

DANIELLA INÁCIO DE BARROS

REQUISITOS DE USABILIDADE E EXTENSÕES UML PARA
APLICAÇÕES EM AMBIENTES MÓVEIS

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para a obtenção do título de *Magister Scientiae*.

VIÇOSA
MINAS GERAIS - BRASIL
2008

DANIELLA INÁCIO DE BARROS

**REQUISITOS DE USABILIDADE E EXTENSÕES UML PARA
APLICAÇÕES EM AMBIENTES MÓVEIS.**

**Dissertação apresentada à
Universidade Federal de Viçosa, como
parte das exigências do Programa de
Pós-Graduação em Ciência da
Computação, para obtenção do título
de *Magister Scientiae*.**

APROVADO: 28 de fevereiro de 2008

Prof. Alcione de Paiva Oliveira
(Co-orientador)

Prof. José Luis Braga
(Co-orientador)

Prof. Vladimir Oliveira Di Iorio

Prof^a Íris Fabiana de Barcelos Tronto

Prof. Mauro Nacif Rocha
(Orientador)

Dedico essa vitória a todas
as pessoas que me ajudaram
durante o tempo que passei em
Viçosa.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que me concedeu essa oportunidade e me deu forças para enfrentar os obstáculos. Obrigada, meu Deus, pela sabedoria, pelo amor e pela proteção. Obrigada pelas pessoas boas que colocou em meu caminho, pois elas me ajudaram a vencer as barreiras e tornar minha caminhada mais tranquila.

Agradeço à minha família, fonte de amor inesgotável, que me incentivou e me carregou no colo nos momentos de fraqueza. À minha mãe, Maria Ângela, pelos sacrifícios feitos para que eu chegasse até aqui, sendo exemplo de mulher guerreira e sonhadora. Ao meu pai, Luiz Carlos, pela torcida, pelo amor incondicional e por todo o apoio. Aos meus irmãos, Gustavo e Leandro, por serem meus anjinhos da guarda e fonte de imensa alegria. Obrigada pelo amor de vocês! Vocês são “as mais brilhantes luzes em minhas mais escuras noites”.

Agradeço ao Prof. Mauro Nacif, por me orientar e acreditar em mim, mesmo nas horas em que nem eu acreditava. Obrigada por todos os ensinamentos e por toda a paciência. Agradeço também aos Profs. José Luís e Alcione, por me auxiliarem nesse trabalho e por todos os conhecimentos compartilhados. Zé, obrigada pelos “puxões de orelha” que me abriram os olhos e me impulsionaram para frente.

Agradeço ao Wiliam, por toda a paciência nos momentos em que tive que trocar nosso namoro pelos meus estudos, por acreditar em mim e sempre me incentivar, me mostrando que eu era capaz. Obrigada por todo o amor, por fazer a minha vida mais feliz e por ser tão especial.

Agradeço a todos os meus amigos, de perto e de longe, que me deram força, torceram por mim e estavam do meu lado sempre que precisei. Agradeço principalmente às Meninas de Ouro, Denise e Gabriela, por serem minhas irmãs em Viçosa, agüentar todo o meu stress e por tornar a nossa convivência algo tão divertido. Agradeço também às meninas da Nossa Casa, por me adotarem e me

darem um teto nessa minha última fase do mestrado, quando não mais morava em Viçosa. Sem a amizade de vocês eu não teria conseguido.

Agradeço à Nidyana, por toda a ajuda nesta fase final do meu trabalho, por toda a atenção dispensada e por estar sempre pronta a ajudar em tudo que era preciso. Agradeço também ao Ítalo, meu eterno companheiro de mestrado, pela ajuda e por toda a alegria compartilhada.

Agradeço ao Departamento de Informática da Universidade Federal de Viçosa, a todos os professores e funcionários, por toda a atenção, ajuda e conhecimento compartilhados. Agradeço em especial ao Altino, por ajudar a resolver todos os “pepinos” e por sempre me receber em sua sala com um sorriso no rosto e um abraço apertado.

BIOGRAFIA

Daniella Inácio de Barros é natural de Conselheiro Lafaiete, Minas Gerais, e filha de Luiz Carlos Salim de Barros e Maria Ângela Inácio de Barros.

Em 2004 graduou-se em Ciência da Computação pela Universidade Federal de Ouro Preto, obtendo o título de Bacharel em Ciência da Computação. No mesmo ano, se tornou professora substituta da mesma instituição, onde lecionou, até janeiro de 2006, para os cursos de Engenharia de Produção e Sistemas de Informação, no Campus Avançado de João Monlevade, Minas Gerais.

No ano de 2005, foi para Viçosa cursar o mestrado em Ciência da Computação, do Departamento de Informática – DPI da Universidade Federal de Viçosa – UFV, defendendo sua dissertação em fevereiro de 2008.

Em fevereiro de 2006, começou a trabalhar na Universidade Presidente Antônio Carlos, campus de Conselheiro Lafaiete, Minas Gerais, (UNIPAC – Lafaiete), onde leciona para o curso de Sistemas de Informação, na área de Engenharia de Software.

SUMÁRIO

LISTA DE FIGURAS	viii
LISTA DE TABELAS	ix
RESUMO	x
ABSTRACT	xi
Introdução	1
1.1 O Problema e sua Importância	2
1.2 Objetivos do Trabalho	4
1.3 Trabalhos Relacionados	5
1.4 Organização deste Documento.....	6
Revisão Bibliográfica.....	8
2.1 Introdução	8
2.2 Computação Móvel	8
2.3 Usabilidade	12
2.3.1. Técnicas de Avaliação de Usabilidade	17
2.4 Interface.....	21
2.5 UML	24
2.6 Perfis UML	26
2.6.1. Estereótipos.....	27
2.6.2. <i>Tagged Values</i>	28
2.6.3. Restrições.....	29
2.7 Extensões UML para Aplicações Web.....	30
Requisitos de Usabilidade para Aplicações em Ambientes Móveis	34
3.1. Introdução	34
3.2. Atributos de Usabilidade	35
3.3. Atributos de Usabilidade para Aplicações Móveis	38
Perfil UML para Aplicações em Ambientes Móveis.....	41
4.1 Introdução	41
4.2 Perfil UML para Aplicações Móveis	42
4.2.1. Meta-modelo	43
4.2.2. <i>Server</i>	43
4.2.3. <i>Mobile Device</i>	44
4.2.4. <i>Adaptable Interface</i>	46
4.2.5. <i>Require</i>	47
4.3 Extensões Web adaptadas para aplicações móveis.....	47
4.4 Comentário Final.....	50
m-PVANET – Um Exemplo de Aplicação.....	53
Conclusões e Perspectivas Futuras	60
UML (<i>Unified Modeling Language</i>).....	62
A.1 – Diagrama de Casos de Uso	62
A.2 – Diagrama de Classes	64
A.3 – Diagrama de Seqüência.....	66
Extensões UML para Aplicações Web	67
B.1 – Estereótipos	67

B.1.1 – <i>Server Page</i>	67
B.1.2 – <i>Client Page</i>	68
B.1.3 – <i>Form</i>	68
B.1.4 – <i>Frameset</i>	69
B.1.5 – <i>Target</i>	70
B.1.6 – <i>JavaScript Object</i>	70
B.1.7 – <i>ClientScritp Object</i>	71
B.1.8 – <i>Link</i>	71
B.1.9 – <i>Targeted Link</i>	71
B.1.10 – <i>Frame Content</i>	72
B.1.11 – <i>Submit</i>	72
B.1.12 – <i>Builds</i>	73
B.1.13 – <i>Redirect</i>	73
B.1.14 – <i>IIOP</i>	74
B.1.15 – <i>RMI</i>	74
B.1.16 – <i>Input Element</i>	75
B.1.17 – <i>Selected Element</i>	75
B.1.18 – <i>Text Area Element</i>	76
B.1.19 – <i>Web Page</i>	76
B.1.20 – <i>ASP Page</i>	77
B.1.21 – <i>JSP Page</i>	77
B.1.22 – <i>Servlet</i>	78
B.1.23 – <i>Script Library</i>	78
Referências Bibliográficas.....	80

LISTA DE FIGURAS

Figura 1 - Topologia do sistema celular (MATEUS & LOUREIRO, 2004)	9
Figura 2 - Teclado numérico (celulares)	15
Figura 3 - Teclado QWERTY (<i>smartphones</i>)	15
Figura 4 - Celular operado a caneta (<i>stylus</i>)	16
Figura 5 - iPhone com tecnologia <i>touchscreen</i>	16
Figura 6 - Visões (perspectivas) de um sistema de software (BEZERRA, 2007) ..	25
Figura 7 - Exemplo da utilização das extensões para aplicações Web (CONALLEN, 1999)	32
Figura 8 - Estrutura da Usabilidade (ISO 9241-11, 1998)	34
Figura 9 - Meta-modelo do perfil UML para aplicações em ambientes móveis	43
Figura 10 - Ícone para o estereótipo <i>Server</i>	44
Figura 11 - Ícone para o estereótipo <i>Mobile Device</i>	45
Figura 12 - Ícone para o estereótipo <i>Adaptable Interface</i>	46
Figura 13 - Associação <i>Require</i>	47
Figura 14 - Ícone para o estereótipo <i>Form</i>	48
Figura 15 - Associação <i>Submit</i>	49
Figura 16 - Associação <i>Build</i>	49
Figura 17 - Associação <i>Redirect</i>	49
Figura 18 - Tela de disciplina no PVAnet em um <i>desktop</i> (LADEIRA, 2007)	54
Figura 19 - Tela de disciplina (sem adaptação) em um dispositivo móvel (LADEIRA, 2007)	54
Figura 20 - Tela de disciplina adaptada para um dispositivo móvel (LADEIRA, 2007)	55
Figura 21 - Diagrama de Casos de Uso – m-PVANET	56
Figura 22 - Diagrama de Classes – Consultar conteúdo da disciplina	58
Figura 23 - Diagrama de Seqüência – Consultar conteúdo da disciplina	59
Figura 24 - Notação UML para ator	62
Figura 25 - Herança entre atores	63
Figura 26 - Relacionamento entre casos de uso: inclusão e extensão	64
Figura 27 - Notação UML para classe	64
Figura 28 - Relacionamento de herança entre classes	65
Figura 29 - Diagrama de Seqüência	66

LISTA DE TABELAS

Tabela 1 - Aspectos da interação móvel em relação aos da interação fixa (CYBIS et al., 2007)	17
Tabela 2 - Estereótipos gráficos WAE (NETO & GRAHL, 2007).....	32
Tabela 3 - Perfil UML para aplicações em ambientes móveis.....	52
Tabela 4 - Fluxo Lógico do Caso de Uso Consultar Conteúdo da Disciplina	57

RESUMO

BARROS, Daniella Inácio de, M.Sc., Universidade Federal de Viçosa, fevereiro de 2008. **Requisitos de usabilidade e extensões UML para aplicações em ambientes móveis.** Orientador: Mauro Nacif Rocha. Co-Orientadores: Alcione de Paiva Oliveira e José Luis Braga.

Computação móvel é uma tecnologia que permite que as pessoas obtenham serviços computacionais independente de sua localização física. A fim de facilitar a mobilidade do usuário, os dispositivos utilizados são pequenos, com o intuito de facilitar sua locomoção. Com isso, origina-se uma limitação na interação usuário-máquina, pois as interfaces de entrada e saída dos dispositivos ficam limitadas, devido à redução do tamanho dos mesmos. Assim, a interface gráfica com o usuário também fica limitada, pois as telas são pequenas, o que impede de explorar todos os recursos gráficos disponíveis. Encontram-se na literatura vários estudos referentes à usabilidade em sistemas que envolvem mobilidade. O presente trabalho apresenta o estudo da usabilidade no desenvolvimento de aplicações para dispositivos móveis e, a partir das limitações observadas, foi criado um perfil UML, a fim de facilitar a modelagem dentro da Computação Móvel. Após a criação do perfil, foi feita a aplicação das extensões através da modelagem de uma aplicação real.

ABSTRACT

BARROS, Daniella Inácio de, M.Sc., Universidade Federal de Viçosa, february 2008. Usability **Requirements and UML Extensions for Mobile Environment Applications**. Adviser: Mauro Nacif Rocha. Co-Advisers: Alcione de Paiva Oliveira and José Luis Braga.

Mobile Computing is a technology that allows people to obtain services independent of their physical location. The devices are small enough to facilitate user mobility, but may cause serious limitations on human-computer interaction. Among the most serious limitations would be input (such as limited keyboard or touch screen, lack of mouse or stylus) and output (small screen, fewer display colors) resources. Therefore, there are plenty of studies about usability in development of mobile systems in the literature. This work presents a study on usability for the development of mobile systems and, because of the limitations observed, a UML profile was created to help the mobile application modeling. After the creation of the UML profile, application of these extensions was done through modeling of a real application.

Capítulo 1

Introdução

Computação móvel é um paradigma computacional que tem como objetivo prover ao usuário acesso permanente a uma rede fixa ou móvel independente de sua posição física. É a capacidade de acessar informações em qualquer lugar e a qualquer momento (LOUREIRO et al, 2003). Através dela, o usuário, portando dispositivos como *palmtops* e *notebooks*, tem acesso a uma infra-estrutura compartilhada, através de redes sem fio, independente da sua localização física. Com isso, percebe-se que cada vez mais pessoas estão utilizando esses dispositivos. Estima-se que só em 2007 foram vendidos 118 milhões de *smartphones* (celulares com recursos computacionais avançados) em todo o mundo, representando um crescimento de 53% sobre as vendas de 2006, e que hoje existem cerca de 3,5 bilhões de celulares, cerca de um para cada dois habitantes do planeta¹.

O uso de equipamentos portáteis que utilizam tecnologia de comunicação sem fio vem alterando a maneira como as pessoas interagem com informações e serviços que antes só eram acessados por meio de computadores fixos. Novos equipamentos, aplicações e serviços estão surgindo para atender às necessidades do usuário móvel (CYBIS et al., 2007).

Com o aumento da mobilidade, o que se percebe é que existe um grande interesse em se desenvolver serviços e aplicações cada vez mais elaboradas e com alto grau de qualidade. Com isso, um grande número de pesquisadores tem trabalhado no sentido de buscar soluções que atendam e melhorem os serviços móveis, e muitas empresas têm investido no desenvolvimento de dispositivos que atendam às necessidades dos usuários.

¹ Fonte: www.teleco.com.br

Algumas áreas que ganham cada vez mais a atenção dos pesquisadores são as de desenvolvimento de interfaces gráficas para dispositivos móveis e o estudo da usabilidade em sistemas deste tipo.

Interface gráfica é um mediador entre o usuário e o sistema. Em última análise, determina também a forma como se dará a interação entre eles. Ela envolve todos os aspectos de um sistema com o qual se mantém contato (MORAN, 1981 apud SOUZA et al., 2005) “e é através da interface que os usuários têm acesso às funções da aplicação” (SOUZA et al., 2005). Essa interação deve acontecer da forma mais natural possível, por isso a interface deve ser natural, ou seja, de fácil manuseio para o usuário. Dessa forma, ele não deve se prender em como utilizá-la, mas sim em como explorar os recursos do sistema.

Partindo do princípio de que o sucesso de um software depende diretamente de sua aceitação pelos usuários, torna-se claro o porquê da preocupação com a maneira de como utilizar o software, pois caso o usuário não se sinta bem utilizando um sistema, é muito provável que ele não o utilizará (SOUZA et al., 2005).

Na ótica dos desenvolvedores, a usabilidade de software é uma medida de qualidade que pode criar o diferencial de um produto, já que um software de fácil entendimento e uso é aceito em um espaço de tempo mais curto. Além disso, os custos de treinamento de usuários durante sua implantação e/ou quando uma nova funcionalidade é adicionada se tornam mais baixos e o treinamento mais rápido.

1.1 O Problema e sua Importância

Qualidade em desenvolvimento de softwares é um dos assuntos mais discutidos no meio acadêmico, gerando várias pesquisas neste tema (KOSCIANSKI & SOARES, 2006). Isso porque é cada vez maior a demanda por softwares que fazem exatamente as tarefas que se propõem a fazer de maneira eficiente.

A fim de obter softwares de qualidade, existem os processos de desenvolvimento de software, que dividem o desenvolvimento em fases como

Engenharia de Requisitos, Análise, Codificação e Testes. No projeto do software, tem-se uma descrição de como o sistema será implementado. Nesse projeto, tem-se a modelagem do sistema, que é feita utilizando ferramentas de modelagem, como por exemplo, a UML¹ (*Unified Modeling Language*) quando se trata de sistemas orientados a objetos.

A UML possui elementos básicos de modelagem orientada a objetos, como casos de uso, classes, dentre outros. Em outras palavras, a UML fornece os principais elementos para modelagem de um sistema, mas para a modelagem de um sistema que envolve restrições e limitações alguns elementos extras são necessários. Com isso, a UML oferece recursos de extensões que servem para modelar as restrições impostas por um determinado tipo de sistema.

No desenvolvimento de softwares que envolvem mobilidade, percebe-se que a UML não oferece todos os recursos necessários para a sua modelagem, o que acaba resultando em projetos que ignoram as limitações impostas pelo ambiente móvel, como, por exemplo, a transferência de dados através da rede sem fio, entre outras. Dessa forma, o que se tem hoje é apenas a migração dos sistemas tradicionais, ou seja, sistemas desenvolvidos para computadores pessoais para sistemas em dispositivos portáteis de mão, isto é, apenas uma adaptação desses sistemas para os novos dispositivos.

Existem diferenças significativas no desenvolvimento de *softwares* para dispositivos móveis e para *desktops* que não são bem claras em um primeiro instante, ficando mais visíveis à medida que se refina o projeto (CYBIS et al., 2007).

Com base nos conceitos estudados em Interação Homem-Computador, pode-se inferir que a usabilidade desses sistemas fica prejudicada com essa migração e faz com que o usuário não interaja com este software de maneira adequada, gastando muito tempo para aprender a se comunicar com a interface e não obtendo, assim, os resultados desejados.

¹ www.uml.org

Outro fator que deve ser levado em consideração é o tamanho dos *displays* (telas) e dos teclados dos dispositivos móveis, que são bem reduzidos a fim de facilitar o transporte do mesmo. Esse fato dificulta o desenvolvimento de uma interface bem elaborada e de fácil navegação.

Desenvolver softwares com grande aceitação pelos usuários é o principal motivo dos estudos em usabilidade de software. Uma possível maneira de melhorar a usabilidade em sistemas móveis é a criação de extensões UML com o propósito de antecipar e evitar problemas da interação com o usuário já na fase de modelagem do *software*, permitindo, assim, um projeto mais cuidadoso e o conseqüente aumento de sua qualidade.

1.2 Objetivos do Trabalho

Esta pesquisa tem como objetivo geral propor um conjunto de requisitos de usabilidade para sistemas móveis, bem como extensões de UML próprias para a descrição de problemas de usabilidade em softwares desenvolvidos para esses ambientes, permitindo um projeto mais cuidadoso e o conseqüente aumento de qualidade desses sistemas.

Detalhadamente, os objetivos são:

- Fazer o levantamento bibliográfico sobre as tecnologias usadas em Computação Móvel.
- Fazer o levantamento bibliográfico sobre usabilidade e aplicações das técnicas de usabilidade em sistemas móveis.
- Fazer o levantamento bibliográfico sobre construção de interfaces para dispositivos móveis.
- Propor requisitos de usabilidade e extensões de UML para Sistemas Móveis.
- Fazer a aplicação e teste do perfil usando uma aplicação real.

