UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação
Departamento de Computação e Automação

MÉTRICAS OBJETIVAS PARA CONSISTÊNCIA DE INTERFACE:

explorando a relação entre

Interação Humano-Computador e Engenharia de Software

Luciano Krob Meneghetti

Orientadora:

Profa. Dra. Beatriz Mascia Daltrini

UNICAMP
BIBLIOTECA CENTRAL
SEÇÃO CIRCULANTE

Julho de 1998

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, como parte dos requisitos exigidos para obtenção do título de Mestre em Engenharia Elétrica.

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

M525m

Meneghetti, Luciano Krob
Métricas objetivas para consistência de interface: explorando a relação entre interação humano-computador e engenharia de software / Luciano Krob Meneghetti.--Campinas, SP: [s.n.], 1998.

Orientadora: Beatriz Mascia Daltrini
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Engenharia de software. 2. Interação homem-máquina. 3. Engenharia de software auxiliado por computador. 4. Interfaces de usuário (Sistema de computador). I. Daltrini, Beatriz Mascia. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

AGRADECIMENTOS

À Bia, minha orientadora, por todo o apoio e paciência, recebido durante esses anos da dissertação. Sobretudo no final é que pude perceber sua sabedoria ao "estender a mão" e "puxar a orelha" por vezes, alternadamente... Obrigado por sua amizade!!!

À "Nóis é Legal", Cherry, Marcito, Phrosa, Vitão e Cesão (nosso morador virtual), pelo amor, carinho e respeito que desenvolvemos uns pelos outros. Obrigado por essa experiência tão importante!!! Uhuuuuu! Acabou galera!

À minha mãe, Rosa, por todo o seu apoio incondicional e "puxadas de orelha", com suas frases célebres: "Ai, meu filho, quando é que voce vai acabar essa dissertação? Assim não é possível!!!" Acabei, mamãe!!! :-))))

Ao meu pai e esposa, Mercio e Inês, por me acompanharem durante toda essa minha caminhada. Obrigado pela torcida... :-)

Aos meus irmãos, Marcelo e Fernando, e irmã, Cristina, pela torcida e por nunca terem aberto a boca para me pressionar...(valeu!!!) :-)

À família Paiva, ao Zé Maria e especialmente ao Daniel, que me apresentou para a Unicamp, além de ter sido e continuar sendo um grande estímulo pela maneira que leva a frente sua pesquisa (agora de doutorado...)!

À galera do LCA, por todos esses anos de convivência, pelas festinhas de aniversário, pelo momentos tão gostosos, pelos papos legais, pelo "crescimento" em conjunto!

À Viviane, do LCA, por toda sua simpatia e responsabilidade, grande responsável por essa dissertação: sem ela não haveria infra-estrutura suficiente para concluir esse trabalho... Muito obrigado!

Ao pessoal do CTI, por todas as "contribuições" com seus "pitacos" e estímulos. Muito obrigado por todos os momentos divertidos e pelas oportunidades de aprendizado!!!

Aos professores da UNIMEP, por terem me passado o conhecimento suficiente para enfrentar essa etapa. Obrigado por acreditarem em mim... e pelas cartas de referência! Ufa!

Ao Marcelo, segurança da Elétrica, que muitas vezes me ajudou a ficar acordado pelas madrugadas nesses últimos meses... obrigado pelas conversas e risadas!!!

À Capes, pelo apoio financeiro.

À todas as outras pessoas, amigos e amigas (que graças a Deus são muitos), que não foram citadas isoladamente, mas sabem que merecem um crédito por terem participado de alguma maneira dessa etapa tão importante da minha vida!

