referem aos critérios adotados para que determinada publicação seja ou não incluída na lista de publicações que serão estudadas de forma mais aprofundada e referenciadas em publicações, relatórios e outros documentos que poderão ser gerados. Refere-se ao conteúdo presente na publicação que realmente interessa. Os critérios de seleção serão usados então, para classificar os artigos que serão ou não estudados. Os critérios são divididos em dois filtros:

- **1º Filtro**: Define os critérios para a seleção das publicações relevantes. Como a seleção prévia das publicações por intermédio da expressão de busca não garante que

todo o material coletado esteja alinhado ao escopo desta pesquisa, é necessário aplicar um filtro que defina porque incluir ou excluir publicações no estudo. Fazem parte do filtro os critérios de inclusão e os critérios de exclusão. A classificação é feita a partir da leitura dos metadados (título, resumo e palavras-chaves) das publicações resultantes da busca.

> Exemplo – Critérios de seleção:
> Critérios de inclusão (CI) do primeiro filtro (1F). Será incluída toda a publicação que:
> CI1F-01: Define e/ou apresenta instrumentos teóricos e/ou práticos voltados para o apoio ao projeto ou à avaliação de interfaces Web
> CI1F-02: Investiga, compara e/ou avalia instrumentos teóricos e/ou práticos voltados para o apoio ao projeto ou à avaliação de interfaces Web.
> CI1F-03: Apresenta a aplicação de instrumentos teóricos e/ou práticos no projeto ou na avaliação de interfaces Web por meio de testes, experimentos e avaliações.
>
> Critérios de exclusão (CE) do primeiro filtro (1F). Será excluída toda a publicação que:
> CE1F-01: Não faz nenhum tipo de avaliação ou análise.
> CE1F-02: Refere-se ao objeto de estudo desta RS apenas como proposta para trabalho futuro.
> CE1F-03: Não está relacionado com estética de interfaces.
>
> *Dica: É necessário rigor na definição dos critérios de seleção, mas especialmente na aplicação dos critérios durante a leitura dos metadados. A regra é que, se a publicação apresentar pelo menos um critério de exclusão, esta deve ser desclassificada, não tendo que ser lida na íntegra.*

- **2º Filtro**: Empregado na seleção e extração dos dados relevantes. Esta etapa se dá mediante a leitura na íntegra, de todas as publicações que foram incluídas segundo os critérios do 1º filtro. A identificação dos critérios durante a leitura irão auxiliar na extração de conteúdo das publicações, criando também um padrão de dados extraídos o que facilitará na análise comparativa dos trabalhos relacionados.

> Exemplo – Critérios de Seleção:
> Critérios de seleção (CS) do segundo filtro (2F). Será incluída toda a publicação que contiver:
> ETP (elemento teórico/e ou prático): informações relacionadas à qual elemento teórico e/ou prático a publicação aborda
> PRO (projeto): possui foco ou informações sobre a fase de projeto de interfaces Web
> AVA (avaliação): possui foco ou informações sobre a fase de avaliação de interface Web.
> EST (estética): aborda a estética da interface
> USA (usabilidade): aborda questões de usabilidade
>
> Uma publicação que aborda todos os critérios supracitados receberá a seguinte sigla: CS2F+ETP+PRO+AVA+EST+USA.
> Se a publicação aborda todos os critérios acima com exceção dos itens estética e usabilidade receberá a seguinte sigla: CS2F+ETP+PRO+AVA-EST-USA
>
> *Dica: É recomendável o uso das siglas na tabela de catalogação das publicações para representar a informação contida no artigo que foi lido. As siglas adotadas devem fazer sentido no contexto da pesquisa.*
> *São esperados nesse passo, procedimentos metódicos de leitura e registro de informações. Pode não ser apropriado deixar o registro de informações importantes sobre cada publicação somente para após a leitura completa do mesmo. Uma boa estratégia é que, ao identificar um dos critérios do segundo filtro em uma publicação, durante a leitura, o registro da informação seja feita imediatamente, já podendo ser, inclusive, na forma de citação. Esse cuidado facilitará muito o trabalho posterior de redação de relatórios, da revisão do estado da arte e de artigos. As anotações, também chamadas de extração de*

dados podem ser feitas de diversas formas, inclusive usando as ferramentas de apoio relacionadas mais adiante.

- **Definição das bases de dados para consulta**: Qualquer base de dados pode ser selecionada para que seja efetuada a busca. Contudo recomendável observar que a qualidade das publicações resultantes pode estar também vinculada à qualidade e confiabilidade da base de dados escolhida. Também é importante observar que a busca em um única base de dados não oferecerá amplitude de resultados e também que a escolha das bases de dados deve fundamentar-se em critérios de escolha justificados no PE. Para cada base de dados, deverá ser feita a adaptação da expressão de busca, tendo em vista que, cada engenho possui sua própria sintaxe.

Exemplo – Bases de dados para busca
Justificativa: Foram selecionadas bibliotecas digitais que possuam sistema de busca avançada para expressão de busca personalizada e que também forneça o arquivo no formato Bibtex para o conjunto de publicações retornadas na busca. As buscas serão realizadas nas 4 máquinas de busca, já que algumas publicações são indexadas em apenas uma ou duas delas.
Seleção:
Engineering Village (Compendex) (http://www.engineeringvillage2.org)
IEEE (http://ieeexplore.ieee.org)
Scopus (http://www.scopus.com)
Science Direct (http://www.sciencedirect.com).

Dica: Apesar da máquina de busca da ACM possuir um sistema de busca avançada, esta não suporta uma expressão de busca complexa. É possível realizar buscas em apenas um único campo por vez (título, resumo, palavras-chave, etc) e o processo de concatenação dos resultados nem sempre é viável. Além disto, a máquina de busca não possui o recursos para download do arquivo no formato BibTEX com os dados e os resumos de todas as publicações selecionadas pela expressão de busca. Por este motivo, as publicações da ACM serão catalogadas por meio da busca manual.

A partir desta metatarefa, os pesquisadores já podem utilizar como apoio ferramental alguns sistemas computacionais gratuitos para auxiliar na condução da RS. Uma importante ferramenta de apoio é a StArt (State of the Art through Systematic Review). O StArt é uma ferramenta gratuita, desenvolvida no LaPES (Laboratório de Pesquisa em Engenharia de Software) da UFSCAR (Universidade Federal de São Carlos) encontrando-se disponível em *http://lapes.dc.ufscar.br/ferramentas/start-tool*. A ferramenta contempla as três etapas da RS: planejamento, execução e sumarização, englobando os processos de filtragem, extração de conteúdo das publicações e algumas análises quantitativas. Na Figura 1.4 ilustra-se a tela inicial de cadastro da revisão sistemática na ferramenta StArt.

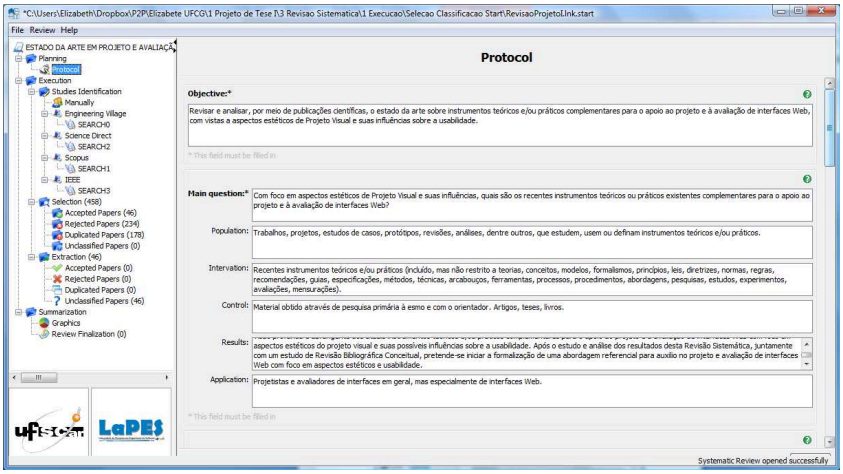


Figura 1.4 Cadastro do PE da na ferramenta StArt

A criação do PE na ferramenta StArt é realizada no módulo de "Planejamento", no qual se definem: (i) o objetivo; (ii) as questões de pesquisa; (iii) os engenhos de busca; (iv) os critérios de inclusão e exclusão; (v) o formulário de extração de informações, dentre outros. As informações cadastradas nesta etapa são utilizadas durante as análises da expressão de busca e das publicações, pois tais informações ficam disponíveis nas telas utilizadas durante a RS.

1.2.2 Segunda etapa: Execução do Protocolo de Estudo

As metatarefas de execução incluem o *planejamento da execução do PE*, sua *execução propriamente dita* e sua *atualização* (vide Figura 1.5). Esta etapa só deverá iniciar após a definição de todos os itens referentes ao PE, bem como o cadastro dos itens na ferramenta StArt.

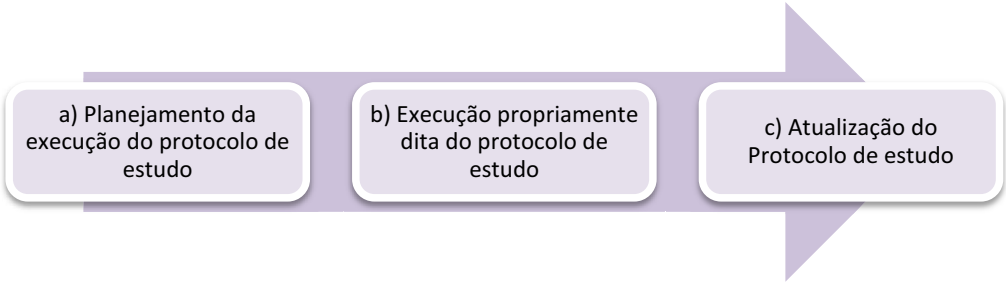


Figura 1.5 Pipeline das metatarefas associadas à Etapa 2 (*Execução do Protocolo de Estudo*)

a) Planejamento da execução do protocolo de estudo: Etapa destinada e ajustes finais no PE antes de iniciar a busca. Dentre os procedimentos associados a esta metatarefa destacam-se:

- **Expressão de busca**: Também denominada de chave de busca ou string de busca, trata-se de um conjunto composto por palavras-chave, utilizado para realizar pesquisas de publicações em bibliotecas digitais. A escolha inicial das palavras-chave deve ter sido definida a partir dos estudos primários. As publicações usadas nos estudos primários fazem parte do chamado *grupo de controle*. A expressão deve ser testada e

ajustada em um processo denominado *calibração da expressão de busca*.

- **Calibração da expressão de busca**: A expressão deve ser modelada segundo a sintaxe de uma das bases de busca e testada iterativamente. Dentre as publicações resultantes da busca, devem-se coletar aquelas que estiverem estreitamente relacionadas com o tema de interesse e adicionadas ao grupo de controle. A partir de uma leitura dinâmica dessas publicações, novas palavras-chaves e sinônimos são identificados sendo, então, adicionados à expressão, a qual deverá ser novamente testada. O processo deve ser repetido até que o pesquisador, com o aval de pelo menos um segundo pesquisador ou orientador entendam que os resultados da busca constituem um grupo de publicações relevantes.

> Exemplo – Expressão de busca antes da calibração:
> ((aesthetics) AND (web OR interface) AND (evaluation OR design) AND (usability))
> Retornou 9 publicações (2007 a 2012), 4 adicionadas ao grupo de controle para extração de novos termos.
> Expressão de busca após o processo da calibração:
> ((aesthetic OR aesthetics OR aesthetically OR aesthetical OR aestheticism OR "aesthetic design" OR attractive OR attractiveness OR attractively OR attractivity OR attraction OR beauty OR beautiful OR good-looking OR "visual design" OR "beauty design" OR "attractive design" OR graceful OR "graceful design" OR "visual appeal") AND (web OR "WWW" OR "World Wide Web") AND (usability OR usabilities OR usable OR "easy of use" OR "easy to use" OR "easy-of-use" OR "easy-to-use") AND ((evaluation OR evaluate OR evaluating OR evaluated OR assess OR assessment OR estimation OR inference OR design OR project OR measure OR measuring OR measurement OR estimate OR estimation OR estimating)))
> Ao final retornou 29 publicações (2007 a 2012)
>
> *Dica: É importante que durante as rodadas do processo de calibração, as palavras-chave sejam adicionas uma a uma à string, a fim de que se possa fazer o acompanhamento da quantidade e da relevância das publicações retornadas. Assim, é possível identificar termos que podem estar resultando em publicações enviesadas. A cada teste anota-se a palavra adicionada e o resultado, bem como se foi adicionada alguma publicação no grupo de controle.*
> *Outro cuidado importante é cautela com o uso de palavras-chave que pertencem a diversos temas de pesquisa, a exemplo a palavra "visual". Se for necessário o uso de palavras-chave desta natureza, tentar associar a palavra a outra, e.g."visual design" ou "visual interface", a fim de delimitar o escopo da busca. Adicionando-se a palavra-chave "visual" na string acima, o retorno de publicações salta de 29 para 81 publicações, trazendo por exemplo, muitos resultados sobre o tema de visualização de dados, que não é de interesse.*

- **Adaptação da expressão de busca para cada engenho**: Após a calibração da expressão de busca em uma das bases de dados escolhida, a string deve ser adaptada a cada engenho de busca, segundo sintaxe própria.

> *Dica: É importante observar que cada engenho de busca tem sintaxe própria. Alguns, por exemplo, permitem usar fragmentos de palavras, de tal modo sejam completadas com algumas variações. Outras, já recuperam singular e plural, bem como alguns sinônimos. É importante estar atento a essas variações.*

- **Cadastro da expressão de busca na ferramenta de apoio**: O aplicativo StArt permite que sejam cadastradas e gerenciadas as expressões de buscas utilizadas em cada engenho de busca das bibliotecas digitais, conforme ilustrado na Figura 1.6.

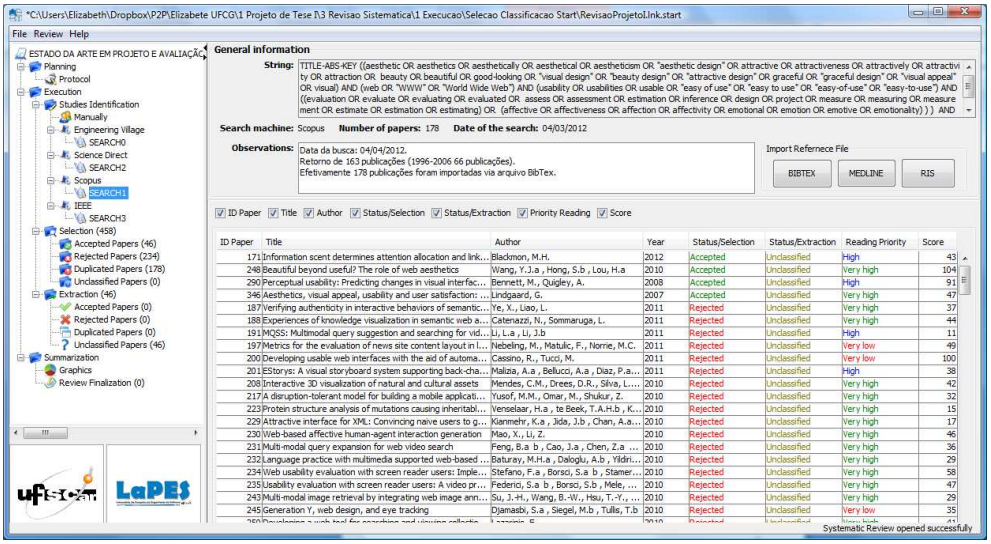


Figura 1.6 Gerenciamento das expressões de busca no sistema StArt

- **Atualização do grupo de controle**: O grupo de controle contém todas as publicações que serviram como base para a extração de palavras-chaves, desde os estudos primários até o final do processo de calibração da expressão de busca. Por conterem estudos relevantes, tais publicações também poderão fazer parte da RS, devendo ser posteriormente cadastradas posteriormente na ferramenta StArt no campo de busca manual.

b) Execução propriamente dita do protocolo de estudo com seleção, classificação e leitura das publicações.

Dentre os procedimentos presentes nesta metatarefa destacam-se:

- **Execução do processo de busca**: É a busca propriamente dita das publicações de interesse em cada uma das bases de dados, a partir das expressões de busca. As bases de dados selecionadas fornecem um arquivo no formato BibTEX contendo os metadados das publicações selecionadas pela expressão de busca nos engenhos das bibliotecas digitais, permitindo a catalogação automática na ferramenta StArt, por meio da importação destes arquivos de metadados.

> *Dica: É importante documentar todos os dados da execução da RS. Em relação à busca nos engenhos, procure anotar a data da busca, quantidade de publicações retornadas em cada plataforma.*
> *Os engenhos de busca fornecem arquivos em outros formatos que não o BibTEX. Contudo, muitos formatos são incompatíveis entre si, dado que a ferramenta StArt efetua um cruzamento de metadados entre os engenhos para identificar publicações que estavam indexadas em mais de uma base de dados. Para tanto, todos os arquivos devem seguir o mesmo padrão.*

- **Catalogação dos dados das publicações**: As publicações devem ser catalogadas no documento do PE. A catalogação é feita a partir de tabulação dos resultados, com a relação de dados considerados importantes para o pesquisador.

> *Dica: Dada a necessidade de geração de relatórios sobre a RS, é importante que os dados das publicações juntamente com os dados extraídos durante leitura sejam*

catalogados de forma documentada. Sugere-se o uso de anotações sob a forma de tabelas conforme os modelos Tabela 1 e Tabela 2, e também de fichas catalográficas, conforme modelo apresentado na Figura 1.10.

- **Seleção, leitura e extração dos dados**: Sem dúvida, trata-se da tarefa mais árdua e também a mais importante. Este procedimento deve ser executado seguindo rigorosamente todos os critérios de seleção definidos no PE, e durante a leitura, deve-se buscar as respostas para as questões de pesquisa.

> *Dica: Manter a sistematização durante o processo de desenvolvimento de uma RS é desafiador. Muitas vezes, o pesquisador em busca de ritmo na pesquisa, opta por ler previamente um conjunto de publicações, e fazer apontamentos somente ao final, sendo muitas vezes necessário reler algumas publicações. Recomenda-se fortemente que, durante o processo de leitura e extração de dados na RS, cada nova informação percebida como sendo de interesse seja imediatamente anotada, e sempre que possível, a anotação pode ser feita na forma de citação. Esse procedimento garante melhor aproveitamento de conteúdo e de tempo.*

A ferramenta StArt contém um módulo destinado à extração de dados, contendo formulários que permitem associar cada publicação aos critérios de inclusão, exclusão e qualidade definidos na metatarefa anterior. Além disto, o pesquisador pode registrar na ferramenta a disponibilidade e o endereço eletrônico da publicação, assim como se a publicação foi aceita ou rejeitada durante a avaliação da publicação. Na Figura 1.7 ilustram-se os formulários utilizados durante a extração dos dados de interesse nas publicações selecionadas pelas bibliotecas digitais.

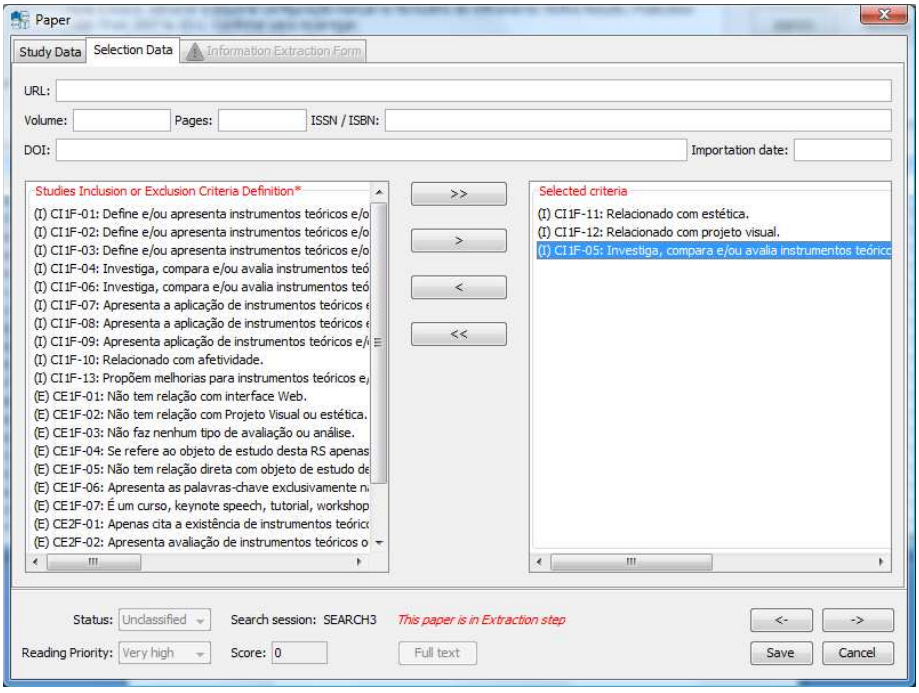


Figura 1.7 Seleção dos critérios das publicações analisadas

Ao final do processo de seleção dos dados relevantes (2º filtro), as publicações selecionadas para o escopo do estudo podem ser catalogadas a partir da ferramenta Mendeley®.

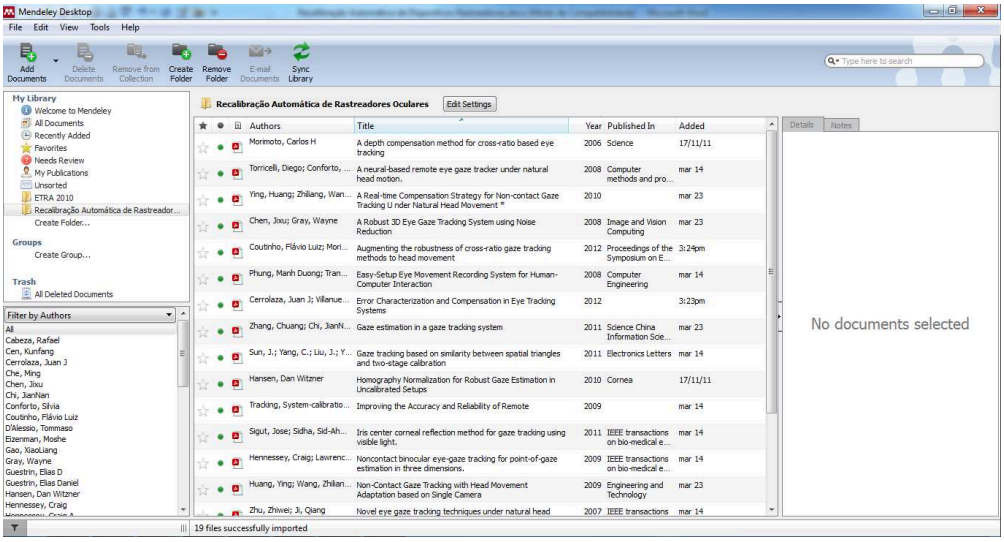


Figura 1.8 Arquivamento das publicações selecionadas na RS

A ferramenta Mendeley® é gratuita e pode ser utilizada para gerenciar e compartilhar artigos científicos e dados de pesquisa. Além da versão Web, há uma versão cliente disponível em *http://www.mendeley.com*. Na Figura 1.8 ilustra-se a catalogação de publicações selecionadas no cliente Mendeley®.

c) Atualização do Protocolo de estudo

A atualização das tabelas de dados e fichas de anotações. É imprescindível que cada detalhe do trabalho seja documentado e atualizado metodicamente quando uma tarefa for executada ou algum evento ocorrer. O documento do PE deve refletir exatamente o que foi desenvolvido, inclusive acertos e erros.

> *Dica: Ainda sobre manter a sistematização durante o processo da RS, é importante lembrar que, qualquer mudança de status em relação ao PE, seleção, classificação ou leitura de publicações, seja imediatamente atualizada no documento do PE, bem como na ferramenta StArt. Manter-se neste método significa dizer que, quando finalizar a leitura da última publicação selecionada, o trabalho, juntamente com relatórios e outros documentos estará praticamente finalizado.*

1.2.3 Terceira etapa: Sumarização dos dados coletados

As metatarefas de sumarização incluem a organização de indicadores quantitativos e qualitativos de interesse à pesquisa, a partir do processo de extração de dados das publicações analisadas, bem como a geração de documentos para organizar e apresentar os resultados obtidos com a execução da RS. Na Figura 1.9 ilustram-se as metatarefas relacionadas ao processo de sumarização dos dados coletados.

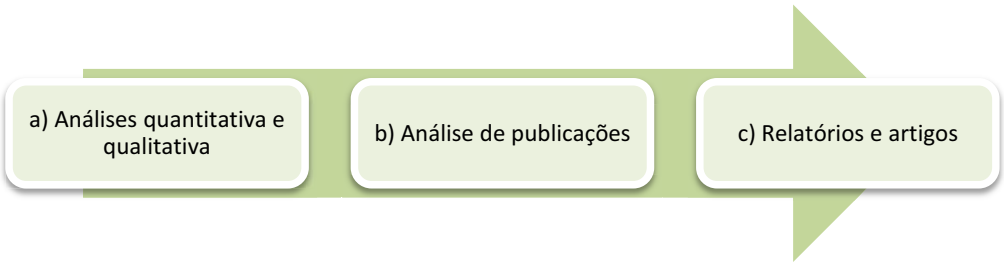


Figura 1.9 Pipeline das metatarefas associadas à Etapa 3

Nessa etapa são gerados gráficos, tabelas, figuras, relatórios, mapas e a documentação final com os resultados obtidos pela execução da RS. A sumarização dos dados coletados, por meio de um documento final, tem o intuito de delimitar o escopo do estudo, bem como auxiliar os pesquisadores na redação de publicações científicas para apresentação dos resultados obtidos.

a) Análises quantitativa e qualitativa

A análise quantitativa tem como objetivo levantar e organizar dados estatísticos relacionados à execução do protocolo de estudo da RS. Nesse caso, essa metatarefa pode ser realizada a partir da extração de informações disponíveis nas bases de dados das bibliotecas digitais selecionadas ou a partir da coleta de informações nas publicações analisadas durante a RS. Destacam-se as seguintes informações a serem analisadas e catalogadas durante a execução desta metatarefa:

- A quantidade de publicações selecionadas por cada engenho de busca das bibliotecas digitais;
- A quantidade de publicações selecionadas nas pesquisas manuais;
- A quantidade de publicações duplicadas, já que a mesma publicação pode estar disponível em duas ou mais bibliotecas digitais;
- A quantidade de publicações aceitas e rejeitadas após a execução de cada uma das etapas da RS;
- A quantidade de novas publicações adicionados ao estudo, caso a RS venha a ser atualizada com a execução do protocolo de estudo num período de meses/anos remanescentes.

Além dos dados supracitados, os pesquisadores também devem catalogar alguns dados quantitativos relacionados à sua pesquisa, e.g., quantidade de elementos teóricos ou práticos identificados na revisão.

b) Análise das publicações

Durante a análise das publicações, devem ser registrados os comentários do pesquisador nos formulários, a fim de documentar as decisões tomadas nos processos de seleção de publicações relevantes (1º filtro) e seleção de dados relevantes (2º filtro).

c) Relatórios e artigos

A ferramenta StArt não contém nenhum módulo exclusivo para criar automaticamente relatórios com os dados coletados durante a execução do PE. Por esse motivo, é necessário gerar manualmente os relatórios com as informações mais relevantes obtidas durante a análise do conteúdo de cada publicação.

Nas tabelas 1.1 e 1.2 ilustram-se dois exemplos que podem ser utilizados na catalogação das publicações analisadas durante a aplicação do 1º e 2º filtro do PE, baseadas na RS desenvolvida por Munzlinger (2012). Nelas, todas as informações foram geradas manualmente e as tabelas devem conter informações sintetizadas de todas as publicações analisadas durante a RS desenvolvida.

Tabela 1.1: Exemplo de tabela para catalogar o 1º filtro do PE

ID	Título	Autores	Dados	1º Filtro	Critérios de Inclusão e Exclusão	Base de Busca				
						Engineering Village	IEEE	Scopus	Science Direct	Manual
001	An exploration of relations between visual appeal, trustworthiness and perceived usability of homepages	Lindgaard, Gitte and Dudek, Cathy and Sen, Devjani and Sumegi, Livia and Noonan, Patrick	2011. ACM. ACM Transactions on Computer-Human Interaction	Sim	CI1F-02 CI1F-08 CI1F-09 CI1F-10 CI1F-11	X		X		X
N	Dominant pattern of visual design in online clothing websites	Anitawati, M.L. and Nor Laila, M.N. and Nagamachi, M.	2007. Webist 2007 - 3rd International Conference on Web Information Systems and Technologies, Proceedings	Não	CI1F-04 CI1F-05 CI1F-10 CI1F-11 CE1F-05	X	X			
277	Total de Publicações avaliadas de acordo com os critérios do 1º Filtro									
47	Total de Publicações resultantes da aplicação dos critérios do 1º Filtro									

Tabela1.2: Exemplo de tabela para catalogar o 2º filtro do PE

ID	Título	Autores	Dados	Critérios do 1º Filtro	2º Filtro	Critérios do 2º Filtro
000	Getting the message across: Visual attention, aesthetic design and what users remember	Sutcliffe, Alistair and Namoune, Abdallah	2008. ACM. Proceedings of the Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques, DIS	CI1F-01 CI1F-05 CI1F-06 CI1F-09 CI1F-11	Sim	(+PRO+AVA)(+WEB) (+PVS+EST)(+USA-AFE)
003	Balancing between usability and aesthetics of Web design	Yusof, Umi Kalsom and Khaw, Leng Khang and Ch'ng, Hui Yang and Neow, Boon Jiann	2010. IEEE. Proceedings 2010 International Symposium on Information Technology - Visual Informatics, ITSim'10	CI1F-01 CI1F-02 CI1F-03 CI1F-07 CI1F-09 CI1F-11	Não	(-PRO+AVA)(+WEB) (-PVS+EST)(+USA-AFE)

Além das tabelas supramencionadas, é necessário criar relatórios contendo informações cadastrais das publicações selecionadas, comentários gerais do pesquisador, critérios de seleção e de análise que foram aplicados nas publicações analisadas durante a execução do PE. Em geral, é necessário criar apenas os relatórios analisados durante a aplicação do 2º filtro do PE, sendo que os pesquisadores devem documentar apenas os relatórios das publicações selecionadas para o escopo do seus estudos. A Figura 1.10 contém um exemplo de modelo a ser utilizado para gerar os relatórios com as informações coletadas durante a aplicação do 2º filtro de seleção nas publicações selecionadas.

Para cada publicação que passou pelo 2º filtro, é necessário detalhar, quando possível, o título, o(s) autores, o veículo, o resumo, o ano de publicação, o volume e o número da publicação, as páginas, o número da conferência, a URL do DOI da publicação e comentários gerais do pesquisador (vide Figura 1.10). Além disto, devem ser catalogados também os critérios que levaram a publicação a ser selecionada no 1º e no 2º filtro, os critérios de análise derivados das questões secundárias e os critérios de

análise derivados da questão principal da pesquisa.

Título: Título do publicação analisada
Autores: Nome dos autores (com índice-H de cada autor, se possível)
Veículo: Nome do periódico
QUALIS: Avaliação QUALIS/CAPES do periódico ao qual o artigo foi publicado
Fator de Impacto: Fator de impacto do periódico ao qual o artigo foi publicado
Ano: – **Volume:** – **Número:** – **Páginas:** – **ISSN/ISBN:** –
Abstract:
Resumo em inglês do artigo, que pode ser extraído do BiBTex
Classificação: Informar se a publicação foi selecionada para o escopo do estudo
URL: Informar o DOI da publicação analisada
1º Filtro:
- Inserir como item todos os filtros selecionados na metatarefa de 2º filtro

2º Filtro:
- Inserir como item todos os filtros selecionados na metatarefa de 2º filtro

Comentários Gerais:
:: Nesse campo, os pesquisadores devem inserir comentários e/ou citações que possam utilizar no futuro, durante a redação de artigos
Critérios de Análise:
:: Nesse campo, os pesquisadores devem inserir os critérios de análise utilizados durante a execução do PE
Itens de Análise:
:: Nesse campo, os pesquisadores devem inserir as respostas das questões de pesquisa da RS

Figura 1.10 Exemplo de ficha para coletar dados durante a aplicação do 2º filtro

Outra forma de catalogar e gerenciar as informações coletadas nas publicações avaliadas é a partir do uso de mapas mentais. Os mapas mentais auxiliam os pesquisadores na organização e estruturação de informações relevantes da RS, por exemplo: (i) informações relacionados diretamente ao escopo dos seus estudos; (ii) análise qualitativa das publicações; (iii) nomes dos autores das publicações; (iv) países e instituições nas quais os trabalhos apresentados foram desenvolvidos, dentre outros.

Os mapas mentais (vide Figura 1.11) podem ser criados com a ferramenta FreeMind, de código livre e gratuito utilizada exclusivamente para apresentar um conjunto de idéias sob uma forma visual. A ferramenta está disponível sob os termos GNU/GPL em: *http://freemind.sourceforge.net/wiki/index.php/Main_Page*. Todos os mapas mentais criados durante uma RS devem ser disponibilizados no apêndice do documento final da pesquisa.

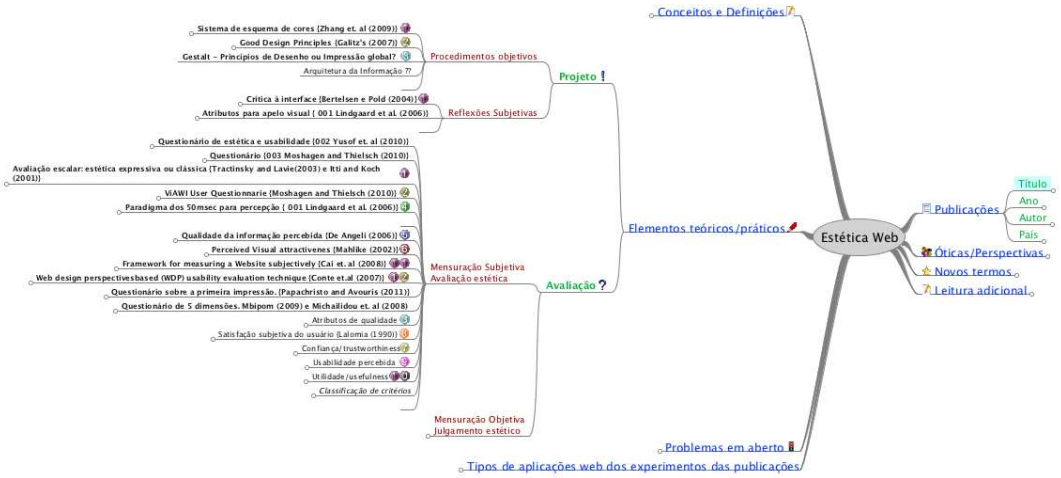


Figura 1.11 Exemplo de mapa mental

De modo complementar, outro tipo de catalogação de informações que pode ser utilizado pelos pesquisadores é o mapa de citações. O mapa tem o objetivo de gerenciar as relações entre as citações dos trabalhos selecionados. Os mapas de citações podem ser criados com a ferramenta CM (*Citation Map*) disponível no portal *Thomson Reuters* (http://apps.webofknowledge.com).