1.3 Trabalhos Relacionados

É cada vez maior o interesse dos pesquisadores pela Engenharia de Software aplicada à Computação Móvel. Vários estudos sobre perfis UML para aplicações que envolvem mobilidade são encontrados na literatura.

Em (BAUMEISTER et al., 2003), (KOSIUCZENKO, 2003) e (WIRSING et al., 2003) são apresentados estudos sobre perfis UML e a adaptação de Diagramas de Seqüência e de Atividades para aplicações que envolvem mobilidade, a fim de atender às restrições impostas pela Computação Móvel.

(GRASSI et al., 2004) apresentam um *framework* para a modelagem de aplicações em computação móvel, onde é modelada tanto a localização física quanto a localização lógica do usuário. Nesse estudo, é criado um perfil UML que modela a mobilidade, utilizando, principalmente, o Diagrama de Atividades.

Em seu trabalho, (ITO et al., 2005) afirmam que os sistemas desenvolvidos para Web devem ser capazes de “moldar suas interfaces para os diversos dispositivos móveis existentes”. Nesse estudo, analisam o *framework* de reconhecimento de dispositivos chamada *Composite Capability Preference Profile* (CC/PP), reconhecido pelo W3C¹ (*World Wide Web Consortium*), capaz de enviar para um servidor as informações relativas às capacidades de um dispositivo cliente. Com base nessa técnica, desenvolveram um *servlet* onde, quando um cliente faz uma requisição a um servidor WEB, recupera o perfil CC/PP e gera um arquivo contendo as capacidades do dispositivo, que “servirá como base para a adaptação de conteúdos”.

O desenvolvimento de aplicações WEB e aplicações em ambientes móveis são semelhantes, uma vez que ambos em geral se utilizam da arquitetura Cliente-Servidor. Estudos relacionados a extensões UML para aplicações WEB também são constantes na literatura.

(CONALLEN, 1999) e (CONALLEN, 2000) apresentam um perfil para modelagem de aplicações WEB, onde são definidos vários estereótipos para

¹ www.w3.org

modelar os diversos elementos presentes nas aplicações WEB. (KOCH et al., 2007) complementa as extensões propostas por Conallen, onde são criados estereótipos que representam elementos de interface gráfica.

Em seu trabalho, (BERGH & CONINX, 2005) afirmam que o número de dispositivos e suas capacidades são muito diversos e, por isso, criam um perfil UML para modelar interface gráfica sensível ao contexto. Para a criação do perfil, (BERGH & CONINX, 2005) se inspiraram no Modelo Baseado em *Design* de Interfaces, que é uma metodologia proposta para apoiar o *design* e o desenvolvimento multiplataforma e, também, interfaces gráficas sensíveis ao contexto.

Uma intensa busca foi realizada nos principais sites de busca disponíveis, tais como scholar.google.com, google.com, periódicos.capes.gov.br etc., e mesmo com os constantes avanços nessa área, não foi encontrado um trabalho que trate especificamente de extensões UML que visem a usabilidade em ambientes móveis.

1.4 Organização deste Documento

- No Capítulo 2, é apresentada a revisão bibliográfica, que norteou o desenvolvimento deste trabalho.
- No Capítulo 3, são apresentadas as extensões UML para aplicações em ambientes móveis.
- No Capítulo 4, é apresentada a exemplificação do perfil UML aqui desenvolvido através da modelagem de uma aplicação real.
- No Capítulo 5, são apresentadas as conclusões e as perspectivas futuras para aprimoramento deste trabalho.

Ao final do documento, ainda são apresentados os Apêndices e as Referências Bibliográficas, onde:

- No Apêndice A, é apresentada uma descrição resumida da UML.

- No Apêndice B, são apresentadas as Extensões UML para Aplicações Web, descritas em (CONALLEN, 1999) e (CONALLEN, 2000).

Capítulo 2

Revisão Bibliográfica

2.1 Introdução

Durante a fase de revisão bibliográfica, foram pesquisados métodos e modelos de desenvolvimento de interface gráfica para aplicações desenvolvidas para dispositivos móveis e algumas limitações impostas pela mobilidade. Além disso, fez-se um levantamento das pesquisas disponíveis acerca da usabilidade nesse tipo de sistema.

Constatou-se que esse é um assunto que vem ganhando cada vez mais o interesse de pesquisadores e muitos trabalhos relacionados à usabilidade e desenvolvimento de interface gráfica em sistemas móveis têm sido desenvolvidos com o intuito de facilitar a navegação dos usuários através dos diversos dispositivos.

2.2 Computação Móvel

Computação móvel é uma tecnologia que permite que os usuários obtenham serviços computacionais independente de sua localização física. “Existem três elementos que caracterizam a computação móvel: o tipo e capacidade de processamento do dispositivo portátil, a mobilidade do usuário e da unidade móvel e a comunicação com outro elemento computacional através de um canal de comunicação sem fio” (MATEUS & LOUREIRO, 2004).

Denomina-se **célula** a área de alcance de uma antena de rádio que serve como ponto de acesso da rede sem fio com a rede cabeada. A Figura 1 ilustra a topologia de um Sistema Celular, onde:

ERB = Estação Rádio Base

CCC = Central de Comutação e Controle

RPT = Rede Pública de Telefonia

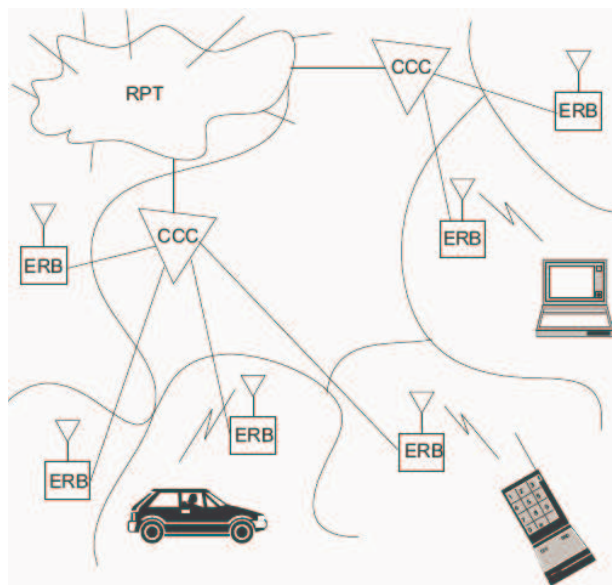


Figura 1 - Topologia do sistema celular (MATEUS & LOUREIRO, 2004)

Com a introdução do conceito de mobilidade à computação, certas limitações acabaram surgindo, como limitações de energia, tamanho de memória e capacidade de processamento do dispositivo (SAHA, 2003). O fato de os usuários estarem em constante movimento e migrando entre várias redes faz com que os serviços fornecidos tenham que ter a capacidade de adaptação. “Adaptação significa dizer que uma aplicação ou algoritmo não tem agora uma única especificação de saída, mas possivelmente um conjunto válido de saídas ou resultados que são aceitáveis em função das condições existentes em um determinado momento do tempo” (MATEUS & LOUREIRO, 2004).

Algumas dessas restrições impostas pela mobilidade são citadas em (FORMAN & ZAHORJAN, 1994) e são listadas a seguir:

- Energia – A fim de facilitar o transporte, os dispositivos móveis possuem baterias com tamanho reduzido, o que diminui seu poder de carga, aumentando a quantidade de recargas.
- Riscos de perda e extravio de dados – “A portabilidade aumenta o risco de danos físicos, acesso não autorizado, perda e roubo do dispositivo móvel, tornando tais computadores menos seguros e confiáveis”.

- Interface com o usuário limitada - O uso de mouses é inviável nos dispositivos móveis, que realizam interface com o usuário através de botões ou canetas *stylus*. Além disso, os dispositivos móveis apresentam telas com tamanhos reduzidos, o que dificulta a visualização de interfaces gráficas mais complexas.
- Capacidade de armazenamento em disco – “O espaço de armazenamento em um computador móvel é comprometido pelo tamanho físico e gasto de energia”. Uma solução para o problema é o uso de *chips* de memória que são acoplados aos dispositivos, aumentando, assim, sua capacidade de armazenamento.
- Rede heterogênea – Devido à capacidade de mobilidade, o usuário móvel se conecta a diferentes tipos de rede. Com a mudança de célula, o usuário pode encontrar diferentes tipos de protocolos e velocidades de transmissão.
- Informação dependente da localidade – Alguns serviços dependem da localização do usuário e “prover mecanismos que obtenham a configuração de dados apropriada a cada localidade é um desafio importante para a computação móvel”.

O paradigma da computação móvel introduz conceitos interessantes, como adaptação, operação desconectada e *handoffs*, explicados a seguir.

“À capacidade de ajuste e à possibilidade de produzir resultados apesar de condições adversas dá-se o nome de **adaptação**” (AMADO, 2002). A adaptação em computação móvel pode se dar em dois extremos: ser de total responsabilidade das unidades móveis ou ser responsabilidade das estações base, sendo que o ideal é que a adaptação esteja em um nível intermediário entre estes. Em outras palavras, “a adaptação pode ser feita tanto no dado que está sendo transmitido para a unidade móvel quanto no próprio processamento solicitado pelo usuário” (MATEUS & LOUREIRO, 2004).

Devido às limitações de energia e de custos, as atividades nas unidades móveis podem ser realizadas desconectadas da rede, o que introduz o conceito de **operação desconectada**.

Handoff é a capacidade de permitir a migração do usuário entre diferentes pontos de acesso à rede sem que haja uma desconexão do mesmo com a rede, ou seja, perdendo o contato com uma estação base e estabelecendo com outra sem se desconectar da rede como um todo.

A arquitetura de um sistema de computação móvel é dividida em uma rede com fio, chamada rede fixa, onde se encontram as estações base, e uma rede sem fio, chamada rede móvel, onde se encontram as unidades móveis. Usando os conceitos definidos em (AMADO, 2002), “a unidade móvel (ou *mobile host*) é um dispositivo capaz de se conectar com a rede fixa através de uma conexão sem fio (*wireless link*). Hosts estacionários são conectados através de uma rede de alta velocidade interligada por fios (por exemplo, *Fast Ethernet* com taxa de 100 Mbps ou *Gigabit Ethernet* com 1 Gbps) podendo ser hosts fixos ou estações base (*base stations*)”.

Com o avanço da Computação Móvel, várias áreas estão adotando essa tecnologia em suas atividades, como, por exemplo, a medicina, através de aplicações onde médicos e enfermeiros têm acesso ao prontuário de um paciente através de *palmtops* e outros dispositivos, e as companhias aéreas, através de aplicações que enviam informações sobre o voo de um determinado passageiro para o seu celular (UMAR, 2004). Outra área beneficiada pelos avanços da tecnologia móvel é a automotiva. É comum que hoje em dia as empresas projetem seus carros com um pequeno computador de bordo que utiliza a tecnologia GPS (*Global Positioning System*) para informar ao motorista rotas e localizações de estabelecimentos.

“Normalmente quando uma empresa decide disponibilizar uma aplicação para um ambiente de Computação Móvel, é feita uma análise da versão/site disponível na rede fixa, que envolve, dentre outros aspectos, aquelas características adequadas para o novo ambiente” (LOUREIRO et al., 2003).

Muitas vezes, essa adaptação não é feita da melhor maneira, o que gera alguns problemas de usabilidade e interação com o usuário. A próxima seção introduz conceitos de usabilidade e discute alguns problemas de usabilidade e suas soluções em ambientes que envolvem mobilidade.

2.3 Usabilidade

Usabilidade é um conceito conhecido desde a década de 80. Porém, com o surgimento da Internet e popularização dos computadores, ela passou a ser estudada com mais atenção, se tornando um fator crítico de uma aplicação (MARTINEZ, 2003).

Segundo (McCALL et al., 1977 apud PRESSMAN, 2006), usabilidade pode ser definida como “o esforço para aprender, operar, preparar entradas e interpretar saídas de um programa”. Vários autores seguem esta mesma linha de raciocínio. Assim, pode-se dizer que usabilidade é a característica do software que o torna agradável e natural ao usuário final.

(NIELSEN, 1993) possui uma definição mais voltada para métricas de avaliação, onde ele considera usabilidade como sendo um conceito de qualidade de uma aplicação sob uma perspectiva de uso, tradicionalmente relacionado a cinco atributos: facilidade de aprendizado, eficiência, facilidade de reter o conhecimento sobre a aplicação obtida em usos anteriores (memorização), baixo índices de erros e satisfação dos usuários. Um resumo desses atributos é dado abaixo:

- **Facilidade de Aprendizado:** grau de facilidade de realizar determinada tarefa na primeira vez que o usuário utiliza o sistema.
- **Eficiência:** rapidez com que podem efetuar determinada tarefa, uma vez que já tenham aprendido a usar o sistema.
- **Memorização:** grau de facilidade do usuário em utilizar o sistema, tendo ficado algum tempo sem operá-lo.
- **Índice de Erros:** quantidade de erros cometida pelos usuários e grau de facilidade de o usuário sair de uma situação de erro.

- Satisfação: o quão agradável é para o usuário utilizar o sistema.

Existem também dois padrões internacionais da ISO que definem a usabilidade. O primeiro deles é o padrão ISO / IEC 9126 que dá ênfase aos atributos internos e externos, que contribuem com a usabilidade, funcionalidade e eficiência do produto. Já o segundo padrão, o ISO / IEC 9241, é centrado no conceito de qualidade de utilização, ou seja, refere-se a como o usuário realiza tarefas específicas em cenários específicos com efetividade (PINHEIRO et al., 2003).

É importante destacar que a usabilidade da interface do software é somente parte da experiência do usuário móvel, uma vez que “os componentes emocionais da experiência do usuário também desempenham papel relevante na interação com os computadores de mão” (CYBIS et al., 2007).

Um dos grandes desafios dos projetistas de interfaces, principalmente em aplicações Web, é desenvolver uma interface gráfica usável pelos diversos tipos de usuários existentes, respeitando diferenças de nível de conhecimento, idade, sexo, cultura, deficiências (motora, visual, auditiva etc.) entre outras (SHNEIDERMAN, 2002). De acordo com (FSP, 2007) a popularidade de estudos em usabilidade “vem do fato de que um produto fácil de usar dá retorno financeiro”.

Fazer uma análise quantitativa da usabilidade não é tarefa fácil, pois a qualidade de uma interface é “uma quantidade psicológica” difícil de ser aferida (BORGES et al., 2002). Essa tarefa envolve vários domínios do comportamento humano, correspondentes aos diversos processos mentais de tratamento da informação (percepção, raciocínio, representação mental) que são do campo de estudo da ergonomia cognitiva.

Alguns atributos devem ser considerados para garantir a usabilidade de uma interface. Dentre eles estão (SOUZA et al., 2005):

- Facilidade de aprendizado do sistema: “tempo e esforço necessários para que os usuários atinjam um determinado nível de desempenho”.

- Facilidade de uso: “avalia o esforço físico e cognitivo do usuário durante o processo de interação, medindo o número de erros cometidos durante a execução de uma determinada tarefa”.
- Satisfação do usuário: avalia o grau de satisfação do usuário em trabalhar com determinado sistema.
- Flexibilidade: “avalia a possibilidade de o usuário acrescentar e modificar as funções e o ambiente iniciais do sistema”.
- Produtividade: “se o uso do sistema permite ao usuário ser mais produtivo do que seria se não o utilizasse”.

Muitos trabalhos relacionados à usabilidade em sistemas móveis têm sido desenvolvidos com o intuito de facilitar a navegação dos usuários através dos diversos dispositivos, como (JONES et al., 1999), (KIM et al., 2002), (CARVALHO & PELISSONI, 2004), entre outros. Estudos de usabilidade em aplicações em dispositivos móveis devem ser centrados nas restrições de design impostas por uma limitação de banda passante e pelos pequenos *displays* dos dispositivos (CHAN et al., 2002).

É muito importante entender o usuário móvel e a dinâmica do contexto em que ele está inserido. O tempo é um fator muito importante para esse tipo de usuário, que “tende a usar serviços que permitam a manipulação rápida da interface e o acesso à informação por meio de um número reduzido de passos” (ERICSSON et al., 2001 apud CYBIS et al., 2007). Além disso, o usuário móvel está sujeito a interrupções durante a sua interação, como o recebimento de uma chamada ou queda da bateria.

Outro fator que deve ser considerado quando se estuda a interação do usuário com dispositivos que utilizam de tecnologia móvel são as interfaces de entrada e saída de dados, pois essas interfaces em dispositivos móveis são muito diversas e, com isso, a interação do usuário com esses aparelhos pode ocorrer de diversas maneiras, muito diferente da interação com um computador pessoal, que consiste, basicamente, na entrada de dados através do mouse e do teclado.

Atualmente, a maioria dos aparelhos celulares utiliza teclados numéricos, conforme ilustra a Figura 2, onde o usuário pressiona as teclas numéricas um determinado número de vezes para a inserção das letras, por exemplo, se o usuário deseja digitar a letra c ele pressiona três vezes a tecla 2 do aparelho.



Figura 2 - Teclado numérico (celulares)

Outro tipo de teclado utilizado em dispositivos móveis, principalmente em *smartphones*, é o teclado QWERTY, que se assemelha aos teclados utilizados em *desktops*. Esse tipo de teclado é apresentado na Figura 3. Outros aparelhos, com tecnologia mais moderna, suprimem o teclado QWERTY e utilizam outras formas de interação com o usuário, como *touchscreen*. As Figuras 4 e 5 apresentam um celular com teclado virtual operado por meio de uma caneta especial (*stylus*) e um iPhone com tecnologia *touchscreen*, respectivamente.



Figura 3 - Teclado QWERTY (*smartphones*)



Figura 4 - Celular operado a caneta (*stylus*)



Figura 5 - iPhone com tecnologia *touchscreen*

(CYBIS et al., 2007) comparam as principais características e diferenças entre a interação fixa e a interação móvel. O resultado dessa comparação é apresentado na Tabela 1.

Aspectos da Interação	Interação Fixa	Interação Móvel
Ambiente	Normalmente interno, pouca variação	Interno e externo, variação freqüente
Tempo de duração da interação	Médio a longo	Médio a curto

Mobilidade do usuário	Baixa, normalmente sentado	Alta, qualquer posição e movimentos do corpo
Hierarquia das tarefas	A interação é tarefa primária	A interação pode ser tarefa secundária
Manipulação de outros objetos que não estão relacionados à interação	Rara	Frequente
Estilos de interação	Alta dependência de manipulação direta; outros estilos são complementares	Seleção de menus, formulários, apoiados por manipulação direta e linguagem natural

Tabela 1 - Aspectos da interação móvel em relação aos da interação fixa (CYBIS et al., 2007)

2.3.1. Técnicas de Avaliação de Usabilidade

Quando se navega entre páginas na Internet, vários são os motivos para um usuário deixar de acessar um site: dificuldade no uso, dúvidas em relação às funcionalidades oferecidas pelo site e em relação ao que o usuário pode fazer através dele, dificuldades na compreensão do texto e falta de respostas às questões dos usuários. Já na Intranet, usabilidade está relacionada à produtividade do funcionário, pois se a interface for difícil de usar, o funcionário gastará um tempo considerável tentando entendê-la e utilizá-la (NIELSEN, 2003).

A fim de solucionar tais problemas, foram desenvolvidas técnicas de avaliações de usabilidade e interfaces com o usuário. Para avaliar se uma interface possui uma boa usabilidade, existem diferentes métricas, tanto objetivas quanto subjetivas, de avaliação de usabilidade, uma vez que “o comportamento humano é observável e o desempenho humano é mensurável” (PÁDUA, 2008).