RESUMO

Avaliar a interface de um software, antes de entregá-lo ao usuário final, é muito importante. O uso de ferramentas CASE construtoras de interface para auxílio ao *designer* de interface não é condição suficiente para garantir uma interface consistente. Essas ferramentas, no máximo, conseguem garantir a disponibilização de *widgets* através de uma *toolkit* pré-definida. Em outras palavras, incorporam somente conceitos da Engenharia de Software (ES), ignorando conceitos de Interação Humano-Computador (IHC). Como esses conceitos de IHC não são incorporados na fase de desenvolvimento da interface, através de ferramentas CASE construtoras de interface, essa dissertação pretende disponibilizá-los para que sejam incorporados em ferramentas CASE para avaliação de interface, oferecendo uma base de métricas. Mais especificamente, o objetivo dessa dissertação é explorar a relação entre as grandes áreas de ES e IHC, através da geração de uma base de métricas objetivas, cuja meta é favorecer (não necessariamente garantir) a usabilidade do produto final que chegará às mãos do usuário.

Palavras-Chave: Engenharia de Software, Interação Humano-Computador, ferramentas CASE, interface, métricas.

ABSTRACT

Evaluating the software interface before delivering it to the end user is very important. Using interface builders CASE tools for helping interface designer is not a sufficient condition to guarantee a consistent interface. These tools, at most, guarantee widgets available through a pre-defined toolkit. In other words, it incorporates only software engineering (SE) concepts, ignoring human-computer interaction (HCI) concepts. As the HCI concepts are not incorporated in interface development phase in interface builders CASE tools, this thesis intends to let them available, so that they can be incorporated in interface evaluation CASE tools, offering a metrics repository. More specifically, the thesis goal is to exploit the relation between SE and HCI areas, by the generation of a repository of objective metrics. Its intention is to favour (not necessarily guarantee) the final product usability that will be given to the end user.

Keywords: Software Engineering, Human-Computer Interaction, CASE tools, interface, metrics.

ÍNDICE

LISTA DE TABELAS

LISTA DE FIGURAS

1. INTRODUÇÃO

1.1. OBJETIVO

O objetivo dessa dissertação de mestrado é explorar a relação entre as grandes áreas de Engenharia de Software (ES) e Interação Humano-Computador (HCI, acrônimo de *Human-Computer Interaction*), introduzindo os conceitos de HCI, mostrando sua importância frente ao *design* de interação visual. Essa relação pretende ser explorada através da geração de uma base de métricas objetivas, baseadas nos conceitos que serão passados, que possam consistir a interface gerada por uma ferramenta *CASE* construtora de interface.

Com isso, pretende-se mostrar que a utilização de uma ferramenta dessas não é condição suficiente para o sucesso de uma interface e consequentemente do produto de software do qual ela faz parte. É necessário que diversos conceitos relacionados à HCI, como as métricas levantadas nessa dissertação, estejam "empacotados" e disponíveis aos *designers* de interface através de, por exemplo, ferramentas *CASE* voltadas para avaliação de interface. A existência de uma ferramenta como essa minimizaria a quantidade de recursos (custo, pessoal e tempo) alocados à tarefa de avaliação de interface, prevista em qualquer ciclo de desenvolvimento de produto de software, além de favorecer (não necessariamente garantir) a usabilidade do produto final que chegará às mãos do usuário.

1.2. COMPUTAÇÃO VERSUS COMUNICAÇÃO

O computador é o primeiro invento humano sem função fixa – seu impacto depende do que se quer dele [VEJA95]. Se alguém for escrever a história da tecnologia de hoje, não poderá fazê-lo sem medir o impacto do computador, em comparação ao telefone, carro, televisão, ou ao próprio rádio. No início desse século, em 1911, o rádio tornou possível ao cidadão comunicar-se através de grandes distâncias, criando uma nova época de intercâmbio de informação e transmissões de mensagens. Essa afirmação é perfeitamente viável nos dias de hoje, trocando-se a palavra rádio por computador, ou ainda, por internet...

Uma vez que o micro migrou para dentro das casa das pessoas, percebe-se que o nome dado à máquina talvez esteja errado: **a prioridade não é mais computar mas comunicar!** Quem está em casa não quer computar. Quer ter acesso a conteúdo, serviços, informações.