O mapa de citação é representado por meio de uma árvore interativa, que contém galhos com citações feitas e/ou recebidas, e folhas com as publicações relacionadas. É possível criar mapas de citação com até 2 (duas) gerações, organizando as informações por cores para facilitar a análise dos resultados. A Figura 1.12 ilustra um exemplo do mapa de citação de 1ª geração de uma publicação da área de rastreio ocular. A ferramenta CM permite que os nós do mapa sejam manipulados interativamente e esta característica fornece uma alternativa para melhor visualizar os relacionamentos existentes entre as publicações selecionadas.

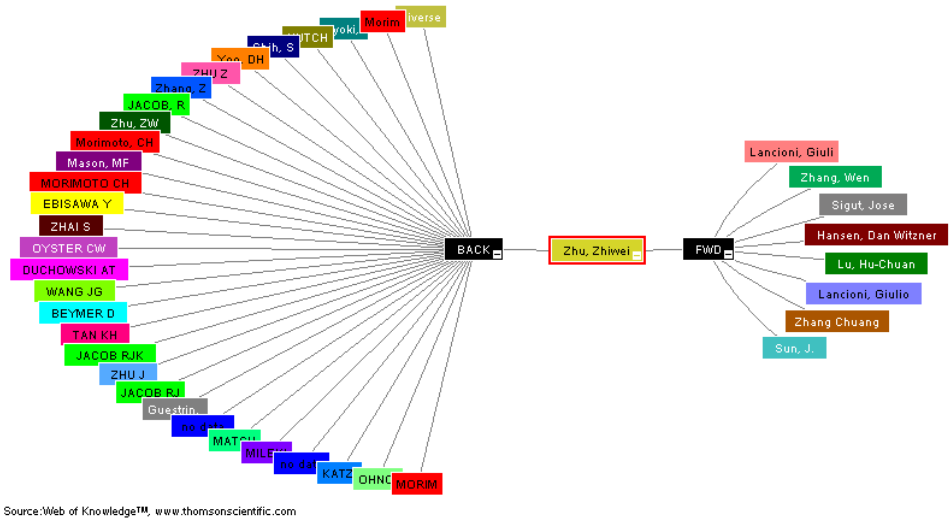


Figura 1.12 Exemplo de mapa de citação

Os mapas de citações gerados a partir da ferramenta CM podem ser exportados para imagens no formato PNG, devendo todos eles constarem em um apêndice no documento final da RS.

1.2.4 Considerações finais

Como toda a RS é procedimental, recomenda-se fortemente o uso de ferramentas auxiliares, para o acompanhamento e o controle do trabalho. Todas as etapas e atividades desenvolvidas devem ser imediatamente documentadas. Este é um dos aspectos que tornam a revisão *sistemática*. O fato se ter um protocolo de estudo formal garante que a mesma pesquisa possa ser reproduzida por outros pesquisadores, seguindo o mesmo delineamento, de modo a obter resultados similares. Também, permite a rápida atualização do estado da arte de um tema ou problema pesquisado recentemente, apenas reproduzindo a RS, a partir da mesma expressão de busca, mas somente para o período de tempo remanescente. Ainda, acrescida de alguns critérios e com algumas alterações nas palavras-chave da expressão de busca pode-se obter o estado da arte do mesmo problema em outro contexto. Todas as vantagens do uso da RS já relacionadas no início deste capítulo são reais e tornam o trabalho do pesquisador ainda mais bem fundamentado e consistente.

Material de apoio pode ser encontrado em: http://www.elizabete.com.br/rs/

Referências

Alderson P, Green S, Higgins JPT, editors. “Cochrane Reviewers’ Handbook 4.2.2” [updated March 2004]. In: The Cochrane Library, Issue 1, 2004. Chichester, UK: John Wiley & Sons, Ltd.

Biolchini, J. C., Mian, P. G., Natali, A. C. C., and Travassos, G. H. (2005). “Systematic review in software engineering”. Technical Report ES 679/05, Federal University of Rio de Janeiro, System Engineering and Computer Science Departament, Federal University of Rio de Janeiro, BR.

Khan, K. S., ter Riet, G., Glanville, J., Sowden, A. J., and Kleijnen, J. (2001). “Undertaking systematic review of research on effectiveness”. crd’s guidance for those carrying out or commissioning reviews. CRD Report Number 4 (2nd Edition), University of York, NHS Centre for Reviews and Dissemination, University of York, USA.

Kitchenham, B. (2004). “Procedures for performing systematic reviews”. Technical Report TR/SE 0401, Keele University and NICTA, Department of Computer Science, Keele University, UK.

Mafra, S. N. and Travassos, G. H. (2006). “Estudos primários e secundários apoiando a busca por evidências em engenharia de software”. Relatório Técnico ES 687/06, Federal University of Rio de Janeiro, System Engineering and Computer Science Departament, Federal University of Rio de Janeiro, BR.

Munzlinger, E. (2012). “Revisão sistemática: estado da arte em projeto e avaliação de interfaces web sob influências e perspectivas do projeto visual”. Relatório de Projeto de Doutorado 01, Universidade Federal de Campina Grande, Departamento de Sistemas e Computação, Universidade Federal de Campina Grande, BR.

Narcizo, F. B. (2012). “Revisão sistemática: calibração automática de dispositivos rastreadores oculares”. Relatório de Projeto de Doutorado 01, Universidade Federal de Campina Grande, Departamento de Sistemas e Computação, Universidade Federal de Campina Grande, BR.

Pai, M., McCulloch, M., and Colford, J. (2002). “Systematic review: A road map version 2.2”. Guidelines 2.2, University of California, Systematic Reviews Group, University of California, USA.

Sampaio, R. F. and Mancini, M. C. (2007). “Estudos de revisão sistemática: um guia para síntese criteriosa da evidência científic”. In *Revista Brasileira de Fisioterapia*, volume 11, pages 83–89.

Capítulo 2

Desenvolvendo Soluções com Interface Baseada em Voz

Alexandre Maciel, Rodrigo Lins, Pablo Barros e Edson Carvalho

Abstract

Oral communication is undoubtedly the most natural form of human communication. As human-computer interaction becomes increasingly common, there is a natural demand for systems capable of interacting with users in a more practical and intuitive way. This tutorial aims to demonstrate the process of building applications using speech interfaces. This is achieved by using a development environment called FIVE (Framework for an Integrated Voice Environment) that allows an easy instantiation of speech engines abstracting complexities. Given this scenario, researchers in the areas of speech processing and human-computer interaction will have the possibility to develop applications using technologies through an environment that fully offers productive mechanisms for the popularization of applications with voice interfaces.

Resumo

A comunicação oral é, sem dúvida, a forma mais natural de comunicação humana. Em virtude da interação humano-computador se tornar cada vez mais comum, surge uma demanda natural por sistemas capazes de interagir com os usuários de maneira mais prática e intuitiva. Para isso, este tutorial tem como objetivo a demonstração do processo de construção de aplicações utilizando interfaces de voz. Para tal é utilizado um ambiente de desenvolvimento intitulado FIVE (Framework for an Integrated Voice Environment) que permite de forma fácil a instanciação de motores de fala abstraindo complexidades inerentes. Diante desse cenário, pesquisadores da área de processamento de voz e da área interação humano-computador terão a possibilidade de desenvolver aplicações utilizando tecnologias através de um ambiente que oferece de forma completa e produtiva mecanismos para a popularização das aplicações com interface de voz.

2.1 Introdução

O homem sempre buscou meios de comunicação que facilitassem a sua interação com a máquina e a interface com o usuário é parte fundamental dessa comunicação. Segundo Moran (1981), a interface com o usuário deve ser entendida como sendo a parte de um sistema computacional com a qual uma pessoa entra em contato – física, perceptiva ou conceitualmente. Ou seja, é por meio da interface que o usuário pode manipular um dispositivo, perceber ou interpretar comandos e respostas desencadeadas pela interação com um sistema.

O desenvolvimento de uma boa interface com o usuário é um fator crítico para o sucesso de qualquer sistema, visto que pode se tornar uma fonte de motivação para o usuário, ou então, se mal projetada, um ponto de forte rejeição ao sistema. Diante dessa preocupação, o desenho de uma interface amigável deve levar em consideração a maneira natural com a qual o homem interage com o próprio homem, tentando levar essa naturalidade aos sistemas computacionais. Isso é alcançado através de interfaces que simulem o sistema sensorial humano (visão, audição, olfato, paladar e tato).

A imaginação é o limite para definir as aplicações que este tipo de interface pode ter. Vários meios de interação natural vêm sendo desenvolvidos ao longo dos anos e utilizados em aplicações, seja isoladamente ou de modo integrado. São as chamadas aplicações com interface multimodal. Como exemplo de interfaces naturais pode-se citar telas sensíveis ao toque, luvas (*datagloves*), sistemas de câmeras que captam gestos, sistemas que controlam o movimento dos olhos, ou as interfaces de voz, que possibilitam a realização de diálogos por meio de reconhecimento e síntese voz.

Nos últimos anos, a área de interface de voz tem recebido grande atenção da academia, em virtude de dois fatores: primeiro, devido à melhoria na performance dos sistemas de processamento automático de fala, incluindo reconhecimento de fala, tradução de idiomas falados e síntese de voz; segundo, devido à convergência de dispositivos e à massiva produção de conteúdos multimídia, que passaram a requerer modos mais rápidos e eficientes de interação com os usuários (Salvador *et al.*, 2010).

Além de ser uma das formas mais naturais de comunicação entre os homens, a interface baseada em voz oferece inúmeras vantagens quando comparadas a outras formas de interface, por exemplo: a velocidade (a maioria das pessoas pode falar facilmente a taxas de 200 palavras por minuto, enquanto poucas conseguem digitar mais de 60 palavras por minuto); a mobilidade (em diversos ambientes não cabe a utilização de teclados e mouse ou os olhos do usuário devem permanecer fixos em um *display*); e a segurança (a voz é um mecanismo biométrico que pode ser usado para verificar o acesso a sistemas e a ambientes restritos) (Fechine, 2002).

Apesar destas vantagens, o processo de construção de uma Interface de Voz com o Usuário (no inglês: *Voice User Interface* - VUI) apresenta diferenças substanciais quando comparado a uma Interface Gráfica do Usuário (no inglês: *Graphical User Interface* - GUI). Como a voz não está "visível" ao usuário, como no caso das GUIs, e também devido seu caráter transiente, a interface de voz normalmente requer uma a carga cognitiva consideravelmente maior do usuário. Outro fator importante é que entradas por voz são muito mais rápidas que as entradas via teclado, no entanto, as saídas por voz são muito mais lentas do que leituras textuais feitas superficialmente.

Desse modo, cabe ao projetista da interface balancear a melhor forma de conjugar as opções de interface aproveitando as principais vantagens que cada uma delas oferece (Martins e Brasiliano, 2012).

Diante desse cenário, as pesquisas na área de interfaces de voz têm evoluído bastante no sentido de oferecer mecanismos que auxiliem o desenvolvimento de aplicações com interface de voz. Este tutorial, cujo público alvo são os projetistas de aplicações com interface de voz, irá demonstrar a utilização do ambiente FIVE (*Framework for an Integrated Voice Environment*), ferramenta construída com o objetivo de auxiliar o desenvolvimento integrado de aplicações com interfaces de voz com rápida curva de aprendizagem, independência de plataforma, extensibilidade a novas técnicas.

O restante deste documento está organizado da seguinte forma: a seção 2 aborda a fundamentação teórica acerca da área de interface de voz, bem como a apresentação do processo de construção e instanciação de motores de fala; a seção 3 apresenta o framework FIVE e um tutorial básico para sua utilização; a seção 4 apresenta um conjunto de exemplo de motores criados com o FIVE; a seção 5 apresenta aplicações de demonstração com interface baseada em voz; e por fim, a seção 6 apresenta as considerações finais sobre este tutorial.

2.2 Interface de Voz

Interface do usuário baseada em voz, em inglês *Voice User Interface* (VUI) consiste na interação de uma pessoa com um sistema através de voz, utilizando uma aplicação de linguagem falada (Shneiderman, 2000).

As VUI's tiveram sua origem em pesquisas de Inteligência Artificial, especialmente no desenvolvimento de "conversational interfaces", na década de 1950. Mas somente depois dos anos 90 é que a tecnologia contou com uma significativa melhoria (Cohen *et al.*, 2004).

A área de interface de voz com o usuário abrange uma grande variedade de subáreas que visam atender as diversas demandas da sociedade. Segundo Campbell (1997), as tecnologias de voz podem ser classificadas como pertencentes a uma das seguintes subáreas:

- **Codificação:** processo de compressão da informação em um sinal de voz assim como a sua transmissão de forma econômica por intermédio de um canal cuja largura de banda seja menor do que um sinal não comprimido.
- **Síntese de Voz:** processo de criação de uma réplica sintetizada de um sinal de fala a partir de uma mensagem textual, com o propósito de fazer saber a informação contida na mensagem.
- **Reconhecimento de Fala:** processo de extração de informação de uma mensagem contida em um sinal de fala que, na sua forma mais simples, permite controlar as ações de uma máquina em resposta aos comandos falados.
- **Reconhecimento de Locutor:** processo tanto de verificação como de identificação do locutor com o propósito de restringir acesso à informação (arquivos privados), a redes (computador ou telefônicas), ou permissões físicas.

- **Identificação de Linguagem:** processo de identificação do idioma que o locutor está falando a partir da voz do locutor.

Todas as tecnologias supracitadas podem desempenhar um papel importante no domínio das aplicações, contudo as áreas que têm recebido maior atenção por parte da comunidade acadêmica são: reconhecimento de fala, síntese de voz e reconhecimento de locutor (Maciel, 2012).

De acordo com Zukerman e Litman (2001), é possível construir sistemas baseados em comandos de voz interativos e em tempo-real, em que as entradas do usuário sejam capturadas por um reconhecedor automático de voz e as saídas do sistema sejam enviadas através de um sintetizador de voz, ou através de mensagens previamente gravadas. Desse modo, para que uma aplicação com interface baseada em voz seja efetiva é preciso ela integre motores de fala diversos que "entendam" o que o usuário diz, desempenhem um processo de computação/transação, e respondam ao usuário de tal forma que dê prosseguimento à conversação e cumprimento dos objetivos do usuário.

Segundo Huang *et al.* (2001), a arquitetura típica para o desenvolvimento de uma aplicação com interface de voz possui três componentes principais: o primeiro representa o conjunto de motores que são responsáveis pelo reconhecimento de fala ou pela síntese de voz, o segundo consiste numa API (*Application Programming Interface*) que normalmente é usada para facilitar a comunicação entre os motores e a aplicação, e, o último consiste num conjunto de possíveis aplicações que podem ser desenvolvidas. A Figura 2.1 apresenta uma visão geral destes componentes dentro da arquitetura mencionada.

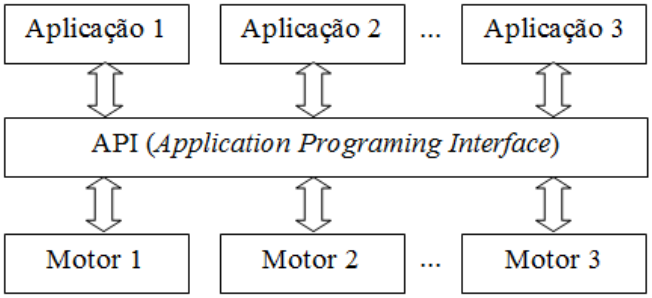


Figura 2.1. Arquitetura de aplicação com interface de voz (Huang *et al.*, 2001)

Esta arquitetura possibilita uma manutenção mais simplificada, já que é possível alterar um componente de uma camada sem influenciar a outra, possibilitando uma clara separação entre a tarefa de construção de motores e a instanciação deles na camada de aplicação. Desse modo, uma equipe integrada por especialistas em processamento de voz pode cuidar do desenvolvimento dos motores, seja de reconhecimento de fala, síntese de voz ou reconhecimento de locutor, enquanto outra equipe composta por projetistas de interfaces, artistas visuais, cientistas da computação e usuários pode ser direcionada para o desenvolvimento da interface (Martins, 2011).

2.2.1. Construção de Motores

A construção dos motores, seja para o reconhecimento fala, seja para a síntese de voz, consiste fundamentalmente em um processo de reconhecimento de padrões. Segundo Duda *et al.* (2000), este processo normalmente segue uma arquitetura comum a qualquer padrão (imagem estática, voz, dados financeiros, etc) e pode ser particionado em quatro módulos principais, como mostra a Figura 2.2.

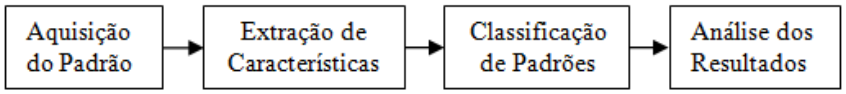


Figura 2.2. Componentes de um sistema de reconhecimento de padrões

Especificamente para os sistemas de reconhecimento de fala, o módulo de aquisição do padrão consiste na captura do sinal de voz e na conversão do mesmo em amostras digitais. O processo de captura das ondas de som normalmente é realizado por meio de um microfone e a conversão é realizada em conjunção com uma placa digitalizadora de som. Associada ao áudio adquirido, os sistemas da fala requerem a transcrição textual do conteúdo do áudio como um modo de ajustar os modelos de linguagem às amostras de áudio (McLoughlin, 2009).

O módulo de extração de características é o responsável por computar os dados do sinal de áudio e gerar as informações representativas necessárias ao módulo de classificação de padrões. Usando apenas as características importantes do sinal, a quantidade de dados utilizada para comparação é reduzida consideravelmente, assim, menos poder computacional é requerido e menos tempo de processamento é necessário. Segundo O'Shaughnessy (2008), os dois principais tipos de parâmetros de fala são os Coeficientes por Predição Linear (*Linear Prediction Coefficients* – LPC) e os Coeficientes Cepstrais em Escala Mel (*Mel-Frequency Cepstral Coefficients* – MFCC).

O módulo de classificação dos padrões consiste na utilização de abordagens algorítmicas para estabelecer representações consistentes dos padrões de voz a partir de um conjunto de amostras de treinamento rotuladas, e para realizar uma comparação fiável em conjunto de amostras de teste (Gaikwad *et al.*, 2010). Segundo O'Shaughnessy (2008), as abordagens mais utilizadas para a classificação de padrões são baseadas nos Modelos Ocultos de Markov, nas Redes Neurais Artificiais e nas Máquina de Vetor de Suporte. Especificamente para problemas de verificação de locutor, Holmes (2002) afirma que as principais abordagens associadas à classificação de padrões são baseadas na técnica de Quantização Vetorial ou nos Modelos de Misturas Gaussianas.

Por fim, o módulo de análise dos resultados consiste num conjunto de métricas utilizadas para oferecer uma melhor apresentação das saídas dos testes de classificação de padrões. Para os sistemas de reconhecimento de fala, a Taxa de Erro por Palavra (*Word Error Rate* – WER) associada a uma Matriz de Confusão é uma das métricas mais comumente usadas (Stehman, 1997). Já para os sistemas de síntese de voz há diversas abordagens para avaliar as vozes sintéticas. A escolha de qual abordagem escolher depende do propósito da avaliação e da aplicação específica a que se destina a voz sintética (Cryer and Home, 2010).

2.2.2. Instanciação de Motores

A etapa de instanciação dos motores de fala numa aplicação é normalmente suportada por uma camada de software, que oculta do desenvolvedor os detalhes de implementação de uma aplicação com interface baseada em voz. Este conjunto de software, habitualmente, designado API, permite o controle em tempo real dos motores de reconhecimento de fala e de síntese voz, bem como as interfaces de entrada/saída de áudio (Maciel *et al.*, 2008).

No contexto das aplicações com interface de voz, Huang (2001) afirma que o uso da API é importante porque garante que múltiplas aplicações possam trabalhar com uma grande quantidade de componentes (motores) disponibilizados por diferentes fornecedores de tecnologia de fala, em ambientes computacionais diversos.

Atualmente há uma grande variedade de iniciativas visando à disponibilização de API's para instanciação de motores de fala. A Microsoft Speech API e a Loquendo API são soluções proprietárias que permitem a interação com diversos tipos de motores independentes de fabricantes. Já a Java Speech API e o W3C VoiceXML (McGlashan *et al.*, 2010) são soluções criadas com o objetivo de oferecer um padrão que seja capaz de reconhecer e sintetizar fala independente de plataforma.

De modo geral, as funções disponíveis numa API são acessíveis somente por meio de programação. Ou seja, as requisições que os desenvolvedores desejam fazer à API são necessariamente realizadas por meio de chamadas inseridas no código fonte de suas aplicações. A sintaxe necessária para a realização dessas chamadas normalmente é definida na documentação disponibilizada junto com a API, sem a qual tal tarefa torna-se inviável.

A maioria da API's de fala disponibiliza suas funcionalidades segundo duas grandes áreas: reconhecimento e síntese. Isso se deve ao fato de que o projeto de uma aplicação com interface baseada em voz deve levar em consideração o modo natural como o homem se comunica com o próprio homem, ou seja, a interação por voz deve simular, de maneira natural, um diálogo com conversas caracterizadas por mudanças de iniciativas e feedbacks verbais e não verbais para poder indicar um verdadeiro entendimento (Maciel e Carvalho, 2007).

A Figura 2.3 mostra um exemplo de um diálogo entre um usuário e uma aplicação com interface de voz.

```
[sistema]: Bom dia. Bem vindo ao serviço de apoio ao passageiro. O que
deseja?
[usuário]: Qual o horário do próximo ônibus do Central Parque para o
Brooklin?
[sistema]: O próximo ônibus com origem Central Parque e destino
Brooklin sai às catorze horas e trinta minutos. Ajudo em algo mais?
[usuário]: Qual o preço do bilhete?
[sistema]: O bilhete custa dois dólares e quinze cents. Algo mais?
[usuário]: Não, obrigado.
[sistema]: Nós é que agradecemos. Tenha um bom dia.
```

Figura 2.3. Exemplo de diálogo com iniciativa mista

2.3 Framework FIVE

O FIVE (*Framework for an Integrated Voice Environment*) é um framework construído com o objetivo de auxiliar o desenvolvimento integrado de aplicações com interfaces de voz. Com ele, é possível gerar motores de reconhecimento de fala, de verificação de locutor e de síntese de voz e, por meio de uma API (*Application Programming Interface*) própria, instanciar estes motores em aplicações desenvolvidas em múltiplas plataformas (Maciel e Carvalho, 2010).

Os requisitos utilizados na construção do FIVE levam em consideração uma pesquisa realizada junto à comunidade acadêmica da área de processamento de voz em (Maciel, 2012). Segundo esta pesquisa, os especialistas consultados mencionaram que os requisitos essenciais para uma ferramenta produtiva que auxilie o desenvolvimento de aplicações com interface baseadas em voz são: rápida curva de aprendizagem, portabilidade entre ambientes computacionais, e capacidade de extensão para as técnicas de extração de características e de classificação de padrões.

Diante desses requisitos, a arquitetura geral do FIVE foi desenhada de modo independente, baseada em três módulos: o CORE, que consiste num conjunto de classes abstratas parcial ou completamente implementadas definidas de modo sequencial e extensível para auxiliar o processo de criação dos motores de fala; o módulo API, que consiste numa implementação simples, robusta, transparente e portável que otimiza a instanciação dos motores na camada de aplicação; e o módulo GUI (*Graphical User Interface*) que consiste numa interface gráfica que auxilia o desenvolvedor através de uma interface sequencial e de fácil compreensão. A Figura 2.4 mostra os módulos da arquitetura geral do FIVE.

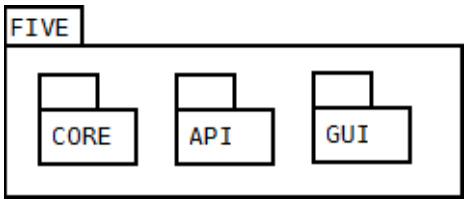


Figura 2.4. Arquitetura geral do FIVE

O FIVE é um framework orientado a objetos, desenvolvido em linguagem de programação Java, utilizando o modelo de divisão de responsabilidades MVC (*Model-View-Controller*) (Gamma *et al.*, 1995) e o framework de persistência Hibernate para proporcionar independência de sistema de banco de dados.

Desenvolvido como parte da tese de doutorado intitulada *Investigação de um Ambiente para o Desenvolvimento Integrado de Interface de Voz* (Maciel, 2012), o FIVE é um software livre, registrado no Instituto Nacional de Propriedade Intelectual do Brasil (INPI), com os direitos autorais pertencentes à Universidade Federal de Pernambuco. A ideia dos mantenedores do FIVE é criar uma grande rede de colaboração para que a comunidade acadêmica possa ajudar no aperfeiçoamento e na implementação de melhorias para a ferramenta. Mais informações sobre os termos de uso e normas de colaboração estão disponíveis no site oficial: www.cin.ufpe.br/~five.

A partir da consolidação da ferramenta, foi construído um tutorial como modo de auxiliar os desenvolvedores desde sua inicialização até a geração do motor de fala. As seções a seguir apresentam o passo a passo deste tutorial.

2.3.1. Inicialização

Para iniciar o FIVE, basta fazer o registro no site oficial do framework, descompactá-lo e executar o arquivo *FIVE_GUI.bat*, caso esteja utilizando o ambiente Windows, ou *FIVE_GUI.sh*, caso esteja utilizando o ambiente Linux. Cabe observar se a pasta *lib* se encontra com os arquivos FIVE_CORE.jar e FIVE_API.jar e todas as outras bibliotecas que acompanham a distribuição do FIVE.

2.3.2. Projeto

Para criar um projeto no framework FIVE, o usuário deve clicar no menu *Arquivo* e escolher a opção *Novo*. A tela *Novo Projeto* será apresentada e, nela, o usuário deve escolher o tipo de projeto que deseja criar: Reconhecimento de Fala, Reconhecimento de Locutor ou Síntese de Voz. Em seguida, o usuário deve clicar no botão *Avançar* e informar o nome e o local que deseja dar ao seu projeto. Por fim, é preciso especificar o banco de dados que deseja utilizar para armazenamento das informações dos cadastros realizados e dos resultados obtidos. A Figura 2.5 mostra a sequência de abas da tela *Novo Projeto*.

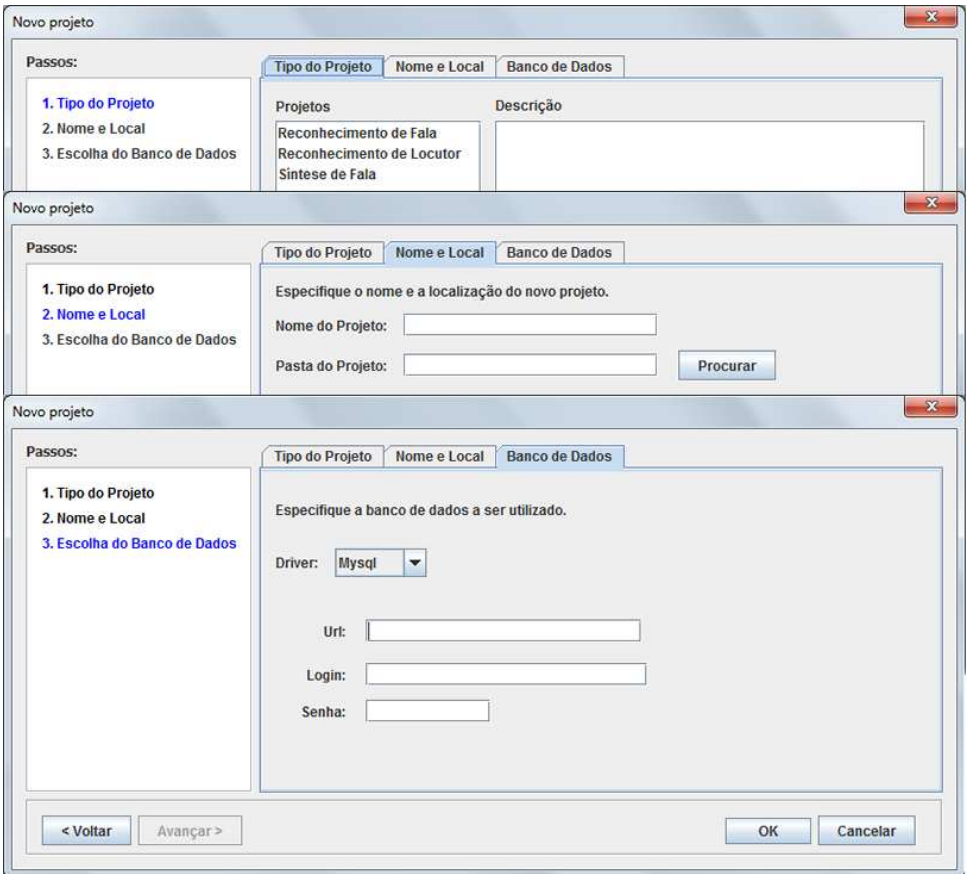


Figura 2.5. Tela Novo Projeto

Assim que for criado o projeto, o painel principal do FIVE apresentará um conjunto de abas com as seções necessárias à construção de um motor de fala e uma estrutura de arquivos e pastas será criada no caminho informado.

2.3.3. Locutor

Para cadastrar um locutor, o usuário deve clicar no botão *Incluir*, na aba *Locutores*, e informar os dados solicitados da tela *Locutores*. Informações como Região, Idade e Gênero são úteis caso o usuário deseje especializar o seu motor de fala, fazendo para isso o uso de opções de filtros que possibilitam essa especialização. A Figura 2.6 mostra a Tela *Locutores*.

2.3.4. Locução

Para cadastrar uma locução o usuário deve clicar no botão *Incluir* na aba *Locuções* e escrever a locução desejada na tela *Locuções*. O conteúdo escrito neste espaço pode variar de acordo com o tipo de projeto que o usuário esteja criando. Para motores de reconhecimento de fala, podem ser inseridas palavras isoladas ou palavras contínuas. Para motores de reconhecimento de locutor, podem ser inseridas palavras isoladas, que funcionam como uma senha para o reconhecimento dependente do texto ou textos foneticamente balanceados, pelos quais os modelos fonéticos são extraídos para o reconhecimento independente do texto. Para motores de síntese de voz, devem ser inseridos textos foneticamente balanceados buscando extrair os melhores modelos fonéticos para a geração de fala sintética. A Figura 2.7 mostra a tela *Locuções*.

Figura 2.6. Tela de Locutores

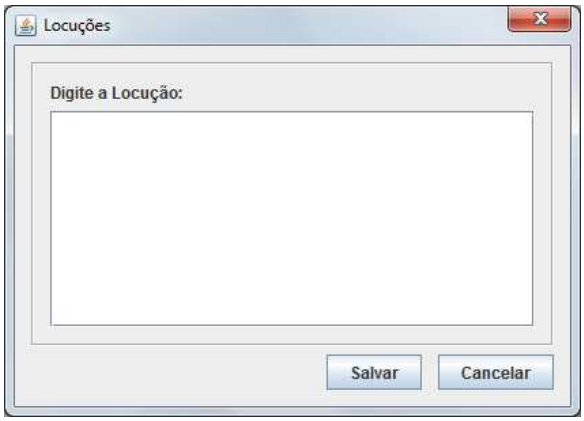


Figura 2.7. Tela de Locuções

A princípio, o processo de cadastro de locuções parece uma ação bastante simplista, mas, na verdade ele realiza um passo fundamental para o desenvolvimento do restante do sistema: o Processamento da Linguagem Natural (PLN). Este processo é dividido em duas fases, a primeira consiste no pré-processamento do texto, transformando abreviações, siglas, e números em representações textuais. A segunda consiste na realização da conversão grafema-fonema, da divisão silábica e da determinação da sílaba tônica das palavras contidas na locução.

2.3.5. Amostras de Áudio

Para cadastrar uma amostra de áudio o usuário deve clicar no botão *Incluir* na aba *Base de Dados* e preencher os campos informados na tela *Amostra*. Para que uma amostra de áudio seja inserida com sucesso, o usuário deve informar todos os campos solicitados na tela: o locutor que irá realizar a gravação, a locução que será falada, o ambiente em que a gravação está sendo realizada, se é locução é genuína ou impostora (informação necessária para motores de verificação de locutor), as características da amostra, e, por fim, o nome do arquivo de áudio que será salvo na pasta *samples* do projeto. A Figura 2.8 mostra a tela *Amostra*.

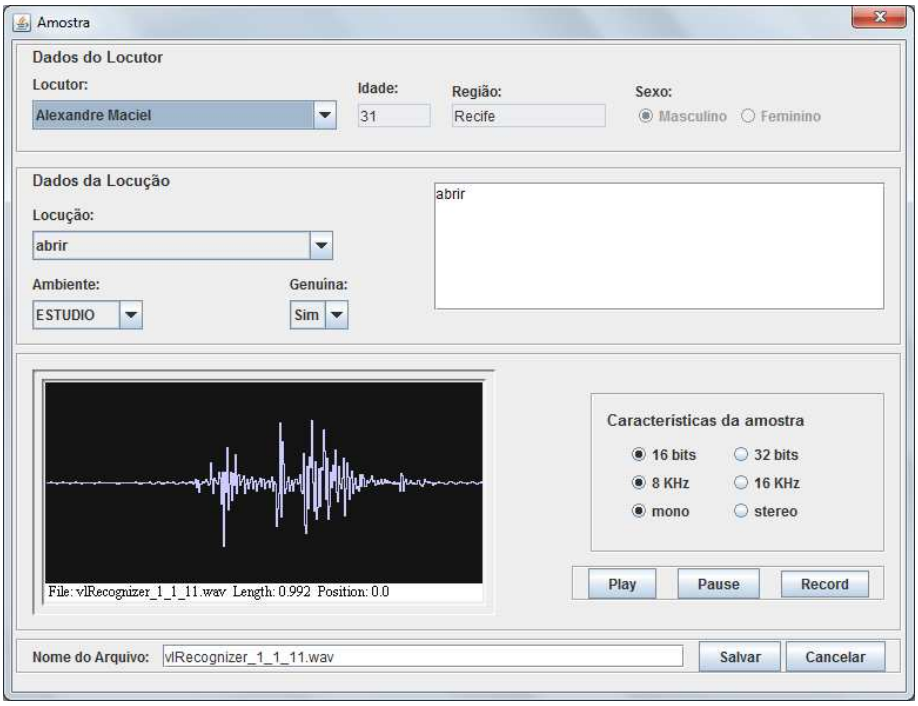


Figura 2.8. Tela de Amostras de Áudio

Para realizar a gravação do áudio, o usuário deve clicar no botão Record, que, assim que é clicado, muda o nome para Stop. Caso deseje ouvir o áudio gravado, o usuário deve clicar no botão Play.