Algumas dessas técnicas são apresentadas pelo site *UsabilityNet*¹ e são descritas a seguir.

2.3.1.1. Entrevistas

As entrevistas para avaliação da usabilidade podem ser de duas naturezas: Entrevistas Tradicionais e Entrevistas Contextuais.

Através das Entrevistas Tradicionais, os projetistas obtêm informações sobre as necessidades das pessoas em relação a um projeto, a partir de um roteiro com tópicos a serem respondidos pelo entrevistado. “É importante que o entrevistador assuma uma postura neutra e analítica”. A entrevista começa com um “aquecimento”, onde o entrevistador se apresenta para o entrevistado. Logo após, ele explica o contexto da entrevista e, então, inicia a entrevista. A partir dos dados coletados, o entrevistador elabora um relatório que servirá como base para a próxima fase do desenvolvimento (CYBIS, 2007).

As Entrevistas Contextuais são realizadas durante a utilização do sistema pelo entrevistado. “Trata-se de uma combinação produtiva entre a técnica de entrevista e de observação do usuário” (CYBIS, 2007). Sua etapa de execução pode ser dividida nas seguintes fases: **entrevista tradicional**, quando o entrevistador solicita ao entrevistado uma visão geral do trabalho, **relação de mestre-aprendiz**, quando o entrevistado deve descrever seu trabalho ao entrevistador, **observação**, quando o entrevistado faz quaisquer perguntas que desejar, e **resumo**, quando o entrevistador apresenta ao entrevistado um resumo do que extraiu durante a entrevista, devendo, esse, ser aprovado pelo mesmo.

2.3.1.2. Questionários

Os questionários podem ser de dois tipos: de perfis e uso e de satisfação.

Os questionários de perfil e de uso servem para “obter informações sobre as características reais dos usuários de um software, ou aplicação Web, e saber como eles efetivamente usam tais ferramentas” (CYBIS, 2007). Em sua

¹ www.usabilitynet.org

elaboração deve-se tomar cuidado em relação à apresentação das questões, que devem ser objetivas e não trazer dúvidas na hora das respostas.

Os questionários de satisfação servem para medir o grau de satisfação de um usuário ao utilizar um sistema. Os dados obtidos a partir desse tipo de questionário podem ser usados para análise estatística, uma vez que produzem “dados quantitativos e objetivos” (CYBIS, 2007).

Segundo (PÁDUA, 2008a), os questionários também podem ser usados para “guiar o desenho ou melhorias de desenho da interface, identificar áreas potenciais para introdução de melhorias e validar avaliações corporativas”.

A fim de facilitar a elaboração de questionários, existem modelos pré-definidos de questionários, como QUIS¹ (*Questionary for User Interface Satisfaction*), SUMI² (*Software Usability Measurement Inventory*), entre outros. Em seu trabalho, (MEDEIROS et al., 1999) propõem uma versão em português do questionário ISONORM, que é um questionário baseado na parte 10 da norma ISO 9241.

2.3.1.3. Observação do Usuário

Essa técnica consiste na observação da interação de um usuário por um avaliador, onde o avaliador toma nota de toda a interação e, a partir dos dados coletados, podem-se obter dados quantitativos (tempo gasto para concluir determinada tarefa) e qualitativos (“práticas e estratégias do usuário”). A observação pode ser direta, quando o observador está presente e assiste de perto à interação, ou indireta, quando o observador assiste a interação através de um vídeo, por exemplo (CYBIS, 2007).

As “regras do jogo” devem ficar claras desde o início para que o usuário tenha consciência de que não é ele quem está sendo avaliado, mas sim o software utilizado por ele. O usuário também pode ficar constrangido de ter uma pessoa observando tudo que ele faz e isso pode atrapalhar na interação.

¹ <http://lap.umd.edu/quis/>

² <http://www.ucc.ie/hfrg/questionnaires/sumi/index.html>

Nesse tipo de técnica é interessante mesclar vários tipos de observação e anotações, como gravação em vídeo, gravação da interação por um software que grave os movimentos do mouse, anotações em papel, fotografias, dentre outros. Ao final da observação, é interessante mesclar outra técnica de avaliação, como entrevista ou questionário.

2.3.1.4. Análise de Tarefa ou Tarefa de *Benchmark*

A análise de tarefa consiste na elaboração de uma tarefa para que o usuário realize usando o software analisado. O observador deve elaborar a tarefa de forma que não gere dúvidas ao usuário na hora de seu cumprimento e deve definir o que o usuário irá fazer e não como ele irá fazer. “Tarefas de *benchmark* devem ser específicas para que o participante não desvie sua atenção para detalhes irrelevantes durante o teste” (PÁDUA, 2008a).

O objetivo da análise de tarefas é “avaliar como o usuário realiza a tarefa utilizando a interface que, no caso, é o instrumento para a sua realização” (PÁDUA, 2008a).

2.3.1.5. Avaliação Heurística

Segundo (NIELSEN & MOLICH, 1990) avaliação heurística é um método informal de avaliação, onde especialistas, a partir de uma lista de heurísticas, avaliam uma interface em relação à sua usabilidade. De acordo com (MEDEIROS et al., 1999), a “Avaliação Heurística analisa a usabilidade da interface a partir de uma lista de recomendações heurísticas”.

(NIELSEN, 2008) propõe um conjunto de heurísticas de usabilidade. São elas:

- Visibilidade do *status* do sistema – o sistema deve sempre informar para o usuário seu *status* atual.
- Faça a correspondência do sistema com o mundo real – o projetista deve sempre usar palavras e termos familiares para o usuário, evitando termos técnicos e de difícil entendimento.

- Controle e liberdade do usuário – o sistema deve oferecer mecanismo de refazer e desfazer.
- Consistência e padrões – Não use termos diferentes significando a mesma coisa, pois os usuários não devem se prender a “descobrir” se determinado termo significa o mesmo que um outro.
- Prevenção de erros – O ideal é que erros não ocorram. A fim de evitar situações de erro, o sistema deve sempre solicitar ao usuário que ele confirme a ação antes de executá-la.
- Reconhecer ao invés de recordar – Deixar objetos, ações e opções visíveis para o usuário, a fim de minimizar a carga de memória do usuário. “Instruções para uso do sistema devem ser visíveis ou facilmente recuperáveis sempre que for necessário”
- Flexibilidade e eficiência de uso – Fornecer teclas de atalho a fim de agilizar o acesso de um usuário “experiente”.
- Estética e design minimalista – Evitar informações desnecessárias no sistema. “Cada unidade extra de informação em um diálogo compete com as unidades de informação e diminui sua relativa visibilidade.”
- Ajudar usuários a reconhecer, diagnosticar e se recuperar de erros – As mensagens de erro devem conter uma linguagem simples, entendível pelo usuário, e explicar precisamente o que é e o que ocasionou o erro, além de sugerir uma solução precisa.
- Ajuda e documentação – O sistema deverá oferecer ao usuário ajuda e documentação. Essa ajuda deve ser centrada no usuário e de fácil entendimento pelo mesmo, além de oferecer soluções de fácil acesso.

2.4 Interface

A importância das interfaces de aplicações computacionais nas últimas décadas tem sido cada vez maior. Ela envolve todos os aspectos de um sistema com o qual se mantém contato (MORAN, 1981 apud SOUZA et al., 2005). É através da

interface que os usuários têm acesso às funções da aplicação. Fatores de satisfação subjetiva, de eficiência, de segurança, de custo de treinamento, de retorno de investimento, todos dependem de um bom projeto de interface (SOUZA et al., 2005).

Criar interfaces para usuários experientes não é tarefa fácil e essa dificuldade aumenta quando uma aplicação é destinada a diversos tipos de usuários, inclusive iniciantes ou usuários sem muitos conhecimentos computacionais (SHNEIDERMAN, 2000).

Não existe um modelo único de interface gráfica que satisfaça a todos os tipos de usuários. Por isso, deve-se estudar o perfil do usuário final de seu produto para que a interface do sistema seja “otimizada” e promova uma maior produtividade do usuário final. A partir do momento que o projetista conhece o perfil do usuário, como grau de escolaridade, aptidão computacional, atividades profissionais, perfil social etc., ele obtém características relevantes para o design de sua interface (MAYHEW, 1999).

“Uma visão geral de como proceder na construção de uma interface adequada ao usuário é abordada através de métodos para desenvolvimento de projetos de interface” e o ideal é que esses métodos sejam centrados no usuário (ZAINA et al., 2002).

Segundo (ZAINA et al., 2002) é possível dividir o processo de desenvolvimento de interfaces em três etapas:

1. Levantamento das informações sobre o usuário final do sistema: onde é feito o levantamento das informações sobre os usuários finais do sistema, em que informações como perfis de usuários, análise de tarefas, princípios gerais de projeto e desenho do sistema são definidos.
2. Projeto da interface: onde são realizados “testes visando checar itens relacionados à usabilidade e correções dos possíveis problemas encontrados nos testes”.

3. Implantação supervisionada da interface: etapa em que é feita a análise de quais fatores devem ser considerados para fazer com que o desenho da interface impacte o menos possível no ambiente de trabalho do usuário.

Um dos conceitos relacionados ao desenvolvimento de interfaces gráficas é a estética, que “se refere ao apelo visual da aplicação, à sua atratividade para o usuário” (CYBIS et al., 2007). No desenvolvimento de aplicações para dispositivos móveis, “as limitações físicas das telas podem restringir as opções do desenvolvedor em relação à quantidade e à qualidade gráfica da interface”.

Quando se tem uma interface desenvolvida para sistemas que irão rodar em dispositivos com tamanho reduzido de tela, onde se tem falta de espaço para se explorar todos os recursos de interface gráfica disponíveis, uma boa opção é o uso de ícones para que o usuário possa interagir com o sistema sem comprometer a usabilidade.

Segundo a Semiótica, que é o estudo dos signos, ou seja, as representações das coisas do mundo que estão em nossa mente (AMSTEL, 2008), um ícone é “uma imagem que tem alguma relação de semelhança entre a representação e o objeto que, se for convincente, permite que se use a representação sem tomar conhecimento do objeto” (AMSTEL, 2006).

Infelizmente, o conteúdo de exibição em dispositivos móveis apresenta várias dificuldades de adaptação, como a grande variação nos tamanhos das telas, o que faz com que os sistemas desenvolvidos para *desktop* se tornem impróprios para esse tipo de dispositivos. Assim, esforços consideráveis são necessários para entender como obter uma visualização efetiva neste tipo de contexto (NOGUEIRA et al., 2007).

A fim de minimizar tais esforços, o W3C estabeleceu algumas diretrizes, ou boas práticas, para o desenvolvimento de *websites* a fim de torná-los facilmente visualizados em mais de um tipo de dispositivo. Dessa forma, ao invés de os desenvolvedores criarem mais de uma versão do site, uma para *desktop* e outra para dispositivos móveis, criaria apenas uma versão, que seria adaptada às

características do dispositivo utilizado para acessá-lo. Algumas dessas diretrizes são listadas a seguir:

- Explorar as capacidades do dispositivo para proporcionar uma melhor experiência do usuário.
- Realizar testes em dispositivos reais, além de usar emuladores.
- Assegurar-se de que o conteúdo é adequado para uso em um contexto móvel.
- Usar linguagem clara e simples.
- Dividir as páginas em porções usáveis mas limitadas.
- Assegurar-se de que o tamanho da página é adequado às limitações físicas e de memória do dispositivo.

2.5 UML

Unified Modeling Language (UML) é uma linguagem utilizada para modelagem de sistemas orientados a objetos, desenvolvida na década de 90, por James Rumbaugh, Grady Booch e Ivar Jacobson, com o intuito de se criar uma linguagem unificada de especificação de software. Atualmente a UML está em sua segunda versão (UML 2.0) e é “amplamente aceita pelas indústrias e instituições de pesquisa” (FILHO & BRAGA, 2007).

Segundo (PÁDUA, 2007), a UML é uma linguagem para **visualizar**, uma vez que “provê formas gráficas para representação de modelos”, **especificar**, pois “permite uma especificação precisa, não ambígua e completa”, **construir**, uma vez que “permite geração de código em uma linguagem de programação a partir de modelos”, e **documentar**, pois “modelos UML podem ser usados como documentação do sistema”.

A UML modela aspectos dinâmicos e estáticos de um sistema. Na UML 2.0, têm-se treze tipos de diagramas divididos em três categorias: Estrutural (Diagrama de Classes, Diagrama de Objetos, Diagrama de Casos de Uso, Diagrama de Estrutura Interna, Diagrama de Colaboração e Diagrama de Componentes),

Dinâmica (Máquina de Estados, Diagrama de Atividades, Diagrama de Seqüência, Diagrama de Tempo e Diagrama de Interação) e Física (Diagrama de Implantação e Diagrama de Pacote).

A Figura 6 ilustra as visões da UML. Segundo (BEZERRA, 2007), a **visão de casos de uso** “descreve o sistema de um ponto de vista externo, como um conjunto de interações entre o sistema e os agentes externos a ele”. A **visão de projeto** “ênfatisa as características do sistema que dão suporte às funcionalidades externamente visíveis do sistema”. A **visão de implementação** “abrange o gerenciamento de versões do sistema construídas pelo agrupamento de componentes e subsistemas”. A **visão de implantação** “corresponde à distribuição física do sistema em seus subsistemas e à conexão entre essas partes”. A **visão de processo** “ênfatisa as características de concorrência, sincronização e desempenho do sistema”.

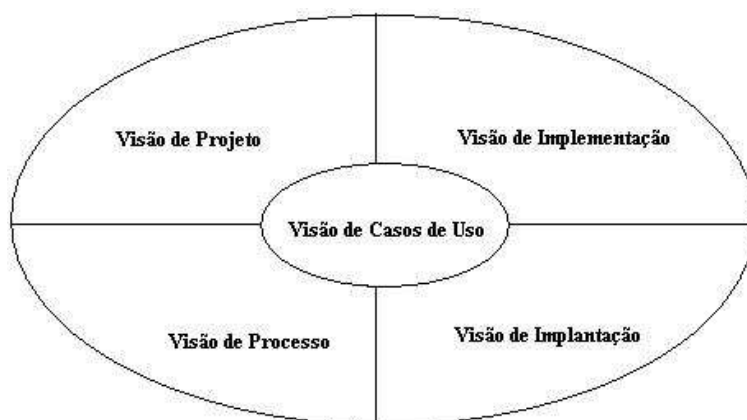


Figura 6 - Visões (perspectivas) de um sistema de software (BEZERRA, 2007)

“A UML provê um número elevado de conceitos e notações particularmente concebidos de forma a satisfazer os requisitos típicos de modelação de sistemas de informação”. Entretanto, existem casos em que a “introdução de conceitos e/ou de notações adicionais” se torna necessária (SILVA, 2003).

Em outras palavras, nem sempre a UML oferece todos os recursos necessários para modelar determinada aplicação. Por esse motivo, a UML é uma linguagem extensível, ou seja, oferece mecanismos de extensão que modelam

uma aplicação especificamente. Os mecanismos de extensão presentes na UML são os estereótipos, os valores rotulados (*tagged values*) e as restrições. A partir desses mecanismos são criados os Perfis UML. Esses mecanismos de extensão serão descritos na próxima seção.

O Apêndice A apresenta mais informações sobre a UML.

2.6 Perfis UML

Para que a UML não se tornasse muito complexa, seus criadores deixaram de lado alguns detalhes, mas criaram mecanismos de extensão que a torna adaptável para sistemas com características específicas (ERIKSSON et al., 2004).

A UML provê mecanismos que permitam que novos elementos possam ser definidos e integrados à linguagem sem que a mesma seja afetada (FILHO & BRAGA, 2007). Com isso, a UML oferece mecanismos que permitem estender a linguagem de uma maneira controlada. Dessa forma, as extensões UML permitem que usuários refinem a sintaxe e semântica da notação para adequação a projetos de sistemas específicos.

A UML provê duas maneiras de estender a linguagem (ERIKSSON, 2004):

1. Adicionando estereótipos, *tagged values* e restrições. Em outras palavras, utilizando os mecanismos nativos de extensão.
2. Utilizando uma variante da UML como uma nova instância de MOF (*Meta-Object Facility*).

Existem três mecanismos nativos de extensão na UML (ERIKSSON et al., 2004):

- Estereótipo: cria um novo elemento do meta-modelo e pode adicionar uma nova semântica (ou semântica extra) a qualquer elemento da UML.
- Valores rotulados (*tagged values*): Propriedades adicionadas aos elementos da UML.

- Restrições: Regras que definem a “semântica de um ou mais elementos da UML” e podem estar associadas a classes e relacionamentos (associações).

A partir desses mecanismos, os desenvolvedores podem “empacotar” suas extensões em perfis, criando novos “dialetos” dentro da UML (ERIKSSON et al., 2004).

Alguns passos para se criar um Perfil UML são propostos em (FERNANDEZ & MORENO, 2004):

1. Em um primeiro momento, deve-se definir o conjunto de elementos que irão compor o sistema e os relacionamentos entre eles, sendo expressos através do meta-modelo. Caso esses elementos não possam ser modelados através de elementos nativos da UML, novos elementos são criados através dos mecanismos de extensão e são associados ao meta-modelo.
2. Uma vez definido o meta-modelo, o Perfil UML poderá ser criado. Nesta fase, os estereótipos para os elementos relevantes da aplicação são criados.
3. Cada estereótipo será aplicado à meta-classe pertencente ao meta-modelo para a definição de um conceito ou de um relacionamento.
4. Após a definição dos estereótipos, são definidos seus *tagged values*, incluindo seus tipos e valores iniciais.

Por último, as restrições são definidas e associadas a cada elemento que compõe o meta-modelo.

2.6.1. Estereótipos

Os estereótipos são usados para criar novos elementos de modelagem, a partir dos elementos já existentes na UML, ou seja, eles “estendem a semântica, mas não a estrutura do elemento” (PÁDUA, 2007). Em outras palavras, a partir dos estereótipos uma pessoa pode “especializar” a UML (ERIKSSON et al., 2004).

Um estereótipo estende uma meta-classe ou um outro estereótipo criando uma nova meta-classe no modelo. Uma meta-classe é uma classe cujas instâncias são classes, e não objetos, e é representada pela notação <<metaclass>>. Estereótipos podem estar associados a classes e associações (ERIKSSON et al., 2004).

A notação de um estereótipo é <<nome do estereótipo>> e o desenvolvedor pode adicionar um ícone para representar o estereótipo criado.

A UML possui vários estereótipos pré-definidos, como <<boundary>>, que representa as classes de interface (limite do sistema), <<control>>, que representam as classes de controle do sistema, e <<entity>>, que representam as classes persistentes do sistema, entre outros.

Os desenvolvedores “podem criar novos estereótipos para classes, definindo regras comuns usadas em um domínio específico, cujas classes possuem seus próprios ícones e significados”. Dessa forma, *tagged values* e restrições podem ser associados a determinado estereótipo. Dessa forma, um novo tipo de elemento é definido (ERIKSSON et al., 2004).

Para que um estereótipo seja definido, é necessário descrever (ERIKSSON et al., 2004):

- Em qual elemento o estereótipo é baseado.
- A nova semântica do estereótipo adicionado ou refinado.
- Um ou mais exemplos de como implementar o estereótipo.

2.6.2. *Tagged Values*

“Uma propriedade representa um valor associado a um elemento” (ERIKSSON et al., 2004). As propriedades contêm informações dos elementos no modelo, sem necessariamente aparecer no sistema final.

Os *tagged values* são propriedades associadas a um elemento e são representados por um par nome-valor. “Eles permitem associar arbitrariamente informação a qualquer elemento do modelo, sendo o seu significado

intencionalmente específico dentro do contexto da sua utilização, podendo ser determinado pelo seu utilizador ou por convenções das ferramentas de suporte” (STEINMACHER, 2003).