E a interface é a parte do software responsável pela comunicação! É pela interface que o usuário efetuará sua interação com o software. Dedicar tempo e dinheiro à interface com o usuário era considerado um frívolo desperdício [NEGRO95]. Hoje, a visão é diferente. O desafio da proxima década é fazer computadores que conheçam o usuário, aprendam quais são suas necessidades e entendam linguagens verbais e não verbais, fazendo com que o *design* de interface desapareça, isto é, torne-se imperceptível ao usuário.

Nos dias de hoje, a comunicação eletrônica é pior que a comunicação humano-humano porque ela ignora questões não verbais presentes na comunicação humana (e.g., linguagem do corpo, gestos) e enriquecimentos da comunicação (e.g., entonação, estresse de algumas palavras, sílabas) [ANDER88], ou seja, **a comunicação via computador tem suas limitações**.

A melhor interface seria aquela que dispusesse de canais diversos e concorrentes de comunicação, mediante os quais o usuário pudesse expressar sua intenção a partir de uma série de aparatos sensoriais diferentes... A idéia é simples: o falar, o apontar e o olhar devem trabalhar juntos, como parte de uma interface multimodo que tem menos a ver com o envio e o recebimento de mensagens e mais com o diálogo cara a cara, de ser humano para ser humano [NEGRO95].

Estudos acadêmicos sugerem que, no caso da maioria dos aplicativos, a fala e a linguagem natural não constituem canais apropriados de comunicação entre as pessoas e os computadores. Tais relatórios técnicos estão repletos de tabelas e grupos de controle, provando que a linguagem natural é confusa quando se trata da comunicação homem-computador.

As futuras interfaces poderão ser baseadas em delegação de tarefas, e não no vernáculo da manipulação direta e do comando por *mouse*. A facilidade de uso tem constituído uma meta tão obrigatória que, às vezes, os *designers* se esquecem de que muitas pessoas simplesmente não querem usar a máquina, querem que ela desempenhe uma tarefa. O que se chama de "interfaces baseadas em agentes" poderá emergir como a maneira predominante de computadores e pessoas se comunicarem.

Esses comentários são fundamentais para se ter um rumo à frente, mas são extremamente avançados para o que existe de tecnologia disponível hoje (com baixo custo para o usuário). Ainda se engatinha em diversos pontos que poderiam levar à realização dessa forma de interação tão sonhada. Essa dissertação propõe-se a trabalhar a interação com enfoque na comunicação cujo meio seja visual, partindo da premissa de que a experiência visual isolada, com um bom *design*, possa produzir efeitos tão satisfatórios como quando acrescidas de multimodo.

1.3. COMPREENSÃO: EM BUSCA DA INFORMAÇÃO

Wurman [WURMA91] trabalha o conceito de arquitetura de informação e muito dessa dissertação tem o objetivo de usar suas idéias, tentando aplicá-las ao *design* da interação visual. Existem estatísticas estarrecedoras a respeito da quantidade de dados com a qual as pessoas lidam no dia-a-dia. Essa quantidade contribui amplamente para a diferença existente entre dados e conhecimento, que é denominada pelo autor como "ansiedade de informação".

Informação é significado, redução de incerteza! O momento que se vive atualmente é uma explosão de não-informação (dados sem significado) e o aumento da tecnologia é um grande culpado disso! Com dados chegando de todos os lados, existem três grandes ramos de negócios relacionados à tecnologia da difusão de informação: transmissão (televisão, rádio, fax, fone, etc), armazenamento (banco de dados, etc) e compreensão.

Dos três ramos, o ramo da compreensão é o que interessa para essa dissertação, pois é a ponte entre os dados e a informação. Compreender a estrutura e a organização da informação permite extrair dela algum valor e significado.