2.3.6. Extração de Características

Para realizar a extração de características das amostras de áudio existentes no projeto o usuário deve clicar no botão *Incluir* na aba *Extração de Características* e preencher os campos informados na tela *Parâmetros da Extração de Características*. Nesta tela, o usuário encontra alguns parâmetros necessários à extração de características. Caso deseje filtrar apenas algumas amostras de áudio a serem extraídas, pode utilizar as opções de Filtro disponibilizadas no topo da tela. Alguns parâmetros solicitados para a extração são fixos, como: Superposição, Duração do Frame, Pré-ênfase e Janelamento. A depender da técnica escolhida, outros parâmetros exclusivos dela devem ser informados. A Figura 2.9 mostra a tela *Parâmetros da Extração de Características.*

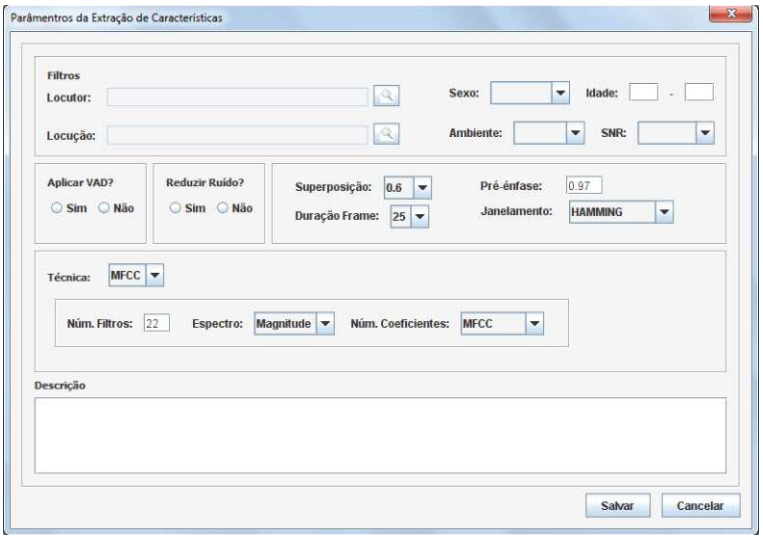


Figura 2.9. Tela Parâmetros de Extração de Características

2.3.7. Classificação de Padrões

Para realizar a classificação dos padrões das características extraídas no projeto o usuário deve clicar no botão *Incluir* na aba *Classificação de Padrões* e preencher os campos informados na tela *Parâmetros da Classificação de Padrões*. Nesta tela, o usuário encontra alguns parâmetros necessários à classificação dos padrões. Caso deseje filtrar apenas algumas amostras de áudio a serem classificadas, pode utilizar as opções de Filtro disponibilizadas no topo da tela. Alguns parâmetros solicitados para a classificação são fixos como: Percentual de Treino, de Teste e *Threshold* (limiar de aceitação de reconhecimento). A depender da técnica escolhida outros parâmetros exclusivos dela devem ser informados. A Figura 2.10 mostra a tela *Parâmetros da Classificação de Padrões.*

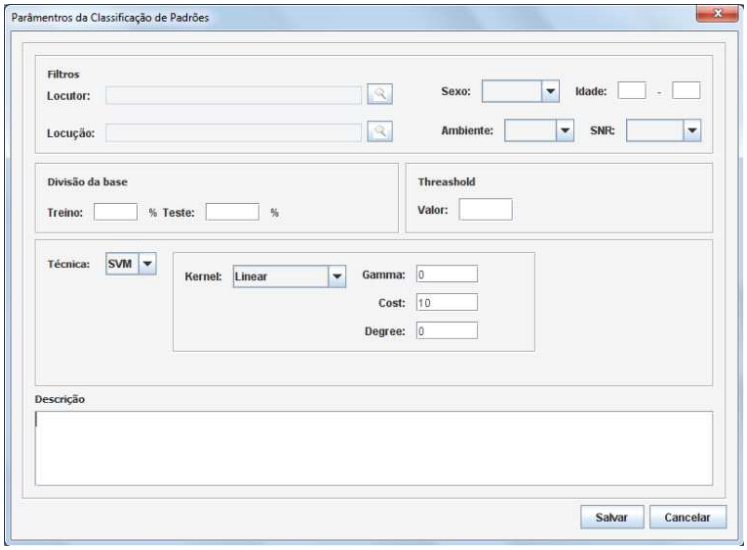


Figura 2.10. Tela de cadastro de classificação de padrões

2.3.8. Geração do Motor

Um motor de fala consiste no agrupamento de alguns artefatos produzidos ao longo do processo construção do motor. Esses artefatos variam de acordo com o tipo de projeto escolhido pelo usuário. Para gerar o motor de fala o usuário deve clicar no botão *Incluir* na aba *Geração do Motor* e preencher os campos informados na tela *Geração do Motor.* Nesta tela, o usuário encontra os parâmetros, nome do motor, arquivo de configuração e de modelos acústicos e, a depender do tipo de projeto escolhido, outros parâmetros exclusivos dele devem ser informados. A Figura 2.11 mostra a tela *Geração do Motor.*

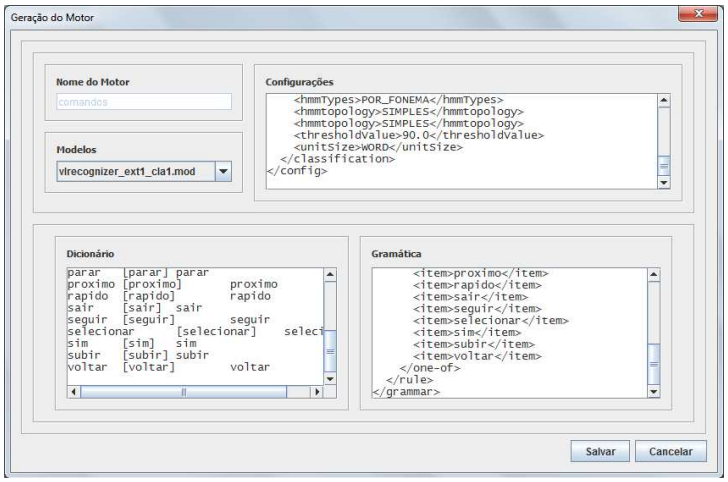


Figura 2.11. Tela de cadastro de motor

Ao final do processo de geração do motor, o usuário deve observar os arquivos gravados na pasta *engine* do projeto. É esta pasta que junto ao FIVE_API leva a interface de voz ao nível de aplicação. A Figura 2.12 mostra o conteúdo da pasta *engine*.

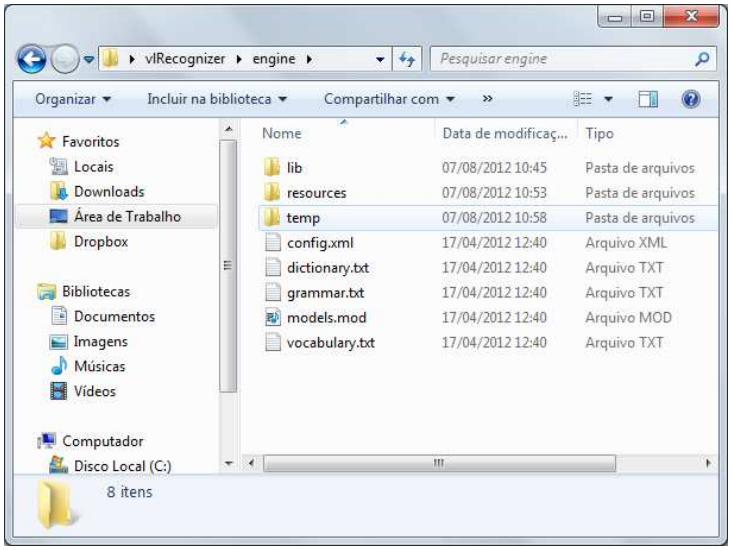


Figura 2.12. Conteúdo da pasta engine

2.4 Exemplos de Motores

Com o propósito de otimizar a demonstração de instanciação de motores em aplicações de exemplo, decidimos utilizar dois motores previamente desenvolvidos. Estes motores fazem parte do trabalho de Maciel (2012). As seções a seguir apresentam os parâmetros e os resultados utilizados na criação de um motor de reconhecimento de fala isolada e de um motor de síntese de voz.

2.4.1. Motor de Reconhecimento de Fala

Para validar os resultados dos motores de fala construídos usando o FIVE, decidiu-se pela criação de um motor de reconhecimento de palavras isoladas. 40 locutores voluntários (30 masculinos e dez femininos) com faixa etária média de 23,5 anos, da região nordeste do Brasil foram selecionados e 20 palavras isoladas, que representam comandos de controle, foram escolhidas para fazer parte do motor de fala.

O processo de aquisição das amostras de áudio ocorreu com a gravação de cinco amostras por locutor para cada uma das palavras, com a frequência de amostragem de 8 kHz, 16 bits por amostra, mono, em ambiente com ruído controlado. Um total de 4000 amostras foi gravado e, após uma averiguação amostral foram eliminadas algumas gravações de má qualidade, restando 3.933 arquivos.

Em seguida, realizou-se a extração de características utilizando o algoritmo MFCC padrão, o algoritmo MFCC seguindo o formato do HTK e, por fim, o algoritmo MFCC baseado no padrão ETSI. Depois de vários testes, numa escala intervalar de parâmetros empíricos, achou-se a configuração paramétrica ideal: pré-ênfase de 0,97, superposição de 60%, duração do frame de 25 milissegundos, e um banco de 22 filtros.

Por fim, realizaram-se os experimentos de classificação dos padrões utilizando as técnicas HMM e SVM. Para os experimentos com HMM, as características utilizadas foram as MFCC_HTK e HTK_ETSI e os modelos fonéticos adotados foram: palavra inteira, fonemas e trifonemas. Para os experimentos utilizando SVM as características utilizadas foram as MFCC e as funções Kernel utilizadas foram: Linear, Polinomial, Sigmoide. A Tabela1.1 mostra as melhores taxas de acertos obtidos nesses experimentos.

Tabela 1.1. Taxas de acerto para reconhecimento de fala.

Técnica	Unidade Fonética	Taxa de acerto
HMM-HTK	Palavra	88,12%
	Fonema	95,72%
	Trifonema	87,89%
HMM-ETSI	Palavra	96,21%
	Fonema	97,28%
	Trifonema	89,10%
SVM	Linear	98,30%
	Polinomial	98,30%
	Sigmoide	97,67%

2.4.2. Motor de Síntese Voz

A validação dos resultados dos motores de síntese foi realizada em duas etapas: a primeira consistiu no treinamento para obtenção dos modelos acústicos de duas vozes sintéticas (uma masculina e uma feminina), e a segunda consistiu numa pesquisa de opinião para avaliar a qualidade dessas vozes.

A etapa de treinamento iniciou com a seleção de dois locutores profissionais, com tratamento fonoaudiológico adequado para evitar sotaques e regionalismos, e com a seleção de 800 locuções foneticamente balanceadas, que foram gravadas pelos locutores em um ambiente de estúdio profissional. O formato das amostras de áudio foi: frequência de amostragem de 44.100 kHz, 16 bits por amostra e modo estéreo.

O processo de treinamento utilizado no FIVE para a obtenção dos modelos acústicos das vozes sintéticas seguiu o modelo baseado em HMM proposto por Maia (2006). Para isso, algumas adaptações no processo de extração de características foram realizadas, a fim de se obter os parâmetros de aperiodicidade e a frequência fundamental. Também foram realizados ajustes no processamento de linguagem natural a fim de se obter informações contextuais das locuções utilizadas no treinamento.

Finalizada a etapa de treinamento, iniciou-se a etapa de avaliação, com a sintetização de 30 frases foneticamente balanceadas com quantidade de palavras variadas, interrogativas e afirmativas. Estas frases foram sintetizadas com as vozes masculina e feminina baseadas nos modelos de difones do Mbrola (Mbrola, 2012); nos modelos acústicos obtidos com o FIVE; e no sintetizador comercial da Loquendo.

O processo de avaliação foi realizado por 30 voluntários e iniciou com um cadastro para a avaliação do perfil dos entrevistados, necessário para avaliar o nível de percepção deles para realizar a transcrição das amostras de áudio, o segundo passo da avaliação. Nesta etapa foi solicitado aos entrevistados que ouvissem uma sequência de áudios, transcrevessem o conteúdo e avaliassem a qualidade segundo três critérios de qualidade: volume, dicção e pontuação.

A Figura 2.13 apresenta os resultados para as seis vozes analisadas (MM:Mbrola masculino, MF:Mbrola feminino, FF:FIVE feminino, FM:FIVE masculino, LF:Loquendo feminino e LM:Loquendo masculino). Para avaliar a qualidade da transcrição das frases faladas calculou-se o percentual de frases transcritas por inteiro e palavras individuais transcritas corretamente.

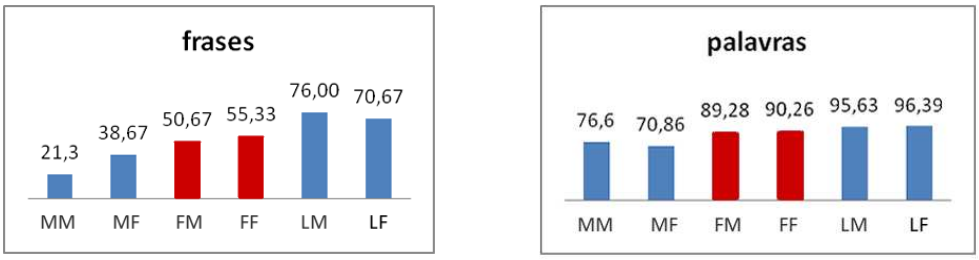


Figura 2.13. Taxa de acerto para transcrição de fala sintética

2.5 Aplicações de Demonstração

Com o objetivo de realizar a validação do processo de instanciação de motores criados a partir do framework FIVE foram desenvolvidas duas aplicações de demonstração que buscaram testar e avaliar o módulo API quanto as suas funcionalidades de reconhecimento de fala e de síntese de voz.

Estas aplicações foram desenvolvidas utilizando o ambiente de desenvolvimento integrado *NetBeans* e, por uma questão de otimização de tempo e esforço, tiveram seu escopo reduzido. A aplicação de reconhecimento de fala é apresentada em duas telas para demonstrar os dois modos de instanciação da API de reconhecimento: a partir de uma amostra de áudio pré-gravada, e a partir da captura de áudio por meio de um microfone. Já a aplicação de síntese de fala consiste numa tela simples onde o usuário digita a frase que deseja sintetizar.

Para criar um projeto de demonstração no *NetBeans* o usuário deve clicar na opção de Menu *Arquivo* e em seguida escolher a opção *Novo Projeto*. Feito isto será mostrada a tela *Novo Projeto*. Nela o usuário deve escolher a opção "Aplicativos Java" e em seguida informar na tela *Novo Aplicativo Java* o nome e a localização do projeto. A Figura 2.14 mostra as telas *Novo Projeto* e *Novo Aplicativo Java*.

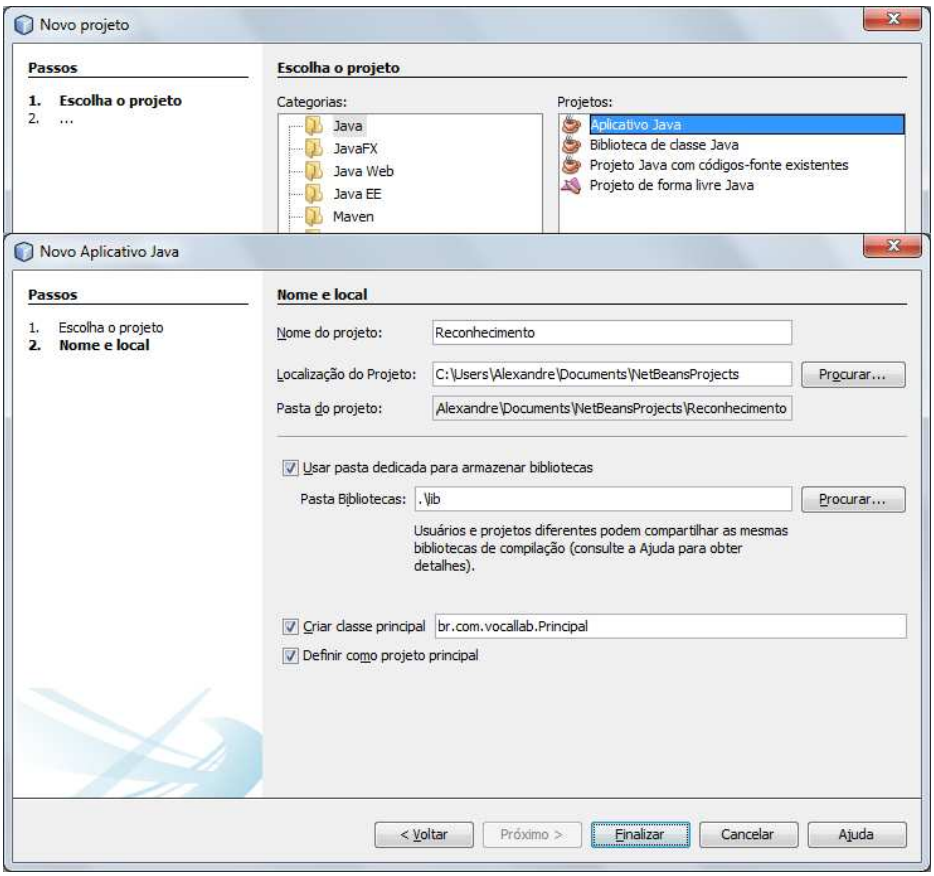


Figura 2.14. Passo a passo da tela Novo Projeto

Criado o projeto, o segundo passo para a construção de uma aplicação de demonstração, que seja capaz de reconhecer ou sintetizar fala, consiste na importação das bibliotecas do FIVE. Para inserir as bibliotecas do FIVE, o usuário deve clicar com o botão direito do mouse sob a pasta *Bibliotecas* na árvore de conteúdo do projeto e escolher a opção *Adicionar JAR/pasta*. Todas as bibliotecas disponibilizadas pelo FIVE devem ser selecionadas. A Figura 2.15 mostra o passo a passo para a inclusão das bibliotecas do FIVE.

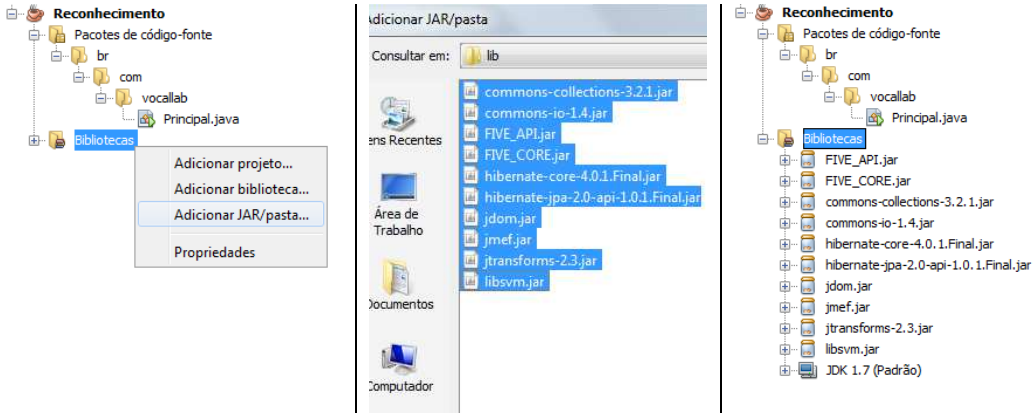


Figura 2.15. Passo a passo para inclusão das bibliotecas do FIVE

Finalizada a importação das bibliotecas, o usuário está livre para codificar sua aplicação. As seções a seguir apresentam trechos de código utilizados para a instanciação da API utilizando as funcionalidades de reconhecimento de fala e de síntese de voz, além de apresentar as telas resultantes das aplicações de demonstração.

2.5.1. Aplicação de Reconhecimento de Fala

A aplicação desenvolvida para validação dos recursos de reconhecimento de fala da API do FIVE levou em consideração o motor de reconhecimento de palavras isoladas construído conforme apresentado na seção 1.4.1. A Figura 2.16 mostra o trecho de código do modo de instanciação baseado em amostras de áudio pré-gravado.

```
try{
   String enginePath = "C:\Users\Alexandre\engine";
   Recognizer rec = new Recognizer(enginePath);

   Result res = rec.recognize(selectedFile);

   taText.append(res.getAnswer ();

catch(RecognizerException ex){
   System.out.println(ex.getMessasge());
}
```

Figura 2.16. Trecho de código para reconhecimento de amostra de áudio

No código apresentado, vê-se a especificação do atributo *enginePath* informando o local que o motor de reconhecimento de fala encontra-se e em seguida a instanciação do objeto *Recognizer*, que recebe o atributo *enginePath*. O processo de reconhecimento

a partir de uma amostra de áudio ocorre por meio da chamada do método *recognize*, passando como parâmetro o arquivo de áudio selecionado, e, por fim, o resultado obtido é apresentado numa área de texto através da chamada do método *getAnswer*. É importante salientar que, durante o reconhecimento, é possível que algum erro cause uma exceção do tipo *RecognizerException*, que pode ser capturada e tratada.

A Figura 2.17 mostra a tela construída para exibição dos resultados deste modo de instanciação. Nesta tela, o usuário busca o arquivo de áudio a ser reconhecido por meio do botão *Buscar*, e em seguida clica no botão *Reconhecer*, que apresenta o resultado no campo *Texto*.

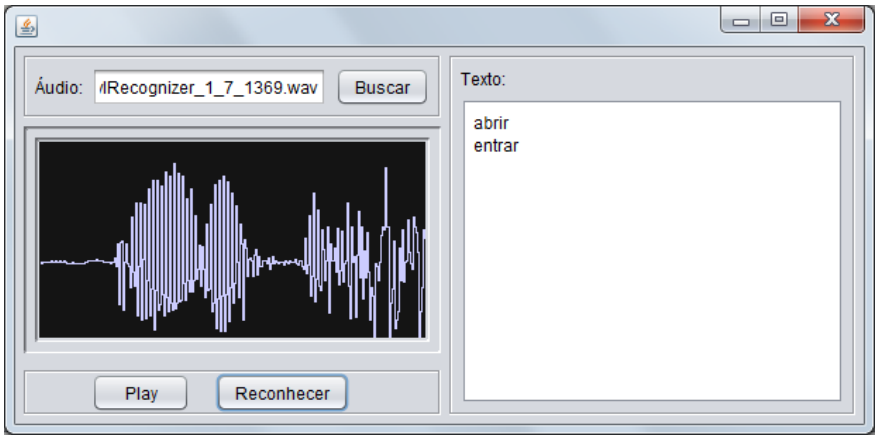


Figura 2.17. Tela de reconhecimento de fala a partir de amostra de áudio

A segunda aplicação para validação dos recursos de reconhecimento de fala da API do FIVE também levou em consideração o motor de reconhecimento de palavras isoladas construído conforme apresentado na seção 1.4.1. A Figura 2.18 mostra o trecho código responsável pelo reconhecimento de fala a partir da captura do som através de um microfone.

```
public class Principal implements RecognizerListener {

   public static void main (String args[]) {
      try{
         String enginePath = "C:\Users\Alexandre\engine";
         Recognizer rec = new Recognizer(enginePath);

         rec.addRecognizerListener(this);
         rec.startRecognition();

      catch(RecognizerException ex){
         System.out.println(ex.getMessasge());
      }
   }

   public void onRecognition(Result res){
      System.out.println(res.getAnswer());
   }
}
```

Figura 2.18. Reconhecimento de fala a partir de microfone

No código apresentado, a classe *Principal* implementa o recurso da classe *RecognizerListener* da API para a criação de eventos. Desse modo, o método *addRecognizerListener* direciona todos os eventos ocorridos na API para a classe *Principal*. Em seguida é apresentada chamada do método *StartRecognition*, responsável pelo abertura do microfone e início de captura de eventos. Como a classe *RecognizerListener* é uma classe abstrata, a classe *Principal* precisa implementar o evento *onRecognition* para que sempre que houver um reconhecimento, o resultado dele (*Result*) seja impresso para o usuário.

A Figura 2.19 mostra a tela construída para a exibição dos resultados deste modo de instanciação. Nesta tela, o usuário clica no botão *Start* e ao falar alguma palavra do dicionário, o sistema a reconhece e apresenta na seção *Palavras Reconhecidas*. Para obtenção de melhores resultados, ao iniciar o microfone o usuário deve esperar alguns segundos para calibração do nível de ruído ambiente.

Figura 2. 19. Tela da aplicação de reconhecimento de fala

2.5.2. Aplicação de Síntese de Voz

A aplicação desenvolvida para apresentação dos recursos de síntese da API do FIVE levou em consideração o motor o motor de síntese de voz construído conforme apresentado na seção 1.4.2. A Figura 2.20 mostra o trecho código responsável pela síntese de voz.

```
try{
   String enginePath = "C:\Users\Alexandre\engine";
   Synthesizer syn = new Synthesizer (enginePath);

   String phrase = taPhrase.getText();
   String speaker = cbSpeaker.getSelectedItem();

   syn.Synthesize(phrase, speaker);

catch(RecognizerException ex){
   System.out.println(ex.getMessasge());
}
```

Figura 2.20. Trecho de código para síntese de voz

Assim como no reconhecimento, na síntese, para instanciar o objeto *Synthesizer*, é preciso informar o caminho (*enginePath*) para acessar o motor. Com o objeto *Synthesizer* criado, o desenvolvedor pode realizar a chamada ao método *synthesize* fornecendo os parâmetros: *phrase* (frase a ser sintetizada) e *speaker* (voz do locutor de que irá narrar a frase). Durante o processo de síntese, algum erro pode causar uma exceção do tipo *SynthesizerException*, que pode ser capturada e tradada.

A Figura 2.21 mostra a tela para exibição dos resultados deste modo de instanciação. Nela, são informados os parâmetros: locutor, escolhido através do campo *Voz*, e a frase a ser sintetizada. Com os parâmetros preenchidos, o usuário clica no botão *Falar* e o sistema toca o áudio sintetizado.

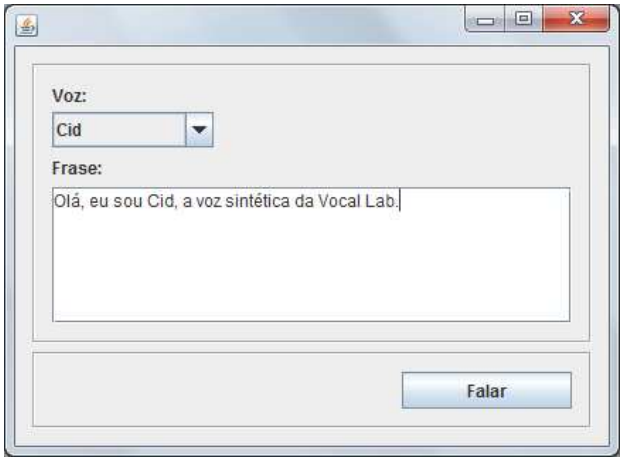


Figura 2.21. Tela da aplicação de síntese de voz

2.6 Considerações Finais

A comunicação oral é, sem dúvida alguma, a forma mais natural de comunicação humana. Em virtude da comunicação homem-máquina se tornar cada vez mais comum, surge uma demanda natural por sistemas capazes de interagir com os usuários de maneira mais prática e intuitiva.

Apesar da evolução da área de interface de voz nas últimas décadas, as aplicações com este tipo de recurso ainda são escassas no mercado e isso se deve a dois fatores: a tecnologia de processamento de voz requer avançados conhecimentos específicos na geração de motores e as ferramentas existentes não se encontram aptas a integrar as principais subáreas da interface de voz.

Este tutorial apresentou o ambiente produtivo FIVE para o desenvolvimento de aplicações com interface de voz que integra motores de fala diversos, com rápida curva de aprendizagem, de modo extensível a várias técnicas e independente de plataforma. Com essa ferramenta espera-se que este tutorial tenha proporcionado aos profissionais da área de Tecnologia da Informação, seja especialistas na área de processamento de voz, seja projetistas de interface, um rápido entendimento deste poderoso recurso, e que em estudos aprofundados posteriormente possam proporcionar a disponibilização de novas aplicações com interface baseada em voz no mercado.

Referências

Campbell, J.P. (1997) "Speaker Recognition: A Tutorial", In: Proceedings of the IEEE, v.85, p.1437-1462.

Cohen, M. H., Giangola, J. P., Balogh, J., Voice User Interface Design, Addison Wesley, 2004.

Cryer, H. and Home, S. (2010) "Review of Methods for Evaluating Synthetic Speech", In: RNIB Centre for Accessible Information (CAI), Technical report # 8.

Duda, R.O., Hart, P.E., Stork, D.G., Pattern Classification, Wiley-Interscience. 2000.

Fechine J. M. (2002) "Reconhecimento Automático de Identidade Vocal utilizando Modelagem Híbrida: Paramétrica e Estatística", Tese (Doutorado em Engenharia Elétrica). Universidade Federal da Paraíba.

Gaikwad, S.K., Gawali, B.W., Yannawar, P. (2010) "A Review on Speech Recognition Technique", In: International Journal of Computer Applications, v.10, No 3.

Gamma, E., Helm, R., Johnson, R. Vissides, J., Design Patterns: Elements of a Reusable Object-Oriented Software, Addison Wesley, 1995.

Holmes, J., Speech Synthesis and Recognition, CRC Press, 2002.

Huang, X., Acero, A., Hon, H.W., Spoken Language Processing – A Guide to Theory, Algorithm, and System Development, Prentice Hall, 2001.

Maciel, A. M. A. (2007) "Investigação de um Ambiente de Processamento de Voz utilizando VoiceXML". Dissertação (Mestrado em Ciência da Computação). Universidade Federal de Pernambuco.

Maciel, A. M. A., Veiga, A., Neves, C., Lopes, J., Lopes, C., Perdigão, F., Sá, L. A. (2008) "Robust Speech Command Recognizer for Embedded Applications". In: Proceedings of International Conference on Signal Processing and Multimedia Application. Porto.

Maciel, A. M. A., Carvalho, E. C. B. (2010) "FIVE – Framework for an Integrated Voice Environment". In: Proceedings of International Conference on Systems, Signal and Image Processing, Rio de Janeiro.

Maciel, A. M. A. (2012) "Investigação de um Ambiente de Desenvolvimento Integrado de Interface de Voz". Tese (Doutorado em Ciência da Computação). Universidade Federal de Pernambuco.

Martins, V. F. (2011) "Avaliação de Usabilidade para Sistemas de Transcrição Automática de Laudos em Radiologia". Tese (Doutorado em Engenharia) Universidade de São Paulo.

Marins, V. F. e Brasiliano, A. (2012) "Interface do Usuário Baseada em Voz como Ferramenta para Promover o Ensino/Aprendizagem de Língua Estrangeira". In: Revista Eletrônica do Alto Vale do Itajaí, vol. 1.

Maia, R. S. (2006) "Speech Synthesis and Phonetic Vocoding for Brazilian Portuguese based on Parameter Generation from Hidden Markov Models". Tese (Doutorado em Engenharia). Nagoya Institute of Technology.

MBROLA, "The Mbrola Project". (2012) Disponível em <http://tcts.fpms.ac.be/synthesis>. Setembro.

Mcglashan, S. *et al*. (2010) "Voice Extensible Markup Language (VoiceXML) 3.0". Disponível em <http://www.w3.org/TR/voicexml30>. Julho.

Mcloughlin, I., Applied Speech and Audio Processing. Cambridge, Cambridge University Press, 2009.

Moran, T. (1981) "The Command Language Grammars: a Representation for the User Interface of Interactive Computer Systems", In: International Journal of Man-Machine Studies. v.15, p.3-50.

O'Shaughnessy, D. (2008) "Automatic Speech Recognition: History, Methods and Challenges". In: Pattern Recognition, v.41, p.2965-2979.

Salvador, V.F.M., Oliveira NETO, J.S., Paiva, M.G. (2010) "Evaluating Voice User Interfaces in Ubiquitous Applications". In: Designing Solutions-Based Ubiquitous and Pervasive Computing Editado por Milton Mendes, Pedro Fernandes. New Issues and Trends. Natal, RN: IGI Global.

Shneiderman, B. (2000) "The limits of speech recognition", In: Communications of the ACM, v. 43, n. 9, p. 24-27.

Stehman, P. (1997) "Selecting and Interpreting Measures of Thematic Classification Accuracy", In: Remote Sensing of Environment, v.62, p.77–89.

Zukerman, I., Litman, D. (2001) "Natural language processing and user modeling: synergies and limitations". In: User Modeling and User-Adapted Interaction n. 11, p.129-158, Khuwer Academic Publishers.

Capítulo 3

Algoritmos de Clusterização e Python Científico apoiando Modelagem de Usuário

Andrey Araujo Masiero, Leonardo Anjoletto Ferreira e Plinio Thomaz Aquino Junior

Abstract

Python is an interactive, interpreted, object oriented, multiplataform programming language that has various applications in industry and in scientific works. There are many libraries available for Python that aid the development of many applications, including in Human-Computer Interaction, by making use of machine learning algorithms. These algorithms, mainly the data clusterization algorithms, are going to be presented as a way to construct the user model as Personas. Personas may aid both the initial user interface development and its enhancement during use. The moment when Personas will help the developer depends on how the information will be acquired. The information obtained by surveys or application logs may be used in pre or post development. However, the latter requires a working version of the system to collect the usage information. In this tutorial, we show how to capture these kinds of information and how to use them with clustering algorithms so that, by automating the clustering of the collected profiles, we are able develop adaptive interfaces. Lastly, techniques used to acquire Personas will be demonstrated based on dispersion metrics methods and the Python toolkits available to present the results in a graphical manner, so that it can also be analytically compared.

Resumo

Python é uma linguagem de programação interativa, interpretada, orientada a objeto e multiplataforma que possui diversas aplicações no mercado de trabalho e científicas. Existem diversas bibliotecas disponíveis para Python que auxiliam o desenvolvimento das diversas aplicações, inclusive em Interação Humano-Computador, através de algoritmos de aprendizado de máquina. Esses algoritmos, em especial os algoritmos de clusterização de dados, serão apresentados para auxiliar na construção de modelos de usuários do tipo Personas. As Personas podem auxiliar o projetista de interface durante o projeto

desta, ou posteriormente, na melhoria de interfaces que já estão em utilização. O momento em que as Personas irão auxiliar o projetista dependerá de como as informações serão capturadas. Essas informações podem ser capturadas através de surveys *ou logs de aplicação. Esses dois tipos de informação podem ser utilizados em momentos pré e pós projeto, contudo o segundo requer uma versão funcional do sistema para coletar as informações de uso. Ao longo deste tutorial será demonstrado como capturar essas informações e ainda como realizar o tratamento desses dados para aplicar os algoritmos de clusterização para agrupar os perfis coletados. Também será mostrado como automatizar esse processo para que os dados sejam utilizados em interfaces adaptativas. Por fim, serão demonstradas as técnicas para obter as Personas com base em métodos de medida de dispersão utilizadas em estatística dos dados e ainda as ferramentas disponíveis no Python para demonstração dos resultados de maneira gráfica para sua comparação analítica.*

3.1. Instalação do Enthought Python Distribution

O Enthought Python Distribution (EPD) é uma distribuição de bibliotecas e interpretadores de Python reunida pela Enthought, Inc., a empresa que desenvolve e mantém as bibliotecas NumPy e SciPy.

Dentre as versões disponibilizadas pela empresa, o EPD Free oferece grande parte das bibliotecas disponibilizadas nos outros pacotes, porém não provê suporte pela empresa nem acesso a conteúdos como webinars e repositórios para atualizações. Entretanto, é possível obter a distribuição completa utilizando a EPD Academic que, apesar de necessitar de um e-mail de uma instituição de ensino, é fornecido gratuitamente para Windows, Linux e OS X e OS X 10.4 para PowerPC.