O desenvolvedor da aplicação pode definir seus próprios *tagged values* dentro de um perfil, criando seus pares nome-valor como parte de um estereótipo. Esse mecanismo de extensão se torna um método útil para acrescentar informações adicionais a um modelo, além de acrescentar semântica extra a um elemento (ERIKSSON et al., 2004).

Para definir um *tagged value*, os seguintes passos devem ser seguidos (ERIKSSON et al., 2004):

1. Investigue o propósito do *tagged value*.
2. Defina o elemento ao qual o *tagged value* estará associado.
3. Nomeie adequadamente o *tagged value*.
4. Defina o tipo de seu valor.
5. Conheça quem ou o que irá interpretá-lo (um computador ou uma pessoa). Se for um computador, o valor deve ser especificado em uma sintaxe mais formal, como, por exemplo, através de uma linguagem de programação.
6. Documente um ou mais exemplos mostrando o uso do *tagged value*.

2.6.3. Restrições

“Uma restrição é uma condição semântica ou restrição de um elemento e são associadas aos elementos através de uma expressão”. A notação para uma restrição é escrevê-la entre chaves ({expressão da restrição}). As restrições podem ser associadas a classes e associações (ERIKSSON et al., 2004).

Restrições “estendem a semântica de um elemento UML, permitindo a definição de novas regras ou modificar regras existentes” (PÁDUA, 2007). Uma restrição pode ser definida de forma textual (linguagem natural) ou através de

expressões OCL (*Object Constraint Language*), sendo que a escolha deve ser feita de acordo com as necessidades do projeto.

Restrições também “são usadas para limitar a semântica de um elemento em termos de condições ou restrições de sua semântica e, em alguns casos, podem representar características avançadas do modelo” (ERIKSSON et al., 2004).

Para definir uma restrição, o desenvolvedor deve descrever (ERIKSSON et al., 2004):

- A qual elemento a restrição se aplica.
- O impacto semântico da restrição no elemento.
- Um ou mais exemplos de como aplicar a restrição.
- Como a restrição pode ser implementada.

2.7 Extensões UML para Aplicações Web

O desenvolvimento de aplicações móveis muito se assemelha ao desenvolvimento de aplicações WEB, que se baseia no modelo Cliente-Servidor, onde clientes requisitam acessos ou serviços a um determinado servidor através de uma rede.

O modelo de arquitetura Cliente-Servidor é um modelo de sistema distribuído, cujos componentes principais são um conjunto de servidores, que oferece os serviços a outros subsistemas, um conjunto de clientes, que solicita os serviços oferecidos pelos servidores, e uma rede, que permite aos clientes acessar esses serviços (SOMMERVILLE, 2003).




A UML pura não oferece todos os recursos necessários para modelar esse tipo de aplicação. Por isso, no final da década de 90, Jim Conallen criou as extensões UML para aplicações WEB¹, também chamadas de WAE (*Web Applications Extensions*).

¹ Mais informações sobre as extensões UML para aplicações móveis em (CONALLEN, 1999) e (CONALLEN, 2000).

Com essas extensões, Conallen modelou os elementos básicos contidos em uma aplicação WEB, tais como: página cliente, página servidor, formulários, *frames*, *links*, entre outros.

A Página Servidor representa uma página WEB que contém *scripts* executados no servidor da aplicação. Já a Página Cliente representa uma página visualizada no navegador (*browser*) do cliente (ARAÚJO, 2007). Os Formulários modelam os campos de entrada de dados que fazem parte de Páginas Cliente. As relações entre essas páginas também são modeladas, através de associações *build*, usada quando uma Página Servidor cria uma Página Cliente, *link*, que representa a ligação entre duas páginas, *submit*, que envia os dados de um formulário para um servidor, entre outras.

A Tabela 2 mostra os estereótipos gráficos criados por Conallen para a modelagem de aplicações WEB.

Classe	Ícone
Página Servidor	
Página Cliente	
Formulário	

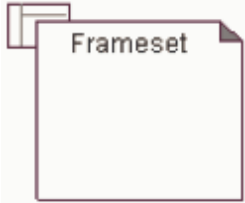

Frame set	
Target	

Tabela 2 - Estereótipos gráficos WAE (NETO & GRAHL, 2007)

Um exemplo de modelagem utilizando essas extensões pode ser visto na Figura 7, onde são ilustradas relações entre páginas cliente e páginas servidor:

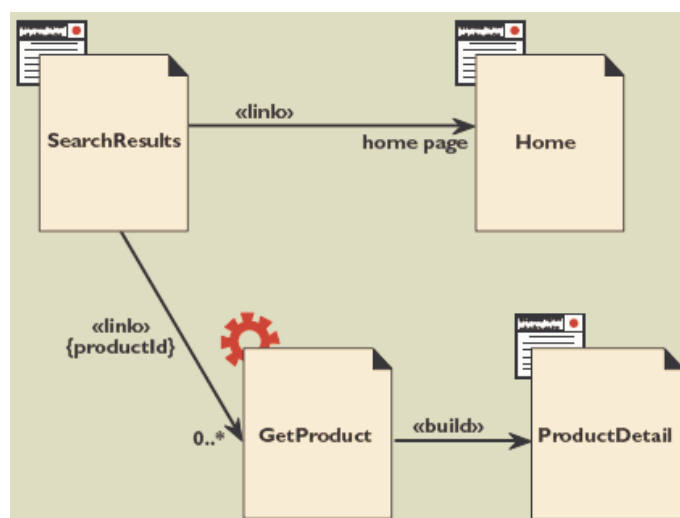


Figura 7 - Exemplo da utilização das extensões para aplicações Web (CONALLEN, 1999)

Devido à semelhança entre o desenvolvimento WEB e o desenvolvimento móvel, muitas dessas extensões podem ser facilmente utilizadas ou adaptadas quando se pretende modelar aplicações que envolvem mobilidade. Com isso, o

presente trabalho se baseou nas extensões WEB para criar as extensões para aplicações móveis.

O Apêndice B apresenta mais informações a respeito das extensões UML para aplicações Web.

Capítulo 3

Requisitos de Usabilidade para Aplicações em Ambientes Móveis

3.1. Introdução

Segundo (PÁDUA, 2008a), “Atributo de usabilidade é uma característica geral de usabilidade a ser usada como critério para avaliação da interface”. A Norma ISO 9241-11 (ISO 9241-11, 1998) afirma que para medir usabilidade devem-se levar em consideração os seguintes atributos: objetivos, eficácia, eficiência e satisfação do usuário, além de considerar os componentes de contexto da interação, como o usuário, a tarefa a ser realizada, o equipamento utilizado e o ambiente onde se dá a interação. A Figura 8 ilustra a estrutura da usabilidade de acordo com a Norma ISO 9241-11 (ISO 9241-11, 1998).

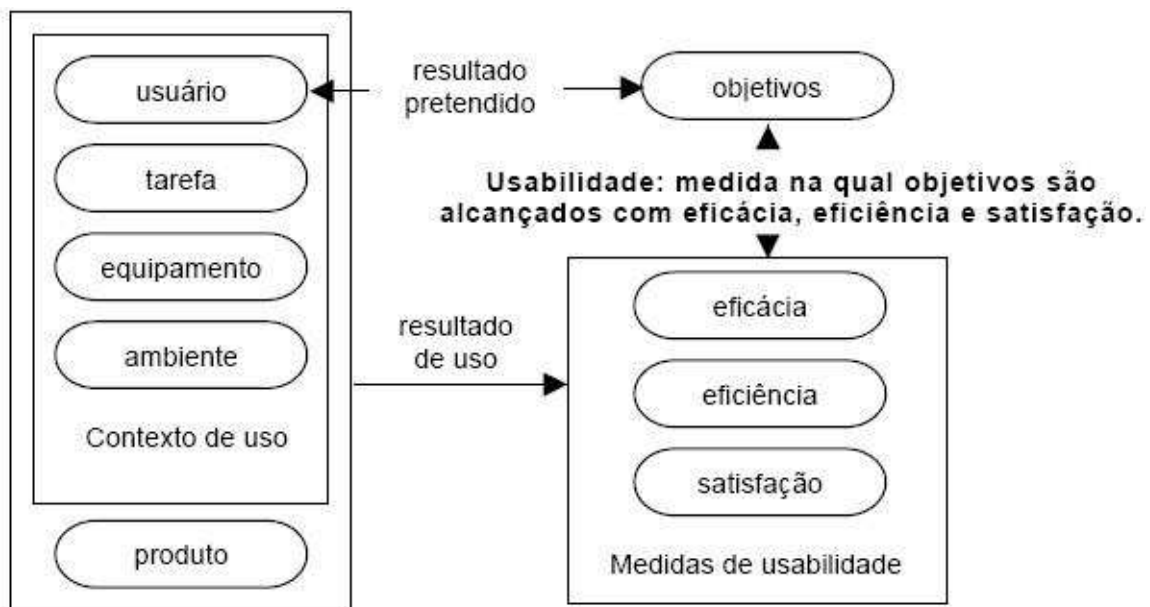


Figura 8 - Estrutura da Usabilidade (ISO 9241-11, 1998)

3.2. Atributos de Usabilidade

A fim de se obter um software com melhor usabilidade e aceitação do usuário, e, por conseguinte, melhor qualidade, alguns requisitos de usabilidade são levantados durante a fase de projeto do software. Alguns desses atributos são sugeridos em (PÁDUA, 2008a) e são listados a seguir:

- Desempenho Inicial – diz respeito ao desempenho inicial do usuário durante a sua primeira interação (primeira vez de uso) com o sistema.
- Desempenho a longo prazo – diz respeito ao desempenho do usuário durante o uso mais freqüente do sistema a um período longo.
- Aprendizagem – “rapidez e facilidade com que o usuário aprende a lidar com o sistema”.
- Retenção – capacidade de o usuário reter na memória o que aprendeu em determinado tempo de uso quando volta a utilizar o sistema (interface).
- Uso de características avançadas – determina a usabilidade de funções complexas da interface.
- Primeira impressão – “avaliação inicial do usuário”.
- Satisfação do usuário a longo prazo – o quão o usuário está satisfeito em utilizar o sistema após um prazo maior de uso.
- Prevenção de erros – “capacidade do sistema de evitar erros em sua utilização”.
- Acurácia – “capacidade de oferecer resultados na precisão desejada pelo usuário”.
- Confiabilidade – o quão o software executa com confiança o que se propôs a fazer.
- Clareza – avalia a clareza com que o software se “comunica” com o usuário. Em outras palavras, o quão claras são as informações que são passadas para o usuário.

(CYBIS et al., 2007), listam atributos ergonômicos para o projeto de interfaces gráficas visando uma maior usabilidade. Esses atributos foram propostos originalmente em 1993 por Dominique Scapin e Christian Bastien e são listados a seguir:

- Condução – esse atributo busca favorecer a utilização do sistema por usuários novatos e é subdividido em quatro outros atributos: convite, agrupamento e distinção entre itens, legibilidade e *feedback* imediato.
 - Convite – diz respeito aos “meios utilizados” para facilitar a realização de ações pelo usuário e a identificação do estado ou contexto em que está inserido durante a interação.
 - Agrupamento e distinção de itens – diz respeito ao posicionamento, ordenação e formato dos objetos que compõem a interface, a fim de facilitar a “intuitividade” do usuário.
 - Legibilidade – diz respeito à apresentação dos caracteres para o usuário, a fim de facilitar a legibilidade da tela.
 - *Feedback* imediato – diz respeito à resposta do sistema ao usuário, a fim de facilitar o entendimento do mesmo acerca do funcionamento do sistema durante a interação.
- Carga de trabalho – “diz respeito a todos os elementos da interface que têm papel importante na redução da carga cognitiva e perceptiva do usuário” e é subdividido em dois outros atributos: brevidade e densidade informacional.
 - Brevidade – é composto por dois outros atributos: **concisão**, que diz respeito à concisão das informações apresentadas pela interface, minimizando a carga perceptiva do usuário, e **ações mínimas**, que diz respeito à simplificação das ações para que o usuário realize uma função do sistema.
 - Densidade informacional – “diz respeito à carga do trabalho do usuário, de um ponto de vista perceptivo e cognitivo, com relação ao conjunto total de itens de informação apresentados”.

- Controle explícito – diz respeito ao controle do usuário sobre suas ações dentro do sistema e é composto de dois outros atributos: ações explícitas do usuário e controle do usuário.
 - Ações explícitas do usuário – o sistema “executa somente aquilo que o usuário quiser e somente quando ele ordenar”, facilitando, assim, o aprendizado e entendimento do usuário acerca do sistema que está utilizando.
 - Controle do usuário – diz respeito ao controle do sistema pelo usuário, como, por exemplo, interromper um processamento.
- Adaptabilidade – diz respeito à adaptação da interface aos diversos usuários que a utilizam e é composto de dois outros atributos: flexibilidade e experiência do usuário.
 - Flexibilidade – diz respeito às diversas maneiras oferecidas para a realização de determinada tarefa pelo usuário.
 - Experiência do usuário – a interface deve oferecer meios para usuários experientes e novatos realizar a mesma tarefa de diferentes maneiras, levando em consideração o tempo de experiência do usuário com a utilização do sistema.
- Gestão de erros – diz respeito ao gerenciamento de situações de erro dentro do sistema e é composto por três outros atributos: prevenção contra erros, qualidade das mensagens de erro e correção de erros.
 - Prevenção contra erros – refere-se à capacidade de o sistema proporcionar ao usuário a prevenção de erros.
 - Qualidade das mensagens de erro – refere-se à “pertinência, legibilidade e à exatidão” das mensagens de erro dadas ao usuário.
 - Correção de erros – diz respeito aos meios que o sistema oferece ao usuário para que ele saia de uma situação de erro.

- Homogeneidade/Coerência – “refere-se à forma na qual as escolhas no projeto de interface são conservadas idênticas em contextos idênticos e diferentes para contextos diferentes”.
- Significado de códigos e denominações – diz respeito à conformidade entre uma informação (ou objeto) apresentada para o usuário e seu significado na interface.
- Compatibilidade – “diz respeito ao grau de similaridade entre diferentes sistemas que são executados em um mesmo ambiente operacional”.

3.3. Atributos de Usabilidade para Aplicações Móveis

Restrições impostas pela mobilidade, como tamanho reduzido das telas, capacidade de processamento, capacidade de armazenamento, entre outros (FORMAN & ZAHORJAN, 1994), implicam em diferenças significativas no desenvolvimento de interfaces gráficas para aplicações executadas em computadores *desktop* e computadores móveis. Isso faz com que o estudo da usabilidade nesses dois ambientes seja diferenciado e novos atributos sejam inseridos e/ou adaptados pela mobilidade.

Segundo (TERRENGHI et al., 2005), o processo centrado no usuário, para criação de interfaces gráficas para ambientes que envolvem mobilidade, faz com que aspectos que não eram contemplados tradicionalmente pela análise da usabilidade “venham à tona” e apóiem o *design* criativo para essas aplicações e, com isso, “prevêem” uma melhor aceitação por parte do usuário e garantindo uma melhor usabilidade do sistema.

Nesse estudo, (TERRENGHI et al., 2005) propõem alguns requisitos de usabilidade para aplicações que envolvem mobilidade baseados na idéia de que a mobilidade pode ser entendida com base em três aspectos: **mobilidade do dispositivo**, se refere ao acesso a serviços através do dispositivo enquanto o usuário se movimenta, **mobilidade do usuário**, se refere à localização e ao acesso a serviços independente do dispositivo, e **mobilidade do serviço**,

“capacidade de prover serviços independente do usuário e do dispositivo”. Esses requisitos são listados a seguir:

- Portabilidade – as diversas mudanças de ambiente e as restrições impostas pelos diversos tipos de dispositivos fazem com que a característica **portabilidade** se torne evidente em aplicações móveis. Portabilidade pode ser entendida como a “capacidade de o software ser transferido de um ambiente a outro” (ISO/IEC 9126-1, 2001 apud TERRENGHI et al., 2005).
- Adaptabilidade – devido às diversas mudanças de ambiente, o sistema deve possuir a capacidade de adaptação aos diversos ambientes. Assim, uma maneira de prover boa usabilidade às aplicações que envolvem mobilidade é criar interfaces previsíveis ao usuário e que se adaptem ao contexto da interação. Adaptabilidade é uma subcategoria de Portabilidade e se refere à capacidade de o sistema operar independente das condições de uso.
- Facilidade de aprendizado do usuário – esse atributo deve ser analisado levando em consideração o fato de o usuário interagir com o dispositivo móvel de maneira diferente da que interage com um computador *desktop*. Como mencionado na Seção 2.3 a interação móvel está sujeita a transferências constantes do ambiente externo. Isso faz com que a percepção do usuário e a capacidade de aprendizado também sofram interferências. Enquanto se move, o usuário tende a usar serviços que possuem interfaces gráficas de fácil manipulação e com um número reduzido de passos para acessar a informação.
- Segurança – existem diferenças significativas quando se trata segurança em dispositivos móveis e segurança em computadores *desktop*. Os dispositivos portáteis estão mais vulneráveis a perda e danos e essas perdas podem afetar a privacidade do usuário devido ao fato de os dispositivos serem pessoais. Diferentemente de um computador *desktop* geralmente ser acessado por diversas pessoas, o dispositivo móvel geralmente é acessado apenas por uma pessoa.

- Confiabilidade – independente dos diversos ambientes em que se encontra, o sistema deve prover o mesmo nível de performance para o usuário.
- Atratividade – além de fornecer fácil acesso às funções e ser de fácil manipulação, a interface gráfica de um sistema deve ser atraente para o usuário.
- Interoperabilidade – o sistema deve ser capaz de operar em diversos ambientes e com diversos tipos de rede.

Capítulo 4

Perfil UML para Aplicações em Ambientes Móveis

4.1 Introdução

Durante a fase de levantamento bibliográfico, constatou-se, como dito anteriormente, que a UML não possui todos os recursos necessários para a modelagem das restrições impostas pela mobilidade, ou seja, ela possui os elementos básicos para a modelagem de um sistema, como casos de uso, classes, entre outros, mas esses elementos não modelam todos os requisitos presentes em determinados tipos de problemas. Com isso, viu-se que seria necessária a criação de extensões UML que permitissem modelar esses casos. Durante essa fase, percebeu-se que o desenvolvimento de aplicações envolvendo mobilidade é semelhante ao desenvolvimento de aplicações para WEB, uma vez que em geral seguem o mesmo modelo Cliente-Servidor.

Após a fase de levantamento bibliográfico, percebeu-se que as extensões UML para desenvolvimento de aplicações WEB, apresentadas na Seção 2.7, propostas em (CONALLEN, 2000), poderiam ser adaptadas para o desenvolvimento de aplicações móveis. No entanto, as particularidades e dificuldades inerentes às aplicações desenvolvidas para dispositivos móveis, em particular com relação à interface homem-máquina, não são contempladas por essas extensões. Outros estudos, como o apresentado em (KOCH et al., 2007), também propõem extensões para o desenvolvimento para WEB, como já foi descrito na Seção 1.3.

Diante disso, o presente estudo se baseou nas extensões propostas em (CONALLEN, 2000) e acrescentou a elas algumas novas extensões a fim de modelar as limitações relativas à interface gráfica e interação com o usuário

impostas pela mobilidade, melhorando, dessa forma, a usabilidade desses sistemas.