Numa linguagem as palavras podem significar uma coisa para uma pessoa e algo bem diferente para outra. Não existe uma forma correta de se comunicar e, pelo menos num sentido absoluto, é impossível partilhar os pensamentos com outras pessoas, pois jamais são compreendidos de forma exatamente igual. O significado completo de uma palavra só aparece quando ela é colocada em seu contexto, que pode ter uma função extremamente sutil como nos trocadilhos e no uso do duplo sentido. E mesmo assim o significado irá depender do ouvinte, do orador, de toda a experiência deles com a linguagem, de seu conhecimento um do outro e da situação. As palavras não significam coisas numa relação de um para um, como num código. Se a comunicação é algo tão fundamental, então todo esforço feito no sentido de aperfeiçoar o conhecimento da língua e seu uso melhorará a capacidade de compreender e ministrar informação.

Wurman declara: "Comunicar é lembrar como era quando não se sabia". Quando alguém passa a conhecer alguma coisa, perde a capacidade de se identificar com aqueles que ainda não conhecem. Uma estimativa interessante: as pessoas lembram 90% do que fazem, 75% do que dizem e 10% do que ouvem. Por essa estimativa fica óbvio que a melhor maneira das pessoas aprenderem a trabalhar com algum tipo de software é interagindo com ele. E mais uma vez o papel da interface é destacado...

A mente está em constante formação e mudança, e as idéias com as quais se tem contato podem redefinir completamente o modo de pensar. A interface, no final, precisa ser encarada metaforicamente como um mapa! Os mapas são como palavras e, se não tiverem um significado universal, perdem seu poder de comunicação. O usuário não pode ficar "perdido". A metáfora precisa ser suficiente para que o usuário ache o caminho através da informação. Com esse objetivo, um bom trabalho de comunicação visual deverá sempre possuir clareza no desenho e complexidade na informação.

Cabe ao *designer* aprender uma linguagem visual efetiva o suficiente para comunicar ao usuário final o que é necessário, sempre sob o ponto de vista do próprio usuário. E cabe a esse, insistir para que a comunicação seja compreensível.

1.4. PROPOSTA DA DISSERTAÇÃO

A avaliação da interface é fundamental! O uso de ferramentas *CASE* construtoras de interface para auxílio ao *designer* de interface não é condição suficiente para garantir uma interface consistente. Essas ferramentas, no máximo, conseguem garantir a disponibilização de *widgets* através de uma *toolkit* pré-definida. Em outras palavras, incorporam somente conceitos da Engenharia de Software, ignorando conceitos de Interação Humano-Computador. Como esses conceitos de HCI não são incorporados na fase de desenvolvimento da interface, através de ferramentas *CASE* construtoras de interface, essa dissertação pretende disponibilizá-los para que sejam incorporados em ferramentas *CASE* para avaliação de interface, oferecendo uma base de métricas. Mais especificamente, o objetivo dessa dissertação é explorar a relação entre as grandes áreas de ES e HCI, através da geração de uma base de métricas objetivas, cuja meta é favorecer (não necessariamente garantir) a usabilidade do produto final que chegará às mãos do usuário.

Partindo dos conceitos passados nesse capítulo e aprofundados no capítulo seguinte de ES e HCI, pretende-se mostrar que as ferramentas *CASE* voltadas para o *design* da interface não são condição suficiente para o sucesso da interface gerada.

Avaliação de Interface é fundamental e, dentre as diversas formas de se avaliar interface, existe uma grande abertura para o estudo de avaliações objetivas, que levem em conta a coleta automática de dados, através de métricas relacionadas a fatores não subjetivos da interface. Esse tipo de avaliação permite que o *designer* interaja somente no início da avaliação, com a entrada de parâmetros para a interface, e no seu final, na hora de verificar os dados coletados. Poupar recursos (humanos, tempo e dinheiro) é uma das maiores vantagens desse tipo de avaliação.