O instalador do pacote pode ser obtido no site da empresa,[1], exibido na figura 3.1a ou através da chave EPD Free nos mecanismos de busca disponíveis *on-line*. Na área de *download* é possível realizar o cadastro para o recebimento de notícias e atualizações sobre a distribuição gratuita e mais abaixo localiza-se o botão para escolha da versão que será obtida (figura 3.1b).

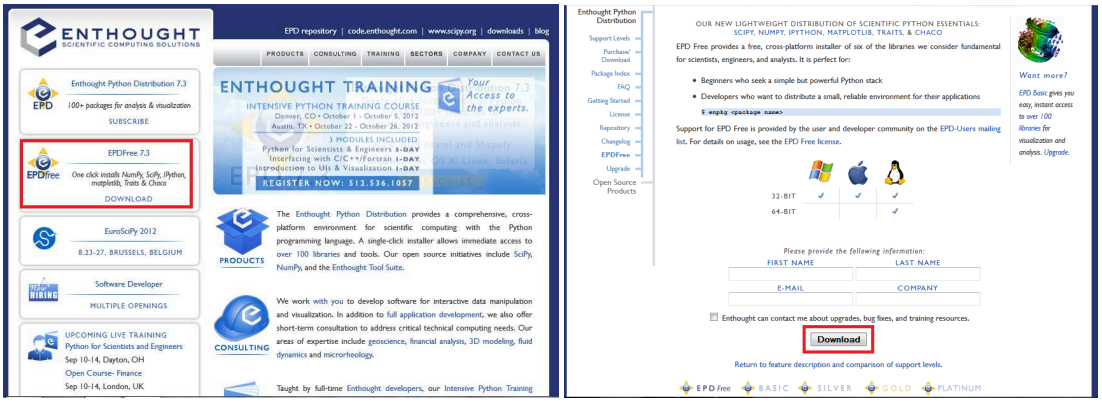
Para realizar o download da versão disponível (32 bits para Windows e OS X e 32/64 bits para Linux – conforme mostrado na figura 3.2), basta escolher a respectiva versão e esperar que o *download* termine. Ao final do *download* do instalador, basta executá-lo seguindo as especificações do sistema escolhido e seguir as instruções fornecidas pelo próprio instalador.

3.2. Programando com Python

Python é uma linguagem de programação de alto nível, assim como C, C++, Perl e Java. Ela também é uma linguagem interpretada, com tipagem dinâmica, interativa e possui suporte a orientação a objetos. Assim como o Java, a interpretação do Python é realizada através de *bytecodes* executados em uma máquina virtual, o que torna os programas portáveis.

A linguagem possui uma sintaxe clara e concisa, tornando o código legível e de

[1]http://www.enthought.com/products/epd_ free.php



(a) Site da Enthought Inc.

(b) Página para o cadastro e *download* da distribuição EPD Free.

Figura 3.1: Site da Enthought, Inc. (a) página principal e (b) página para o *download* da distribuição EPD Free. Último acesso: 18 de setembro de 2012

EPD Free installers

Name	Size	MD5
epd_free-7.3-2-macosx-i386.dmg	72.36 MB	2e0e47e346bd5bb4b57799536a4f705f
epd_free-7.3-2-rh5-x86.sh	91.80 MB	272c82acd37bacc4256af6433e556446
epd_free-7.3-2-rh5-x86_64.sh	101.76 MB	4ae74b49152e28c45c0d1c050d06097e
epd_free-7.3-2-win-x86.msi	89.28 MB	0c2db71b9d30571d49cf5f25c366e0df

Updated: 2012-08-16 15:45:07 UTC

Figura 3.2: Versões disponíveis para *download* do EPD Free. Último acesso: 18 de setembro de 2012

fácil compreensão ao desenvolvedor. Diversos tipos de estruturas, utilizadas em programação de alto nível, compõem o Python, como, por exemplo, listas e dicionários, além de um grande número de bibliotecas e *frameworks* de terceiros que podem aumentar o poder computacional dessa linguagem de programação. Python tem seu código aberto com licença compatível a GPL (*General Public License*) e é mantido pela *Python Software Foundation*.

Outro ponto forte no Python é a possibilidade de integração com outras linguagens de programação como o C. Tais características tornam o Python uma linguagem muito utilizada no mercado de trabalho, codificando *plugins* e *scripts* para ferramentas como OpenOffice, PostgreSQL, GIMP e Blender. Está presente em algumas gigantes do mercado de tecnologia como o Google, Yahoo!, Microsoft e Nokia, e também em empresas de entretenimento como a Disney, na criação de animações 3D.

3.2.1. Primeiro programa em Python

Para executar e escrever programas, o Python possui um *shell* e editor nativo, que se parece em alguns aspectos com o *prompt* de comando. O *shell* do Python é chamado de IDLE. Nele é possível executar expressões e visualizar seus resultados em tempo de execução, e também criar programas inteiros antes de executá-los. Todos os exemplos deste capítulo serão executados a partir do IDLE.

O nosso primeiro programa apresenta a instrução para exibir textos na tela. Vejam a seguir como funciona o comando `print` para exibirmos a frase "Hello, Python".

```
Python 2.7.2 |EPD 7.2-2 (32-bit)| (default, Sep  7 2011,
09:16:50)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more infor-
mation.

>>> print "Hello, Python"
Hello, Python
```

Notem no exemplo que as três primeiras linhas são compostas pelo cabeçalho do IDLE, a linha iniciada por três sinais de maior (> > >) é a instrução do programa e a linha subsequente o resultado obtido. Note que não há necessidade de nenhum ponto nem ponto-e-vírgula para encerrar a linha de código. Em Python, cada linha representa uma única instrução de programa. Caso seja necessário continuar a mesma instrução em uma linha diferente utiliza-se a barra invertida(\) para informar isso ao interpretador.

Alguns outros pontos importantes sobre a sintaxe do Python são: (I) o caractere # é utilizado para realizar comentários no código; (II) existem comentários funcionais que são pré-condições para o interpretador realizar antes de executar o programa, como por exemplo, para determinar o unicode do código "$\#-*-coding:<tipodoencoding>$"; e (III) os blocos de código no Python são delimitados através de endentação e qualquer instrução antes de começar um bloco é finalizada com dois pontos(:).

Ao longo desta seção os exemplos da sintaxe do Python serão demonstrados em detalhes. Continuando os exemplos, será apresentada a atribuição de valores em variáveis. Como não existe tipagem explícita de variáveis em Python, pode-se atribuir qualquer tipo de valor. Para identificar o tipo da variável em algum ponto do código existe uma função chamada `type`. O código a seguir demonstra as atribuições e o uso da função `type`.

```
>>> x = 5
>>> print x
5

>>> y = 3.
>>> print y
3.0
```

```
>>> z = 'texto'
>>> print z
texto

>>> type(x)
<type 'int'>

>>> type(y)
<type 'float'>

>>> type(z)
<type 'str'>
```

Com os pontos básicos apresentados, serão abordados nas próximas subseções os pontos fundamentais sobre a linguagem para que consigamos atingir o objetivo de gerar o modelo de usuário Personas a partir dos algoritmos de *clustering* com Python.

3.2.2. Operadores Matemáticos

Python contém sete operadores matemáticos básicos. Na tabela 3.1 são apresentados cada um dos operadores e as descrições que representam as operações por eles executadas.

Alguns exemplos de uso dos operadores matemáticos podem ser observados no trecho abaixo.

```
>>> 3 + 5
8
>>> 3 ** 2
9
>>> 5.4 // 2
2.0
>>> 8 - 3
5
```

Além dos operadores matemáticos básicos existe a combinação de cada um desses com a atribuição de valores, facilitando alguns tipos de operações, como na tabela 3.2.

Veja o exemplo abaixo:

```
>>> base = 4
>>> expoente = 2
>>> print base
4
>>> base**=expoente
```

```
>>> print base
16
```

Tabela 3.1: Operadores Matemáticos em Python

Operador	Descrição
+	realiza a operação de adição de valores
-	realiza a operação de subtração de valores
*	realiza a operação de multiplicação de valores
/	realiza a operação de divisão de valores
%	realiza a operação de módulo na divisão de valores
**	realiza a operação de potência de valores
//	realiza a operação de divisão com arredondamento para o valor inteiro mais baixo

Tabela 3.2: Operadores Matemáticos Combinados a Atribuição

Operador	Descrição
+=	quando temos uma operação b+=a significa b = b + a
-=	quando temos uma operação b-=a significa b = b - a
*=	quando temos uma operação b*=a significa b = b * a
/=	quando temos uma operação b/=a significa b = b / a
%=	quando temos uma operação b%=a significa b = b % a
**=	quando temos uma operação b**=a significa b = b ** a
//=	quando temos uma operação b//=a significa b = b // 4

3.2.3. Operadores de Comparação

Além dos operadores matemáticos, Python possui operadores de comparação utilizados quando se quer verificar alguma condição durante o programa. Eles são muito utilizados com elementos de controle de fluxo e laços de repetições, que serão apresentados nas próximas subseções. Na tabela 3.3 pode-se verificar quais são os operadores de comparação existentes no Python.

3.2.4. Controle de Fluxo

O controle de fluxo é utilizado para possibilitar tomadas de decisão ao longo de um programa. Tais tomadas de decisão determinam a sequência em que o programa será executado. Dessa forma, controles de fluxo são utilizados para controle do caminho feito pelo programa.

Em Python existe apenas um tipo de declaração para controle de fluxo de um programa, essa declaração é a `if..else`. Ela funciona de forma semelhante a qualquer outra linguagem. Observe o exemplo abaixo.

```
>>> x = 3
>>> y = 3
```

Tabela 3.3: Operadores de Comparação em Python

Operador	Descrição
==	Verifica se o valor de um elemento é igual ou não ao valor do outro
!=	Verifica se o valor de um elemento é diferente ou não ao valor do outro
<>	Semelhante ao operador !=
>	Verifica se o valor do elemento à esquerda é maior do que o valor do elemento à direita
<	Verifica se o valor do elemento à esquerda é menor do que o valor do elemento à direita
>=	Verifica se o valor do elemento à esquerda é maior ou igual do que o valor do elemento à direita
<=	Verifica se o valor do elemento à esuqerda é menor ou igual do que o valor do elemento à direita

```
>>> if x == y:
        print "X e Y possuem o mesmo valor"

X e Y possuem o mesmo valor
```

Note que é importante manter a endentação no bloco que deseja executar quando a condição for verdadeira, e ao final da linha onde é realizada a declaração da condição deve ser colocado o sinal de dois pontos para que o interpretador do Python entenda que existe a necessidade de uma linha endentada abaixo. Existe ainda a variação com a instrução `else`.

```
>>> if x!=y:
        print "X eh diferente de Y"
    else:
        print "X eh igual a Y"

X eh igual a Y
```

3.2.5. Laço de Repetição

Laços de repetição são inseridos em um programa quando se deseja que um bloco de instruções sejam executadas um certo número de vezes. O número de repetição é determinado através de uma condição ou critério de parada. No Python existem dois tipos de laços de repetição: o `while` e o `for`. O `while` repete o bloco de instrução enquanto a condição de parada retornar como verdadeira, como no exemplo abaixo.

```
>>> i = 0
>>> while i < 10:
        print i
        i+=1
```

```
0
1
.
.
.
8
9
```

No programa acima, foi solicitado que o número contido na variável i seja exibido enquanto for menor que 10, e depois disso termine o programa.

O `for` no Python é um pouco diferente das outras linguagens de programação. Nas linguagens de programação em geral, o `for` possui uma condição de parada e um incremento. Já no Python, esse laço de repetição percorre uma lista inteira, atribuindo o valor de cada posição a uma variável auxiliar. Veja no exemplo uma lista de 0 a 9 como se comporta esse tipo de declaração.

```
>>> for item in lista:
        print item

0
1
.
.
.
8
9
```

Perceba que todo o conteúdo dentro da lista é passado um por um para a variável item, e essa é utilizada no bloco de instrução. Para trabalhar com o `for` como nas demais linguagens é necessário o auxílio de uma função chamada `range`. A função `range` possui 3 argumentos: (I) valor de início; (II) valor final; e (III) incremento. O valor final é o único parâmetro obrigatório da função, e tem seu valor real igual a $n-1$, pois o índice no Python inicia-se em 0, como na linguagem C. Um exemplo da função `range` será demonstrado a seguir.

```
>>> for i in range(1,10,2):
        print i

1
3
5
7
9
```

3.2.6. Listas

Listas ou sequências são as estruturas de dados mais básicas que podemos encontrar em Python. Cada elemento da lista possui um índice para que possa ser acessado individualmente em qualquer ponto do programa. O índice do primeiro elemento da lista é zero, do segundo é um e assim por diante. Contudo, é possível acessar um elemento através de um índice negativo, dessa forma é possível acessar o último elemento através do índice (-1).

```
>>> listaDeNomes = ['Jose','Maria','Joao','Leopoldina']
>>> listaDeNomes[1]
'Maria'
>>> listaDeNomes[-3]
'Maria'
```

Uma *string*, em Python, também é uma lista e pode-se acessar qualquer um de seus caracteres através do seu índice. Esse tipo de operação pode auxiliar o desenvolvedor caso possua uma *string* de parâmetro e em determinada posição gostaria de gravar algum valor para comparação, por exemplo.

```
>>> 'Leopoldo'[3]
'p'
```

Para manipulação das listas, existe uma técnica em Python chamada de *slicing*, onde é possível obter intervalos da lista sem a utilização de um laço de repetição. Esse tipo de operação é realizada a partir da inclusão de dois pontos no índice da lista, como nos exemplos a seguir.

```
# Lista utilizada para o exemplo.
>>> exemplo
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Selecionando um intervalo de valores.
# Posição inicial é o índice de número 4.
# Posição final é o índice de número 7, pois é percorrido
# o valor do índice final - 1.
>>> exemplo[4:8]
[5, 6, 7, 8]

# Os índices negativos também funcionam.
>>> exemplo[-3:-1]
[8, 9]

# Dessa forma você inclui todos os elementos antes do
```

```
# índice final.
>>> exemplo[:4]
[1, 2, 3, 4]


# Dessa forma você inclui todos os elementos após o
# índice inicial.
>>> exemplo[2:]
[3, 4, 5, 6, 7, 8, 9, 10]

# Um terceiro parâmetro pode ser inserido para determinar o
# passo da seleção.
>>> exemplo[1:8:2]
[2, 4, 6, 8]

# Também pode ser utilizado para determinar a ordem da
# seleção, tornando-a crescente ou decrescente.
>>> exemplo[8:1:-2]
[9, 7, 5, 3]

>>> exemplo[::-1]
[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
```

Com a apresentação das listas, todos os elementos básicos da linguagem Python foram apresentados nesse capítulo. Nas próximas seções serão apresentadas algumas das bibliotecas científicas do Python para que seja possível a manipulação dos dados e criação do modelo de usuário Persona.

3.3. Numpy - Trabalhando com Números e Matrizes em Python

NumPy[2] é uma biblioteca ou módulo dos mais importantes para o trabalho de pesquisas científicas, pois contém funções de alto desempenho para manipulação de dados multidimensionais, como matrizes. Esse módulo disponibiliza operações matemáticas, lógicas, de ordenação, I/O, álgebra linear e estatística básica, entre outras (Jones et al.).

As funções disponíveis no NumPy foram preparadas para trabalhar com informações numéricas e matrizes com maior desempenho computacional, o que as tornam mais recomendadas. No trecho de código abaixo é apresentado como criar vetores ou *arrays* com o NumPy, porém é necessário informar ao interpretador que o módulo será utilizado. Para isso, utiliza-se a palavra `import` mais o nome do módulo que deseja.

```
import numpy
```

Ainda é possível aplicar um *alias* para esse módulo, facilitando o seu uso durante o programa.

[2]NumPy Website - http://numpy.scipy.org/

```
# np é o alias para o módulo do numpy neste caso.
import numpy as np
```

Agora que o módulo do NumPy foi incluído no programa, pode-se continuar com o exemplo de *arrays*.

```
>>> import numpy as np

# Criando um array com o numpy. Não é diferente da lista
# tradicional, só deve passá-la como parâmetro da função.
>>> a = np.array([2,3,5,6])

# Visualizando o array
>>> a
array([2, 3, 5, 6])

# Existe uma outra maneira de criar este array, determinando
# o tipo de seu conteúdo.
# Para isso, basta utilizar o parâmetro dtype.
>>> a = np.array([1,3,5,7],dtype = float)

# Visualizando o array
>>> a
array([ 1.,  3.,  5.,  7.])
```

O parâmetro `dtype` também é um atributo do *array*, e pode ser utilizado para identificar o tipo de conteúdo que está armazenado durante a execução do programa, como a seguir.

```
# o atributo dtype do array equivale à função type()
>>> a.dtype
dtype('int32')
```

Outras funções possibilitam a conversão do tipo do conteúdo (`astype`) ou verificar o formato do *array* (`shape`). Um ponto de atenção que se deve considerar quando trabalha-se com *array* é a operação matemática. Ao realizar uma operação matemática entre *arrays*, o NumPy simplesmente realiza a operação matemática entre os elementos deste de maneira simples.

```
# Exemplo da operação de multiplicação entre dois arrays
>>> vetor1 = np.array([2,2,2])
>>> vetor2 = np.array([3,3,3])
```

```
>>> vetor1*vetor2
array([6, 6, 6])
```

Note que a multiplicação foi realizada da seguinte maneira: o primeiro elemento de vetor1 com o primeiro elemento de vetor2, segundo elemento de vetor1 com o segundo elemento de vetor2, e assim por diante. Para realizar a multiplicação entre vetores utiliza-se a função `dot`.

```
>>> vetor1.dot(vetor2)
18
```

A função `array` é aconselhada quando se trabalha com vetores. Para trabalhar com matrizes é recomendado utilizar a função `matrix`.

```
>>> A = np.matrix('1 2;3 4')
>>> A
matrix([[1, 2],
        [3, 4]])
```

Ao se criar um objeto com a função `matrix`, pode-se acessar os valores de matriz transposta ou de matriz inversa através dos atributos `T` e `I`, respectivamente.

```
>>> A.T
matrix([[1, 3],
        [2, 4]])

>>> A.I
matrix([[-2. ,  1. ],
        [ 1.5, -0.5]])
```

Contudo, as operações da função `matrix` são inversas à função `array`. Veja como fica o exemplo da operação de multiplicação.

```
# Multiplicação Matricial
>>> A*A
matrix([[ 7, 10],
        [15, 22]])

# Multiplicação Elemento a Elemento
>>> np.multiply(A,A)
matrix([[ 1,  4],
        [ 9, 16]])
```

Além de uma melhor estrutura para trabalhar com vetores e matrizes, o NumPy possui algumas funções de estatística que são úteis durante a modelagem do usuário como, por exemplo, as medidas de dispersão dos dados. As quatro principais que iremos utilizar ao longo do capítulo são a média (`average`), a mediana (`median`) e o desvio padrão (`std`). Exemplos de cada uma dessas funções serão demonstrados a seguir.

```
# Vetor para demonstração das medidas de dispersão
>>> exemplo = np.array([1,2,3,4,5,6,7,8,9,10])

# Média
>>> np.average(exemplo)
5.5

# Mediana
>>> np.median(exemplo)
5.5

# Desvio Padrão
>>> np.std(exemplo)
2.8722813232690143
```

O NumPy possui diversas funções que auxiliam principalmente na manipulação de números e operações matemáticas, desenvolvidas para obter maior desempenho computacional. As funções apresentadas nesta seção são fundamentais para a modelagem de usuário que será apresentada nesse capítulo.

3.4. Scipy - Biblioteca Científica

SciPy[3] é um módulo de Python com foco em cientistas, engenheiros e matemáticos. Ele é uma extensão do NumPy, porém oferece uma gama maior de métodos para estatística, otimizações, integrações, álgebra, processamento de sinais e imagens, dentre outras áreas de utilização de algoritmos (Jones et al.).

Para modelagem de usuário precisamos conhecer algumas funções presentes no SciPy, que podem auxiliar durante a execução do algoritmo ou tratamento e validação das informações coletadas a fim de gerar um modelo de usuário. As funções são encontradas nos módulos `stats`, de estatística, e `linalg`, de álgebra linear.

No módulo `stats` pode-se usar, para a criação do modelo de usuário, a função de moda (`mode`) para verificar qual valor dentre os usuários ocorre com maior frequência. Ainda nesse módulo, são encontradas as funções para auxiliar na realização de teste de hipótese, gerar números aleatórios e tambem a distribuição normal ou binomial dos dados. Veja a seguir o exemplo da moda.

```
# Outra boa maneira para importar um módulo
>>> from scipy import stats
```

[3]SciPy Website - http://www.scipy.org/

```
>>> import numpy as np
>>> exemplo = np.array([3,4,3,3,2,1,5,6,7,4,3,4,6,6])
>>> stats.mode(exemplo)
(array([ 3.]), array([ 4.]))
```

Este outro exemplo apresenta como gerar uma distribuição normal aleatória.

```
>>> r = stats.norm.rvs(size=100)
```

Existem diversas funções no módulo `stats` que podem auxiliar na análise de qualquer grupo de informações. É um complemento do que já existe no NumPy, apresentado na seção 3.3. Outro módulo é o `linalg`, que possui as funções de manipulação de matrizes. Veja o exemplo de cálculo da matriz seno.

```
>>> from scipy import linalg as la
>>> M = np.matrix('1 2;4 5')
>>> la.sinm(M)
array([[-0.31500257,  0.18115826],
       [ 0.36231652,  0.04731395]])
```

O SciPy possui um grande número de módulos que podem ser úteis em diversos tipos de aplicações. Contudo, para o propósito desse capítulo, os exemplos apresentados nesta seção são os mais importantes, dependendo da manipulação que for necessária das informações escolhidas para a modelagem de usuário.

3.5. SciKits: Expandindo o Python com Bibliotecas Científicas

Existem diversas maneiras de expandir as funcionalidades que o Python oferece em sua biblioteca padrão, desde a simples instalação de pacotes cujos instaladores podem ser obtidos nas páginas de seus projetos até a instalação e atualização a partir de gerenciadores de pacotes (como os normalmente encontrados em distribuições Linux).

Algumas dessas bibliotecas foram desenvolvidas com o objetivo específico de complementar o SciPy e oferecer novas funcionalidades em diversas áreas, como processamento de imagens, modelos estatísticos e aprendizado de máquina. O projeto que reúne essas bibliotecas específicas foi dado o nome de *SciPy Toolkits*, mais conhecido como SciKits [4].

Na área de aprendizado de máquina, um dos projetos que fazem parte do SciKits e que possui um desenvolvimento muito ativo é o SciKit-Learn, que disponibiliza diversas técnicas de aprendizado supervisionado e não-supervisionado, validação cruzada e pré-processamento de dados.

Algumas dessas técnicas serão apresentadas de forma mais aprofundada na seção 3.6, mas uma lista extensiva de técnicas oferecida atualmente pelo SciKit-Learn é:

[4]O site oficial do projeto encontra-se em http://scikits.appspot.com/ e as bibliotecas que atualmente fazem parte do projeto estão no endereço http://scikits.appspot.com/scikits

1. Aprendizado Supervisionado: Método dos Mínimos Quadrados, Regressão Linear, Logística e Bayesiana, Gradiente Descendente Estocástico, Perceptron, Análise de Discriminante Linear e Quadrática, Máquinas de Vetores de Suporte (SMV), Processo Gaussiano, Árvores de Decisão e *Naïve* Bayes, entre outros.

2. Aprendizado Não-Supervisionado: Mistura Gaussiana, *Manifold Learning*, K-Means, *Affinity Propagation*, DBSCAN, Cluster Hierárquico, Análise de Componentes Principais (PCA) e Independentes (ICA) e Modelos Escondidos de Markov, entre outros;

3. Transformação de base de dados: pré-processamento de dados (normalização, binarização e escalonamento), extração de informações em texto e imagens e aproximação de kernel (RBF, Chi Quadrado Aditivo), entre outros;

4. Validação Cruzada: *Leave-One-Out*, *Leave-P-Out* e Permutação Aleatória;

5. Base de dados de exemplos instaladas juntamente com a biblioteca, juntamente com a possibilidade de *download* direto do código de certas bases disponíveis na internet.

3.6. SciKit-Learn – Aprendizado de Máquina em Python

O Scikit-Learn é uma biblioteca para Python que oferece uma grande variedade de métodos de aprendizado de máquina, tanto nas áreas de aprendizado supervisionado quanto em aprendizado não-supervisionado.

O objetivo do desenvolvimento do Scikit-Learn é oferecer soluções simples e eficientes para problemas de aprendizado de máquina (Pedregosa et al., 2011). Por se tratar de uma biblioteca que visa a viabilização de reutilização de código de forma acessível (Pedregosa et al., 2011), o Scikit-Learn se integra facilmente com o NumPy, SciPy e Matplotlib.

Apesar do Scikit-Learn possuir funções para a realização de diversos tipos de aprendizado, neste tutorial serão apresentadas apenas três técnicas relacionadas a clusterização de dados (K-Means, DBSCAN e Affinity Propagation) e como podem ser aplicadas à área de IHC para o reconhecimento de Personas.

Fundamentalmente, o Scikit-Learn provê duas formas de realizar a classificação de uma amostra. A primeira é utilizando funções disponíveis que recebem como parâmetros, além da amostra, informações específicas do algoritmo que será utilizado, e retornam a classificação da amostra segundo os parâmetros informados.

A segunda maneira é utilizando as classes disponíveis. Estas se baseiam na classe `estimator` e implementam os métodos `fit()` e (muitas vezes) `predict()`, que são os responsáveis por realizar o aprendizado e a classificação (ou previsão, no caso de algoritmos de aprendizado supervisionado), respectivamente. A única diferença na implementação do método `fit()` entre os métodos de aprendizado supervisionado e não-supervisionado é que, no primeiro, deve-se passar como parâmetros os valores observados e as respostas, enquanto que no segundo passa-se apenas os dados da amostra.

A vantagem de se utilizar classes em vez das funções implementadas no Scikit-Learn é que as funções somente realizam a classificação durante o processo de aprendizado e não realizam a predição de um novo dado, enquanto que as classes armazenam as informações pertinentes à classificação realizada e podem ser utilizadas para verificar a qual grupo pertence um novo dado que não está contido na amostra utilizada para o treinamento.

3.6.1. Clusterização de Dados Usando o Scikit-Learn

O objetivo da clusterização é identificar similaridades entre dados, de forma que seja possível agrupar automaticamente os dados que pertençam à mesma classe (Marsland, 2009).

3.6.1.1. K-Means

Proposto por Lloyd (1982); MacQueen et al. (1967), o K-Means realiza a classificação de uma amostra através da relação entre a distância Euclidiana (equação 1) do centro de um cluster existente e cada ponto da amostra.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{1}$$

Por se tratar de um algoritmo iterativo, a cada ponto da amostra que é classificado, o K-Means realiza a atualização da posição de todos os centros de clusters existentes. Sendo o centro do cluster representado pela média de todos os pontos existentes no cluster, a nova posição deste centro pode ser calculada através da equação $\mu_k = \frac{\sum_{i=1}^{n_k} x_k}{n_k}$.

Uma vez que cada ponto pode ser relacionado com todos os centros de clusters existentes, o K-Means classifica um novo ponto não incluso na amostra inicial como pertencente ao cluster que possui a menor distância Euclidiana, como na equação $d_P = \min_k (x_P, \mu_k)$.

Uma das dificuldades na utilização do K-Means é a escolha da quantidade de núcleos k necessária para a realização da classificação e que deve ser definida previamente. Apesar de ser um valor necessário para a utilização do algoritmo, não há uma maneira empírica de calcular o valor de k que garanta a melhor classificação possível.

Desta forma, é possível obter grupos vazios, nos quais não há nenhum elemento, ou grupos que poderiam ser classificados como apenas um, mas que pela quantidade de núcleos foram divididos em dois ou mais. Da mesma maneira, um valor de k baixo pode fazer com que grupos que deveriam ser separados sejam classificados como um grupo único.

Entretanto, utilizar o K-Means oferecido pelo Scikit-Learn é simples, necessitando apenas de uma instância do objeto encontrado em `sklearn.cluster.KMeans`, para o qual pode-se passar o número de núcleos (k) que se deseja encontrar (por padrão, a biblioteca assume 8 clusters). Para realizar o aprendizado sobre a amostra obtida, utiliza-se o método `fit(amostra)`, em que se passa os valores de amostra medida sem os resultados da classificação (quando conhecidos). Para obter o grupo em que um novo

ponto pode ser classificado, utiliza-se o método `predict()`, passando como parâmetro o(s) ponto(s) que se deseja classificar.

A seguir, encontra-se um exemplo de utilização do K-Means oferecido pelo Scikit-Learn. Na seção 3.6.3 são apresentados exemplos da utilização destes métodos para a classificação dos dados, juntamente com os outros métodos que serão apresentados ao longo do texto, bem como uma forma de carregar a amostra armazenada em um arquivo texto para utilizar nos algoritmos demonstrados.

```
#importa o K-Means implementado pelo Scikit-Learn
from sklearn.cluster import KMeans

#instancia um objeto do K-Means utilizando 3 núcleos (k=3)
kmeans = KMeans(k=3)

#considerando que a variável "amostra" possui os valores
#medidos, realiza-se o treinamento utilizando o método fit
kmeans.fit(amostra)

#classificação de um novo ponto P não existente na amostra
#inicial utilizada para realizar o treinamento
kmeans.predict(P)

#um método interessante disponibilizado pelo Scikit-Learn
#no K-Means é o fit_predict, que realiza o treinamento
#utilizando a amostra informada e retorna o resultado da
#classificação nela realizada
kmeans.fit_predict(amostra)
```

3.6.1.2. DBSCAN

Uma outra maneira de classificar os dados obtidos é a partir da densidade dos pontos da amostra, em que é possível considerar que existe uma densidade maior de pontos dentro do cluster do que fora dele (Ester et al., 1996). Entretanto, é possível que exista ruído fora do cluster, interferindo no treinamento do algoritmo de clusterização. Para tentar resolver este problema, Ester et al. (1996) propuseram o *Density Based Spatial Clustering of Applications with Noise* (DBSCAN).

Ao contrário do K-Means, o DBSCAN não precisa que o número k de clusters seja definido previamente, pois este definirá a quantidade de grupos em que a amostra será dividida à medida que itera sobre os dados utilizados para a classificação. Entretanto, duas outras informações devem ser estimadas previamente para que seja possível realizar a classificação com o DBSCAN: Eps e $MinPts$.

O valor de Eps indica a distância máxima que um ponto pode estar para ser considerado do mesmo grupo, e $MinPts$ (nomeado de `min_samples` no algoritmo imple-

mentado no Scikit-Learn) o número mínimo de pontos na vizinhança para que o dado atual possa ser considerado o centro do grupo (*core point*).

A seguir encontra-se um exemplo classificando a mesma base utilizada no exemplo do K-Means, porém utilizando o DBSCAN.

```
#importa o DBSCAN implementado pelo Scikit-Learn
from sklearn.cluster import DBSCAN
#distância euclidiana disponível no Scikit-Learn
from sklearn.metrics import euclidean_distances as \
 euclidean

#instancia um objeto do DBSCAN
dbscan = DBSCAN()

#considerando que os dados estão armazenados na
#variável "amostra", calcula distância entre os
#valores da amostra
damostra = euclidean(amostra)

#realiza-se o treinamento utilizando o método fit
#usando Eps e MinPts padrões da classe
dbscan.fit(damostra)

#exibe resultado da classificação
dbscan.labels_
```

3.6.1.3. *Affinity Propagation*

A terceira e última técnica de clusterização que será apresentada neste trabalho é conhecida como *Affinity Propagation*, proposta por Frey and Dueck (2007).

Da mesma forma que o DBSCAN, o *Affinity Propagation* não precisa ser informado do número de núcleos que se deseja formar sobre a base de dados que está sendo classificada. Porém, ao contrário dos dois métodos apresentados anteriormente, o *Affinity Propagation* utiliza uma matriz de similaridade (afinidade) calculada para cada par de pontos pertencentes à amostra para realizar a classificação, não utilizando os valores originais pertencentes à amostra.

Uma das maneiras de calcular a matriz de similaridade necessária para realizar a classificação com o *Affinity Propagation* é utilizando a distância Euclidiana, já apresentada na equação 1.

Uma vez que a afinidade entre os pontos tenha sido calculada, o *Affinity Propagation* itera sobre os pontos em busca do *exemplar* (que o centro do grupo encontrado pelo *Affinity Propagation*) e então marca a distância entre o *exemplar* e os outros pontos que pertencem ao grupo.

Assim como apresentado para o DBSCAN, o exemplo a seguir introduz uma forma de utilizar o *Affinity Propagation* a partir dos mesmos dados utilizados para realizar a classificação com o K-Means. Neste caso, foi utilizada a distância Euclidiana para calcular a matriz de afinidade.

```
#importa o Affinity Propagation do Scikit-Learn
from sklearn.cluster import AffinityPropagation
#importa o cálculo da distância euclidiana
from sklearn.metrics import euclidean_distances as \
 euclidean

#instancia um objeto do Affinity Propagation
ap = AffinityPropagation()

#calcula-se a disância euclidiana entre os
#pontos existentes na amostra
damostra = euclidean(amostra)

#realiza-se o treinamento utilizando o metodo fit
#com os dados contidos na variável amostra "damostra"
ap.fit(damostra)
```

A principal função do Scikit-Learn é oferecer uma vasta biblioteca de algoritmos de aprendizado de máquina que sejam fáceis de utilizar. Entretanto, o Scikit-Learn também provê métodos para transformação dos dados, para que se torne mais fácil trabalhar com eles.

3.6.2. Transformação de Dados

Em todos os algoritmos disponibilizados pelo Scikit-Learn, é necessário que certas informações lhes sejam passadas para que o aprendizado possa ser feito. Em alguns desses casos, é importante que os dados sejam transformados (normalizados, binarizados ou tenham as médias subtraídas) para que o resultado obtido se torne mais simples de ser analisado. Para realizar essas tarefas, o Scikit-Learn oferece algumas funções dentro de seu módulo `preprocessing`.

Assim como os algoritmos de aprendizado, o módulo de pré-processamento de dados oferece objetos e funções. A principal diferença entre os dois métodos de pré-processamento é o objetivo final de cada um: enquanto as funções oferecem uma forma rápida de transformar os dados, os objetos retêm as informações da transformação inicial, permitindo que novos dados possam ser pré-processados mantendo as mesmas métricas que a transformação inicial.

Os três métodos existentes no Scikit-Learn que serão apresentados são o `scale` (que centra os dados na média com um valor de variância unitário, o `normalize` que realiza a normalização dos dados e os apresenta com valores entre 0.0 e 1.0. O último método apresentado é o `binarize`, que recebe dois parâmetros: as amostras que se deseja transformar e o valor de limiar que será utilizado para dividir os dados entre 0 e 1.

Assim como nas seções anteriores, o exemplo a seguir realiza as transformações na mesma base de dados utilizada anteriormente.

```
#importa biblioteca de pré-processamento
from sklearn import preprocessing
from numpy import mean

#considerando que os dados desejados estão armazenados
#na variável "amostra"

#realiza a normalização
norm = preprocessing.normalize(amostra)

#realiza o escalonamento
scale = preprocessing.scale(amostra)

#realiza a binarização
binarizado = preprocessing.binarize(amostra, mean(amostra))
```

Utilizando as funções como mostrado no exemplo anterior, tem-se os resultados armazenados nas variáveis `norm`, `scale` e `bin`. Porém, se for desejado que uma nova amostra seja transformada da mesma forma que a anterior, é necessário que se utilize objetos no lugar nas funções.