Como mencionado anteriormente, duas grandes preocupações ao se desenvolver aplicações para dispositivos móveis são a forma como as informações serão apresentadas na tela e a forma como o usuário irá interagir com a aplicação. A fim de resolver esses problemas, este estudo propôs a criação de métodos de classes que buscam nos dispositivos móveis dados técnicos sobre suas interfaces de entrada e saída de dados. Usando esses dados, a interface gráfica poderia ser customizada para determinado dispositivo, ou seja, poderia ser apresentada ao usuário atendendo às restrições impostas pelo aparelho que ele estiver utilizando. Com isso, o desenvolvimento seria feito de acordo com as boas práticas da WEB móvel, propostas pelo W3C, disponíveis em (RABIN & McCATHIENEVILE, 2006).

4.2 Perfil UML para Aplicações Móveis

De acordo com o W3C, deve haver um equilíbrio no tamanho das páginas que são exibidas em dispositivos móveis. Se as páginas são muito grandes, podem levar muito tempo para carregá-las e apresentá-las ao usuário. Além disso, muitos dispositivos têm restrição em relação ao tamanho máximo da página que pode ser apresentado. Por outro lado, se as páginas são muito pequenas, o usuário terá que fazer várias requisições a fim de visualizar todo o conteúdo disponível, o que acarreta em um atraso desnecessário, uma vez que cada requisição demanda um tempo para ser completada (RABIN & McCATHIENEVILE, 2006).

Outro fator importante em relação à apresentação das interfaces gráficas de determinada aplicação nas telas dos dispositivos é a distorção que pode haver pelo fato de haver vários tamanhos de telas disponíveis. Com isso, a interação do usuário com o sistema fica prejudicada, uma vez que o usuário não consegue ter uma visualização perfeita das telas.

Visando a atender às boas práticas propostas pela W3C e à melhoria da usabilidade de aplicações que são executadas em ambientes móveis, o presente

trabalho apresenta extensões UML que visam à adaptação das interfaces gráficas das aplicações ao tamanho da tela do dispositivo utilizado para o acesso.

4.2.1. Meta-modelo

A Figura 9 representa o meta-modelo do perfil UML para aplicações móveis. As próximas seções apresentam os estereótipos criados.

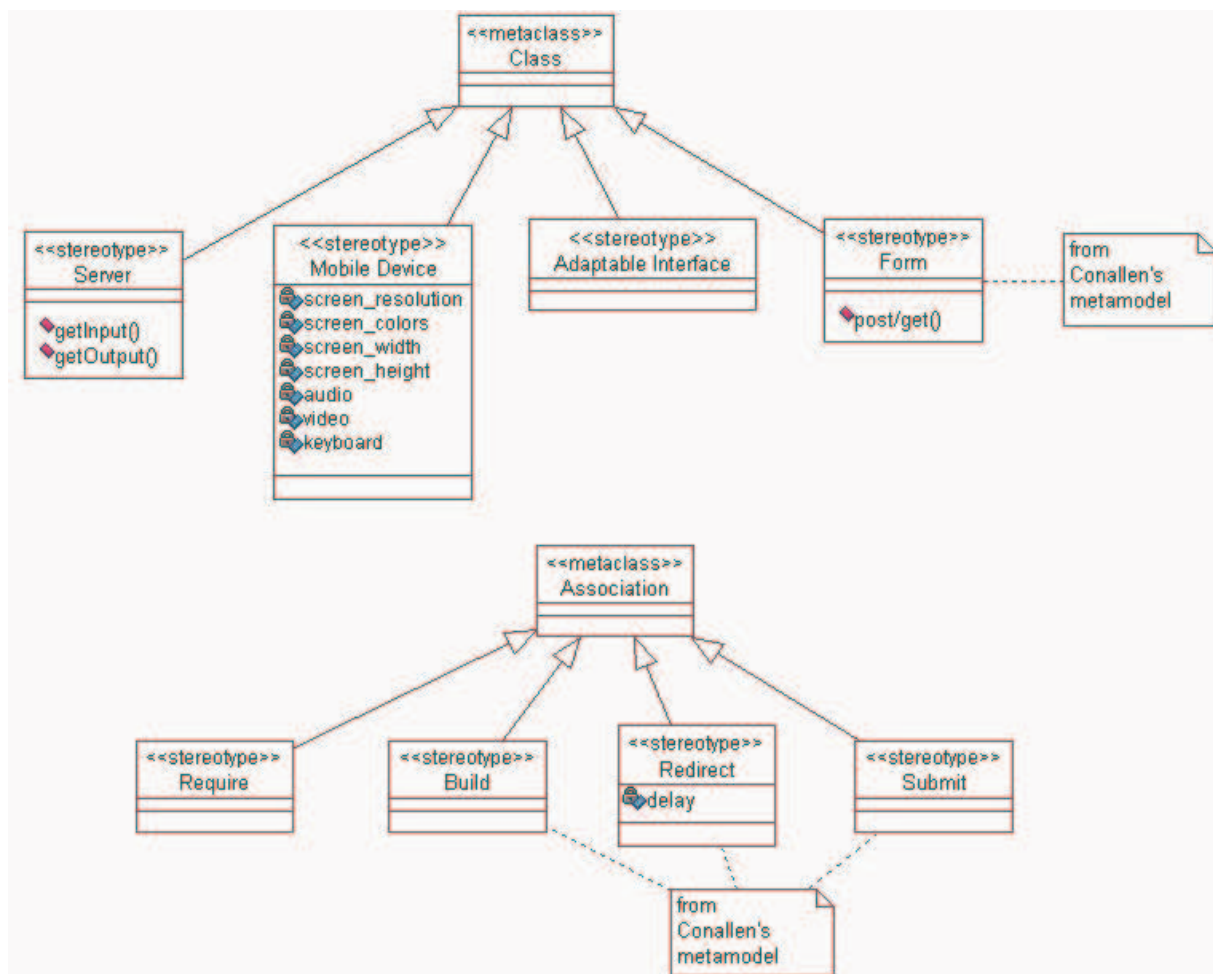


Figura 9 - Meta-modelo do perfil UML para aplicações em ambientes móveis

4.2.2. Server

O estereótipo *Server* representa o servidor da aplicação. Esse servidor busca as informações referentes às interfaces de entrada e saída no dispositivo móvel que está solicitando determinado serviço e constrói a interface adaptada para aquele dispositivo, respeitando as interfaces de entrada e saída nele presentes.

No estereótipo *Server*, estão presentes os métodos *getInput* e *getOutput*. Os métodos *getInput* e *getOutput* são modelados como *tagged values* associados ao estereótipo *Server*. Através desses métodos, toda vez que for feita uma requisição ao servidor, são buscados, no dispositivo solicitante, as interfaces de entrada e saída de dados. A partir do retorno desses métodos, a interface gráfica é adaptada e apresentada ao usuário atendendo às limitações impostas pelo dispositivo utilizado.

O ícone que representa o estereótipo *Server* é apresentado na Figura 10:

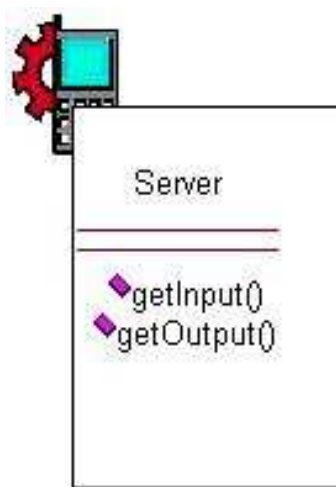


Figura 10 - Ícone para o estereótipo *Server*

4.2.3. *Mobile Device*

Mobile Device descreve o estereótipo que contém as características do dispositivo móvel que solicita determinado serviço, executa determinada aplicação ou utiliza uma aplicação WEB. Esse estereótipo contém características de interação do usuário com o dispositivo, ou seja, contém as informações referentes às interfaces de entrada e saída presentes no dispositivo, a partir das quais a interface gráfica será construída e adaptada ao dispositivo pelo estereótipo *Server*.

Essas informações são atributos, modelados como *tagged values*, associados ao estereótipo *Mobile Device*. São eles:

- *screen_resolution*: atributo que contém a informação referente à resolução da tela do dispositivo.
- *screen_colors*: atributo que contém a informação referente ao sistema de cores adotado pelo dispositivo.
- *screen_width*: atributo que contém a largura da tela do dispositivo.
- *screen_height*: atributo que contém a altura da tela do dispositivo.
- *audio*: atributo que contém a informação referente ao tipo de áudio suportado pelo dispositivo.
- *video*: atributo que contém informação referente ao tipo de vídeo suportado pelo dispositivo.
- *keyboard*: atributo que contém informação referente ao tipo de teclado presente no dispositivo.

A Figura 11 ilustra o ícone para o estereótipo *Mobile Device*:

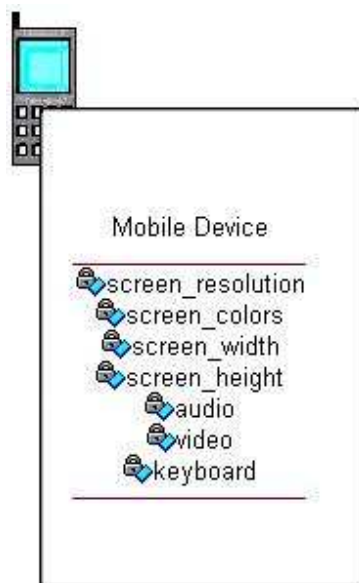


Figura 11 - Ícone para o estereótipo *Mobile Device*

4.2.4. *Adaptable Interface*

O estereótipo *Adaptable Interface* modela a interface adaptável construída pelo estereótipo *Server* através das informações obtidas através do estereótipo *Mobile Device*.

Associados a esse estereótipo estão os seguintes requisitos de usabilidade:

- Concisão: concisão das informações apresentadas na tela da interface gráfica.
- Ações Mínimas: simplificações para que os usuários realizem operações com ações mínimas.
- Significado de Códigos e Denominações: conformidade entre a informação (objeto) e seu significado.
- Facilidade de Aprendizagem: interface gráfica de fácil manipulação e número reduzido de passos para acessar a informação.
- Atratividade: a interface deve ser atraente para o usuário.

O ícone que o representa o estereótipo *Adaptable Interface* é ilustrado pela Figura 12:

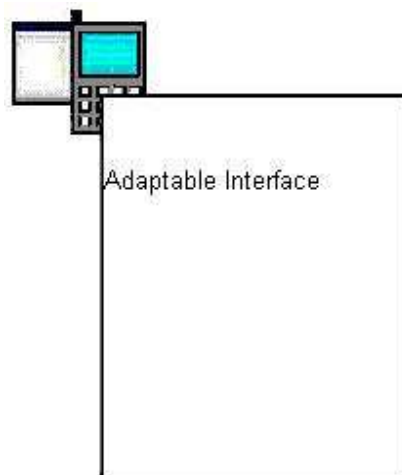


Figura 12 - Ícone para o estereótipo *Adaptable Interface*

4.2.5. *Require*

Require é a associação unidirecional entre os estereótipos *Server* e *Mobile Device* e representa a busca de informações referentes às interfaces de entrada e saída existentes no dispositivo. Esta associação possui como restrição o fato de ser possível somente entre *Server* e *Mobile Device*, sendo que a mesma parte de *Server* para *Mobile Device*.

A Figura 11 ilustra a associação *require*.



Figura 13 - Associação *Require*

4.3 Extensões Web adaptadas para aplicações móveis

Algumas das extensões apresentadas na Seção 2.7 e no Apêndice B foram adaptadas neste perfil. São elas: *Form*, *Build*, *Redirect* e *Submit*.

O estereótipo ***Form*** representa um formulário composto por uma coleção de campos através do qual o usuário envia dados ao sistema. Um formulário faz parte da interface gráfica com o usuário. No caso do perfil aqui apresentado, faz parte de uma classe com o estereótipo *Adaptable Interface*. O estereótipo *Form* possui como *tagged value* o método *post* (ou *get*), usado para enviar os dados para o servidor. O estereótipo ***Submit*** é uma associação que representa esse envio de dados de um *Form* para o sistema.

Associados ao estereótipo *Form* estão os seguintes requisitos de usabilidade:

- Concisão: concisão das informações apresentadas pelo formulário.

- Ações Mínimas: simplificações para que o usuário realize operações com ações mínimas.
- Facilidade de Aprendizado: interface gráfica de fácil manipulação e número reduzido de passos para acessar a informação.

O ícone que representa o estereótipo *Form* é ilustrado pela Figura 14 e a associação *submit* é ilustrada pela Figura 15.

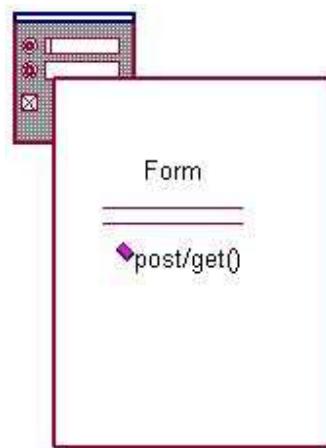


Figura 14 - Ícone para o estereótipo *Form*

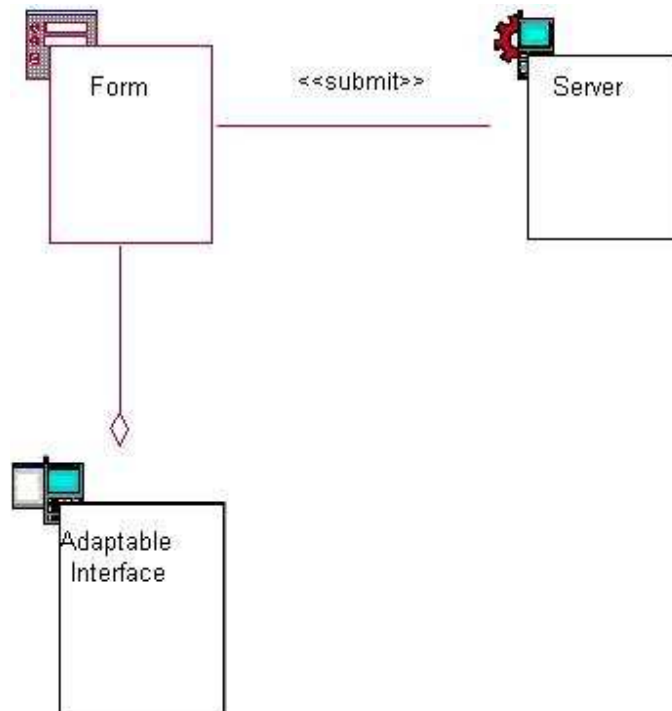


Figura 15 - Associação *Submit*

Build é uma associação entre um *Server* e um *Adaptable Interface* que representa a criação de uma tela de interface gráfica pelo servidor. A Figura 16 representa essa associação.

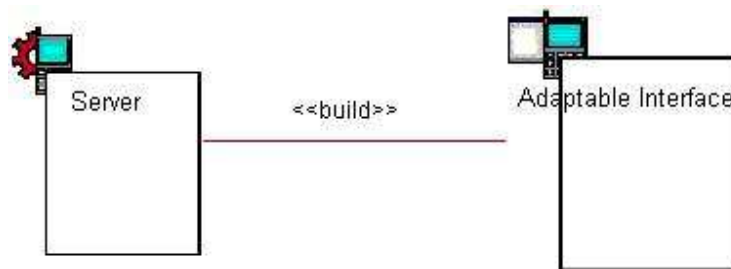


Figura 16 - Associação Build

A associação **Redirect** é uma associação unidirecional entre *Server* e *Adaptable Interface* e representa o redirecionamento de uma tela para outra. Essa associação contém como *tagged value* um atributo, chamado *delay*, que representa o tempo que a classe deve esperar até o redirecionamento para a outra. A Figura 17 ilustra a associação *redirect*.

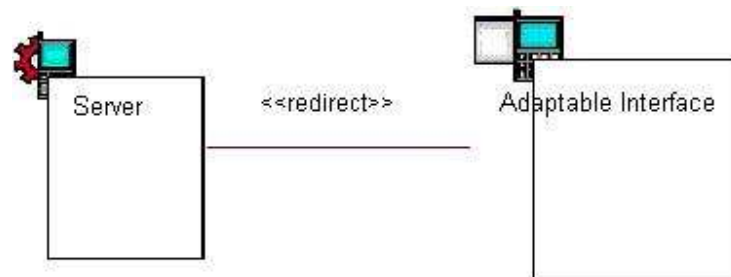
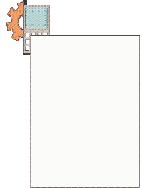
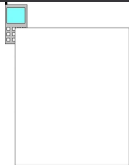
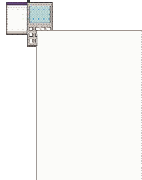
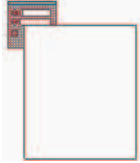


Figura 17 - Associação Redirect

4.4 Comentário Final

A Tabela 3 apresenta um resumo dos estereótipos usados no Perfil UML para aplicações em ambientes móveis, apresentado neste capítulo.

	Ícone	Descrição	Tagged values	Restrições
<i>Server</i>		Modela o servidor da aplicação, que busca as informações referentes às interfaces de entrada e saída no dispositivo móvel e constrói a interface adaptada para aquele dispositivo.	Métodos: getInput getOutput	-
<i>Mobile Device</i>		Contém as características do dispositivo móvel que solicita determinado serviço, executa determinada aplicação ou utiliza uma aplicação Web.	Atributos: screen_resolution screen_colors screen_width screen_height audio vídeo keyboard	-
<i>Adaptable Interface</i>		Modela a interface adaptável construída pela classe <i>Server</i> através das informações referentes à entrada e saída obtidas através da classe <i>Mobile</i>	-	-

		<i>Device.</i>		
<i>Form</i>		Representa um formulário composto por uma coleção de campos através do qual o usuário envia dados ao sistema.	Método: post ou get	-
<i>require</i>	-	Associação unidirecional entre as classes <i>Server</i> e <i>Mobile Device</i> quando o servidor busca as informações referentes às interfaces de entrada e saída existentes no dispositivo.	-	Possível apenas entre uma classe <i>Server</i> e uma classe <i>Mobile Device</i> .
<i>build</i>	-	Associação entre uma classe <i>Server</i> e uma classe <i>Adaptable Interface</i> que representa a criação de uma tela de interface gráfica pelo servidor.	-	-
<i>submit</i>	-	Associação entre as classes <i>Form</i> e <i>Server</i> onde os dados do formulário são enviados para o servidor.	-	-
<i>redirect</i>	-	Associação entre a classe <i>Servidor</i> e uma classe <i>Adaptable Interface</i> e representa o	Atributo: <i>delay</i>	-

		redirecionamento de uma tela para outra.		
--	--	--	--	--

Tabela 3 - Perfil UML para aplicações em ambientes móveis

Capítulo 5

m-PVANET – Um Exemplo de Aplicação

5.1. Introdução

A exemplificação da aplicação do perfil foi feita utilizando uma aplicação real, m-PVANET (*Mobile PVANET*), que consiste na adaptação do PVAnet para dispositivos móveis e está sendo desenvolvida por (MIRANDA et al., 2007). A exemplificação consistiu na modelagem, através da UML e das extensões aqui apresentadas, de toda a interface gráfica com o usuário, através do Diagrama de Casos de Uso, Diagramas de Classes e Diagramas de Seqüência.

O PVANET é uma ferramenta de auxílio ao ensino, desenvolvida e usada na Universidade Federal de Viçosa, que contém como principal finalidade aumentar a integração entre os professores e os alunos desta universidade. Apesar de este ser o principal objetivo, a ferramenta contém também um “módulo” que visa repassar a estudantes localizados a longa distância, conteúdos sobre os cursos a distâncias oferecidos por esta instituição (FERRAZ et al., 2007).