Assim como os algoritmos apresentados anteriormente, as classes de pré-processamento de dados são baseadas na classe `Transformer`, e implementam dois métodos: `fit()` e `transform()`. A seguir, será realizado o mesmo pré-processamento que no exemplo anterior, porém uma nova amostra também será transformada após o pré-processamento da primeira.

```
#importa biblioteca de pré-processamento
from sklearn import preprocessing
from numpy import mean  # para cálculo da média

#considerando que os valores com os quais se
#deseja realizar o treinamento estão armazenados
#na variável "amostra"

#realiza a normalização
normalizer = preprocessing.Normalizer()
normalizer.fit(amostra)

#realiza o escalonamento
scaler = preprocessing.Scaler()
scaler.fit(amostra)
```

```
#realiza a binarização
binarizer = preprocessing.Binarizer(mean(amostra))
binarizer.fit(amostra)

#pré-processamento de uma segunda amostra a partir da
#transformação realizada na primeira

#considerando que os novos dados foram carregados
#na variável "amostra2"

#normalização da segunda amostra a partir da primeira
norm = normalizer.transform(amostra2)

#escalonamento da segunda amostra a partir da primeira
scale = scaler.transform(amostra2)

#binarização da segunda amostra a partir da primeira
binarizado = binarizer.transform(amostra2)

#assim como o K-Means, os objetos de pré-processamento
#apresentam o método fit_transform, que
#transforma os dados e retorna o resultado
norm1 = normalizer.fit_transform(amostra)
scale1 = scaler.fit_transform(amostra)
bin1 = binarizer.fit_transform(amostra)
```

Nesta seção foram apresentados alguns algoritmos de clusterização disponibilizados pelo Scikit-Learn juntamente com o módulo de pré-processamento. A seguir, será apresentado um exemplo, utilizando a mesma base de dados, que reúne todas as técnicas que foram apresentadas.

3.6.3. Exemplo de Utilização do Scikit-Learn

Considerando que os dados referentes à amostra foram carregados na variável `amostra`, o exemplo a seguir realiza o treinamento dos algoritmos de classificação e a previsão de alguns pontos utilizando os métodos descritos anteriormente, juntamente com o pré-processamento dos dados da amostra.

```
#importa os módulos necessários
from sklearn import preprocessing  # normalização
#algoritmos de clusterização apresentados
from sklearn.cluster import KMeans
from sklearn.cluster import DBSCAN
from sklearn.cluster import AffinityPropagation
#importa o cálculo da distância euclidiana
```

```
from sklearn.metrics import euclidean_distances as \
euclidean

#carrega os dados do arquivo 'data.txt' e formata
f = open('data.txt')
temp = f.readlines()
f.close()

amostra = []
for i in temp:
    amostra.append(i.split())
amostra = [[float(i), float(j)] for i, j in amostra]

#realiza a normalização na amostra
norm = preprocessing.Normalizer()
amostra_norm = norm.fit_transform(amostra)
damostra = euclidean(amostra_norm)

#instancia objetos para classificação
kmeans = KMeans(k=3)
dbscan = DBSCAN()
ap = AffinityPropagation()

#treinamento dos algoritmos de classificação
kmeans.fit(amostra_norm)
dbscan.fit(damostranorm)
ap.fit(damostra)

#resultado da classificação
kmeans.predict(amostra_norm)
dbscan.labels_

#prevê os pontos de uma segunda amostra
#pré-processada a partir da primeira amostra

#carrega os dados do arquivo 'data_110.txt' e formata
f = open('data.txt')
temp = f.readlines()
f.close()

amostra2 = []
for i in temp:
    amostra2.append(i.split())
amostra2 = [[float(i), float(j)] for i, j in amostra2]

#normalização da segunda amostra
```

```
amostra2_norm = norm.transform(amostra2)

#classifica nova amostra a partir do aprendizado
#feito na primeira amostra
kmeans.predict(amostra2_norm)
```

Nesta seção foram apresentadas algumas técnicas de classificação e pré-processamento de dados, porém tão importante quanto a utilização dos algoritmos é a visualização dos dados e resultados. Na próxima seção será apresentada a biblioteca Matplotlib, que permite a geração de diversas formas de gráficos, inclusive de animações.

3.7. Personas

Persona é uma técnica que ajuda a entender e aplicar o perfil de usuários no processo de criação de produtos eficientes e efetivos. Na psicologia, Jung (1975) definiu personas na psicologia como a aparência que as pessoas assumem de acordo com suas convenções coletivas. Definiu que persona é a forma como o indivíduo se apresenta ao mundo, expondo seu caráter no relacionamento entre indivíduos, sendo que uma pessoa pode assumir várias personas, de acordo com seu papel. Declara personas como arquétipos da conformidade (Jung and Rocha, 1990; Jung and do Amaral, 1986).

O conceito de personas foi popularizado por Cooper (1999), que definiu personas como: “Personas não são pessoas reais... são arquétipos hipotéticos dos usuários atuais... definidos com rigor e precisão significativos”. Cooper et al. (2007) se referiu a personas como arquétipos hipotéticos porque não há como provar que as personas representam realmente os usuários atuais antes do lançamento de um produto.

Personas também foram são definidas como modelos descritivos de usuários que representam um grupo de pessoas reais e suas características (Aquino Junior and Filgueiras, 2005). Esses modelos agrupam informações sobre conhecimento, habilidades, motivações e preocupações, descrevendo os padrões de ação que um grupo de usuários tem sobre um determinado projeto, permitindo a troca de informações entre os perfis de usuário e a equipe do projeto (Pruitt and Grudin, 2003). Personas podem servir como ferramentas de apoio para documentação de soluções de projeto de interface, armazenando e organizando a representação de usuários (Aquino Junior, 2008). Neste mesmo sentido, Bagnall (2007) descreve a técnica de Personas como uma poderosa ferramenta de design e troca de informações auxiliando a equipe de criação de sistemas interativos, tanto na área de marketing quanto nas equipes de engenharia do conhecimento.

Neste tutorial definimos as personas como modelos criados para representar pessoas reconhecidas por algum processo de estudo do perfil de usuários alvos de um produto ou serviço. As personas ajudam a estabelecer um consenso entre os projetistas e *stakeholders* sobre as necessidades e motivações do público-alvo. Esta técnica produz um conjunto de personagens fictícios ou representações reais identificadas de acordo com as definições iniciais do projeto.

Esses personagens possuem nome e descrições detalhadas sobre suas preferências, comportamentos, expectativas, objetivos, cenários de uso associados, histórias ilus-

trativas, e qualquer informação adicional que ajude o projetista a compreender melhor esse segmento de usuário. Diversos trabalhos demonstram formatos diversificados de representação descritiva das personas em diversas áreas de aplicação, como por exemplo, governo eletrônico (Filgueiras et al., 2005), mercado de celulares (Cirelli et al., 2010), educação (Aquino Junior, 2010, 2012), padrões de interface (Aquino Junior, 2008; Aquino Junior and Filgueiras, 2008), entre outros.

A divulgação das personas pode ocorrer por publicação dos personagens em sites, blogs, cartões de visita, pôster ou qualquer meio que facilite o acesso à informação, tornando o conjunto de personas mais próximo do dia a dia da equipe de projeto e desenvolvimento. De acordo com Cooper (1999), "um dos benefícios mais visíveis das personas é prover uma face humana que coloca o foco da empatia nas pessoas representadas por dados demográficos".

Existem várias maneiras de criar personas. Pruitt and Adlin (2006) apresentaram um ciclo de vida para criação, uso e descarte de personas. Trata-se de um ciclo completo de aplicação de personas no desenvolvimento de produtos. Neste trabalho, considera-se apenas um processo de concepção de personas, com o objetivo de evidenciar a vantagem de integração do Python científico na modelagem de usuários.

Desta maneira, as seguintes fases são consideradas para a concepção de personas: "Planejamento", "Pesquisa", "Criação e Modelagem".

a) Fase de Planejamento:

No planejamento devem-se compor as suposições, as condições e a busca por informações. Define-se o conjunto de suposições possíveis ou não sobre os usuários, e da quais se tirarão as consequências a serem verificadas na fase de pesquisa. Define-se o conjunto de condições que se toma como ponto de partida para desenvolver o raciocínio com base no problema (projeto). Busca-se o maior número de informações possíveis dos envolvidos com o produto alvo. Podem ser informações provenientes de campanhas publicitárias, banco de dados de clientes, registros de suporte, etc.

Com as informações de planejamento defina: (a) o que as pessoas fazem antes, durante e depois de realizarem as tarefas?; (b) esboce os tipos de papéis que as pessoas assumem para realizar as tarefas; (c) suponha o que essas pessoas gostam de fazer, quais são seus hábitos e preferências; (d) agrupe perfis fictícios com as informações produzidas até esse momento; (e) valide com o cliente quais são os grupos mais importantes de acordo com a estratégia da empresa para esse projeto.

b) Fase de Pesquisa:

Deve-se aplicar iniciativas para coleta de dados com base nas definições da fase anterior. A pesquisa pode ser qualitativa ou quantitativa. A pesquisa qualitativa tem caráter exploratório, estimulando os entrevistados a se expressarem livremente sobre assuntos, conceitos e objetivos. Trata-se de uma pesquisa que auxilia a busca por justificativas para questões em geral, apoiando a interpretação de hipóteses. A amostra da pesquisa qualitativa geralmente é pequena, pois não há preocupação em mapear uma população. A pesquisa qualitativa pode ser executada com discussões em grupo, análise de documentos, mesa-redonda, entrevistas de profundidade, questionário baseado em roteiros.

A pesquisa quantitativa busca a representação numérica de informações pertencentes ao domínio, apurando opiniões e atitudes explícitas dos entrevistados. Essa pesquisa é apoiada por questionários em diversos formatos de mídias. O objetivo é medir e apoiar o teste de hipóteses, pois os resultados são concretos e menos suscetíveis a erros de interpretação. As pesquisas quantitativas permitem estabelecer métricas e índices para comparar os resultados das pesquisas em função do tempo. Pode ser aplicada para informações quantificáveis com inferências a partir de amostras numéricas. A amostra exige um grande número de entrevistados para garantir maior exatidão dos resultados que mapeiam uma população. A pesquisa quantitativa pode ser executada por intermédio de entrevista por atributos (idade, escolaridade, faixa salarial, etc) com questionários padronizados ou análise de *logs* de acesso, ambos possibilitando a preparação de tabelas, quadros e relatórios de uma população. A pesquisa quantitativa permite fazer a análise demográfica, estudando a dinâmica populacional da sociedade ou um grupo específico, considerando as dimensões, as estatísticas, as distribuições e as estruturas da população.

c) Fase de Criação e Modelagem

Para criar personas, procura-se agrupar uma série de comportamentos, motivações e características dos usuários, coletadas através de pesquisas. Desta maneira, com as informações preliminares coletadas na fase de planejamento e os dados coletados durante a fase de pesquisa, é possível identificar padrões de comportamento, de desejos e de necessidades que diferenciam grupos de pessoas. Esse padrão identificado é usado para definição da persona. Esses padrões podem ser extraídos das seguintes variáveis:

- Habilidades: descreve as capacidades do usuário de acordo com o contexto alvo analisado
- Aptidões: descreve os conhecimentos que o usuário possui quanto à tecnologia e tarefa envolvida
- Motivações: descrevem quais são as motivações que o usuário possui com o contexto e produto previsto
- Visão: descreve as opiniões do usuário sobre o produto ou serviço considerado
- Tarefas: descrevem quais são os papéis que o usuário assume no produto ou serviço. A informação é complementada com descrição de "como" as tarefas são executadas
- Demografia: dados que caracterizam o perfil etnográfico do grupo de pessoas, como experiência computacional, escolaridade, idade, sexo, profissão, local de trabalho, dados de moradia, dados culturais, etc.

As descrições das personas podem ter maior ou menor grau de detalhamento, de acordo com as condições de realização do projeto e de acordo com o seu escopo. Mas alguns elementos devem constar das descrições para ajudar os projetistas a tomar decisões sobre o produto ou serviço Aquino Junior (2008). Considera-se que a modelagem de personas deve possuir:

- Nome: objetivo de facilitar a referência ao modelo específico, criando uma associação com pessoas reais e suas características. Algumas equipes preferem aplicar apelidos relacionados à ação do usuário ou considerando traços fortes de sua personalidade;

- Imagem: em conjunto com o nome ajuda a equipe a identificar e memorizar o personagem. Algumas equipes usam imagens de pessoas reais que participaram da pesquisa. Outras usam caricaturas produzidas especialmente para ajudar a lembrar de traços da personalidade da persona;

- Pensamentos e afirmações: apresenta pensamentos expressados pelos usuários reais, assim como afirmações realizadas espontaneamente, ajudando a conhecê-lo quanto a suas atitudes e ações;

- Informações demográficas: a narrativa sobre os atributos etnográficos (culturais) e fatores demográficos como localização (onde mora e trabalha), perfil estatístico (idade, sexo, tamanho da família, renda, ocupação, educação), perfil psicológico genérico (hábitos, características de personalidade, atitude) (vide Figura 3.3 – indicação 1);

- Necessidades: descreve as necessidades que motivam a pessoa no uso do produto ou serviço (vide Figura 3.3 – indicação 2);

- Dificuldades: representa as dificuldades que normalmente esse grupo de pessoas encontra no uso de sistemas interativos. Podem-se descrever os recursos com o qual o usuário interage, desde funcionalidades da interface e de bancos de dados à demanda por determinado conteúdo. Descreve familiaridade de uso de computadores e da internet (vide Figura 3.3 – indicação 3);

- Comportamento: apresenta uma história que aborda o comportamento deste grupo de pessoas ao lidar com pessoas e computadores. Apresentam características afetivas ou sociais comuns a alguns tipos de usuários, motivações que os levam a procurar o produto ou serviço (ocasião de uso, frequência, tipo de dispositivo, necessidade de segurança exigência de declarações legais), linguagem oral, hábitos e preferências. Estes traços proveem a textura visual, emocional e sensorial em que a persona atua (vide Figura 3.3 – indicação 4).

A figura 3.3 apresenta um exemplo de persona usando a estrutura acima. Essa persona foi definida no contexto de governo-eletrônico Aquino Junior (2008).

As personas podem ser apoiadas por informações que complementam o conhecimento a ser aplicado. A descrição de cenários de uso, baseados em tarefas, apoia o entendimento de como um grupo de usuários alcança seus objetivos em interfaces diversificadas ou resolvem problemas na execução de seus papéis.

Com o conjunto de personas definido é possível realizar o alinhamento final com os *stakeholders* do produto e serviço encomendado, com objetivo de classificar a persona principal e a persona secundária. A persona principal representa o público prioritário, enquanto que a persona secundária considera os perfis que também usam o produto ou

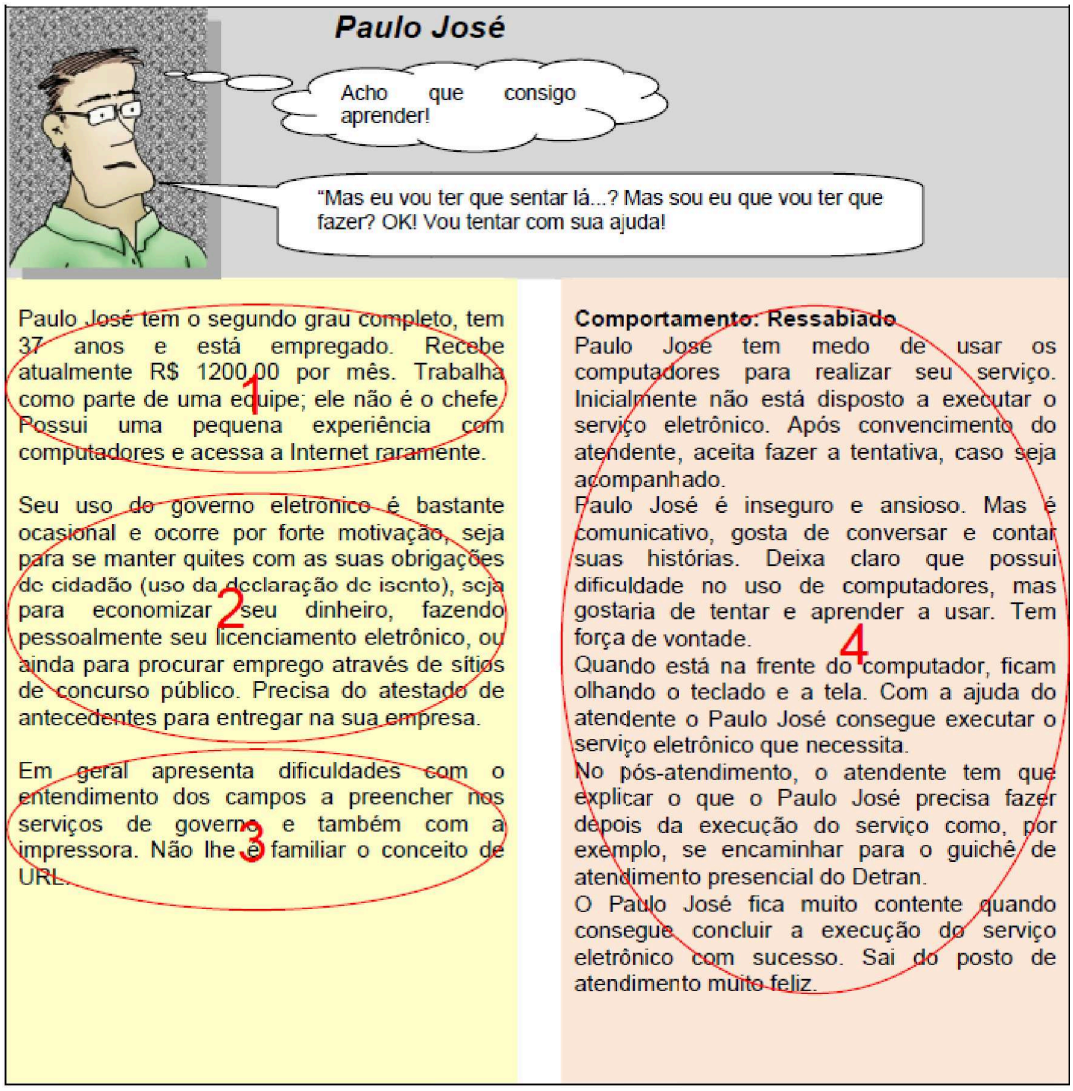


Figura 3.3: Persona definida no contexto de governo-eletrônico Aquino Junior (2008)

serviço, porém suas necessidades não são priorizadas quando sobrepõem requisitos da persona principal. As personas ainda podem ser classificadas em perfis de baixa importância (usam o produto ou serviço esporadicamente), perfis não considerados (não fazem parte do público-alvo) e perfis produzidos (representam pessoas importantes que não foram identificadas na fase de pesquisa, mas identificadas e requisitadas pelos *stakeholders*).

Deve-se ressaltar que a classificação das personas depende do tipo de pesquisa realizado. Quando as personas são criadas apenas por pesquisa qualitativa, normalmente é mais difícil classificar grupos de personas, pois depende apenas da opinião dos *stakeholders*. Quando os projetistas possuem informações provenientes de pesquisa quantitativa é possível identificar prioridades entre personas, com decisões apoiadas por dados estatísticos.

As personas podem apresentar relacionamentos. Os relacionamentos entre perso-

nas podem simplesmente ficar indicados na descrição de cada uma, ou serem apresentados em mapas e diagramas que estabeleçam a ligação de cada um e indiquem as motivações ou preferências com relação a algumas funcionalidades.

A seleção e discriminação de grupos de usuários, bem como a sua tipificação e criação de padrões pode gerar mecanismos arbitrários que reflitam os valores dos agentes envolvidos na sua especificação. Eventualmente pode ser necessário remanejar os resultados para atender as estratégias institucionais ou as visões sobre o alcance do produto ou serviço.

As personas são úteis em todo o ciclo de vida de criação de produtos interativos. As personas fornecem informações desde a concepção do produto/interface até a seleção de perfis para testes de usabilidade. Personas servem de base para os roteiros dos testes de usabilidade com usuários verdadeiros, que a rigor verificam a aderência das narrativas fictícias às situações de uso reais.

3.8. Criando Personas

Agora que o Python, os módulos com algoritmos e o modelo de usuário Personas foram apresentados, será demonstrado um processo completo para criação de Personas. Dessa maneira, pretende-se deixar melhor compreendido o conteúdo apresentado ao longo deste capítulo.

Como exemplo, será utilizado um questionário de pesquisa que contém algumas perguntas sobre a satisfação do usuário relativa a um produto. O questionário completo não será utilizado nessa seção, pois ele é muito extenso; apenas algumas questões serão utilizadas. O questionário é apresentado na sequência do texto.

1. Qual é a sua idade?
2. Você realizou a tarefa de forma fácil e rápida?
   (a) Concordo totalmente
   (b) Concordo em parte
   (c) Discordo em parte
   (d) Discordo totalmente
3. Você utilizaria o sistema frequentemente?
   (a) Concordo totalmente
   (b) Concordo em parte
   (c) Discordo em parte
   (d) Discordo totalmente
4. O sistema atende as suas necessidades?
   (a) Concordo totalmente
   (b) Concordo em parte

(c) Discordo em parte

(d) Discordo totalmente

Com base no questionário acima serão criadas as Personas. Em primeiro lugar precisa-se analisar quais são os tipos das variáveis e quais informações elas trazem. Quando existem variáveis categóricas e alternativas, como no caso das perguntas 2, 3 e 4, deve-se analisar se possui razão entre as respostas, ou seja, há uma hierarquia nas possibilidades de respostas (Hair , 2007).

No exemplo utilizado existem quatro variáveis, uma numérica e contínua, a idade, e três variáveis categóricas que estão divididas em quatro opções de resposta. Nas três variáveis categóricas pode-se notar uma certa hierarquia, ou seja, possuem razão entre as respostas, então fica mais fácil para transformá-las em números, pois o algoritmo trabalha apenas com números. Para transformar as três variáveis em números, como existe razão, será utilizado apenas o método de codificar as repostas em números. Dessa forma as respostas ficão assim:

1. Concordo totalmente
2. Concordo em parte
3. Discordo em parte
4. Discordo totalmente

Codificadas as repostas, há necessidade de carregar as informações dos usuários para uma variável na memória, para que assim possa ser realizada a análise dos dados.

```
#Carregar o arquivo para a memória
arq = open('../BD/persona_exemplo.txt','r')

#Criando a matriz de usuários
usuarios = []

#Separando as variáveis em suas linhas e colunas
#corretamente.
for linha in arq.read().split('\r'):
    usuarios.append(linha.split('\t'))

#Convertendo as variáveis de string para número
listaUsuarios = [[float(i[0]),float(i[1]),float(i[2]),\
                  float(i[3])] for i in usuarios]
```

Com os dados carregados na memória, inicia-se o primeiro passo da análise. Existem quatro variáveis: uma pode conter números entre 0 e ∞, que é a idade, as outras contêm números entre 1 e 4, que são as possibilidades de respostas. Essa diferença de escala pode gerar uma tendência ao algoritmo e diminuir o desempenho de seu resultado.

Para resolver esse problema realiza-se uma normalização dos dados, geralmente entre 0 e 1 (Lattin et al., 2011), mantendo-os dentro de uma mesma escala. A normalização realizada nesse exemplo é apresentada na equação 2.

$$X_{norm} = \frac{\min(X) - X}{\max(X)} \tag{2}$$

Antes de normalizar os dados é necessário verificar como está a distribuição dos valores para não tornar os valores tendenciosos, por exemplo, no caso da idade temos uma concentração de valores entre 15 e 17 anos, porém existe um usuário que tem 58 anos. Se a normalização for realizada considerando como 58 o valor máximo, os usuários que possuem 15 e 17 anos terão o desvio padrão praticamente anulado e serão considerados iguais ao ponto de vista do algoritmo.

Assim, precisa-se definir os melhores valores de máximo e mínimo para cada variável de tal forma, que a distribuição das variáveis fique sem ruídos. Dessa forma, com o auxílio do desvio padrão e a média dos dados pode-se verificar se essa distribuição está adequada ou não para a normalização. Baseando-se na estatística, para a distribuição dos dados ser considerada sem ruído o desvio padrão deve ser menor a 30% da média (McKay , 1932; Verrill , 2003; Hair , 2007). Veja a verificação dessa condição para a idade dos usuários do exemplo.

```
>>> np.std(listaUsuarios[:,0])
6.6755124296150345

>>> np.average(listaUsuarios[:,0])
17.255813953488371

>>> max(listaUsuarios[:,0])
58.0

>>> min(listaUsuarios[:,0])
13.0

>>> 0.3*np.average(listaUsuarios[:,0])
5.1767441860465109

>>> np.std(listaUsuarios[:,0]) < 0.3 * \
               np.average(listaUsuarios[:,0])
False
```

Observa-se que a idade da base de dados utilizada não atende a condição para normalização. Assim, deve-se excluir o valor que está causando o ruído. Para isso, considera-se qualquer valor acima do máximo definido como o valor máximo. Na base de dados do exemplo, basta remover a idade de 58 anos, tornando como o máximo o valor de 27 anos (segundo maior da amostra).

Para as demais variáveis não há a necessidade de alterar o mínimo e o máximo devido ao significado perante o que se espera como resultado. Sendo assim, normaliza-se as informações considerando 13 e 27 como os mínimo e máximo da idade e 1 e 4 para as demais variáveis. A função de normalização fica assim:

```
#Normalizando as informações dos usuários
for usuario in listaUsuarios:
    #Normalizando a variável idade do usuário
    if usuario[0] <= 13:
        usuario[0] = 0
    elif usuario[0] >= 27:
        usuario[0] = 1
    else:
        usuario[0] = abs(13 - usuario[0]) / 27
    #Normalizando a variável tarefa fácil e rápida
    if usuario[1] <= 1:
        usuario[1] = 0
    elif usuario[1] >= 4:
        usuario[1] = 1
    else:
        usuario[1] = abs(1 - usuario[1]) / 4
    #Normalizando a variável utilizaria frenquetemente
    if usuario[2] <= 1:
        usuario[2] = 0
    elif usuario[2] >= 4:
        usuario[2] = 1
    else:
        usuario[2] = abs(1 - usuario[2]) / 4
    #Normalizando a variável atende necessidades
    if usuario[3] <= 1:
        usuario[3] = 0
    elif usuario[3] >= 4:
        usuario[3] = 1
    else:
        usuario[3] = abs(1 - usuario[3]) / 4
```

Com os dados normalizados, pode-se aplicar um dos algoritmos de clusterização apresentados na seção 3.6.3. Para este exemplo, utilizaremos o DBScan, que tem demonstrado um melhor resultado, sem depender de informar o número de grupos que se deseja.

```
#Executando o DBScan
S = np.vstack(listaUsuarios)
core_samples, labels = dbscan(S, eps=0.6, min_samples=1, \
                      metric='euclidean',random_state=None)
```

O DBScan trabalha com a distância euclidiana entre os usuários para determinar se estão ou não em um mesmo grupo. O valor de distância mínima (eps) considerado nesse exemplo foi de 0.6 e o número mínimo para considerar um grupo foi de 1 elemento, para não se excluir ninguém. Ao executar o algoritmo, ele retorna na variável `labels` o valor do índice do grupo ao qual o usuário pertence.

Após esse passo precisamos separar os elementos em seus grupos para gerar as Personas. Para criar os grupos basta realizar o seguinte código.

```
#Criando uma lista de grupos
grupos = []

#Gerando os grupos
for i in set(labels):
    grupos.append([index for index in core_samples \
                   if labels[index] == i])
```

Grupos formados, este é o último passo para criarmos as Personas, que consiste simplesmente de combinar os valores das variáveis e encontrar um valor em comum entre elas. Conforme apresentado por Masiero et al. (2011), para encontrar o valor em comum pode-se utilizar três métodos: a média, a moda ou a mediana. Como já foi realizada a análise da distribuição dos dados antes a normalização, a média deve ser descartada devido ao ruído encontrado na amostra, deixando o desvio padrão maior do que 30% da média. Nesse caso, é indicado que se utilize da mediana ou moda. Para finalizar o exemplo, será utilizada a mediana, conforme ilustrado no código.

```
#Criando uma lista para as personas
personas = []

#Construindo as personas
for grupo in grupos:
    temp = []
    for membro in grupo:
        temp.append(listaUsuarios[membro])
    temp = np.array(temp)
    personas.append([np.median(temp[:,0]), \
                     np.median(temp[:,1]), \
                     np.median(temp[:,2]), \
                     np.median(temp[:,3])])
```

Assim, foram construídas as personas ao longo do processo apresentado nesta seção. As personas construídas nesse exemplo podem ser conferidas na tabela 3.4.

Tabela 3.4: Personas construídas como exemplo

Idade	Frequência de Utilização	Facilidade	Necessidades
16	Concordo em parte	Concordo em parte	Concordo em parte
27	Discordo totalmente	Discordo totalmente	Discordo em parte
58	Concordo em parte	Discordo totalmente	Concordo em parte

3.9. MatplotLib – Gerando Gráficos com Python

A visualização dos dados e dos resultados obtidos através da classificação é importante para a análise dos resultados. Utilizando os exemplos da classificação do K-Means, serão gerados gráficos para a visualização dos dados utilizando o Matplotlib.

O Matplotlib inicialmente desenvolvido por Hunter (2007) tem como objetivo a geração de gráficos em diversos formatos (inclusive vídeos com gráficos animados) com qualidade para publicações em diversas plataformas.

Sendo uma das bibliotecas de Python mais utilizadas para a geração de gráficos, o Matplotlib segue os conceitos de facilidade que são fundamentais para o Python, sendo ainda familiar aos usuários de MATLAB, facilitando o aprendizado e a utilização dos recursos disponíveis para usuários de diversas áreas.

Nesta seção serão apresentadas algumas formas de geração de gráficos disponíveis no Matplotlib.

3.9.1. Geração de Gráficos

Dos diversos tipos de gráficos existentes, pode-se citar o de dispersão, de linha e histograma como alguns dos fundamentais e largamente utilizados. No Matplotlib, esses três tipos de gráficos estão reunidos dentro do objeto `pyplot` que, segundo a descrição apresentada no site da biblioteca,[5] fornece um *framework* para geração de gráficos parecido com as funções do MATLAB.

A seguir, será apresentado um exemplo que, a partir dos dados gerados na seção anterior, plota os gráficos de linha, dispersão e histograma, exibe o gráfico em uma janela e os salva como imagens, para uso posterior.

```
#importa o K-Means implementado pelo Scikit-Learn
from sklearn.cluster import KMeans
from sklearn import preprocessing  # normalização
#funções para gerar os gráficos
from matplotlib import pyplot as plt

#instancia um objeto do K-Means utilizando 3 núcleos (k=3)
kmeans = KMeans(k=3)

#considerando que os dados foram carregados e
#formatados corretamente e que encontram-se
```

[5] http://matplotlib.sourceforge.net/api/pyplot_api.html

```
#na variável "amostra"

#realiza a classificação da amostra e
#armazena o resultado para plotagem
resultados = kmeans.fit_predict(amostra)

X = amostra.T[0]
Y = amostra.T[1]

#plota os dados em um gráfico de dispersão
plt.scatter(X, Y)  # cria o gráfico de dispersão
plt.xlabel('Variavel 1')  # insere nome do eixo X
plt.ylabel('Variavel 2')  # insere nome do eixo Y
plt.savefig('dispersao.png')  # salva o gráfico como figura
plt.show()  # exibe uma janela com o gráfico

#mesmos dados em um gráfico de linha
plt.figure()  # nova figura para não sobrepor o gráfico
plt.plot(X, 'b', Y, 'r')  # cria o gráfico de linha
#os parâmetros para cor e tipo de linha seguem
#o mesmo padrão que o MATLAB
plt.xlabel('Número da variável')
plt.ylabel('Valor da variável na amostra')
plt.legend(('variável 1', 'variável 2'))
plt.savefig('linha.png')
plt.show()

#o resultado da classificação plotado com histograma
plt.figure()  # cria uma terceira figura
#plota o histograma com o resultado da classificação
plt.hist(resultados)
plt.savefig('histograma.png')
plt.show()
```

Os resultados do código anterior encontram-se na figura 3.4.

Entretanto, em alguns momentos se torna mais fácil analisar o gráfico quando plotado em conjunto com os outros gráficos. O exemplo a seguir apresenta a divisão do gráfico de linha da figura 3.4b em dois gráficos plotados na mesma janela. O resultado deste exemplo pode ser visualizado na figura 3.5.

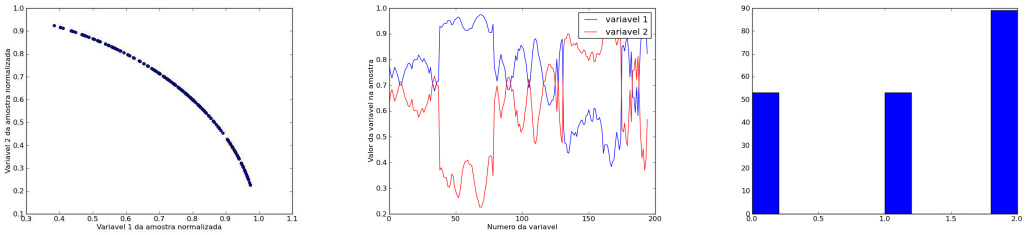
```
#importa o K-Means implementado pelo Scikit-Learn
from sklearn.cluster import KMeans
from sklearn import preprocessing  # normalização
from matplotlib import pyplot as plt
```

```
#instancia um objeto do K-Means utilizando 3 núcleos (k=3)
kmeans = KMeans(k=3)

#considerando que os dados foram carregados e
#formatados corretamente e que encontram-se
#na variável "amostra"

#realiza a classificação da amostra e
#armazena o resultado para plotagem
resultados = kmeans.fit_predict(amostra)

X = amostra.T[0]
Y = amostra.T[1]

#gera um gráfico que aceita subplots
plt.subplot(121)  # 2 posições na horizontal e 1 vertical
plt.plot(X)  # plota o gráfico na primeira posição
plt.subplot(122)  # escolhe a segunda posição
plt.plot(Y, 'r')  # plota o segundo gráfico
plt.savefig('subplot.png')  # salva os gráficos
plt.show()  # exibe os gráficos em apenas uma janela
```

Além destes três tipos de gráficos, é possível também gerar gráficos em 3D utilizando o Matplotlib. Da mesma forma que o 2D, é necessário utilizar o `pyplot`, além do`Axes3D` contido em `mpl_toolkits.mplot3d`. O exemplo a seguir plota o resultado da classificação em relação às variáveis normalizadas da amostra e a figura 3.6 apresenta o resultado dessa classificação.

```
#importa o K-Means implementado pelo Scikit-Learn
from sklearn.cluster import KMeans
from sklearn import preprocessing  # normalização
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D  # gráficos em 3D

#instancia um objeto do K-Means utilizando 3 núcleos (k=3)
kmeans = KMeans(k=3)

#considerando que os dados foram carregados e
#formatados corretamente e que encontram-se
#na variável "amostra"

#realiza a classificação da amostra e
#armazena o resultado para plotagem
resultados = kmeans.fit_predict(amostra)
```

```
X = amostra.T[0]
Y = amostra.T[1]

#gera o gráfico 3D com a amostra e classificação
fig = plt.figure()  # cria uma nova figura
ax = fig.gca(projection='3d')  # para fazer o gráfico 3D
#plota os resultados utilizando dispersão
ax.scatter(X, Y, resultados)
plt.savefig('plot3d.png')
plt.show()
```



(a) Dispersão (b) Linha (c) Histograma

Figura 3.4: Resultados do exemplo de gráficos do Matplotlib

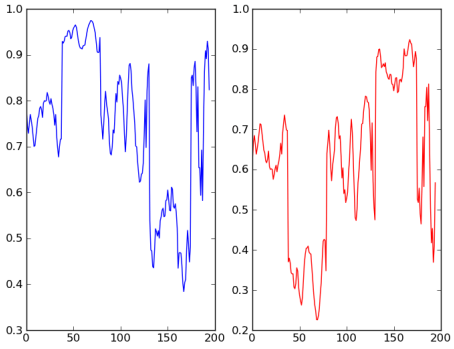


Figura 3.5: Resultado do plotagem de dois gráficos em uma única janela

Os gráficos apresentados até o momento são figuras estáticas que podem ser utilizados para produção de textos e apresentações. Entretanto, o Matplotlib também permite que sejam criadas animações nas quais os dados são plotados ao longo do tempo.