A Figura 18 apresenta a tela principal de uma disciplina no PVAnet versão para *desktop*. A mesma tela, acessada a partir de um PDA (*Personal Digital Assistant*), é apresentada pela Figura 19. O que se percebe é que a interação com o usuário fica comprometida, uma vez que a tela, projetada para visualização em *desktop*, fica com a visualização distorcida em um dispositivo móvel. A fim de resolver esse problema, uma adaptação é necessária. Dessa forma, a tela foi adaptada para o dispositivo móvel que faz o acesso e a interação com o usuário não fique comprometida, favorecendo, assim, a usabilidade do sistema. O resultado dessa adaptação é apresentado pela Figura 20.



Figura 18 - Tela de disciplina no PVANet em um *desktop* (LADEIRA, 2007)



Figura 19 - Tela de disciplina (sem adaptação) em um dispositivo móvel (LADEIRA, 2007)



Figura 20 - Tela de disciplina adaptada para um dispositivo móvel (LADEIRA, 2007)

As próximas seções apresentam a modelagem do sistema descrito acima. Toda a modelagem é feita de acordo com as extensões aqui apresentadas.

5.2. Diagrama de Casos de Uso

“O Modelo de Casos de Uso é uma representação das funcionalidades externamente observáveis do sistema e dos elementos externos ao sistema que interagem com ele, além de ser um modelo de análise que representa um refinamento dos requisitos funcionais do sistema em desenvolvimento”. Assim, uma das finalidades do Diagrama de Casos de Uso é apresentar os elementos externos e as funcionalidades de um sistema e as maneiras segundo as quais eles as utilizam (BEZERRA, 2007).

O Diagrama de Casos de Uso apresentado na Figura 21 representa as funcionalidades do m-PVANET.

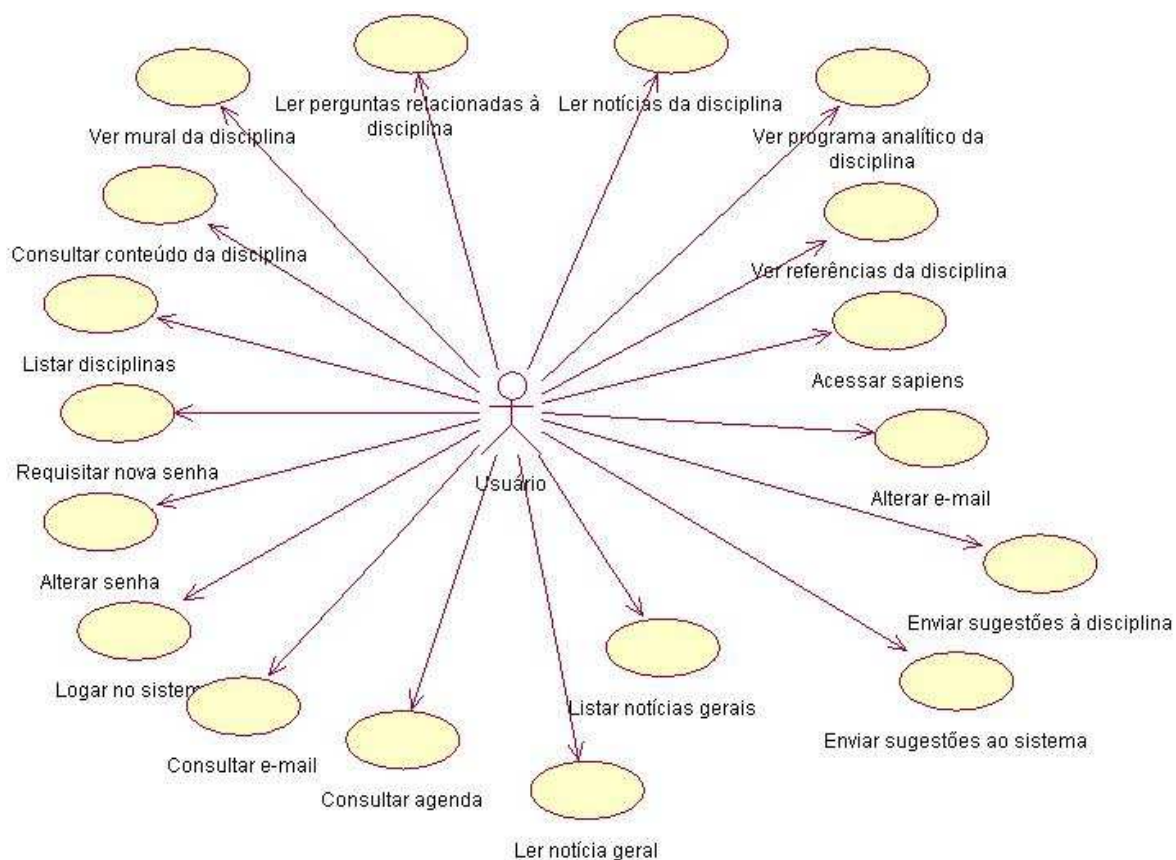


Figura 21 - Diagrama de Casos de Uso – m-PVANET

5.3. Diagrama de Classes

O Diagrama de Classes modela todas as classes existentes no sistema e o relacionamento entre elas. O Modelo de Classes é dividido em três: **classes de análise**, que representam as classes “que se tornam evidentes à medida em que se foca a atenção sobre o que o sistema deve fazer”, **classes de especificação**, onde a atenção é focada sobre “como” o sistema deve fazer, e **classes de implementação**, que “corresponde à implementação das classes em alguma linguagem de programação” (BEZERRA, 2007).

O caso de uso escolhido para a exemplificação do perfil apresentado foi o caso de uso Consultar conteúdo da disciplina. A modelagem foi feita com a utilização dos estereótipos apresentados no Capítulo 4. A Tabela 4 apresenta o fluxo lógico do caso de uso.

Fluxo Lógico de Caso de Uso	
Caso de Uso: Consultar conteúdo da disciplina	
Ator: Usuário	
Seqüência Típica de Eventos	
Ação do Ator	Resposta do Sistema
1 – O caso de uso inicia quando o usuário seleciona a opção de visualizar o material de apoio da disciplina.	
	2 – O sistema busca no acervo a lista de material disponível e informa ao usuário.
3 – O usuário seleciona o material de apoio que deseja visualizar.	
	4 – O sistema busca no dispositivo móvel as informações sobre as interfaces de entrada e saída de dados.
	5 – O dispositivo móvel retorna as informações sobre interfaces de entrada e saída.
	6 – Com base nas informações obtidas, o sistema cria a interface gráfica e exibe para o usuário o material de apoio solicitado.
7 – O usuário seleciona material para <i>download</i> .	
	8 – O sistema busca o material solicitado e efetua o <i>download</i> .

Tabela 4 - Fluxo Lógico do Caso de Uso Consultar Conteúdo da Disciplina

A Figura 22 apresenta o diagrama de classes referente ao caso de uso **Consultar conteúdo da disciplina**. A classe TelaConteudoDisciplina representa a interface gráfica com o usuário que exibe as informações referentes a determinada disciplina. O servidor é representado pela classe ConteudoDisciplina e é responsável pela comunicação entre a interface gráfica e o sistema, além de ser o responsável pela construção das interfaces gráficas com o usuário, obedecendo as restrições impostas pelo dispositivo, e pela comunicação com o dispositivo móvel utilizado na interação. O dispositivo móvel é representado pela classe IOInterface. A tela que exibe o material de apoio da disciplina é

representada pela classe TelaMaterialApoio e é através dela que o usuário faz o *download* do material disponível.

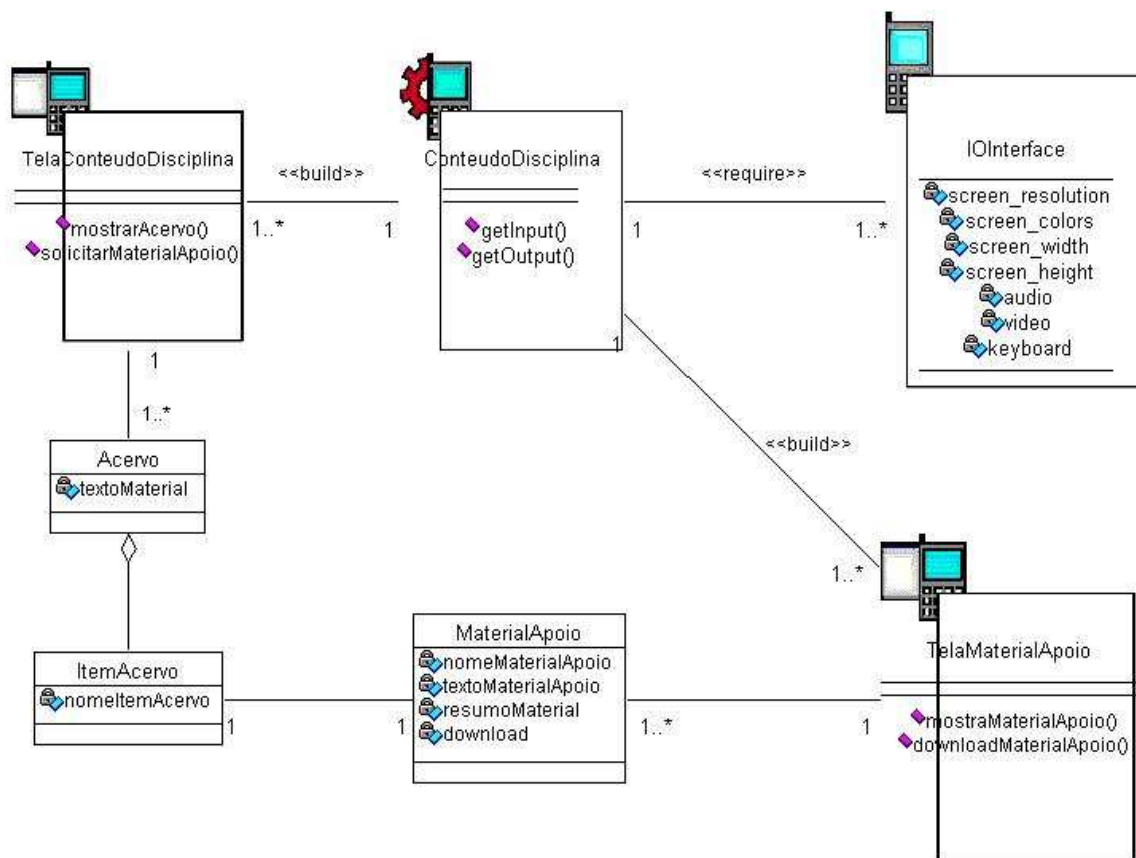


Figura 22 - Diagrama de Classes – Consultar conteúdo da disciplina

5.4. Diagrama de Seqüência

Para que se obtenha um maior entendimento sobre o comportamento interno de um sistema, é construído o modelo de interação. “Esse modelo representa as mensagens trocadas entre os objetos para a execução de cenários dos casos de uso do sistema” (BEZERRA, 2007).

Para modelar a troca de mensagens do caso de uso escolhido para a exemplificação de utilização do perfil apresentado, foi escolhido o Diagrama de Seqüência, que apresenta “as interações entre objetos na ordem temporal em que elas acontecem” (BEZERRA, 2007).

A Figura 23 ilustra o Diagrama de Seqüência do caso de uso **Consultar conteúdo da disciplina**.

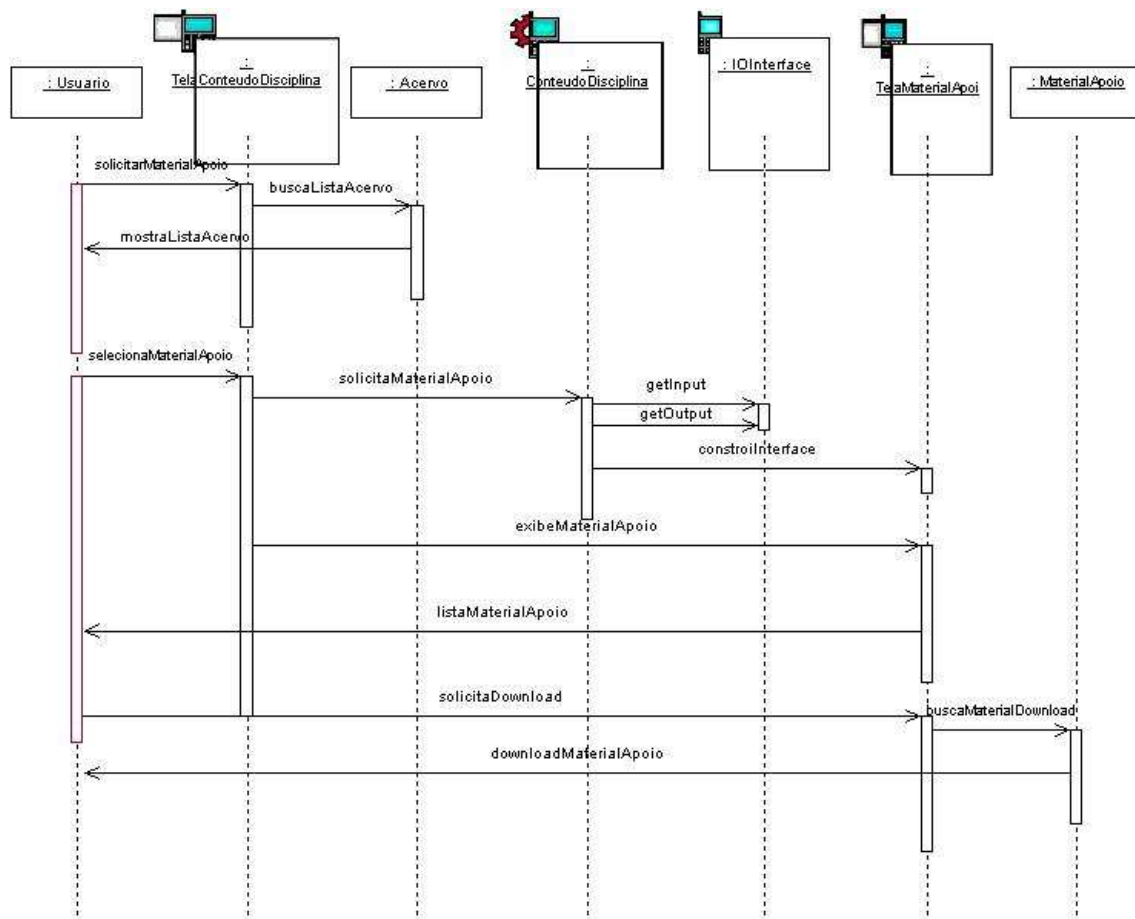


Figura 23 - Diagrama de Seqüência – Consultar conteúdo da disciplina

Capítulo 6

Conclusões e Perspectivas Futuras

Esse trabalho apresentou uma ferramenta de auxílio para a modelagem de interfaces gráficas para aplicações desenvolvidas para dispositivos móveis, através da criação de um perfil UML que modele as principais restrições e limitações em ambientes que envolvem mobilidade. O desenvolvimento deste trabalho foi constituído de três etapas principais, como dito na Seção 3.2. São elas: revisão bibliográfica, desenvolvimento do perfil UML e exemplificação de uso do mesmo.

A partir da etapa de revisão bibliográfica, foram levantados os principais assuntos e problemas envolvendo usabilidade em Computação Móvel e as soluções encontradas por diversos pesquisadores para resolver tais problemas. Constatou-se que ainda não havia um perfil UML direcionado para a modelagem de interfaces gráficas dentro da Computação Móvel. Com isso, o tema para o trabalho foi definido e estudos mais aprofundados foram feitos, constatando que algumas das extensões para aplicações WEB, descritas na Seção 2.7, poderiam ser adaptadas para o desenvolvimento móvel.

A partir das informações obtidas com a revisão bibliográfica, o perfil UML foi desenvolvido, com o intuito de identificar problemas relativos à usabilidade na fase de modelagem do sistema e de atender algumas diretrizes propostas pelo W3C, conforme dito no Capítulo 3. Dessa forma, os principais elementos de uma aplicação desenvolvida para ambientes móveis foram modelados, como o servidor, o dispositivo móvel, a interface gráfica, entre outros, atendendo às restrições impostas pela mobilidade. Neste caso, dando mais ênfase aos elementos de entrada e saída de dados presentes nos dispositivos móveis, que se tornam limitados devido ao tamanho reduzido dos mesmos.

Após a criação do perfil UML, foi feita a sua exemplificação, utilizando uma aplicação real, que está sendo desenvolvida por (MIRANDA et al., 2007). Com

isso, pode-se comprovar que o perfil se torna eficiente na modelagem de interfaces gráficas para aplicações em ambientes móveis, uma vez que as interfaces são criadas levando-se em conta as limitações impostas pelo dispositivo utilizado.

A criação de interfaces gráficas para dispositivos móveis se torna complicada uma vez que os mesmos possuem *displays* com tamanho reduzido, o que compromete a utilização de todos os recursos disponíveis. Outra limitação encontrada é o tempo para carregar a interface no dispositivo, pois o custo da conexão móvel é elevado e o tempo da bateria reduzido. Com isso, a interface gráfica deve ser leve. De acordo com o W3C, o desenvolvedor deve certificar-se se o tamanho da interface criada seja adequado às limitações de memória do dispositivo utilizado. Dessa forma, deve-se tomar cuidado para que a interface não seja tão grande que demore muito tempo para carregar totalmente nem tão pequena que o usuário precise fazer várias requisições para visualizar a interface completa. Assim, o perfil aqui apresentado resolve este problema, pois as telas são criadas de acordo com as restrições de cada dispositivo, inclusive tamanho do *display*.

6.1. Trabalhos Futuros

Como proposta para trabalhos futuros, podemos destacar:

- A implementação do sistema aqui modelado, que já está sendo feita por (Miranda et al., 2007).
- A avaliação do modelo através de outros estudos de casos.
- A implementação das extensões em ferramentas CASE (através de *plugins*, por exemplo).
- A implementação de *plugins* para ambientes de MDA (*Model Driven Architecture*) para possibilitar a geração automática de códigos adaptáveis a interfaces diversas.

Apêndice A

UML (*Unified Modeling Language*)

Conforme dito na Seção 2.5, a UML é uma linguagem utilizada para modelagem de sistemas orientados a objetos, desenvolvida na década de 90, por James Rumbaugh, Grady Booch e Ivar Jacobson. Este apêndice apresenta algumas de suas notações e alguns de seus diagramas, a fim de explicar, de maneira resumida, alguns conceitos da UML.

A.1 – Diagrama de Casos de Uso

O Diagrama de Casos de Uso, conforme dito anteriormente, apresenta os elementos externos e as funcionalidades de um sistema e as maneiras segundo as quais eles as utilizam (BEZERRA, 2007).

Os atores são os elementos UML que representam os elementos externos a um sistema. Em outras palavras, os atores representam os papéis de usuários de um sistema (FILHO, 2003). Cada funcionalidade que o sistema oferece representa um caso de uso. O Diagrama de Casos de Uso especifica o relacionamento entre os atores e os casos de uso do sistema. Suponha um sistema para uma biblioteca. Um provável ator seria o bibliotecário e alguns casos de uso possíveis seriam Registrar empréstimo e Cadastrar livro, entre outros. A Figura 24 ilustra parte do Diagrama de Casos de Uso para o exemplo citado.

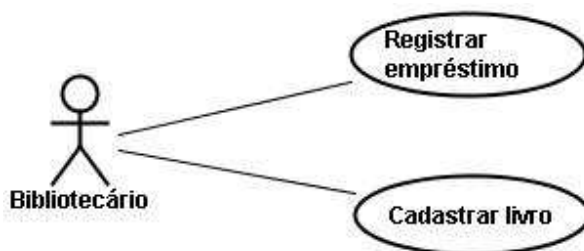


Figura 24 - Notação UML para ator

Quando existe um número elevado de atores em um sistema, o ideal é que eles sejam agrupados em atores genéricos que representem as características

comuns a todos eles. Cada ator é ligado ao ator genérico através do relacionamento de generalização, também conhecido como herança (FILHO, 2003). Para o exemplo citado anteriormente, poderia existir um administrador do sistema que, além de ter as mesmas funcionalidades (casos de uso) que um bibliotecário, fosse responsável por outras funções exclusivas de um administrador, como, por exemplo, o cadastro de novos usuários para o sistema. Esse relacionamento de herança entre o Bibliotecário e o Administrador é ilustrado na Figura 25.



Figura 25 - Herança entre atores

Os casos de uso também podem relacionar entre si. Nesse caso, a representação descreve uma funcionalidade mais complexa. Com isso, surgem os conceitos casos de uso primários, que são realizados diretamente por um ator, e casos de uso secundários, que são invocados por um outro caso de uso (FILHO, 2003). Entre esses relacionamentos estão o de extensão, que representa a invocação de um caso de uso secundário pelo primário sob determinadas condições, e o de inclusão, que representa a invocação de um caso de uso secundário pelo primário sempre que este for invocado pelo ator. A Figura 26 ilustra os relacionamentos de inclusão (*include*) e o de extensão (*extend*) para o sistema de biblioteca citado anteriormente, onde o bibliotecário, ao enviar os dados de entrada no sistema, aciona a funcionalidade de validação dos mesmos e, ao consultar determinado livro, possui a funcionalidade de alterar seus dados, se isso for necessário.

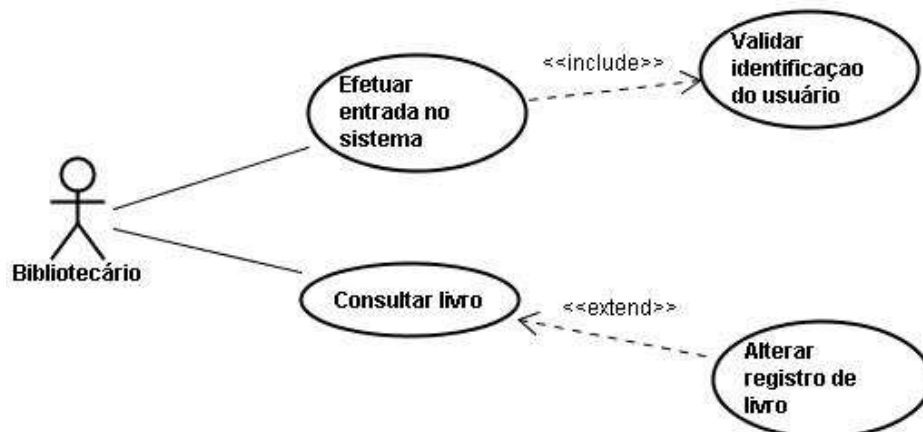


Figura 26 - Relacionamento entre casos de uso: inclusão e extensão

A.2 – Diagrama de Classes

“Classe é o descritor de um conjunto de objetos que compartilham os mesmos atributos, operações, métodos e comportamento” (PÁDUA, 2008b). O Modelo de Classes descreve o aspecto estrutural estático de um sistema, ou seja, descreve a forma como o sistema é estruturado internamente (BEZERRA, 2007). Dessa forma, o Modelo de Classes “captura a estrutura estática de um sistema ao caracterizar os objetos no sistema, o relacionamento entre eles e os atributos e as operações para cada classe dos objetos” (BLAHA & RUMBAUGH, 2006).

O Diagrama de Classes é composto pelas classes do sistema e pelo relacionamento entre elas. Cada classe é representada por uma caixa que contém seu nome, seus atributos e suas operações. A Figura 28 ilustra a notação UML para classe.



Figura 27 - Notação UML para classe

Uma classe pode se relacionar com outras através de associações, dependências, generalizações, agregações e composições. Uma associação representa a troca de mensagem entre duas classes. A dependência representa a dependência entre duas classes onde alterações na classe principal implicam em mudanças nas classes dependentes. A generalização representa a ligação entre duas classes onde existe uma classe mais genérica, que contém as informações básicas e comuns a determinado tipo de objeto, e uma classe especializada, que contém, além das características pertencentes à classe mais genérica, características especializadas. Agregação é uma forma de associação “em que um objeto agregado é composto de partes constituintes”, que, por sua vez, são partes do agregado. A composição é um tipo de agregação em que “uma parte constituinte pode pertencer a no máximo uma montagem e a exclusão de um objeto de montagem dispara a exclusão de todos os objetos constituintes por meio da composição” (BLAHA & RUMBAUGH, 2006).

A Figura 28 exemplifica o relacionamento de herança. Neste exemplo, a classe genérica é a classe Publicacao, que contém as características comuns a todos os tipos de publicações disponíveis na biblioteca. As classes especializadas Livro e Revista contêm as informações referentes a cada uma delas, além de todas as características descritas na classe Publicacao.

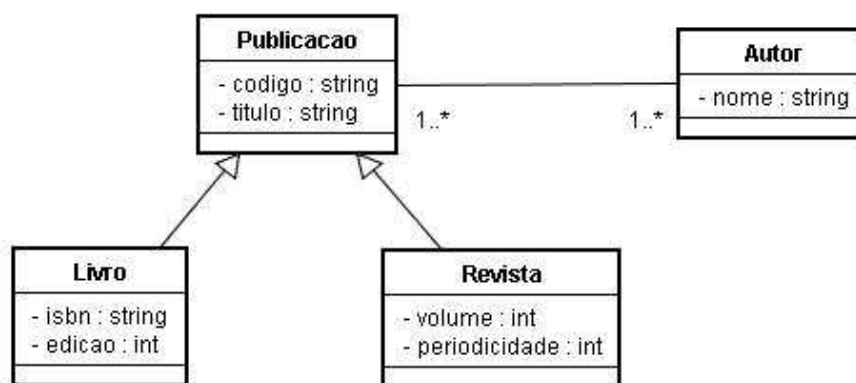


Figura 28 - Relacionamento de herança entre classes

Cada relacionamento entre classes contém sua multiplicidade. A multiplicidade representa o “número de instâncias de uma classe que podem se

relacionar a uma instância de uma classe associada” (BLAHA & RUMBAUGH, 2006). Na UML as multiplicidades possíveis são: 1 (exatamente um), 1..* (um ou mais), * (zero ou mais) e literais, como 2 (exatamente dois) ou 2..6 (dois a seis).

A.3 – Diagrama de Seqüência

O Diagrama de Seqüência é um diagrama de interação que representa a troca de mensagens entre os objetos de um sistema. “Uma mensagem enviada a um objeto invoca a execução de umas de suas operações” (BEZERRA, 2007). O Diagrama de Seqüência mostra a ordem como os eventos constituintes de um caso de uso ocorrem no transcorrer do tempo (BLAHA & RUMBAUGH, 2006).

Em um Diagrama de Seqüência, as “linhas verticais representam os objetos e as linhas horizontais representam as mensagens passadas entre eles” (FILHO, 2003). Opcionalmente, as linhas verticais podem estar associadas a uma escala de tempo. A Figura 29 apresenta o Diagrama de Seqüência para o caso de uso Cadastrar livro.

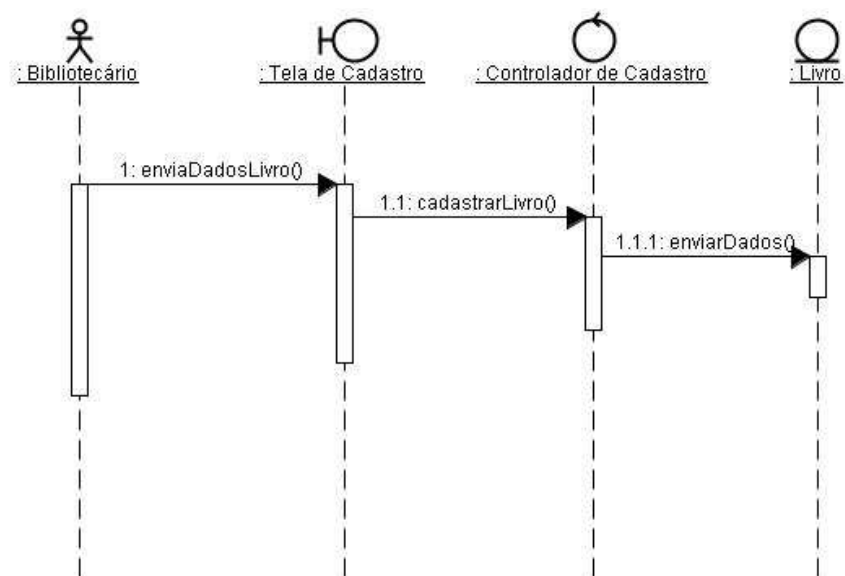


Figura 29 - Diagrama de Seqüência

Apêndice B

Extensões UML para Aplicações Web

As extensões criadas por Conallen (CONALLEN, 1999) e (CONALLEN, 2000) representam os principais elementos que compõem uma aplicação Web. Esses elementos são representados através do Perfil UML para Aplicações Web (WAE – *Web Application Extension*), que, através de estereótipos, *tagged values* e restrições, permite a modelagem de aplicações Web.

A próxima seção apresenta os estereótipos criados por (CONALLEN, 1999) e (CONALLEN, 2000). Essa apresentação corresponde à transcrição do Apêndice A do livro *Building Web Applications with UML* (CONALLEN, 2000).

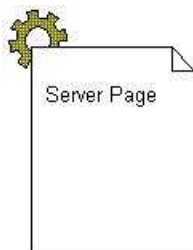
B.1 – Estereótipos

B.1.1 – *Server Page*

Classe do Meta-modelo: classe

Descrição: Uma página do servidor representa um elemento web que possui *scripts* que executam no servidor. Esses scripts interagem com os recursos do servidor, como banco de dados, regras de negócio e sistemas externos. As operações do objeto representam as funções no script e seus atributos representam as variáveis que são visíveis no escopo da página (acessíveis por todas as funções na página).

Ícone:



Restrições: Páginas do servidor se relacionam apenas com objetos no servidor.

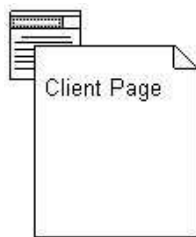
Tagged values: Scripting engine: toda linguagem que é usada para executar ou interpretar essa página (*JavaScript, VBScript* etc.).

B.1.2 – *Client Page*

Classe do Meta-modelo: classe

Descrição: Uma instância de uma página cliente é uma página Web no formato HTML e é uma junção de dados, apresentação e, também, lógica. Páginas clientes são interpretadas por *browsers* e podem conter scripts que são interpretados por esse tipo de programa. Funções da página cliente mapeiam funções em *tags* de *script* da página. Atributos da página cliente mapeiam variáveis declaradas nas *tags* do *script* da página e são acessíveis por qualquer função da página (escopo de página). Páginas cliente podem se relacionar com outras páginas clientes ou páginas do servidor.

Ícone:



Não possui restrições.

Tagged values: TitleTag: o título da página como é apresentado pelo *browser*.

BaseTag: a base URL para “derreferenciar” URLs relativas.

BodyTag: o conjunto de atributos para a *tag* `<body>`, que configuram o plano de fundo e os atributos padrões de texto.

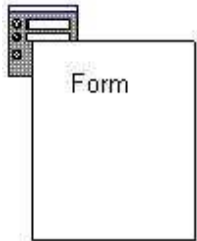
B.1.3 – *Form*

Classe do Meta-modelo: classe

Descrição: Uma classe que contém o estereótipo `<<form>>` é uma coleção de campos de entrada de dados que fazem parte de uma página cliente. Uma classe formulário mapeia diretamente os campos de um formulário HTML (*input boxes*,

text areas, radio buttons, check boxes e hidden fields). Um formulário não possui operações além das que estão encapsuladas em um formulário. Alguma operação que interage com o formulário é uma propriedade da página que o contém.

Ícone:



Não possui restrições.

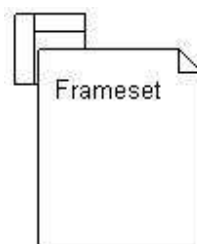
Tagged values: o método, *post* ou *get*, usado para a submissão de dados.

B.1.4 – *Frameset*

Classe do Meta-modelo: classe

Descrição: Um *frameset* é um contêiner de múltiplas páginas Web. A área de exibição retangular é dividida em *frames* retangulares menores. Cada *frame* pode ser associado com um único `<<target>>`, apesar de não necessariamente. O conteúdo de um *frame* pode ser uma página Web ou outro *frameset*. Uma classe com o estereótipo *frameset* mapeia diretamente para um *frameset* de uma página Web e a tag HTML *frame*. Pelo fato de um *frameset* ser uma `<<clientpage>>`, ele também possui operações e atributos, mas essas propriedades são ativadas apenas pelos *browsers* que não interpretam *frames*.

Ícone:



Não contém restrições.

Tagged values: Rows: o valor dos atributos linha da tag HTML `<<frameset>>`. Ele é uma string com vírgulas delimitando a altura das linhas.

Cols: o valor do atributo coluna da tag HTML <<frameset>>. Ele é uma string com vírgulas delimitando a largura das colunas.

B.1.5 – *Target*

Classe do Meta-modelo: classe

Descrição: Um *target* é um compartimento nomeado em uma janela de *browser* na qual páginas Web são interpretadas. O nome da classe que contém esse estereótipo é o nome do *target*. Tipicamente, um *target* é um *frame* em uma janela definido por um *frameset*, porém, um *target* pode ser uma nova instância do *browser* ou janela. Associações <<Targeted link>> especificam *targets* como um lugar onde a nova página Web será interpretada.

Ícone:



Restrições: um nome de um *target* deve ser único para cada cliente do sistema. Portanto, apenas uma instância de um *target* pode existir no mesmo cliente.

Não possui *tagged values*.

B.1.6 – *JavaScript Object*

Classe do Meta-modelo: classe

Descrição: Em um *browser* que suporta *JavaScript* é possível simular objetos definidos pelo usuário com funções *JavaScript*. Instâncias <<JavaScript Object>> existem apenas no contexto de páginas clientes.

Não possui ícone

Não possui restrições.

Não possui *tagged values*.

B.1.7 – *ClientScript Object*

Classe do Meta-modelo: classe

Descrição: Um objeto *ClientScript* é uma coleção separada de *scripts* do lado cliente que existe em um arquivo separado e são incluídos por uma requisição separada na parte do *browser* cliente. Esses objetos são, geralmente, oferecidos por um conjunto de funções do usuário através de uma aplicação ou uma empresa. Alguns *browsers* são capazes de esconder esses arquivos, reduzindo a taxa de tempo de *download* para uma aplicação.

Ícone:



Não contém restrições.

Não contém *tagged values*.

B.1.8 – Link

Classe do Meta-modelo: associação

Descrição: Um *link* é um ponteiro de uma página cliente para outra <<Page>>. Em um diagrama de classes, um *link* é uma associação entre uma <<client page>> e qualquer outra <<client page>> ou <<server page>>. Uma associação *link* mapeia diretamente a *anchor tag* da HTML.

Não possui ícone.

Não possui restrição.

Tagged value: Parameters: uma lista de nomes de parâmetros que deve ser passada junto com a requisição de uma página “linkada”.

B.1.9 – *Targeted Link*

Classe do Meta-modelo: associação

Descrição: Similar à associação *link*, um <<targeted link>> é um *link* cuja página associada é interpretada por outro *target*. Essa associação mapeia diretamente a *anchor tag* da HTML, com o *target* especificado pela *tag target attribute*.

Não possui ícone.

Não possui restrições.

Tagged values: Parameters: uma lista de nomes de parâmetros que deve ser passada junto com requisição para a página “linkada”.

Target Name: o nome do <<target>> que a página desse *link* aponta e deve ser interpretada.

B.1.10 – *Frame Content*

Classe do Meta-modelo: associação

Descrição: Uma associação *frame content* é uma associação de agregação que expressa o conteúdo de um frame de outra página ou *target*. Uma associação *frame content* pode, além disso, apontar para outro *frameset*, indicando *frames* aninhados.

Não possui ícone.

Não possui restrições.

Tagged values: Row: um inteiro indicando a linha específica do *frame* no *frameset* à qual a página está associada ou *target* pertencente a ela.

Col: um inteiro indicando a coluna específica do *frame* no *frameset* à qual a página está associada ou *target* pertencente a ela.

B.1.11 – *Submit*

Classe do Meta-modelo: associação

Descrição: Uma associação <<submit>> é sempre entre um <<form>> e uma <<server page>>. Formulários enviam seus valores dos campos para um servidor através de <<server pages>> para processamento. O servidor Web processa a

<<server page>> que aceita e usa as informações submetidas pelo formulário. Esse relacionamento indica qual página (ou páginas) pode processar o formulário e quais formulários a <<server page>> têm conhecimento.

Não possui ícone.

Não possui restrições.

Tagged value: Parameters: uma lista de nomes de parâmetros que devem ser passadas junto com a requisição da página “linkada”.

B.1.12 – *Builds*

Classe do Meta-modelo: associação

Descrição: O relacionamento <<builds>> é um relacionamento especial que mostra a diferença entre páginas cliente e páginas do servidor. Páginas do servidor existem apenas no servidor. Elas são usadas para criar páginas cliente. A associação <<builds>> identifica qual página do servidor é responsável pela criação da página cliente. Esse é um relacionamento direcional, uma vez que a página cliente não contém conhecimento de como isso acontece. Uma página do servidor pode criar várias páginas cliente, mas uma página cliente só pode ser criada por uma única página do servidor.

Não contém ícone.

Não contém restrição.

Não contém *tagged values*.

B.1.13 – *Redirect*

Classe do Meta-modelo: associação

Descrição: Um relacionamento <<redirect>>, uma associação unidirecional com outra página Web, pode ser dirigida tanto de páginas cliente e páginas do servidor. Se o relacionamento for originado de uma <<server page>>, o processamento da página requerida pode continuar em outra página. Isso não indica que a página de destino sempre participa da construção da página cliente. Esse relacionamento

particular não é completamente estrutural, uma vez que a invocação da operação de redirecionamento precisa ser feita programaticamente no código da página de origem. Se o relacionamento for originado de uma <<client page>>, isso indica que a página de destino será automaticamente requisitada pelo *browser*, sem entrada do usuário. Um valor para o tempo de atraso deve ser configurado para especificar um atraso, em segundos, antes de a segunda página ser requisitada. Esse uso de redirecionamento corresponde à *tag* <<META>> e HTTP-EQUIV valor de atualização.

Não contém ícone.

Não contém restrição.

Tagged value: Delay: o tempo que uma página cliente espera antes do redirecionamento da próxima página. Esse valor corresponde ao atributo *Content* da *tag* <<META>.

B.1.14 – IIOP

Classe do Meta-modelo: associação

Descrição: IIOP (*Internet Inter-ORB Protocol*) é um tipo especial de relacionamento entre objetos no cliente e objetos no servidor. IIOP representa um outro mecanismo HTTP para comunicações cliente/servidor. Tipicamente, esse relacionamento é entre *JavaBeans* no cliente e *Enterprise JavaBeans* no servidor.

Não contém ícone.

Não contém restrições.