3.9.2. Gráficos Animados

Utilizando o mesmo princípio da geração de gráficos em 2D e 3D, o Matplotlib permite a plotagem de gráficos animados utilizando a biblioteca `matplotlib.animation`.

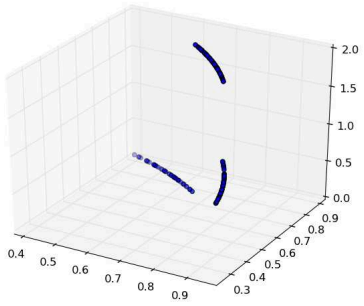


Figura 3.6: Resultado da classificação feita pelo K-Means com $k = 3$

Neste caso, os dados que serão plotados necessitam de atualização a cada passo da animação, de forma que ao longo do tempo o Matplotlib produzirá uma animação que consiste na plotagem sequencial destes dados.

O exemplo a seguir apresenta um código que gera uma animação a partir dos mesmos dados utilizados nos exemplos anteriores e a exportação desse resultado como um vídeo no formato mp4.

```
#importa o K-Means implementado pelo Scikit-Learn
from sklearn import preprocessing  # normalização
from matplotlib import pyplot as plt

#a biblioteca animation contém as funções necessárias
#para o gráfico animado
from matplotlib import animation

#considerando que os dados foram carregados e
#formatados corretamente e que encontram-se
#na variável "amostra"

X = amostra.T[0]
Y = amostra.T[1]

#gera o gráfico animado

#função para atualizar os dados que serão plotados
fig = plt.figure()
ax = fig.add_subplot(111)
linha = ax.scatter([], [])  # gera uma figura vazia
ax.set_xlim(-1.5, 1.5)  # limita os valores em X
ax.set_ylim(-1.5, 1.5)  # limita os valores em Y
x, y = [], []  # dados que serão plotados
```

```
indice = 0


def gera_dados():
    cont = 0
    while cont < len(X) - 1:
        cont += 1
        yield X[cont], Y[cont]


def atualiza_dados(data):
    x0, y0 = data  # adiciona aos dados que serão plotados
    x.append(x0)
    y.append(y0)

    #determina o conjunto de pontos
    linha = ax.scatter(x, y)

    return linha

ani = animation.FuncAnimation(fig, atualiza_dados,
                      gera_dados, interval=10, repeat=False)

ani.save('animacao.mp4')
plt.show()
```

Nesta seção foram apresentadas algumas funções básicas para a produção de gráficos utilizando o Matplotlib, porém existem diversas outras opções disponíveis no Matplotlib, como gráficos de pizza, apresentação de imagens e gráficos com coordenadas polares e boxplot.

Referências

Aquino Junior, P. T. (2008) "Picap: padrões e personas para expressão da diversidade de usuários no projeto de interação", Biblioteca Digital de Teses e Dissertações da USP. http://www.teses.usp.br/teses/disponiveis/3/3141/tde-15092008-144412/, Abril

Aquino Junior, P. T. (2010) "Papéis do docente de ihc: do conhecimento ao mercado", In: IX Simpósio sobre Fatores Humanos em Sistemas Computacionais, IHC '10, Brazil, Brazilian Computer Society. p. 79–82.

Aquino Junior, P. T. (2012) "Personas alunos e papéis do docente no atendimento da diversidade de perfis"

Aquino Junior, P. T. and Filgueiras, L. V. L. (2005) "User modeling with personas", In: Proceedings of the 2005 Latin American conference on Human-computer interaction, CLIHC '05, USA, ACM, p. 277–282.

Aquino Junior, P. T. and Filgueiras, L. V. L. (2008) "A expressão da diversidade de usuários no projeto de interação com padrões e personas", In: Proceedings of the VIII Brazilian Symposium on Human Factors in Computing Systems, IHC '08, Brazilian Computer Society, Brazil, p. 1–10.

Bagnall, P. (2007) "Using personas effectively", In: Proceedings of the 21st British HCI Group Annual Conference on HCI 2008: People and Computers, volume 2.

Borges, L. E. "Python para desenvolvedores 2ª edição".

Cirelli, M.; Aquino, P. T., and Filgueiras, L. V. L. (2010) "Personas celulares do mercado brasileiro", In: Proceedings of the IX Symposium on Human Factors in Computing Systems, IHC '10, Brazilian Computer Society, Brazil, p. 129–138

Cooper, A. (1999) "The Inmates Are Running the Asylum", Macmillan Publishing Co., Inc., 1999.

Cooper, A.; Reimann, R., and Cronin, D. (2007) "About face 3: the essentials of interaction design", Wiley-India.

Deitel, H. M.; Deitel, P. J.; Liperi, J. P., and Weidermann, B. (2002) "Python: how to program", Prentice Hall.

Ester, M.; Kriegel, H.; Sander, J., and Xu, X. (1996) "A density-based algorithm for discovering clusters in large spatial databases with noise", In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data mining, AAAI Press, p. 226–231.

Filgueiras, L. V. L.; Aquino Junior, P. T.; Sakai, R.; Filho, A. G.; Torres, C., and Barbarian, I. (2005) "Personas como modelo de usuários de serviços de governo eletrônico", In: Proceedings of the 2005 Latin American conference on Human-computer interaction, CLIHC '05, USA, ACM, p. 319–324.

Frey, B. J. and Dueck, D. (2007) "Clustering by passing messages between data points",

Hair, J. F. (2007) Análise multivariada de dados Bookman

Hunter, J. D. (2007) "Matplotlib: A 2d graphics environment", Computing In Science & Engineering, p. 90–95

Jones, E.; Oliphant, T.; Peterson, P., and others, . (2001) "SciPy: Open source scientific tools for Python", http://www.scipy.org/

Jung, C. G. (1975) "The archetypes and the collective unconscious", volume 2, Bollingen.

Jung, C. G. and do Amaral, V. (1986) "O desenvolvimento da personalidade", volume 7, Brazil, Vozes.

Jung, C. G. and Rocha, M. R. (1990) "AION-Estudos sobre o simbolismo do si-mesmo", Brazil, Vozes.

Lattin, J.; Carrol, D. J. D., and Green, P. E. (2011) "Análise de dados multivariados", Brazil, Cegrage Learning.

Lloyd, S. P. (1982) "Least squares quantization in pcm", In: IEEE Transactions on Information Theory 28, p. 129–137

MacQueen, J. and others, (1967) "Some methods for classification and analysis of multivariate observations", In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability", volume 1, USA, p. 281–297.

Marsland, S. (2009) "Machine Learning: An Algorithmic Introduction", USA, CRC Press.

Masiero, A. A.; Leite, M. G.; Filgueiras, L. V. L., and Aquino Junior, P. T. (2011) "Multidirectional knowledge extraction process for creating behavioral personas",' In: Proceedings of the 10th Brazilian Symposium on on Human Factors in Computing Systems and the 5th Latin American Conference on Human-Computer Interaction, IHC+CLIHC '11, Brazil, Brazilian Computer Society, p. 91–99.

McKay, A. T. (1932) "Distribution of the Coefficient of Variation and the Extended t Distribution" Journal of the Royal Statistical Society, JSTOR, p. 695–698

Pedregosa, F.; Varoquaux, G.; Gramfort, A., and others. (2011) "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research 12

Pruitt, J. and Adlin, T. (2006) "The persona lifecycle: keeping people in mind throughout product design", USA, Morgan Kaufmann Publishers.

Pruitt, J. and Grudin, J. (2003) "Personas: practice and theory", In: Proceedings of the 2003 conference on Designing for user experiences, DUX '03, USA, ACM, p. 1–15.

Summerfield, M. (2009) "Programming in Python 3: a complete introduction to the Python language", USA, Addison-Wesley Professional.

Verrill, S. P. (2003) "Confidence bounds for normal and lognormal distribution coefficients of variation" US Department of Agriculture, Forest Service, Forest Products Laboratory

Capítulo 4

Projetando Sistemas *Web* com o uso de Técnicas de Interação Humano-Computador

Patricia Cristiane de Souza, Cristiano Maciel, Luciana A. O. de Moraes

Abstract

It is fundamental to improve software development processes towards Web Engineering, which must consider, for example, the possibility of different types of users of an application, as this would have an impact on the way people interact with the systems. Pressman and Lowe (2009) propose an agile framework, employing different methods for designing a web application (WebApp), which are represented in a pyramidal structure whose levels depict project actions. However, these actions are not integrated in Human-Computer Interaction (HCI) activities. Thus, this tutorial aims to enable participants to integrate HCI practices and Web Engineering methods by means of theoretical and practical strategies.

Resumo

O aprimoramento dos métodos de desenvolvimento de software voltados para a Engenharia Web é fundamental. Esta deve considerar, por exemplo, a possibilidade de existirem diversificados tipos de usuários fazendo uso de uma aplicação, o que tem forte impacto nas formas de interação nos sistemas. Em Pressman e Lowe (2009), propõe-se um arcabouço ágil, em que distintos métodos de projeto de uma aplicação Web (WebApp) são utilizados, representado por uma pirâmide dividida em níveis, que correspondem a ações de projeto. Apesar de ser uma proposta inovadora, nota-se que essas ações não estão totalmente integradas às atividades da Interação Humano-Computador (IHC). Neste sentido, o presente tutorial visa capacitar os participantes, por meio de estratégias teóricas e práticas, no projeto de sistemas web com o uso de técnicas de IHC.

4.1 Apresentação

Neste capítulo, apresentamos uma proposta de incorporação das atividades de interação humano-computador (IHC) ao projeto de uma aplicação *web* (WebApp). Sob a forma de

um tutorial introdutório, o conteúdo deste material, com duração de seis horas, tem como público-alvo os acadêmicos de graduação e pós-graduação, além de representantes da indústria de *software* interessados na Engenharia para Sistemas *Web*, nos aspectos da Interação Humano-Computador e na relação entre as duas áreas.

Este tutorial pretende demonstrar como os métodos da Engenharia *Web*, em especial o modelo proposto por Pressman e Lowe (2009), podem ser desenvolvidos de forma alinhada às especificidades da IHC, visando garantir o atendimento aos critérios de usabilidade no desenvolvimento de WebApps.

Desta forma, pode-se afirmar que tal proposta está relacionada ao tópico de interesse *Integração de técnicas de IHC e de Engenharia de Software* (ES) do Simpósio Brasileiro de Fatores Humanos em Sistemas Computacionais, IHC 12, principalmente no que diz respeito ao esforço despendido para que os sistemas *web* sejam desenvolvidos não só para integrar técnicas e procedimentos aliados a sua finalidade como para respeitar as características de seus usuários.

O texto deste capítulo está organizado em sete seções ao todo. A próxima seção apresenta o processo da engenharia *web* proposto por Pressman e Lowe (2009). Ilustrando os modelos de processo desta área, está a terceira seção, que apresenta sucintamente as atividades da IHC. A quarta seção relata o esforço de alguns outros pesquisadores em integrar as áreas de IHC e ES. Na quinta seção apresentamos um estudo piloto desenvolvido com o intuito de analisar as dificuldades em alinhar a prática das duas áreas no projeto de sistemas *web*. Por fim, reservamos a sexta seção para apresentarmos a nossa proposta de incorporação das atividades de IHC ao projeto de WebApp, especificamente na etapa de projeto. O capítulo encerra-se com algumas considerações acerca dos resultados obtidos até o momento e algumas reflexões para trabalhos futuros.

4.2 Processo da Engenharia Web

A Engenharia *Web* (WebE) é um processo que se divide em ações ou tarefas para melhor se estruturar e organizar o desenvolvimento de aplicações *web*.

As tarefas ou ações, por sua vez, são desenvolvidas através de um conjunto de técnicas e métodos, que geram artefatos e assim concluem uma etapa do processo, ficando a equipe de engenheiros disponível para a próxima fase até concluir o produto, qual seja, a aplicação.

Para que uma aplicação *web* obtenha qualidade e sucesso, é essencial a utilização de técnicas e métodos apropriados, bem como o atendimento às recomendações da Engenharia *Web* relativas ao processo de desenvolvimento.

Pressman e Lowe (2009) sugerem um processo de desenvolvimento que contemple as atividades do arcabouço ágil, porém disciplinado, proposto pela Engenharia *Web*. Nas seções 4.2.1 e 4.2.2 explanaremos sobre o arcabouço de processo da WebE, seus métodos e as respectivas atividades genéricas de cada um deles.

4.2.1 Arcabouço de processo de WebE

Para fins didáticos, podemos entender o arcabouço como o esboço, o delineamento inicial do processo da Engenharia *Web* e como ágil a capacidade dos engenheiros de se adequarem rapidamente às mudanças exigidas pelos *stakeholders* e pelo negócio da aplicação, inclusive.

O arcabouço proposto por Pressman e Lowe (2009) estabelece também o alicerce para o processo de desenvolvimento de forma completa, que identifica as atividades a serem aplicadas a todos os projetos de WebApp, independentemente de tamanho e complexidade. A Figura 4.1 ilustra o processo WebE sugerido pelos autores.

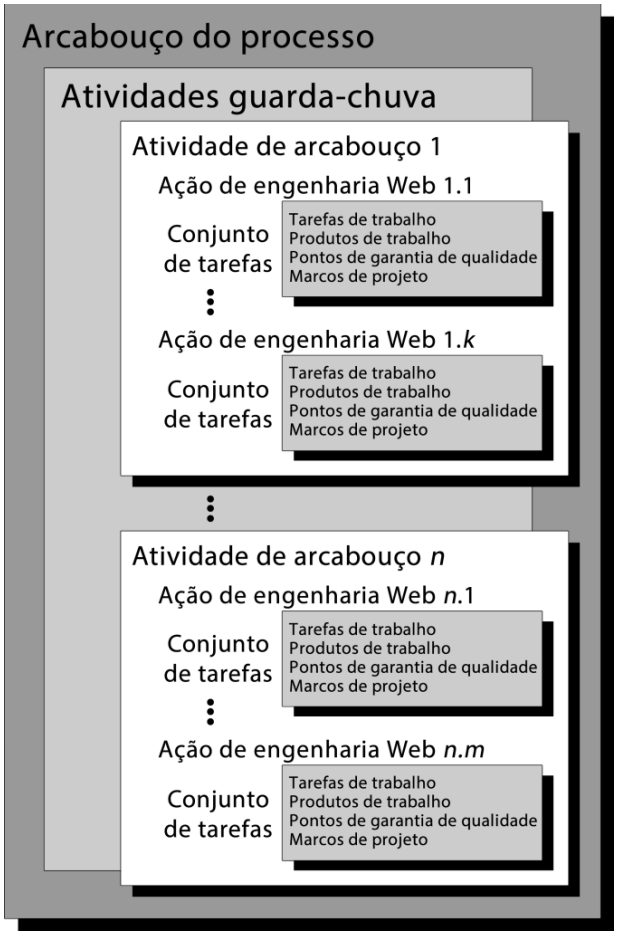


Figura 4.1 - Um arcabouço de processo de WebE

Fonte: Pressman e Lowe (2009, p.13)

Na Figura 4.1 podemos identificar que o processo de WebE engloba, além do arcabouço do processo ou macro atividade, as atividades consideradas guarda-chuvas, cujo escopo abrange as várias atividades de arcabouço, que se compõem de ações de engenharia *Web*, em que cada qual foi organizada em um conjunto de tarefas formado por tarefas e produto de trabalho, pontos de garantia de qualidade e marcos do projeto.

Esse processo de WebE é genérico e pode ser aplicado em vários tipos de projetos de WebApp. Uma vez que as ações *web* já estão definidas, os engenheiros deverão apenas adequar as novas necessidades de negócios aos próximos sistemas a serem desenvolvidos, o que resultará em ganho de produtividade e qualidade.

Levando em consideração que o modelo do processo de uma aplicação *web* deve ser ágil, é importante que os métodos e técnicas atendam eficientemente a um arcabouço do processo da Engenharia *Web*.

De acordo com Pressman e Lowe (2009), os métodos de WebE podem ser categorizados da seguinte forma:

1. **Métodos de comunicação** - onde ocorre a definição das técnicas que contribuem com a comunicação que deve existir entre os envolvidos no processo de desenvolvimento da aplicação, tais como clientes, usuários finais, gerente do projeto, *designers*, desenvolvedores, analistas, entre outros.
2. **Métodos de análise de requisitos** - contribuem para um melhor entendimento das necessidades e funções que a WebApp deve atender, bem como o comportamento dos usuários finais ao utilizar o sistema.
3. **Métodos de projeto** - adotam técnicas para os projetos de conteúdo, arquitetura da informação e da aplicação, interface e navegação.
4. **Métodos de construção** - envolvem as ferramentas, tecnologias, linguagens para o desenvolvimento das funcionalidades e conteúdos da WebApp.
5. **Métodos de teste** - envolvem as técnicas a serem usadas no conjunto de testes necessários ao pleno funcionamento da aplicação, tais como os testes de implementação, usabilidade, segurança, navegação, entre outros.

Todos os métodos são não só extremamente importantes como norteadores do processo da WebE, uma vez que apresentam, de acordo com cada categoria, as tarefas e/ou ações necessárias à elaboração de uma WebApp de qualidade.

Como mencionado anteriormente, o arcabouço de processo WebE é genérico e pode ser utilizado em vários projetos, desde que haja o devido atendimento às necessidades do negócio e aos envolvidos, mediante seu refinamento e/ou adaptação.

Além disso, sempre que o arcabouço for aplicado, deve-se fazê-lo de forma iterativa, ou seja, para cada incremento da aplicação desenvolvido, repete-se o processo para a WebApp, em que certamente as atividades de arcabouço também se repetirão.

Para Pressman e Lowe (2009) existem cinco atividades consideradas de arcabouço, como mostra a Figura 4.2, que configuram as ações necessárias para a implementação do processo de WebE.

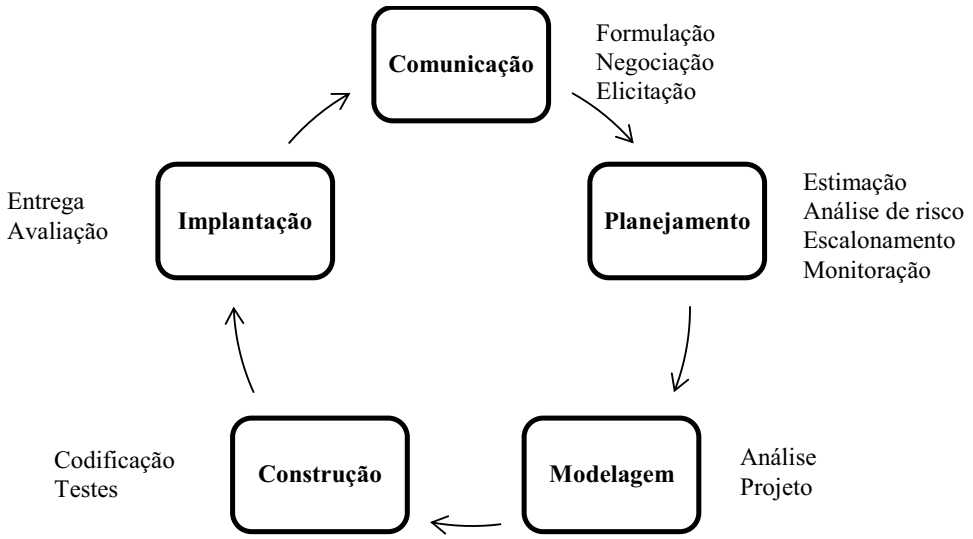


Figura 4.2 - Fluxo de processo com as atividades de arcabouço genéricas da WebE

A seguir apresentamos os conceitos e ações relativos a cada uma das atividades do processo WebE, denominadas fases.

Comunicação: No processo de WebE, a comunicação é definida por três ações: formulação, elicitação e negociação. A formulação caracteriza essencialmente o contexto do negócio e da organização da aplicação. A elicitação é responsável pelo levantamento de requisitos. Já a negociação é utilizada para viabilizar a conciliação das divergências entre os diferentes interessados no projeto.

Planejamento: É composto por ações, tais como plano de trabalho, estimativas dos recursos, riscos, cronograma para acompanhamento e monitoramento das etapas do projeto. Tudo isso, de acordo com a quantidade de incremento exigido pela aplicação.

Modelagem: Nesta etapa são realizadas ações de análise e as tarefas de projeto, que juntas formam a modelagem da WebApp.

Construção: A utilização de ferramentas e tecnologia WebE para a codificação e testes da WebApp, a qual foi modelada na etapa anterior.

Implantação: Esta fase inclui as ações que envolvem a configuração do ambiente para o recebimento da aplicação e para o treinamento dos usuários finais que previamente iniciam um processo de avaliação da WebApp.

Com essas atividades o processo de WebE fica mais compreensível, uma vez que podemos identificar dentro do arcabouço sugerido as ações necessárias, contribuindo assim para o desenvolvimento efetivo da WebApp.

É importante lembrar que as ações advindas das atividades mencionadas são repetidas na medida em que novos incrementos são entregues ao cliente.

A próxima seção apresenta mais informações sobre a atividade de modelagem, uma vez que o foco deste trabalho é a integração das atividades de IHC com os métodos da WebE especificamente na etapa de projeto.

4.2.2 A Atividade de Modelagem

Para Pressman e Lowe (2009), a modelagem é uma atividade que cria uma ou mais representações conceituais de algum aspecto da WebApp a ser construída. Nesta fase ocorrem duas ações de engenharia web: análise e projeto.

A atividade de modelagem envolve tanto a análise dos requisitos necessários para o desenvolvimento da WebApp quanto as ações do projeto, que são imprescindíveis para a próxima etapa do processo da engenharia, qual seja, a de construção.

Na etapa de análise, os requisitos levantados na atividade de comunicação são devidamente verificados, examinados, representados e refinados, formando assim uma aplicação que, de fato, atenda as necessidades dos envolvidos. As ações necessárias para cumprir a análise envolvem vários fatores, que são realizados conforme a aplicação a ser desenvolvida. Normalmente, já existe um modelo com as ações a serem executadas, segundo a realidade da fábrica de software e do negócio.

A etapa de projeto, para engenharia web, tem a função de produzir um modelo ou uma representação, a qual mostre confiabilidade, comodidade e agradabilidade. Por isso, nos baseamos na pirâmide de projeto apresentada por Pressman e Lowe (2009), em que um modelo de projeto completo é composto com base em ações de projeto, em níveis sequenciais, partindo do projeto de interface e finalizando com o início da construção, conforme ilustra a Figura 4.3.

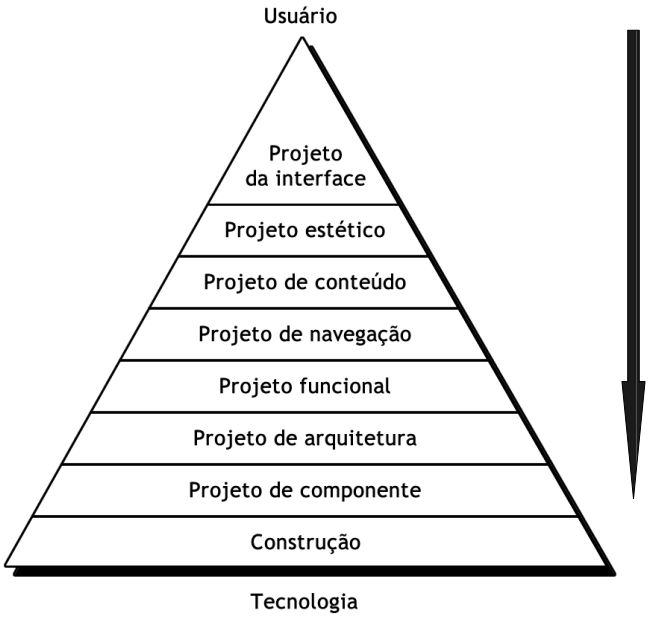


Figura 4.3 - Pirâmide de projeto da WebApp

Fonte: Pressman e Lowe (2009, p.172)

Sabendo-se da importância do modelo de processo requerido pela WebE, convém, antes de se pensar na execução do processo do projeto da WebApp, conhecer detalhadamente as suas ações e tarefas, de acordo com a pirâmide proposta por Pressman e Lowe (2009), as quais apresentamos a seguir:

Projeto de *Interface*: É a ação responsável por descrever a estrutura e a organização da *interface* com o usuário.

Projeto Estético: Trata-se da ação, conhecida também como projeto gráfico, que define a aparência e o estilo da aplicação.

Projeto de Conteúdo: Ação que define o *layout*, a estrutura e o esboço para todo o conteúdo que deve ser apresentado na WebApp, inclusive estabelece a relação entre os objetos de conteúdo.

Projeto de Navegação: Uma ação destinada a representar o fluxo de navegação entre os objetos de conteúdo e para todas as funções da aplicação.

Projeto Funcional: É aquela ação que identifica o comportamento geral e a funcionalidade provida pelo sistema.

Projeto de Arquitetura: Ação que tem a função de encontrar os componentes conceituais para a WebApp e respectivas interconexões, além de identificar a arquitetura técnica.

Projeto de Componente: Esta é a ação que desenvolve a lógica de processamento detalhada, exigida para implementar componentes funcionais que dão apoio a um conjunto completo de funções da aplicação.

Os dois primeiros projetos, quais sejam, de interface e estético, que formam o projeto de interação, são base para a interação do sistema com o usuário. Eles exigem a execução de tarefas de definição do *layout* das telas da aplicação, que envolvem os mecanismos de navegação, cores, *layout* geométrico, tamanho dos textos, fontes, entre outros.

Muitos aspectos de projeto são identificados através de protótipos de telas que acabam se tornando o embasamento para a construção da WebApp. Assim, fica claro que problemas na construção poderão gerar a necessidade de reprojeto, fazendo com que o projeto continue dentro das suas tarefas.

Já as duas ações seguintes, os projetos de conteúdo e navegação, têm foco no conteúdo da aplicação, e executam atividades como: o levantamento e organização das informações que devem fazer parte da WebApp, bem como verificar a maneira como o usuário navega e acessa as informações e as funcionalidades da aplicação.

Por outro lado, as três últimas ações, que formam os níveis mais baixos da pirâmide, referem-se aos projetos funcional, arquitetural e de componente que, em conjunto, formam o suporte para a atividade de construção ao considerar como a interface, navegação, conteúdo e funções da WebApp são integrados e implementados. Em decorrência dessa característica, esses níveis são classificados como técnicos pelos autores.

Aos projetos anteriormente mencionados cabem desenvolver tarefas que ajudem a identificar os componentes conceituais, as interligações destes componentes, os

componentes técnicos, as funcionalidades e, se necessário, o fluxo de trabalho das ações do sistema e dos processos do banco de dados.

Todas as ações citadas no parágrafo anterior estão inseridas no processo de desenvolvimento, que requer uma produção incremental, ou seja, à medida que novos requisitos forem elencados e incrementos projetados, as ações da pirâmide devem ser (re)-elaboradas.

Os projetos de interface e estético, posicionados no topo da pirâmide, representam individualmente os princípios da interação do usuário com o sistema, mas quando considerados no conjunto, formam o projeto de interação. No nível inferior subsequente está o projeto de navegação que, pela sua importância considerável com a IHC, há que se repensar na "reconstrução" ou "desconstrução" da pirâmide, de forma a contemplar a integração das técnicas e métodos aplicados entre as ações da referida pirâmide.

Ainda, se considerarmos métodos estruturados, como o que é proposto por Wazlawick (2011), também se percebe que as camadas de interface e navegação estão em um nível inicial do projeto, ou seja, em fases mais avançadas do desenvolvimento pode ser necessário refazer algum artefato desses níveis iniciais.

Apesar de Pressman e Lowe (2009) e Wazlawick (2011) contemplarem atividades do projeto de interação no processo de desenvolvimento de sistemas, tais abordagens, além de dificultar a incorporação de atividades da IHC, ainda podem gerar reprojeto à medida que os artefatos gerados tenham de ser revistos. Eis então o desafio para a Web: considerar o projeto de interação de forma estruturada até a fase de desenvolvimento do sistema.

4.3 Modelos de Processo da IHC

Os processos de design de IHC, geralmente, começam analisando a situação atual. Quando o projetista considera ter adquirido conhecimento suficiente sobre essa situação e identificado as necessidades e oportunidades de melhoria, ele prossegue seu trabalho sintetizando, concebendo, modelando e construindo uma solução.

Existem vários modelos de processo de design propostos pela IHC. Cada um privilegia uma forma de pensar, uma sequência de atividades ou um emprego de certos artefatos. Alguns modelos propostos são:

Projeto Centrado no Usuário (PCU) - O foco está no entendimento sobre os usuários e suas tarefas desde o princípio e, desta forma, os projetistas devem ser colocados em contato direto com o usuário e esses devem participar das atividades do projeto. De modo geral, é um processo iterativo com ciclos alternantes de design, teste, avaliação e redesign. Neste modelo se encontram algumas abordagens como o Design Participativo (DP) e o Design Contextual (DC).

Estrela - criado em 1989 por Hartson e Hix (Barbosa e Silva, 2010) e é baseado em experimentos empíricos sobre como os designers de interface trabalhavam. Este modelo é bastante flexível, uma vez que não possui ordenamento das atividades, e a

avaliação é o ponto central deste, sendo que a cada atividade finalizada a mesma é avaliada. Como ponto negativo ressalta-se que tamanha flexibilidade do modelo não permite ter uma visão geral do esforço de desenvolvimento, nem monitorar o progresso, previsão de gastos etc.

Engenharia de Usabilidade – o modelo proposto por Mayhew em 1999 visa integrar as tarefas de usabilidade com as atividades dos ciclos de vida tradicionais como abordagens com o uso de prototipação e orientação a objetos. Este processo tem foco no uso de um guia de estilo utilizado em todo o projeto para assegurar o cumprimento de metas de usabilidade (Preece *et al.*, 2005).

Baseado em Cenários – os cenários são histórias sobre pessoas utilizando uma tecnologia em dado contexto. O processo baseado em cenários auxilia no entendimento da situação e do desejo dos usuários; na especificação dos requisitos, na antecipação de ideias para avaliação e ainda especificação da implementação (Benyon, 2011).

Projeto Centrado na Comunicação – tem como base teórica a engenharia semiótica, a qual compreende a IHC como um processo de comunicação entre o usuário e o projetista do sistema, por meio da interface. A motivação principal deste projeto é elaborar uma solução de IHC que transmita a metacomunicação do projetista de forma eficiente e eficaz, ou seja, produzir um sistema interativo com alta comunicabilidade. Este modelo propõe três atividades principais: análise do usuário, do domínio e do contexto de uso; o projeto de interação e interface; e avaliação do que foi projetado (Barbosa e Silva, 2010).

Para Preece *et al.* (2005), o processo de design de interação envolve quatro atividades básicas:

- identificar necessidades e estabelecer requisitos;
- desenvolver designs alternativos que preencham esses requisitos;
- construir versões interativas dos designs, de maneira que possam ser comunicados e analisados; e
- avaliar o que está sendo construído durante o processo.

Além das quatro atividades básicas de design propostas por Preece *et al.* (2005), existem três características-chave quanto ao processo de design de interação:

- os usuários devem estar envolvidos no desenvolvimento do projeto;
- a usabilidade específica e as metas decorrentes da experiência do usuário devem ser identificadas, claramente documentadas e acordadas no início do projeto; e
- a iteração em todas as quatro atividades é inevitável.

Percebe-se, então, que o desenvolvimento de sistemas interativos adequados aos usuários exige tarefas determinadas, realizadas desde o início do ciclo de vida. Essa prática, apesar de não ser nova, ainda não está consolidada. Tentar garantir uma boa interação deve permear todo o processo de desenvolvimento e essa atividade requer estudos específicos (Valverde, Panach e Pastor, 2007). Em especial, o investimento em atividades de usabilidade desde o início do ciclo melhora sensivelmente a qualidade do produto final.

Uma das técnicas que tem sido usada na WebE é a arquitetura da informação (Rosenfeld e Morville, 2002 *apud* Pressman e Lowe, 2009), a qual permite a combinação entre organização, rotulação, sistema de navegação e pesquisa em uma WebApp. Ela define o projeto estrutural de um espaço de informação para facilitar a realização da tarefa e o acesso intuitivo ao conteúdo. Desta forma, permite estruturar e classificar Webapps para ajudar as pessoas a encontrarem e gerenciarem informações. Como elementos a serem considerados na arquitetura da informação citam-se os usuários (necessidades, tarefas, experiências, expectativas, comportamento etc), o contexto (objetivos do negócio, visão, recursos, política etc) e o conteúdo (documentos, dados, serviços, aplicações que os usuários precisam usar ou encontrar etc). Tal arquitetura tem influência direta na modelagem da interação de um sistema Web.

Em um processo de desenvolvimento de software, as etapas do projeto de interface devem ser executadas conjuntamente com as etapas de um processo de software (conforme modelo de desenvolvimento da ES). A primeira etapa é a identificação das necessidades dos usuários e os requisitos, considerando, entre outros, as tarefas quem os usuários devem realizar por meio da interface, o conteúdo que é apresentado como parte da interface e o ambiente em que essas tarefas serão conduzidas. As demais etapas seguem o modelo de desenvolvimento de software escolhido para o projeto, todavia é necessário considerar aspectos do design da interação também nas outras fases do processo. Projetos como os de conteúdos e de navegação são fundamentais para a usabilidade do sistema.

Por fim, resta dizer que é evidente que os modelos de processo, métodos e técnicas da WebE e da IHC expressem o foco da área que os propõem. Porém, nossa crença é de que o alinhamento dos métodos e técnicas de ambas as áreas podem resultar no desenvolvimento de sistemas de forma mais abrangente, que além de satisfazer o usuário/cliente por serem entregues no prazo, dentro do custo previsto, serem de fácil manutenção, confiáveis e de fácil utilização.

A próxima seção apresenta alguns trabalhos que também se ocupam em pesquisar e desenvolver atividades de integração entre as áreas de ES e IHC que, de certo modo, estão relacionados a nossa proposta.