Não contém *tagged values*.

B.1.15 – RMI

Classe do Meta-modelo: associação

Descrição: RMI (*Remote Method Invocation*) é um mecanismo para *applets* Java e *JavaBeans* para enviar mensagens para outro *JavaBeans* em máquinas

diferentes. Tipicamente, esse relacionamento é entre *JavaBeans* ou *applets* no cliente e *Enterprise JavaBeans* no servidor.

Não contém ícone.

Não contém restrições.

Não contém *tagged values*.

B.1.16 – *Input Element*

Classe do Meta-modelo: atributo

Descrição: Um elemento *input*, um atributo de um objeto <<Form>>, mapeia diretamente a *tag* <input> em formulário HTML. Esse atributo é usado para enviar uma palavra simples ou linha de texto. Os *tagged values* associados com esse estereótipo corresponde aos atributos da *tag* <input>. Para completar os valores requisitados pela *tag* HTML, o atributo nome é usado como o nome da *tag* <input> e o valor inicial do atributo é o valor da *tag*.

Não possui ícone.

Não possui restrição.

Tagged values: Type: o tipo de entrada usado: *text*, *number*, *password*, *checkbox*, *radio*, *submit* ou *reset*.

Size: especifica o tamanho da área alocada na tela, em caracteres.

Maxlength: o número máximo de caracteres que pode ser enviado.

B.1.17 – *Selected Element*

Classe do Meta-modelo: atributo

Descrição: Um campo de entrada usado em formulários. Esse controle permite que usuários selecionem um ou mais itens de uma lista. Muitos *browsers* interpretam esse controle como um *combo* ou *list box*.

Não possui ícone.

Não possui restrições.

Tagged value: Size: especifica quantos itens podem ser mostrados ao mesmo tempo.

Multiple: um booleano que indica quantos itens podem ser selecionados simultaneamente de uma lista.

B.1.18 – Text Area Element

Classe do Meta-modelo: atributo

Descrição: um campo de entrada, usado em formulários, que permite múltiplas linhas de entrada.

Não possui ícone.

Não possui restrições.

Tagged values: Rows: o número de linhas visíveis.

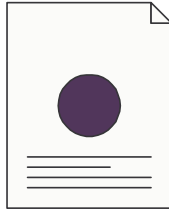
Cols: a largura visível de controle em média da largura dos caracteres.

B.1.19 – Web Page

Classe do Meta-modelo: componente

Descrição: Uma página componente é uma página Web. Ela pode ser requisitada pelo nome por um *browser*. Uma página componente pode ou não conter *scripts* cliente ou servidor. Tipicamente, páginas componente são arquivos de texto acessíveis pelo servidor Web, mas elas podem ser compiladas por módulos que são carregados e invocados pelo servidor Web. Em última instância, quando acessada por um servidor Web tanto como arquivo quanto executável, uma página produz um documento no formato HTML que é enviado em resposta à requisição do servidor.

Ícone:



Não possui restrições.

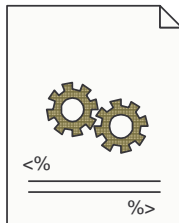
Tagged values: *Path:* o caminho requisitado para especificar a página Web no servidor Web. Esse valor pode ser relativo ao diretório raiz da aplicação (site) Web .

B.1.20 – ASP *Page*

Classe do Meta-modelo: Componente

Descrição: Páginas Web que interpretam código ASP do lado cliente. Esse estereótipo é aplicável apenas em aplicações baseadas em *Microsoft Active Server Pages*.

Ícone:



Não possui restrições.

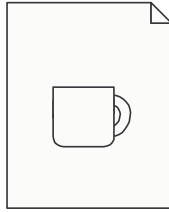
Tagged values: mesmos *tagged values* de *Web Page*.

B.1.21 – JSP *Page*

Classe do Meta-modelo: componente

Descrição: Páginas Web que implementam código JSP do lado servidor. Esse estereótipo é aplicável somente em aplicações Web desenvolvidas em ambientes que utilizam *Java Server Pages*.

Ícone:



Não possui restrições.

Tagged values: mesmos *tagged values* de *Web Page*.

B.1.22 – *Servlet*

Classe do Meta-modelo: componente

Descrição: Um componente *Java servlet*. Esse estereótipo é aplicável apenas em aplicações Web que suportam *servlets Sun*.

Ícone:



Não contém restrições.

Tagged values: mesmos *tagged values* de *Web Page*.

B.1.23 – *Script Library*

Classe do Meta-modelo: componente

Descrição: O componente que provê a biblioteca de rotinas ou funções que podem ser incluídas por outras *Web Pages*.

Ícone:



Não possui restrições.

Tagged values: mesmos tagged values de Web Page.

Referências Bibliográficas

AMADO, Paulo G. F. **Bancos de Dados Móveis: Visão Geral, Desafios e Soluções Atuais**. Projeto de Final de Curso, CIn-UFPE, 2002.

AMSTEL, Frederick V. **Afinal, o que é ícone? Como criar ícones?** Disponível em: http://usabilidoido.com.br/afinal_o_que_e_icone_como_criar_icones.html. Acesso em: 03 jan. 2007.

AMSTEL, Frederick V. **Semiótica**. Disponível em: http://usabilidoido.com.br/cat_semiotica.html. Acesso em: 10 jan. 2008.

ARAÚJO, João. **Modelando Aplicações Web com UML**. Disponível em: <http://ctp.di.fct.unl.pt/~ja/as2/w.ppt>. Acesso em: 30 nov. 2007.

BAUMEISTER, Hubert; KOCH, Nora; KOSIUCZENKO, Piotr; WIRSING, Martin. **Extending Activity Diagrams to Model Mobile Systems**. In NetObject-Days 2002 (M. Aksit, M. Mezini, R. Unland Eds.), LNCS 2591, pp. 287-293, 2003.

BERGH, Jan V.; CONINX, Karin. **Using UML 2.0 and Profiles for Modelling Context-Sensitive User Interfaces**. Workshop on Model Driven Development of Advanced User Interfaces (MDDAUI 2005), Montego Bay, Jamaica, 2005

BEZERRA, Eduardo. **Princípios de Análise e Projeto de Sistemas com UML**. Rio de Janeiro: Elsevier, 2007.

BLAHA, Michael; RUMBAUGH, James. **Modelagem e Projetos Baseados em Objetos com UML 2**. 2ª edição, Rio de Janeiro: Elsevier, 2006.

BORGES, Marcell; ZAMBALDE, André L.; SOUZA, Reginaldo F. **Estudo da Usabilidade de Diferentes Sites Vencedores do Prêmio I-BEST**. Monografia de Graduação, UFLA, 2002.

CARVALHO, José O. F.; PELISSONI, Carla G. **Avaliação da Usabilidade de Interface de Dispositivos Móveis para Aplicações de Apoio ao Ensino: Uma Experiência com Educação a Distância**. 3ª Conferência Iberoamericana em Sistemas, Cibernética e Informática, IIIS – International Institute of Informatic and Systemics, Orlando, Flórida, EE.UU., v. 1, p. 40-44, julho, 2004.

CHAN, Susy S.; ZHOU, Yanzan; FANG, Xiaowen; XU, Shuang; BRZEZINSKI, Jack; LAM, Jean. **Usability for Mobile Commerce Across Multiple Form Factors**. Journal of Electronic Commerce Research, 2002. Vol. 3, Nro. 3, p.187–199. Disponível em: <http://www.csulb.edu/web/journals/jecr/issues/20023/paper7.pdf>. Acesso em: 6 jun. 2006.

CONALLEN, Jim. **Modeling Web Applications Architectures with UML**. Communications of the ACM. Vol. 2, Nro. 10. October, 1999.

CONALLEN, Jim. **Building Web Applications With UML**. Massachusetts: Addison-Wesley, 2000.

CYBIS, Walter; BETIOL, Adriana H.; FAUST, Richard. **Ergonomia e Usabilidade: Conhecimentos, Métodos e Aplicações**. São Paulo: Novatec Editora, 2007.

ERICSSON, T.; CHINCHOLLE, D.; GOLDSTEIN, M. **Both the Cellular Phone and the Service Impact WAP Usability**. Proceedings of IHM-HCI 2001, volume II, HCI. In: VANDERDONCKT, J.; BLANFORD, A. DERYCKE, A. *Practice*. Lille, France, 10-14 September, 2001.

ERIKSSON, Hans-Erik; PENKER, Magnus; LYONS, Brian; FADO, David. **UML2 Tool Kit**. Indianapolis: Wiley Publishing, Inc., 2004.

FERRAZ, Valéria. C.; ARQUETE, Daniela. A. R.; BARTOLOMEU, Tereza. A.; PASSOS, Frederico. J. V. **O Ensino de Graduação da Universidade Federal de Viçosa Ultrapassando os Limites do Campus Universitário**. Disponível em: <<http://www.abed.org.br/congresso2005/por/pdf/125tcf3.pdf>>. Acesso em: 05 jan. 2008.

FERNANDEZ, Lúcia F.; MORENO, Antônio, V. **An Introduction to UML Profiles**. UPGRADE Vol. V, No. 2, April 2004.

FILHO, Jugurta L.; BRAGA, José L. **UML: Unified Modeling Language**. GIS, Sep. 2007.

FILHO, Wilson P. P. **Engenharia de Software, Fundamentos, Métodos e Padrões**. 2ª edição, Rio de Janeiro: LTC – Livros Técnicos e Científicos Editora, 2003.

Folha de São Paulo. **Entrevista com Jakob Nielsen**, 02/02/2007. Disponível em: <http://www1.folha.uol.com.br/folha/informatica/ult124u21504.shtml>. Acesso em 17/03/2008.

FORMAN, George H.; ZAHORJAN, John. **The Challenges of Mobile Computing**. IEEE Computer 27, 4, pp. 38-47. Apr. 1994.

GRASSI, Vincenzo; MIRANDOLA, Raffaella; SABETTA, Antonino. **A UML Profile to Model Mobile Systems**. 7th International Conference on the Unified Modeling Language. Lisbon, Portugal, LNCS 3273, Springer-Verlag, pp. 128-142. Oct. 2004.

ISO/IEC 9126-1: 2001. **Software Engineering – Product Quality – Part 1: Quality Model**. 2001.

ISO 9241-11: 1998. **Ergonomic requirements for office works with visual display terminals (VTDs) – Part 11: Guidance on Usability.** 1998.

ITO, Giani C.; SILVA, Antônio C. B.; JÚNIOR, Ney N. T. R. **Análise de um Repositório de Perfis para Reconhecimento de Dispositivos Móveis.** In XII SIMPEP - Bauru, SP, Brasil, 2005.

JONES, Matt; MARSDEN, Gary; MOHD-NASIR, Norzila; BUCHANAN, George. **Improving Web Interaction on Small Displays.** Computer Networks 31, pp. 1129-1137, 1999.

KIM, Hoyoung; KIM, Jinwoo; LEE, Yeonsoo; CHAE, Minhee; CHOI, Youngwan. **An Empirical Study of the Use Contexts and Usability Problems in Mobile Internet.** Proceedings of the 35th Hawaii International Conference on System Sciences, 2002.

KOCH, Nora, BAUMEISTER, Hubert, HENNICKER, Rolf, MANDEL, Luis. **Extending UML to Model Navigation and Presentation in Web Applications.** Disponível em: <http://www.pst.informatik.uni-muenchen.de/personen/kochn/ExtendingUML.pdf>. Acesso em 30 out. 2007.

KOSCIANSKI, André; SOARES, Michel, S. **Qualidade de Software.** São Paulo: Novatec Editora, 2006.

KOSIUCZENKO, Piotr. **Sequence Diagrams for Mobility.** In Stefano Spaccapietra, editor, 21 International Conference on Conceptual Modeling (ER2002). Springer-Verlag, October 2002.

LADEIRA, Paulo H.; ROCHA, Mauro N. **M-PVANET – Desenvolvimento de uma Ferramenta de Auxílio ao Ensino na UFV para Ambientes Móveis.** Projeto de Final de Curso, UFV, 2007.

LOUREIRO, Antônio A. F.; SADOK, Djamel F. H.; MATEUS, Geraldo R.; NOGUEIRA, José M. S.; KELNER, Judith. **Comunicação Sem Fio e Computação Móvel: Tecnologias, Desafios e Oportunidades.** Trabalho apresentado na XXII Jornada de Atualização em Informática, Anais do XXIII Congresso Brasileiro da Sociedade Brasileira de Computação, pp. 125-244, Campinas, SP, 2003.

MARTINEZ, Maria. L. **Usabilidade no Design Gráfico de Web Sites.** In: GRAPHICA'2000: III International Conference on Graphics Engineering for Arts and Design / 14 Simpósio Nacional de Geometria Descritiva e Desenho Técnico, 2000, Ouro Preto - MG. Anais em CD-ROM, 2000.

MATEUS, Geraldo. R.; LOUREIRO, Antônio. A. F. **Introdução à Computação Móvel,** 2004.

MAYHEW, Deborah J. **The Usability Engineering Life Cycle: A practitioner's handbook for user interface design.** Ed. Morgan Kaufmann, 1999.

McCALL, Jim A.; RICHARDS, Paul K.; WALTERS, Gene F. **Factors in Software Quality.** NTIS AD—A049-014, 015, 055, nov. 1977.

MEDEIROS, Marco A.; CYBIS, Walter A.; BARCIA, Ricardo M. **ISO 9241: Uma Proposta de Utilização da Norma do Grau de Satisfação de Usuários de Software.** Dissertação de mestrado, Programa de Pós-Graduação em Engenharia de Produção, UFSC, 1999.

MIRANDA, Nidyana; MENDES, Thiago; ROCHA, Mauro N. **mPVANet – Desenvolvimento de uma Ferramenta de Auxílio ao Ensino da UFV para Ambientes Móveis,** Projeto de Dissertação, UFV, 2007.

MORAN, T. **The Command Language Grammars: a representation for the user interface of interactive computer systems.** International Journal of Man-Machine Studies, 15, 3-50, 1981.

NETO, Siegfried K.; GRAHL, Everaldo A. **Modelando Aplicações WEB com UML.** Disponível em: <http://www.inf.furb.br/~egrahl/disciplinas/qualidade/material/POSWEB2/UMLWEB.pdf>. Acesso em: 16 jun. 2007.

NIELSEN, Jacob. **Ten Usability Heuristics.** Disponível em: http://www.useit.com/papers/heuristic/heuristic_list.html. Acesso em: 19 mar. 2008.

NIELSEN, Jacob. **Usability 101: Introduction to Usability.** Jakob Nielsen's Alertbox.. Disponível em: <http://www.useit.com/alertbox/20030825.html>. Aug. 2003.

NIELSEN, Jacob. **Usability Engineering.** New York: Academic Press, 1993.

NIELSEN, Jacob; MOLICH, Rolf. **Heuristic Evaluation of User Interfaces.** Conference on Human Factors in Computing Systems, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Empowering People, 1990.

NOGUEIRA, Hugo M.; LADEIRA, Paulo H.; ROCHA, Mauro N. **The Challenge of Displaying WEB Mobile Pages,** accepted for publication in the IADIS International Conference Information Systems 2008, Algarve, Portugal, April 2008.

PÁDUA, Clarindo I. P. S. **UML: Introdução.** Disponível em: <http://homepages.dcc.ufmg.br/~clarindo/arquivos/disciplinas/uml-mpn/material/transparencias/1-uml-introducao.pdf>. Acesso em: 27 ago. 2007.

PÁDUA, Clarindo I. P. S. **Introdução Engenharia de Usabilidade.** Disponível em: <http://homepages.dcc.ufmg.br/~clarindo/arquivos/disciplinas/eu/material/transparencias/topicos/1-introducao.pdf>. Acesso em: 17 mar. 2008.

PÁDUA, Clarindo I. P. S. **Especificação de Requisitos de Usabilidade**. Disponível em:

<http://homepages.dcc.ufmg.br/~clarindo/arquivos/disciplinas/eu/material/transparencias/topicos/5-requisitos-usabilidade.pdf>. Acesso em: 19 mar. 2008 (a).

PÁDUA, Clarindo I. P. S. **UML: Classe e Relacionamento**. Disponível em: <http://homepages.dcc.ufmg.br/~clarindo/arquivos/disciplinas/uml-mpn/material/transparencias/4-uml-classe-relacionamento.pdf>. Acesso em: 15 mai. 2008 (b).

PINHEIRO, Erick. J. F.; MOURA, Hermano P.; GOMES, Alex S. **Análise de Usabilidade do PMBOK EASY**. Disponível em: <http://www.cin.ufpe.br/~tg/2003-1/ejfp.pdf>. Acesso em: 15 abr. 2006.

PRESSMAN, Roger. S. **Engenharia de Software**. 6ª edição, São Paulo: McGraw-Hill, 2006.

RABIN, Jô; McCATHIENEVILE, Charles. **Mobile Web Best Practices 1.0, Basic Guidelines**. W3C Proposed Recommendation, 2 november 2006. Disponível em: <http://www.w3.org/TR/2006/PR-mobile-bp-20061102/>. Acesso em: 14 jan. 2008.

Basic Guidelines W3C Proposed Recommendation. Disponível em: <http://www.w3.org/TR/2006/PR-mobile-bp-20061102/>. Acesso em: 14 jan. 2008.

SAHA, Goutam K. **Fault Management in Mobile Computing**. Ubiquity, v.4 n.32, p.1-1, October 8 - October 14, 2003.

SHNEIDERMAN, Ben. **Universal Usability**. Communications of the ACM. Vol. 43, Nro. 5. May, 2000.

SILVA, Alberto. R. **Abordagem XIS ao Desenvolvimento de Sistemas de Informação**. In: Conferência da Associação Portuguesa de Sistemas de Informação, 4, Porto. Anais. Porto: Universidade Portucalense, 2003.

SOMMERVILLE, Ian. **Engenharia de Software**. 6ª ed., São Paulo: Pearson Addison wesley, 2003.

SOUZA, Clarice. S., LEITE, Jair. C., PRATES, Raquel. O., BARBOSA, Simone. D. J., **Projeto de Interfaces de Usuário – Perspectivas Cognitivas e Semióticas**. Disponível em: http://www.dimap.ufrj.br/~jair/piu/JAI_Apostila.pdf. Acesso em: 28 mai. 2005.

STEINMACHER, Igor F. **Estudo de Princípios para Modelagem Orientada a Aspectos**. Maringá: Centro de Tecnologia, Departamento de Informática, Universidade Estadual de Maringá, 2003.

TERRENGHI, Lucia; KRONEN, Marcus; VALLE, Carla. **Usability Requirements for Mobile Service Scenarios**. Proceeding of HCI International Conference, Las Vegas, USA, 1-10, 2005.

UMAR, Amjad. **Mobile Computing and Wireless Communications**. New Jersey: NGE Solutions, 2004.

ZAINA, Luciana A. M.; PAULA, Marcus V. S. O.; SILVEIRA, Regina M.; CINELLI, Gustavo B.; ALMEIDA, Sérgio R.; BRESSAN, Graça; CARVALHO, Teresa C. M. B.; RUGGIERO, Wilson V., **Implementação de Interfaces para Sistemas de Ensino a Distância Através da Web**. In: InterTech 2002, International Conference in Learning Technology, Santos, 2002.

WIRSING, Martin; MÜCHEN, LMU; BAUMEISTER, Hubert; KOSIUCZENKO, Piotr; KOCH, Nora; STEPHAN, Merz; ZAPPE, Julia. **UML for Global Computing**. In Global Computing, Programming Environments, Languages, Security and Analysis of Systems, LNCS 2874, Springer Verlag, 2003.