4.4 Trabalhos Relacionados

A integração de métodos e técnicas de IHC em processos tradicionais de desenvolvimento de *software* propostos pela ES tem sido muito investigada por pesquisadores (Willshire, 2003; Souza e Freiberger, 2011; Silva *et al*., 2004; Sousa and Furtado, (2003); Ferre, 2003; Ferre e Bevan, 2011). Para Barbosa e Silva (2010), os principais estudos para essa integração consistem em encontrar meios de incorporar atividades e métodos de IHC aos processos propostos pela ES e em investigar, nos processos de desenvolvimento, características que se preocupam com a qualidade de uso.

Em Willshire (2003), o autor expõe sua preocupação sobre a lacuna existente até então entre a engenharia de software e a interação humano-computador. O autor discute e relata seu esforço, como docente em ambas as disciplinas, em disseminar os conceitos,

metodologias e processos destas áreas aos alunos. Ele apresenta algumas associações existentes entre a ES e a IHC, tais como: necessitam levantar e especificar requisitos, podem fazer uso de prototipação, entre outras.

Em Souza e Freiberger (2011) há o relato da experiência prática de um projeto interdisciplinar desenvolvido com uma turma de alunos de graduação envolvendo três disciplinas: Interface Humano-Computador (IHC), Análise e Projeto de Sistemas I e Programação em Ambiente Web I.

Neste projeto o aluno se envolveu em uma situação prática de desenvolvimento de sistemas computacionais. O modelo de processo de desenvolvimento de software escolhido foi o processo unificado (PU) com abordagem ágil. Desta forma, os artefatos puderem ser relacionados por fase e estabelecidos uma marcação da fase que os artefatos foram iniciados (I) e fases que sofreram revisões (R). Muitos destes artefatos foram elaborados visando à integração de atividades do projeto centrado no usuário ao PU.

A Tabela 4.1 apresenta a relação dos artefatos produzidos nas três fases: concepção, elaboração e construção.

Tabela 4.1: Descrição dos Artefatos Produzidos

Artefatos/Fases	Concepção	Elaboração (E1..En)	Construção (C1..Cn)
Documento de Visão	I		
Glossário	I	R	
Relatório das técnicas de levantamento de requisitos	I	R	
Modelo de Caso de Uso	I	R	
Especificação Suplementar	I	R	
Regras de Negócio	I	R	
Modelo de Domínio		I-R	
Modelos de Projeto		I	R
Projeto de Informação e Interação		I-R	R
Modelo de Dados		I	R
Plano de Testes e Plano de Avaliação de Usabilidade		I	R
Protótipos interativos		I	R
Código fonte		I	R

Fonte: Souza e Freiberger (2011)

Alguns artefatos da Tabela 4.1 merecem certo destaque. O modelo de casos de uso continha especificações de casos de uso completas contendo percursos de interação via *wireframe* visando a rastreabilidade entre os dois elementos.

O projeto de informação e interação é um documento composto por mapa de navegação; definição dos padrões de rótulos e mensagens do sistema; descrição da

arquitetura da informação bem como descrição das questões de acessibilidade e usabilidade adotadas no projeto.

Na fase de elicitação de requisitos foi gerado o relatório das técnicas de levantamento de requisitos utilizadas identificando as necessidades básicas do sistema e iniciando a participação de usuários do sistema no projeto e desenvolvimento do sistema. A partir desta fase outros artefatos gerados foram apresentados e discutidos com a participação de usuários, tais como o documento contendo a rastreabilidade entre casos de uso e *wireframes* do sistema.

Durante todas as etapas de especificação, modelagem e desenvolvimento de protótipos foram estudados e desenvolvidos os artefatos discutidos nas disciplinas de IHC e Análise de Projetos I. Desta forma, esta prática propiciou uma visão não sedimentada da área de IHC, fortalecendo a compreensão e a prática dos diferentes conhecimentos necessários ao desenvolvimento de sistemas.

Para Silva *et al.* (2004, p.118) a integração da ES com a IHC não é uma tarefa trivial. Os autores sugerem "uma identificação mais detalhada das tarefas a serem realizadas durante o processo de desenvolvimento, não se limitando somente a responsabilizar o engenheiro de software pela parte tecnológica e o especialista em IHC pela interação e interface com o usuário". Os autores também relatam os problemas de comunicação entre os dois profissionais (engenheiro de software e especialista de IHC) e desta forma propõem o uso de padrões como forma de amenizá-los. Neste trabalho, os autores apresentam diversas aplicabilidades dos padrões de engenharia de software e de interação humano-computador para o desenvolvimento de sistemas interativos e afirmam que o uso de padrões favorece a criação de um sistema de forma mais abrangente, que considera e trata os principais aspectos da ES e da IHC.

A proposta de Sousa and Furtado (2003) tem o objetivo principal de apresentar um estudo sobre as áreas de IHC e ES, com foco na importância da integração entre essas áreas, mais precisamente no Processo de Desenvolvimento de Software (PDS). Em especial, trata-se da adaptação do Rational Unified Process (RUP) para contemplar, de forma integrada, aspectos da IHC em seus fluxos de trabalho principais, gerando o Rational Unified Process para Sistemas Interativos, denominado neste de RUPi.

O RUP é um PDS bem estabelecido, que tem a intenção de garantir a qualidade, controlando o cronograma do projeto, orçamento, comunicação, produtividade e confiabilidade. Por sua vez, o RUPi pretende garantir a acessibilidade, usabilidade e aceitabilidade, mantendo o foco nos usuários e em seu contexto de uso, na modelagem de tarefas dos usuários, considerando guidelines sugeridos durante o PDS e definindo mecanismos para projetar as interfaces de usuário (UIs) e para testar sua usabilidade.

A denominação RUPi foi elaborada com base em da Silva e Paton (2000) e na UMLi (2001) – Linguagem de Modelagem Unificada para Sistemas Interativos, uma extensão da UML. A UMLi aborda a modelagem de aplicações interativas completas, especialmente suas User Interfaces (UIs).

O RUPi é composto por quatro fluxos de trabalho, que são baseados nos fluxos de trabalho do RUP e foram adaptados pela inclusão de conceitos de IHC. Os quatro fluxos de trabalho escolhidos para RUPi são: análise de requisitos, projeto, implementação e teste. Essa adição de conceitos de IHC é feita a fim de realizar atividades que produzam artefatos que considerem:

Fatores humanos: Considerados durante o desenvolvimento de um sistema interativo para ajudar os usuários a executar suas tarefas de forma eficaz.

Requisitos de usabilidade: Usados para decidir qual a opção de UI é mais adequada para o desempenho de uma tarefa específica por um determinado usuário.

Acessibilidade e aceitabilidade: Aplicadas durante a definição de diretrizes, a fim de acomodar as características de diferentes tipos de usuários.

Avaliação de usabilidade: Usada para testar a usabilidade de um sistema interativo considerando as características dos usuários, tarefas executadas, ambiente de trabalho e tecnologias utilizadas.

Cenários gráficos: Gerados para representar graficamente situações específicas vividas pelos usuários ao interagir com o sistema.

A Tabela 4.2 representa a relação entre os quatro fluxos de trabalho RUPi e os respectivos artefatos específicos e profissionais envolvidos.

Tabela 4.2: Relacionamento entre Fluxos de trabalho, Profissionais e Artefatos

Fluxos de Trabalho	Profissionais	Artefatos
Engenharia de requisitos	Ergonomista → Cenário Especialista em Fatores Humanos → Cenário *Designer* de Interface de Usuários	Modelo de Usuário Modelo de Tarefas Protótipo de Interface de Usuários
Análise e Projeto	*Designer* de Interface de Usuários	Modelo de Tarefas MIC Modelo de Classe
Implementação	Programador	Protótipos
Teste	*Designer* de Teste	Modelo de Teste

Fonte: Sousa and Furtado, 2003

A principal contribuição de Sousa and Furtado (2003) é a definição de RUPi e de seus primeiros fluxos de trabalho, levando em conta os conceitos de IHC. Cada fluxo de trabalho RUPi sugere a geração de alguns artefatos que pretendem ajudar as equipes no desenvolvimento de um sistema interativo que considere os fatores humanos, os

requisitos de usabilidade, os problemas de acessibilidade e aceitabilidade, a avaliação de usabilidade, os cenários gráficos, entre outros aspectos.

Ferre (2003) pesquisou sobre a integração de técnicas e atividades de IHC aos processos de desenvolvimento de software. O autor entende que o desenvolvimento iterativo é a melhor escolha para o projeto centrado no usuário (PCU), este último comumente utilizado por pesquisadores de IHC. Conforme Benyon (2011) assinala, a adoção da abordagem centrada no usuário para o projeto de sistemas interativos, apesar de possuir um custo adicional para qualquer projeto, é vantajosa por uma série de razões, dentre as quais destaca: o retorno do investimento, o nível de segurança dos sistemas e a ética dos designers envolvidos no projeto.

Nesse trabalho, Ferre (2003) fez uma minuciosa pesquisa na literatura de IHC, visando a um levantamento sobre as técnicas e atividades de usabilidade que melhor se alinham com o processo de desenvolvimento de software. Com essa pesquisa, foram elencadas, após alguns refinamentos, 51 técnicas e atividades de usabilidade. As atividades de usabilidade foram agrupadas de acordo com as três atividades genéricas de processo de software: atividades de análise, atividades de projeto e atividades de avaliação. A Tabela 4.3 apresenta esse agrupamento.

Tabela 4.3: Agrupamento das atividades de usabilidade com as atividades genéricas de processo de software

Atividades de Análise	Atividades de Projeto	Atividades de Avaliação
1. Especificação do contexto de uso (análise do usuário e análise da tarefa) 2. Especificações de usabilidade	1. Desenvolvimento do conceito do produto 2. Prototipação 3. Projeto de interação	1. Avaliação de usabilidade

A partir desse agrupamento, o autor apresenta um mapeamento das atividades de usabilidade para as atividades de desenvolvimento de software, a que ele se refere como a adaptação das atividades de usabilidade para os conceitos e terminologia do processo de desenvolvimento da engenharia de software. O passo seguinte foi a alocação das técnicas de usabilidade às atividades de desenvolvimento. A Figura 4.4 apresenta as técnicas que são propostas para a fase de análise.

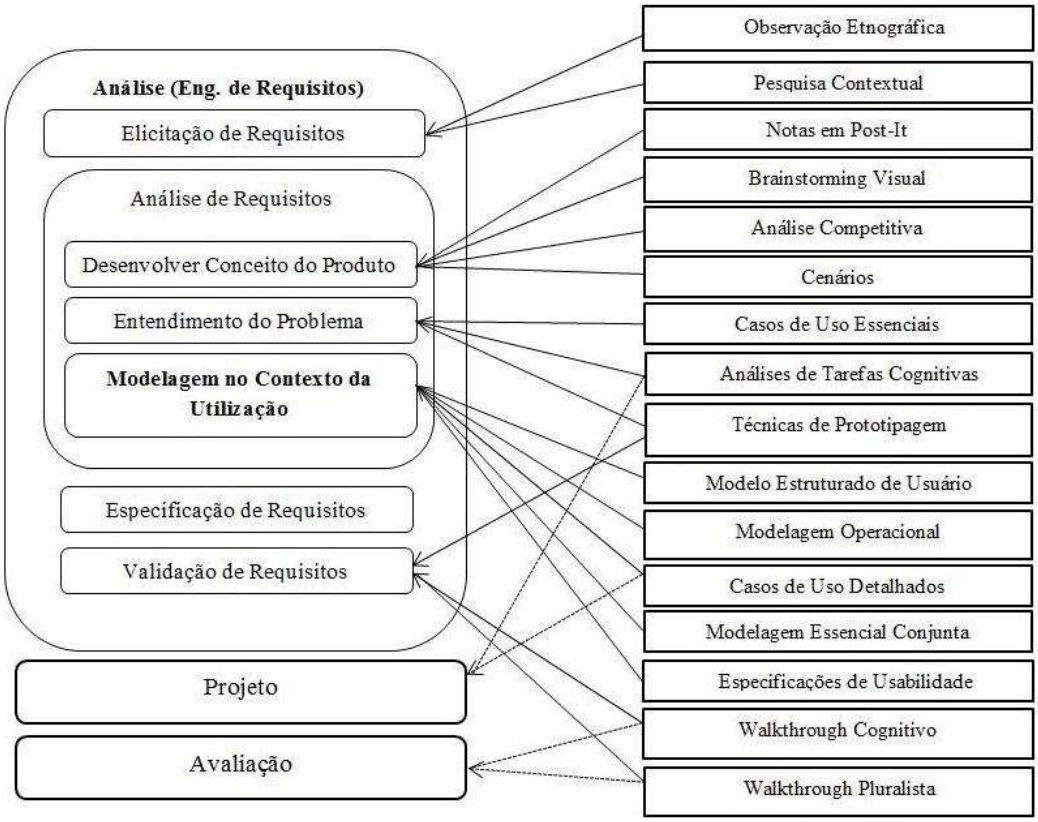


Figura 4.4 – Técnicas de usabilidade aplicadas na etapa de análise

Fonte: Ferre (2003, p.31)

As atividades e técnicas de usabilidade são aplicadas no processo de desenvolvimento como incrementos, chamados de deltas. Cada incremento ou delta é descrito de acordo com as seguintes características: seu propósito, a fase (análise, projeto, avaliação), o estágio do processo de desenvolvimento onde é aplicável, participantes (stakeholders), lista de técnicas de usabilidade do incremento/delta. Ao todo foram definidos sete deltas. O autor afirma que o conjunto de técnicas que formam um delta é organizado de acordo com a atividade para o qual é alocado. Desta forma, o desenvolvedor pode adicioná-lo com mais facilidade ao seu processo de desenvolvimento.

Uma das contribuições deste trabalho foi a criação de um catálogo com breve descritivo das técnicas de usabilidade referenciadas nos deltas. O catálogo tinha como objetivo aumentar a compreensão acerca das técnicas, bem como auxiliar os desenvolvedores na tarefa de criarem sistemas que atendessem aos critérios de usabilidade.

Em Ferre e Bevan (2011) é apresentado o Usability Planner, uma ferramenta on-line desenvolvida para auxiliar na seleção de métodos do PCU para cada estágio do ciclo de vida do processo de desenvolvimento de software. A ferramenta foi construída

considerando as normas ISO PAS 18152:2003 e ISO TR 16982, além das contribuições oriundas da experiência dos próprios autores.

Há uma diversidade de questões que podem influenciar na escolha de um método, como por exemplo, quando a organização depara-se com recursos limitados (quer seja de tempo ou recursos financeiros) para cumprir com o PCU. Desta forma, a organização precisa escolher métodos que sejam comportados pela situação da empresa e que ao mesmo tempo garantam um nível de usabilidade esperado pelo sistema em desenvolvimento.

O *Usability Planner* possui uma sequência de passos para auxiliar o desenvolvedor na tarefa de seleção dos métodos mais apropriados, priorizando os estágios do projeto onde a usabilidade trará mais benefícios ao sistema. Possui duas interfaces distintas: uma para desenvolvedores e outra para profissionais de UX (user experience), pesquisadores e estudantes. O *Usability Planner* está disponível para uso na web gratuitamente[1].

No trabalho de Moraes (2012) foi realizado um estudo com ênfase nas atividades de análise e projeto do processo da WebE, com o intuito de alinhar as ações de ambas as atividades com as técnicas de IHC. A partir de um estudo piloto, recomendações foram propostas com o intuito de assistir ao projeto de sistemas web com as atividades da área de IHC. As seções 5 e 6 apresentam, respectivamente, o estudo piloto e as recomendações para incorporação de atividades de IHC ao Projeto de WebApps

4.5 Estudo Piloto

Durante a nossa pesquisa, realizamos um estudo piloto visando a análise da integração das atividades de IHC com o processo da WebE, através de um projeto interdisciplinar que envolveu várias disciplinas (Engenharia de *Software Web*, Gerenciamento de Projetos, Análise de Pontos por Função, Projeto de Interfaces e Avaliação de Interfaces) do curso de Especialização em Engenharia de Sistemas para *Web* da Universidade Federal de Mato Grosso (UFMT). O estudo foi realizado com duas turmas, uma de 2010 e outra de 2011.

Ao todo foram analisados 11 projetos interdisciplinares, sendo 7 do ano de 2010 e 4 do ano de 2011. A seleção dos trabalhos levou em consideração aqueles cujas atividades de análise e projeto, foram cumpridas na íntegra e que geraram artefatos passíveis de ser analisados.

O estudo piloto teve o papel de identificar e analisar os artefatos gerados na fase de modelagem. Os critérios utilizados nas análises foram baseados na leitura e na verificação daqueles artefatos gerados com métodos, ferramentas e técnicas e que ainda contemplaram as ações significativas para IHC na integração com o processo da WebE.

O projeto interdisciplinar exigia dos alunos a elaboração de artefatos, cumprindo as etapas de comunicação, planejamento, modelagem e parcialmente a etapa de construção de uma aplicação web. Para tanto, ambas as turmas receberam roteiros, os

[1] Disponível em: http://usabilityplanner.org/

quais nortearam a elaboração dos projetos, inclusive com os negócios a serem desenvolvidos na WebApp e com as ações necessárias para cada disciplina envolvida.

Os roteiros deveriam nortear a execução de cada projeto, embora cada equipe tivesse liberdade para elaborar as ações e artefatos, além de utilizar as técnicas e ferramentas que julgassem necessárias.

Como principal objetivo dos projetos interdisciplinares pode-se destacar a vivência prática, se não de todas, pelo menos da maioria das fases da engenharia de sistema para web no desenvolvimento de uma WebApp.

Os negócios abordados pelas WebApps nos projetos interdisciplinares foram os relacionados à divulgação do curso de Especialização em Engenharia para Web, em 2010, e da Copa do Mundo de 2014, em 2011.

Ao longo do projeto interdisciplinar, as atividades de comunicação, planejamento e modelagem foram executadas na íntegra, faltando apenas as tarefas de construção e implantação para contemplar todo o fluxo de processo das atividades do arcabouço ilustrado na Figura 4.2. A cada disciplina cursada, era identificada a atividade em que ela se enquadrava, para entender em qual fase dentro da engenharia de sistemas web as equipes deveriam trabalhar, e ainda, quais ações seriam necessárias.

A utilização de roteiros para orientar os projetos interdisciplinares foi a maneira encontrada pelos professores das disciplinas envolvidas para alinhar as técnicas/métodos da WebE com a IHC. No entanto, se compararmos os roteiros das duas turmas (2010 e 2011), é possível notar um grande avanço dos roteiros do primeiro para o segundo ano, no que se refere ao alinhamento intentado entre as esferas WebE e IHC.

À medida que os grupos iam desenvolvendo as atividades recomendadas, foi possível perceber que várias ações da IHC, as quais deveriam ser realizadas na fase de comunicação, por exemplo, não foram a causa do retrabalho, o que ficou revelado pela diferença significativa entre os roteiros, considerando que atividades antes apresentadas apenas em fases avançadas do projeto ou sequer apresentadas, acabaram sendo devidamente incorporadas e/ou readequadas.

Da percepção dessa diferença, surgiu a ideia de efetuar um estudo bibliográfico e posteriormente exploratório, por meio dos projetos interdisciplinares, para elaborar uma proposta em que, de fato, o processo da WebE e as atividades da IHC estivessem integrados.

Compreender melhor os métodos e técnicas aplicados na atividade de modelagem dentro de um contexto e, com base nisso, identificar fatores que sejam importantes para a incorporação das atividades de IHC e do processo da WebE, é imprescindível para a proposta deste trabalho.

4.5.1 As Ações da WebE e da IHC Aplicadas no Estudo

Como já mencionado, uma das fases do processo de WebE sugerida por Pressman e Lowe (2009) refere-se à modelagem, a qual é composta por atividades de análise e projeto. Considerando que os projetos interdisciplinares cumpriram na íntegra tal fase,

apresentamos nesta seção uma análise dos artefatos gerados a partir das ações de análise e projeto, levando em conta a IHC.

As primeiras ações realizadas na fase de análise foram relacionadas à hierarquia do usuário, ao desenvolvimento de casos de uso e de diagramas e ao refinamento desses casos de uso.

Com a incorporação da etapa de projeto, as ações executadas se pautaram na pirâmide (Pressman e Lowe, 2009, p.172), revelando que os projetos de conteúdo, arquitetura, navegação e interação foram devidamente contemplados em todos os trabalhos. Vale ressaltar que, além desses, ainda existem os projetos de componentes e o funcional; todavia nem todos os trabalhos necessitaram desses projetos.

Assim, apresentaremos primeiramente a atividade de análise e posteriormente a de projeto, onde alguns artefatos serão mostrados com o intuito de contribuir com as recomendações sugeridas neste trabalho para uma melhor integração entre as atividades de IHC e o processo da WebE.

4.5.2 Ações de Análise

Durante a execução dos projetos interdisciplinares, várias ações da etapa de análise foram realizadas e a partir delas foram gerados diversos artefatos, cujas análises serão apresentadas nesta seção com o intuito de contribuir com as recomendações a serem apresentadas neste trabalho para uma melhor integração entre as atividades de IHC e o processo da WebE.

De acordo com Pressman e Lowe (2009, p. 107), a análise visa à produção de modelos, examinando os requisitos dos interessados num esforço para atender o que a WebApp deveria fazer. É justo nessa fase que as necessidades do sistema são compreendidas e modeladas e, quando são gerados os principais artefatos: casos de uso em UML, especificação dos casos de uso, diagramas de objetos em UML, lista com a hierarquia dos usuários e descrição das tarefas por categoria de usuário.

A partir da análise de requisitos realizada, Silva *et al.* (2011-b) geraram artefato com a descrição das características e tarefas, organizando-as por categoria de usuário e baseando-se na hierarquia dos usuários identificada.

Já Penha *et al.* (2011), além do diagrama geral dos casos de uso da aplicação, agregaram outros tipos de diagramas, tais como o diagrama de estado, de atividade e ainda de sequência para demonstrar o comportamento de alguns requisitos do sistema.

Em contrapartida, Oliveira *et al.*(2011) elaboraram o artefato de expansão dos casos de uso, incluindo a regra de negócio, conforme mostra a Figura 4.5. Ou seja, já na análise dos requisitos foi dado o devido destaque para as regras de negócio. A utilização desse recurso contribui para a equipe ficar atenta às especificidades da aplicação.

Caso de Uso Cadastrar Usuário	
Caso de Uso	Cadastrar Usuário
Objetivo	Cadastrar os usuários que terão acesso ao sistema
Requisitos	RF02
Atores	Coordenador/Administrador do sistema
Prioridade	Alta
Frequência de uso	Baixa
Trigger	Administrador seleciona a opção Cadastrar Usuário
Fluxo Principal	1 – O sistema apresenta a tela de cadastro contendo as informações:
	- Nome completo
	- Login
	- Senha
	- Opção Nível de acesso [RN01]
	- Opção Salvar
	- Opção Cancelar
	2 – O ator preenche as informações
	3 – O ator seleciona a opção Salvar. [A1]
	4 – O sistema armazena as informações do novo usuário
	5 – O sistema exibe a mensagem “Usuário cadastrado com sucesso”
	6 – O caso de uso é encerrado
Fluxo Alternativo	[A1] – O ator seleciona a opção Cancelar
	1 – O sistema ignora as informações preenchidas e lista os usuários cadastrados
	2 – O caso de uso é encerrado
Extensões	Não se aplica
Regra de Negócio	[RN01] – Opção dada pelo sistema: “1-Coordenador/Administrador”, “2-Professor”, “3-Funcionário”, “4-Aluno”

Figura 4.5 - Artefato expansão de casos de uso

Fonte: Oliveira *et al.* (2011)

Silva *et al.*(2011-c) criaram um artefato com o uso da técnica de modelo conceitual para a hierarquia dos usuários, onde cada categoria de usuário é representada já com as suas permissões dentro do sistema, ilustrado na Figura 4.6.

No caso das equipes de 2011, os engenheiros de sistemas Teobaldo *et al.* (2012), Januario *et al.* (2012) e Alves *et al.* (2012), fizeram o refinamento dos casos de uso usando a técnica de prototipação, inclusive com *layout* de alta fidelidade. Assim é possível identificar, pelos stakeholders, possíveis equívocos nos requisitos levantados e, consequentemente efetuar o refinamento.

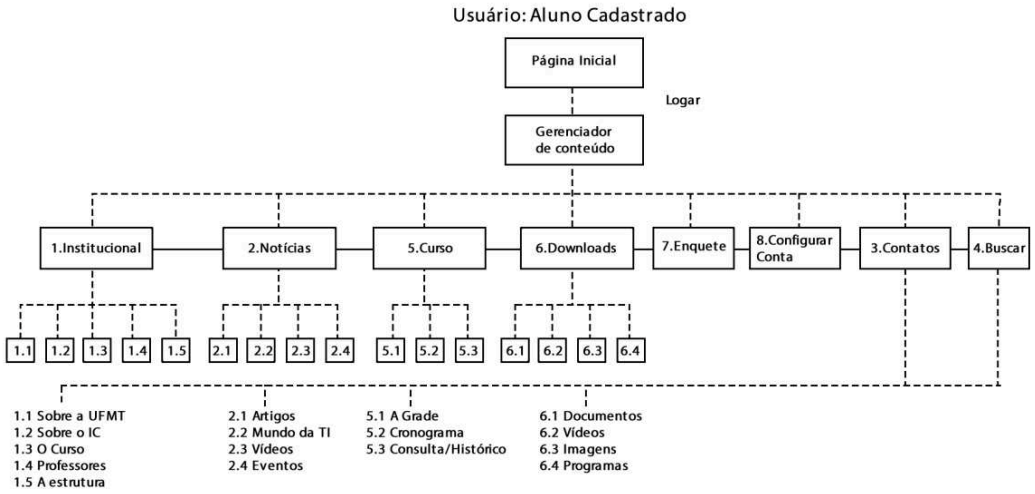


Figura 4.6 - Artefato modelo conceitual da hierarquia do usuário aluno

Fonte: Silva *et al.* (2011-c)

Além disso, outra técnica usada por Teobaldo *et al.* (2012), Januario *et al.* (2012) e Alves *et al.* (2012) foi a de apresentar a modelagem dos dados, campos que fazem parte do sistema em WebML. O uso desse recurso já contribui para a elaboração do banco de dados da WebApp.

Nesta etapa podem ser desenvolvidas também especificações de casos de uso completas contendo percursos de interação via *wireframe* visando a rastreabilidade entre os dois elementos. Essa e outras ações e técnicas utilizadas pelas equipes e apresentadas nesta seção foram de grande valia na medida em atividades de IHC, como por exemplo, modelo conceitual da hierarquia de usuários, foram incorporadas ao processo de desenvolvimento, além de terem sido gerados diversos modelos para melhor compreensão da WebApp.

4.5.3 Ações de projeto

Assim como na etapa de análise, a fase de projeto também gerou diversos artefatos através de várias ações. Tais artefatos também serviram de embasamento para que pudéssemos sugerir uma interligação entre a IHC e o processo da WebE, em especial, na etapa de modelagem.

De acordo com Pressman e Lowe (2009) a atividade de projeto trata de compreender a estrutura interna da WebApp para criar a sua forma a partir do que foi identificado na atividade de análise.

Nas seções subsequentes vamos apresentar as ações do projeto individualmente e de acordo com os artefatos gerados nos trabalhos interdisciplinares. Nem todos os projetos interdisciplinares contemplaram todas as ações da pirâmide, quais sejam, projeto de interface, estético, conteúdo, arquitetura, navegação, componente e funcional. Geralmente, os projetos de interface e estético formam um único projeto, o de interação.

4.5.3.1 Projeto de Conteúdo

O principal artefato gerado nesta ação é a descrição dos conteúdos e elementos necessários para a aplicação.

Para Silva *et al.* (2011-b), o projeto do conteúdo do sistema foi descrito detalhadamente, apresentando todos os rótulos e de acordo com cada página e ambiente da aplicação.

Por outro lado, Silva *et al.* (2011-c) deram destaque ao projeto de conteúdo nas informações de multimídia, o que é de grande valia, pois se formos considerar que cada usuário possui um tipo de computador, bem como um link de internet para realização do acesso ao sistema, mencionar no projeto de conteúdo informações como o tamanho máximo de imagens permitidas no site e sua dimensão, e ainda, que os vídeos aceitos são apenas aqueles postados no YouTube, são fatores dos quais não se pode prescindir.

Podemos observar que é importante a elaboração do projeto de conteúdo, pois apresenta toda a informação que deve conter a WebApp, além desta ser a base para as outras ações de projeto.

4.5.3.2 Projeto de Arquitetura

Nesta ação, os principais artefatos gerados são baseados em prototipação, como *storyboarding* e *wireframe*. Penha *et al.* (2011) utilizaram *wireframes* para apresentar as informações contidas nas páginas do *site*; todavia incluíram notas de esclarecimentos sobre as ações que seriam realizadas pelo usuário. Já Silva *et al.* (2011-c) usaram *wireframes* com informações claras e objetivas que devem ser apresentadas por perfil de usuário. Veja a Figura 4.7.

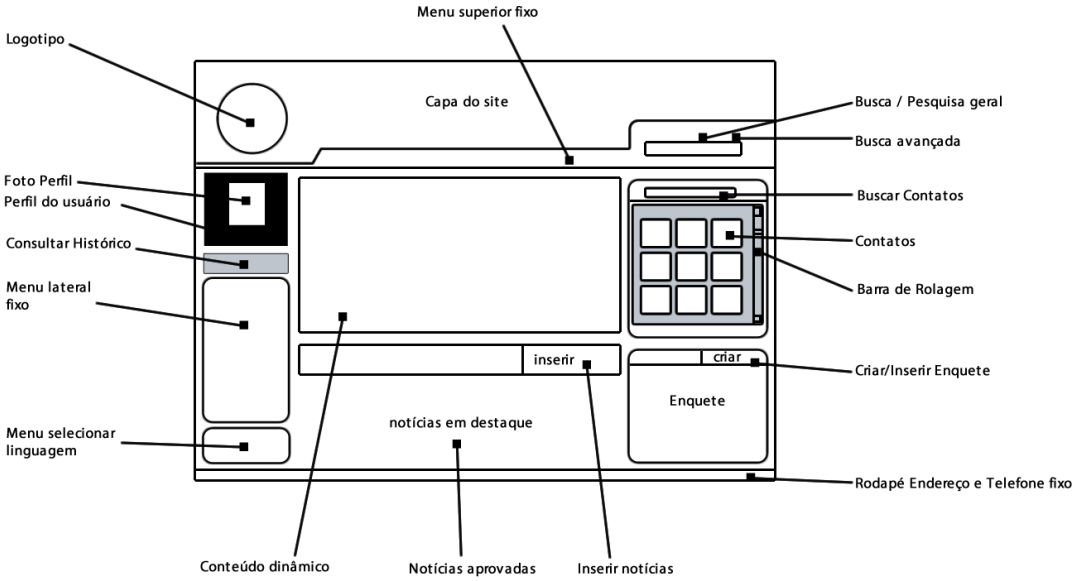


Figura 4.7 - Artefato *wireframe* de conteúdo

Fonte: Silva *et al.* (2011-c)

Como podemos constatar, a disponibilidade das informações e consequentemente o seu refinamento pelos envolvidos, podem ser realizados através das técnicas de *storyboarding* e *wireframes*.

4.5.3.3 Projeto de Navegação

Nesta ação, os artefatos principais gerados são mapas de navegação. O projeto de navegação garante navegabilidade entre os objetos de conteúdos e funções da WebApp.

Em Silva *et al.* (2011-b), o mapa de navegação com ênfase no menu vertical, artefato gerado e incorporado ao projeto, apresenta também o conjunto de ações de determinado perfil de usuário.

De acordo com Januario *et al.* (2012), além do mapa de navegação, é importante que seja desenvolvida a definição da semântica de navegação dos casos de uso, utilizando-se a técnica dos diagramas de sequência. Com essa mesma ideia, Teobaldo *et al.* (2012) propõem também a definição semântica de navegação; só que por meio das técnicas dos diagramas de classes. Vide Figura 4.8.

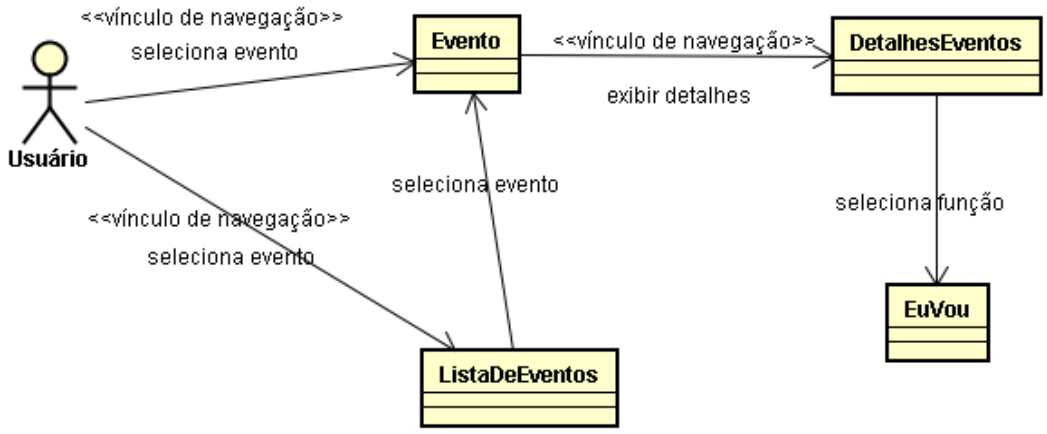


Figura 4.8 - Artefato definição da semântica de navegação

Fonte: Teobaldo *et al.* (2012)

Já Alves *et al.* (2012), usam as ferramentas FreeMind, Astah e Visio para apresentarem os mapas de navegação, garantindo uma estrutura em rede com menu horizontal e um melhor entendimento da navegabilidade na WebApp.

É importante que haja a possibilidade de efetuar a navegabilidade entre as funcionalidades do sistema, para possibilitar uma maior facilidade e acessibilidade pelos usuários da aplicação.

4.5.3.4 Projeto de Interface e Projeto Estético

Os projetos de *interface* e estético formam o projeto de interação e contam com artefatos como *wireframes* e *layouts* em alta fidelidade. É uma das ações mais importantes e trabalhosas da pirâmide, pois é por meio da *interface* que o usuário final acessará a aplicação.

A par dessa complexidade, os engenheiros de sistemas utilizaram várias técnicas e ferramentas gerando diversos layouts e artefatos já em alta fidelidade para avaliação dos stakeholders.

De acordo com Silva *et al.* (2011-b), na interface gerada para a aplicação foram levados em consideração principalmente o mapeamento dos objetivos, e também as principais funções da aplicação. A técnica usada foi o *wireframe* para representar tais recursos, conforme demonstra a Figura 4.9.

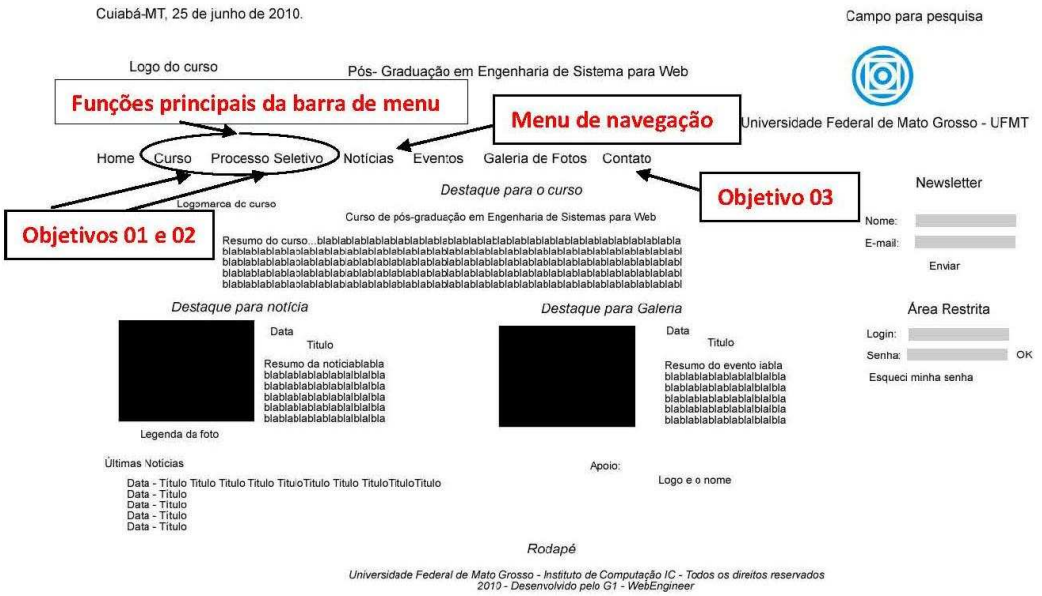


Figura 4.9 - Artefato *wireframe* das funções e objetivos da WebApp

Fonte: Silva *et al.* (2011-b)

Cumpre destacar que o projeto de interação desenvolvido pelos engenheiros Silva *et al.* (2011-b) também incluiu diretrizes, tais como acessibilidade, padronização, legibilidade e usabilidade, e ainda as recomendações sobre os relatórios de mensagens do sistema, que são imprescindíveis na fase de implementação.

Ademais, foi possível efetuar o refinamento do *layout*, a priori sugerido, através da aplicação dos conceitos estéticos: cores, fontes, linguagem, tamanho das fontes e consequentemente o projeto gráfico, resultando em um *layout* de alta fidelidade, como podemos ver na Figura 4.10.

Figura 4.10 - Artefato *layout* de alta fidelidade

Fonte: Silva *et al.* (2011-b)

No projeto de interação, segundo Silva *et al.* (2011-a), deve haver um padrão de posicionamento dos elementos de interface. Tudo isso, para atender as convenções em aplicações web. A descrição das mensagens do sistema por funcionalidade também é um artefato de extrema importância e que foi elaborado.

Para Teobaldo *et al.* (2012), os critérios de usabilidade foram decisivos na elaboração do projeto de interação, o que justificou a pesquisa antecipada acerca dos padrões estéticos em vários sites com a mesma proposta de negócio e ainda com usuários semelhantes. E mais, sempre com a preocupação de atender a usabilidade, acessibilidade e internacionalização.

Nessa mesma vertente, Januario *et al.* (2012) acrescentaram ao projeto de interação a necessidade de identificar padrões estéticos para se adequar a usabilidade.

Identificar problemas e posteriormente corrigi-los a fim de atender satisfatoriamente a IHC, é o objetivo da realização de avaliações e validações mediante uso do *layout* de alta fidelidade ou protótipos interativos.

4.5.3.5 Projeto Funcional

O projeto funcional possui atividades relacionadas às funcionalidades em nível de usuário e em nível de aplicação. Neste caso, iremos apresentar os artefatos gerados em nível de usuários, onde é possível identificar uma representação comportamental da *interface*. Normalmente, os principais artefatos gerados são os diagramas de caso de uso, sequência, atividade e estado.

Para os engenheiros de sistemas Silva *et al.* (2011-b), Wollinger *et al.*(2011), Penha *et al.* (2011), Oliveira *et al.* (2011), Silva *et al.* (2011-a) e Galvão *et al.* (2011), o projeto funcional tem como base fundamental a interação do usuário com a aplicação, levando em consideração as tarefas a serem realizadas por cada perfil de usuário.

Por isso, ao elaborarem os artefatos, as técnicas de diagramas de sequência, estado e atividades foram feitas de acordo com os casos de uso envolvidos em cada negócio. Assim, apresentamos a Figura 4.11 referente ao diagrama de estado.

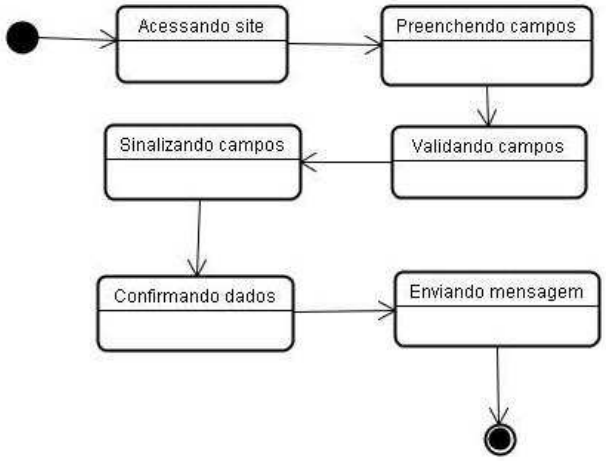


Figura 4.11 - Artefato diagrama de estado

Fonte: Silva *et al.* (2011-b)

Os engenheiros de sistemas Alves *et al.* (2012), por outro lado, elaboraram o projeto funcional associado ao de interação, utilizando-se do software MOLIC Designer para especificar as tarefas dos usuários.

A utilização do projeto funcional no processo de WebE evita o reprojeto, em virtude de representar os comportamentos do sistema de acordo com o perfil de usuário.

4.5.3.6 Projeto de Componente

O projeto de componente é opcional, e só é elaborado caso os engenheiros se utilizem desse recurso na aplicação, o que se verificou em dois projetos interdisciplinares. Normalmente os principais artefatos gerados são diagramas em UML, que melhor

representam o comportamento do sistema junto ao componente, e a descrição das características da ferramenta.

Wollinger *et al.*(2010) abordaram no artefato gerado, além da ferramenta que trata do componente Moodle, os pré-requisitos para a sua instalação, os padrões, as funcionalidades e ainda o mapa conceitual da WebApp integrado ao Moodle, conforme mostra a Figura 4.12:

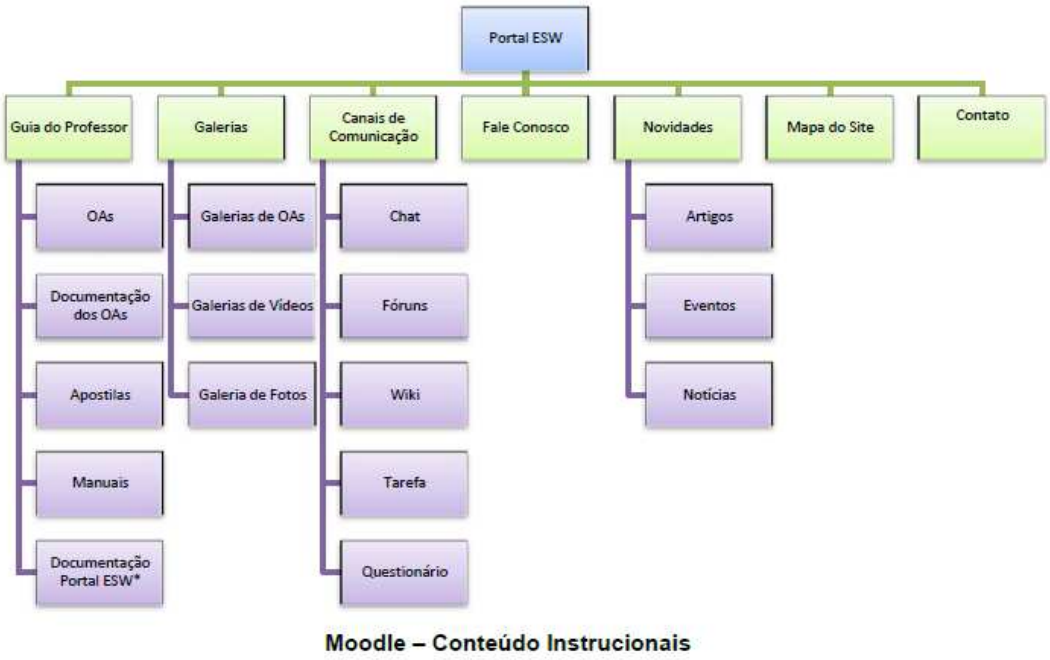


Figura 4.12 - Artefato mapa conceitual

Fonte: Elaborado por Wollinger *et al.* (2011)

Por questão de preferência, Silva *et al.* (2011-a) mencionaram na atividade de levantamento de requisitos que utilizariam o componente Joomla, o qual ficaria responsável pelo gerenciamento do conteúdo da WebApp, por se tratar de uma ferramenta consolidada no mercado e de código aberto.

Sempre que um componente for incorporado a uma aplicação, é importante que o projeto seja elaborado para não permitir que haja retrabalho ao longo do processo de WebE.

Finalmente, os projetos interdisciplinares a que nos referimos neste estudo piloto encontram-se na íntegra no Instituto de Computação (IC) da UFMT, podendo ser consultados para dirimir possíveis dúvidas relacionadas às atividades que envolvem a engenharia de sistema para web.

É importante ressaltar que o roteiro da 1ª turma seguiu os passos da pirâmide de projeto proposto por Pressman e Lowe (2009). Através da percepção dos professores das disciplinas envolvidas no projeto interdisciplinar bem como de suas análises sobre os projetos apresentados, pode-se perceber a necessidade de retrabalhar a organização

dos processos apresentados pelo trabalho dos autores supracitados. Desta forma, com a 2ª turma, o roteiro teve uma nova vertente, onde algumas fases foram redefinidas visando atender claramente as especificidades de um processo iterativo com ciclos alternantes de projeto, teste, avaliação e reprojeto ao mesmo tempo em que se buscava a agilidade do desenvolvimento web.

Com os resultados da segunda experiência percebemos a necessidade de definir uma proposta para a incorporação de atividades de IHC ao projeto de uma WebApp. Esta proposta, apresentada na seção 4.6, tem como foco a fase de projeto.

4.6 Incorporação de Atividades de IHC ao Projeto de WebApp

As atividades de comunicação, planejamento, modelagem, construção e implantação, formam o fluxo do processo da *WebE*, conforme atestam Pressman e Lowe (2009). Neste trabalho, investigamos as ações relativas à etapa de projeto que compõe, junto com a etapa de análise, a fase de modelagem.

Embora entendêssemos que para contemplar de forma efetiva a IHC, as ações devem ser executadas de forma iterativa, tendo o usuário como o centro das atividades, escolhemos trabalhar na etapa de projeto em virtude dela apresentar uma técnica. Esta etapa é baseada em uma pirâmide na qual cada nível representa uma ação de projeto, onde são executadas em sequência. Cada um desses níveis representa um projeto, sendo que aqueles que ficam no nível superior são os mais considerados pela IHC.

Em consequência disso, problemas, como, o reprojeto nas ações de projeto e retrabalho nas atividades de análise podem ocorrer, devido à falta da aplicabilidade de atividades e diretrizes da IHC de forma interligada com a etapa de projeto da *WebE*.

Logo, é essencial que haja uma integração entre todos os níveis da pirâmide levando em consideração a IHC, não apenas evidenciando o usuário nos níveis superiores, mas também na ordem cronológica das execuções das ações do projeto de acordo com o processo da *WebE*.

Para tanto, as ações de análise e especialmente de projeto devem ser refinadas, inclusive com a reformulação do modelo baseado na pirâmide, sendo proposto outro modelo que tenha o usuário do sistema como centro dos projetos, permitindo que as ações dos projetos possam ser executadas de maneira cíclica.

Segundo, Barbosa e Silva (2010),

> Mesmo após a definição da interação e da *interface*, e durante o projeto e a implementação do sistema, pode ser necessário também rever o projeto de IHC quando surgir alguma dificuldade para construir a solução projetada ou quando algo não tiver sido completamente especificado (Barbosa e Silva, 2010, p. 126).

Em suma, é importante que haja o alinhamento das atividades de IHC e do processo da *WebE*, para evitar-se o reprojeto da *WebApp* e consequentemente retrabalho.

Diante do exposto, apresentamos na Figura 4.13, uma nova versão das ações de projeto, baseadas na pirâmide, de forma integrada com os aspectos da IHC:

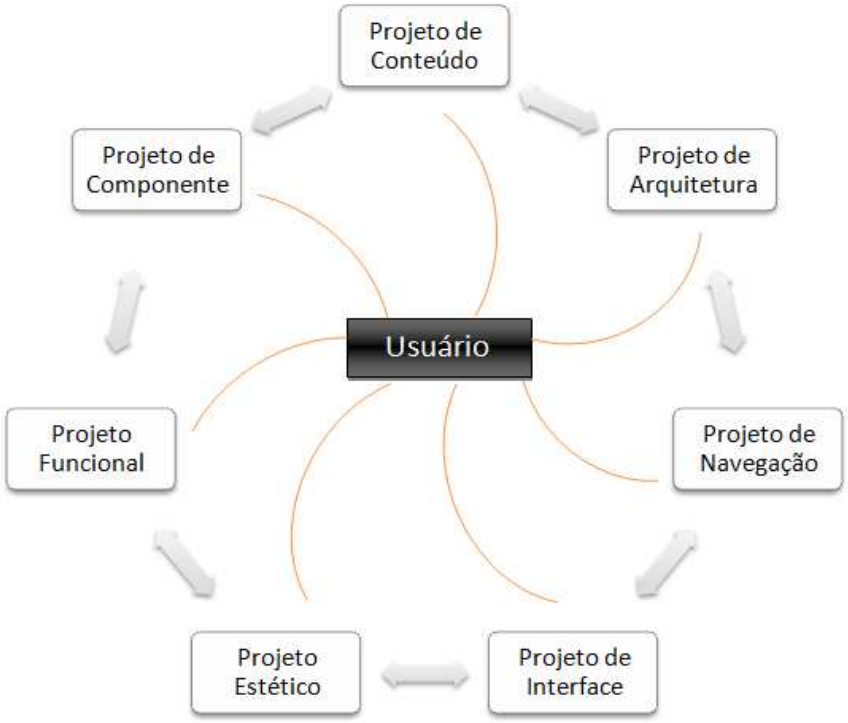


Figura 4.13 - Nova proposta com as ações de projeto da WebApp

Nota-se, que a ação de construção, a qual forma a base da pirâmide, não foi contemplada pela nova proposta. Isto porque consideramos essa ação como a etapa que deve ser executada posterior a de modelagem dentro do processo da *WebE*.

Outro fator relevante é que o primeiro projeto a ser elaborado deve ser o de conteúdo, pois é ele que dará base para os demais e, caso necessário, poderá ser revisto dentro do ciclo, já que as ações seguem um fluxo cíclico.

A nova proposta apresenta os projetos de forma cíclica, onde as ações necessárias para elaboração de cada um deles interagem na medida em que for identificada essa necessidade.

Na próxima seção são detalhados cada um dos projetos do modelo proposto na Figura 4.13.

4.6.1 Recomendações para elaboração de WebApp

Esta seção apresenta algumas recomendações para elaboração de WebApp com o objetivo de integrar as atividades da IHC com o processo de *WebE* utilizando como base as pesquisas bibliográficas e o estudo piloto.

Na etapa de modelagem, as atividades desenvolvidas compõem-se de análise dos requisitos levantados na fase de comunicação e da elaboração das ações de projeto.

Assim, a atividade de análise deve ser executada primeiramente através de ações, tais como:

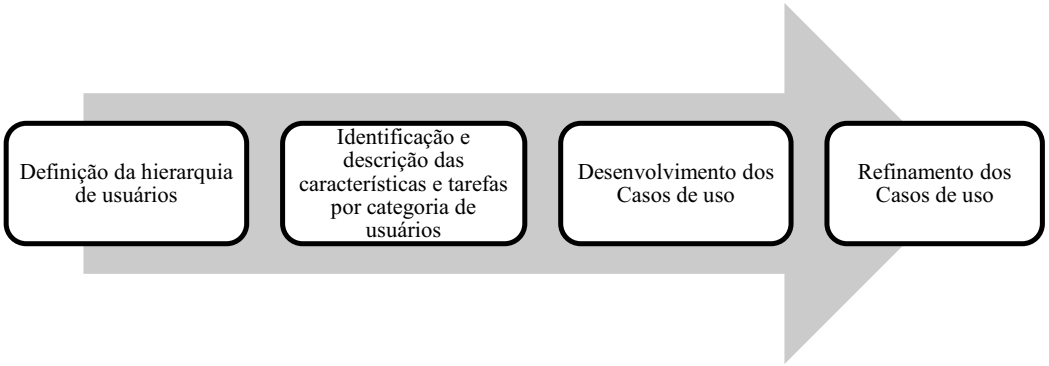


Figura 4.14 - Fluxo com as atividades da fase de modelagem – análise dos requisitos

É essencial que as referidas ações sejam feitas na sequência apresentada e que se utilizem das técnicas e ferramentas descritas a seguir:

Definição da hierarquia de usuários: deve ser elaborado um modelo representativo da hierarquia de usuários, sendo necessário o uso de ferramenta para elaboração de diagramas, por exemplo, o de classe. Com isso, os perfis de usuários já ficam estabelecidos.

Identificação e descrição das características e tarefas por categoria de usuários: apresentação com a identificação e descrição de características e das tarefas permitidas por categoria/perfil de usuário através do uso de *templates* ou de diagramas em UML.

Elaboração dos casos de uso: desenvolver diagramas através de ferramentas como *Astah*, além de realizar a expansão desses casos de uso utilizando informações tais como, grau de importância e as regras de negócios.

Refinamento dos casos de uso: poderá efetuar o uso de percurso de interação via protótipo (*storyboard* ou *wireframe*), bem como de diagramas da modelagem.

Para a atividade de projeto, sugerimos iniciar as tarefas pelos projetos de conteúdo e de arquitetura da informação. Assim, as ações necessárias para a *WebApp* são:

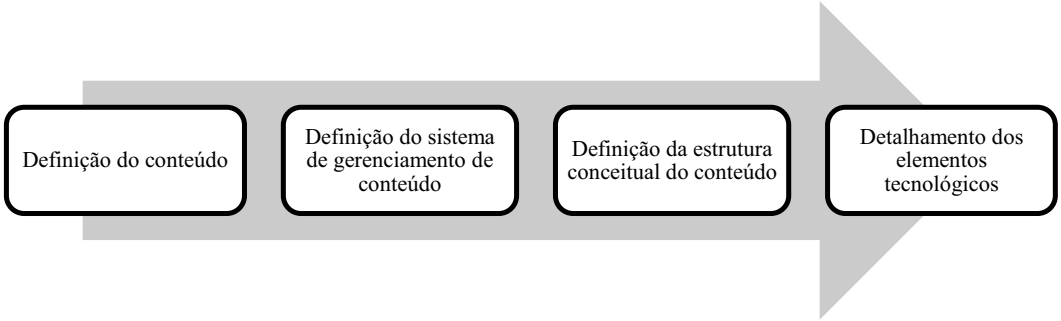


Figura 4.15 - Fluxo com as atividades da fase de modelagem – projetos de conteúdo e de arquitetura

Apresentamos então, algumas técnicas e ferramentas que podem ser utilizadas:

Definição do conteúdo: descrição dos elementos de acordo com os objetivos do negócio, evidenciando os rótulos existentes na aplicação, inclusive fazendo referência às mídias sociais, caso existam, e ainda usando a técnica de metadados, *template* que apresente as informações que a aplicação deve conter de maneira organizada.

Definição do sistema de gerenciamento de conteúdo: tarefa que deve ser realizada mediante pesquisa de mercado para identificar o gerenciador que melhor se adéqua ao negócio e ainda para constatar a viabilidade do seu uso.

Definição da estrutura conceitual do conteúdo: para apresentar a disposição do conteúdo na *WebApp*, é essencial a criação de esquemas identificando os objetivos elencados na definição do conteúdo. Além disso, as técnicas de prototipação podem ser utilizadas através de *storyboardings* e *wireframes*.

Detalhamento dos elementos tecnológicos: existem sistemas em que pode ser necessário disponibilizar ambientes com recursos tecnológicos para atender o negócio. Em função disso, devem-se realizar pesquisas de mercado para identificar e garantir a viabilidade do(s) melhor(es) recurso(s) a ser(em) utilizado(s) e, posteriormente fazer-se a descrição dos elementos selecionados nesta ação.

Em seguida, é recomendável o desenvolvimento do projeto de navegação, o qual permitirá a interação entre usuário e sistema, levando em conta inclusive, as permissões por categoria de usuários. Por isso, as principais atividades a serem realizadas são:

Figura 4.16 - Fluxo com as atividades da fase de modelagem - projeto navegação

Poderão ser utilizadas para contemplar tais atividades, as seguintes ferramentas e técnicas:

Mapa de navegação: a estrutura do sistema deve ser montada para melhor realizar a navegabilidade, podendo ser usadas várias ferramentas, inclusive *Astah*, *Visio, FreeMind* e as técnicas de diagramas em *UML*. Além disso, para nortear a criação desses artefatos são necessárias as respostas para as três perguntas: onde estou?, onde estive? aonde posso ir? (Nielsen, 2000 *apud* Barbosa e Silva, 2010).

Definição da semântica de navegação: efetuar de forma clara a descrição dos rótulos do sistema e ainda, representá-los através de tabelas ou de diagramas em *UML*.

Descrição da sintaxe da navegação: identificar o tipo de menu, se horizontal e/ou vertical e as particularidades na navegabilidade, como, os recursos de ajuda, por exemplo.

Logo após, deve ser elaborado o projeto de interação, o qual deve incluir atividades tais como:

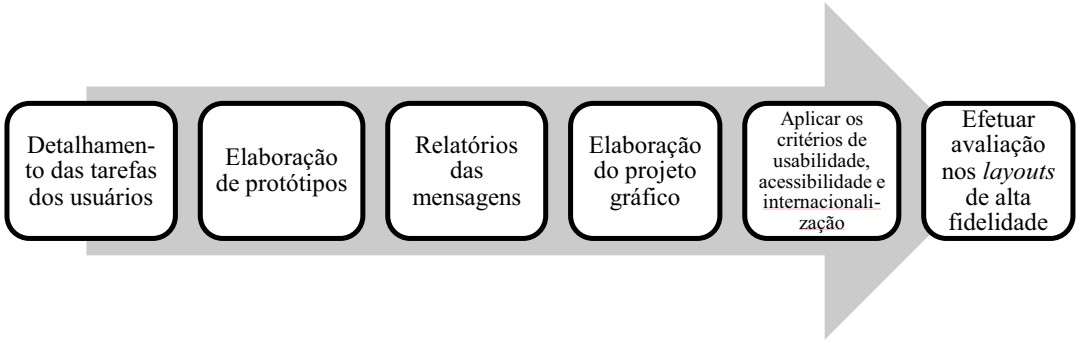


Figura 4.17 - Fluxo com as atividades da fase de modelagem - projetos estético e de interface

Para tanto, algumas técnicas e ferramentas poderão ser usadas nas referidas atividades:

Detalhamento das tarefas dos usuários: poderá ser feito com o uso de diagramas em *UML* e com a elaboração de prototipação.

Elaboração de protótipos: para elaboração de protótipos é essencial o uso da técnica de *wireframe,* já apresentando a estrutura do conteúdo, identificada no projeto de conteúdo e arquitetural. A aplicação, nesses protótipos, de padrões convencionados de elementos de *interface* usados na *web* é igualmente importante.

Relatórios das mensagens: é necessário que as mensagens apresentadas pelo sistema sejam organizadas pelas funcionalidades/páginas existentes com uso de linguagem simples, objetiva, padronizada e ainda de acordo com o público alvo.

Elaboração do projeto gráfico: utilizar palheta de cores, icnografia e tipografia para desenvolver, de acordo com os protótipos já elaborados, o *layout* de alta fidelidade. É importante que as cores sejam escolhidas a partir do perfil do usuário e do negócio.

Aplicar os critérios de usabilidade, acessibilidade e internacionalização: destacar, nos próprios *layouts* de alta fidelidade criados na atividade anterior, os critérios usados.

Efetuar avaliação nos *layouts* de alta fidelidade, conforme as heurísticas sugeridas por Nielsen (1994 *apud* Barbosa e Silva 2010): após a finalização dos *layouts* em alta fidelidade, deve-se aplicar a avaliação das heurísticas.

Em seguida, verificar se há necessidade de elaborar o projeto de componente, que é opcional. Em caso afirmativo, as atividades abaixo contemplam tal projeto:

Figura 4.18 - Fluxo com as atividades da fase de modelagem - projeto de componente

Assim, algumas técnicas e atividades são imprescindíveis para este projeto:

Descrição e análise do componente: descrição do componente deve ser bem organizada para facilitar o entendimento do uso do componente.

Modelo de integração do sistema com o componente adotado: elaboração de diagrama em *UML* para representar a integração da aplicação com o componente.

Pesquisa de mercado: devem ser utilizadas, para levantar informações sobre o uso e a viabilidade do componente para o projeto.

Para finalizar as recomendações, o projeto funcional deve ser desenvolvido de modo a identificar a relação existente entre os usuários e as funcionalidades da aplicação e entre o comportamento das próprias funcionalidades contidas no sistema, conforme fluxo a seguir:

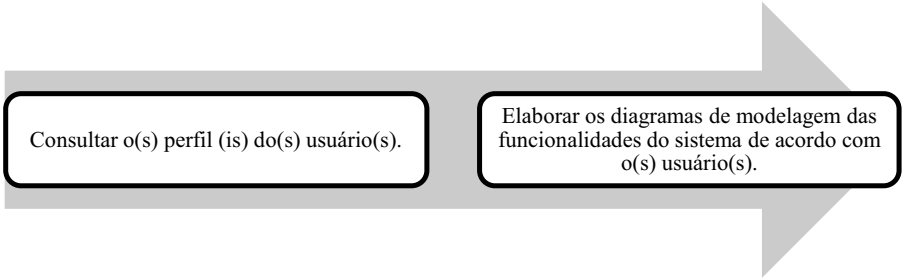


Figura 4.19 - Fluxo com as atividades da fase de modelagem - projeto funcional

As duas seguintes atividades devem ser executadas:

Consultar o(s) perfil(is) do(s) usuário(s): as informações contidas no artefato de requisitos funcionais e não funcionais devem ser consultadas e utilizadas levando sempre em consideração o perfil de usuário e, a representatividade das funcionalidades, do sistema.

Elaborar os diagramas de modelagem das funcionalidades do sistema de acordo com o(s) usuário(s): a partir da consulta realizada na atividade anterior, modelar através de diagramas de atividade, sequência e/ou estado, as funcionalidades que já foram identificadas na fase de formulação do sistema, que também devem ser organizadas por perfil de usuário e/ou sistema.

Esse conjunto de atividades e técnicas formam dentro do processo da *WebE*, o corpo da *WebApp*, lembrando que a próxima etapa já é a de construção, seguida dos testes e enfim da implantação. As recomendações sugeridas neste trabalho evidenciam os usuários finais para que seja contemplada a IHC, conforme o modelo de processo da *WebE.*

Espera-se que a adoção destas recomendações contribua com a academia, de modo que os discentes tenham a possibilidade de vivenciar essa prática em eventuais estudos de casos da *WebE*, e ainda com as fábricas de *software*s *web* no mercado de trabalho.

Além disso, é importante frisar que esta proposta pode ser adaptada a qualquer modelo de processo de desenvolvimento de aplicações *web*.

Finalmente, teremos dessa forma, as atividades, técnicas e ferramentas necessárias para uma Interação Humano-Computador em conformidade e devidamente alinhada com o processo da *WebE*, em especial, na etapa de modelagem.

4.7 Considerações Finais

O principal objetivo deste trabalho foi o de investigar, com base na pesquisa bibliográfica e na exploratória, como devem ser interligadas as atividades da IHC com o processo da WebE para então propor recomendações que melhor contemplem o alinhamento entre ambas as esferas com vistas à eficácia da WebApp.

Neste trabalho efetuaram-se pesquisas que contemplaram a IHC e a WebE, com o propósito de contribuir para uma WebApp de qualidade e que atenda as necessidades dos diversos usuários que acessam diariamente sistemas web.

A pesquisa bibliográfica nos mostrou, de acordo com a proposta de WebE dos autores Pressman e Lowe (2009), que apesar de existirem atividades dentro do processo da WebE que sejam válidas e importantes, elas ainda necessitam ser readequadas para que o usuário final do sistema consiga interagir eficientemente.

Outro fator importante apresentado na referida pesquisa refere-se ao projeto de interação, onde foram apontadas várias técnicas e também a importância de cada uma delas para o usuário final da aplicação, como o uso de prototipação e as heurísticas de Nielsen (1994 *apud* Barbosa e Silva, 2010), por exemplo.

É importante frisar que, mesmo nos baseando em apenas um processo de desenvolvimento da engenharia web, a compreensão das suas ações e ainda a apresentação de possibilidades para atender a IHC já foi extremamente enriquecedora.

Aplicar a pesquisa exploratória através do estudo piloto, em que trabalhos interdisciplinares do curso de especialização da UFMT que se utilizaram do processo de WebE contribuiu satisfatoriamente com as recomendações resultantes.

Além disso, tais recomendações poderão ser aplicadas não somente a sistemas web, mas também a aplicativos móveis, o que as tornam mais significativas para os engenheiros de sistemas. Uma vez compreendidas as recomendações sugeridas, os planos de atividades podem ser criados para serem aplicados, mediante adaptação, em qualquer WebApp.

Das recomendações apresentadas, uma das mais interessantes é o uso de prototipação para interação do usuário com o sistema, antes da sua construção e implantação, o que evita possíveis falhas de requisitos.

Finalmente, resta-nos apontar como desafios para este trabalho o desenvolvimento de mais experimentos seguindo o modelo cíclico e as recomendações apresentadas na seção 4.6 com o intuito refinar e adicionar novos elementos a proposta, incorporando ainda, as técnicas clássicas de usabilidade. Também, a proposta pode avançar na investigação das outras fases da WebE, quais sejam, planejamento, construção e implantação.

Referências

Alves, F; Senci, R., Odenor, V., Maia, J. (2012) "Sistema Lugarpraficar.com", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

Barbosa, S. D. J. e Silva, B.S., Interação Humano-Computador, Elsevier, 2010.

Benyon, D., Interação Humano-Computador, 2ª ed., Pearson, 2011.

Da Silva, P. P. and Paton, N. W. (2000) "UMLi: The Unified Modeling Language for Interactive Applications", In: <<UML>> 2000 - The Unified Modeling Language: Advancing the Standard (3rd International Conference on the Unified Modeling Language, York, United Kingdom), LNCS, Vol. 1939, p. 117-132.

Ferre, X. (2003) "Integration of Usability Techniques into the Software Development Process", In: Bridging the Gaps between Software Engineering and Human-Computer Interaction. International Conference on Software Engineering, ICSE'03. Portland, Oregon, p. 28-35.

Ferre, X. and Bevan, N. (2011) "Usability Planner: A Tool to Support the Process of Selecting Usability Methods", In: Lecture Notes in Computer Science, 2011, Vol. 6949, Human-Computer Interaction – INTERACT 2011, p. 652-655.

Folmer, E., Van Welie, M. and Bosch, J. (2006) "Bridging patterns: An approach to bridge gaps between SE and HCI", In: Information and Software Technology. Vol. 48. Elsevier: Feb, 2006. p. 69–89. http://www.sciencedirect.com/science/ article/pii/S0950584905000273. July.

Galvão, C.; Silva, E. F.; Jesus, F. A.; Barbieri Neto, F.; Pacheco, T. C. (2011) "Projeto EngWeb", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

Gould, J. D., Lewis, C. H. (1985) "Designing for usability: key principles and a what designers think", In: Communications of the ACM, 28 (3), p. 300-311.

Januário, A. J. S., Gomes, S. G., Dias, R. A. (2012) "Copa Voluntária: Uma aplicação para gerir e integrar voluntários", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

Moraes, L. A. O. (2012) "Alinhando a Interação Humano-Computador com o Processo da Engenharia Web", Monografia de Especialização em Engenharia de Sistemas Web, Instituto de Computação, UFMT.

Oliveira, E. J. C.; Anjos, G. A. R.; Souza, R. R.; Santos, R. C.; Oliveira, T. Y. (2011) "Projeto Hermes", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

Penha, A., Borges, C., Luiz, E., Magno, G., Souza, I., Pinheiro, W. (2011) "Oráculo", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

Preece, J., Rogers, Y. and Sharp, H. Design de Interação: Além da interação homem-computador. Possamai, V. (trad.), 1ª ed., Bookman, 2005.

Pressman, S. R. e Lowe, D. Engenharia Web. Vieira, D. (trad.), 1. ed., Livros Técnicos e Científicos, 2009.

Silva, A. C., Silva, J. C. A., Penteado, R. A. D., Silva, S. R. P. (2004) "Aplicabilidade de Padrões de Engenharia de Software e de IHC no Desenvolvimento de Sistemas Interativos", In: IV Congresso Brasileiro de Computação - CBComp, p. 118-123.

Silva, A. L. R., Ramos, A. A. A., Pessoa, T. L., Gois, A. G. V., Freitas, M. M. (2011-a) "Portal web ESW", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

Silva, V. A., Alcoforado Neto, J. G., Moraes, L. A. O., Karling, D. M, Amorim, J. (2011-b) "Projeto G1: WebEngineer", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

Silva, A. B., Evandir, E, Macente, W. (2011-c) "SPGESW", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

Sousa, K. S. and Furtado, E. (2003) "RUPi - A Unified Process that Integrates Human-Computer Interaction and Software Engineering", In: Proceedings of the International Conference on Software Engineering (ICSE), Portland, Oregon.

Souza, P. C. e Freiberger, E. C. (2011) "A Prática do Projeto Interdisciplinar como Suporte ao Ensino de IHC", In: X Simpósio Brasileiro de Fatores Humanos em Sistemas Computacionais, IHC2011, Anais Estendidos X IHC2011, Porto de Galinhas, PE.

Teobaldo, G. R. B., Rodrigues, J. L. A., Moraes, H. A. S., Cerqueira, T. R. (2012) "Pequi de Festa", Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

UMLi. The Unified Modeling Language for Interactive Applications. http://trust.utep.edu/umli/, Acesso em: 10 jul. 2011.

Valverde, F., Panach, I.,; Pastor, O. (2007) "An abstract interaction model for a MDA software production method", In: Tutorials, Posters, Panels and industrial Contributions at the 26th International Conference on Conceptual Modeling, Vol. 83, New Zealand.

Wazlawick, R. S. Análise e Projeto de Sistemas de Informação Orientados a Objetos, 2ª ed., Campus, 2011.

Willshire, M. J. (2003) "Where SE and HCI Meet: A Position Paper". In: Bridging the Gaps between Software Engineering and Human-Computer Interaction. International Conference on Software Engineering, ICSE'03. Portland, Oregon, p. 57-60.

Wollinger, A., Carneiro, S.O. (2011) “Last Minute”, Trabalho interdisciplinar do curso de especialização em Engenharia de Sistema para Web, Instituto de Computação, UFMT.

In Cooperatior

© Sociedade Brasileira de Computação, SBC

View publication stats