Ao desenvolver essa dissertação, procurou-se trabalhar os conceitos em uma plataforma difundida como o *Windows95*, desenvolvida pela *Microsoft*, que disponibiliza comercialmente sua *toolkit* de objetos, através de ferramentas construtoras de interface bastante conhecidas como por exemplo o Visual Basic (VB). A geração das diversas métricas desse trabalho está baseada na forma como o VB disponibiliza os objetos ao designer.

Se trabalhar em cima de um ambiente como o *Windows95* é trabalhar em função das regras que o mercado mundial de PCs (*personal computers*) dita, então descobrir que as ferramentas que oferecem esse objetos comercialmente são incapazes de consistir aspectos básicos da interface é uma descoberta muito importante, pois oferece uma ampla área de pesquisa a ser trabalhada.

Uma resposta possível: os engenheiros de software que construiram essas ferramentas só se preocuparam em "empacotar" conceitos da Engenharia de Software, esquecendo conceitos tão importantes como os de HCI. Ao empacotar os conceitos de HCI, muitos dos problemas encontrados com as métricas seriam eliminados em fases preliminares do desenvolvimento da interface, minimizando custos operacionais de desenvolvimento e manutenção de software.

Esse é um trabalho baseado na pesquisa realizada por Mahajan [MAHAJ97], cuja estrutura de métricas não é clara. Assim, optou-se por melhor especificá-la, o que facilitou a a geração de centenas de outra métricas.

A geração de uma base de métricas objetivas é o passo inicial para algo mais concreto como a sua implementação e disponibilização na forma de uma ferramenta *CASE* de avaliação de interface.

Resumindo, a interface precisa comunicar, passar informação ao usuário. Para tanto, no *design* de interface é necessário levar em conta diversos aspectos da interação humano-computador. As ferramentas *CASE* construtoras de interface não suportam essa necessidade, o que torna necessário a existência de ferramentas *CASE* de avaliação de interface que assim o façam. Uma ferramenta como essa necessita de métricas, a proposta dessa dissertação.

1.5. ORGANIZAÇÃO DA DISSERTAÇÃO

Para que essa dissertação seja compreendida claramente, ela será apresentada em partes, numa abordagem *top-down*, introduzindo o tema da pesquisa em diferente níveis.

O segundo capítulo aborda as duas grandes áreas de conhecimento: a Engenharia de Software e a Interação Humano Computador. Nesse capítulo, serão apresentados os conceitos mais genéricos, incluindo o conceito de interação visual.

O terceiro capítulo apresenta um estudo sobre Ferramentas *CASE* para *Design* de Interface, a relação a ser explorada nessa dissertação. É apresentada uma visão geral sobre ferramentas CASE, sob o enfoque da ES, abordando posteriormente as ferramentas *CASE* voltadas para *Design* de Interface. Ao final do capítulo é apresentado um tipo de ferramenta diferente, cujo objetivo não é o design, mas a avaliação da interface, então sob o enfoque de HCI. Para esse tipo de ferramenta, são propostas as métricas objetivas para consistência de interface.

O quarto capítulo aborda as métricas objetivas para consistência de interface, oferecidas sob uma estrutura cruzada entre contextos e categorias. Cada cruzamento desses oferece uma tabela, que apresenta um grupo de métricas. No total, são apresentadas 29 tabelas com mais de 750 (setecentos e cinquenta) métricas.

O quinto capítulo é a conclusão dessa dissertação, que apresenta também trabalhos futuros.

2. ENGENHARIA DE SOFTWARE E INTERAÇÃO HUMANO-OMPUTADOR

2.1. INTRODUÇÃO

Este capítulo tem o objetivo de oferecer ao leitor uma visão geral sobre as áreas de Engenharia de Software e Interação Humano-Computador, além de mostrar uma necessidade maior de união entre elas. São apresentados definições e paradigmas para ambas as áreas, com o objetivo de embasar e introduzir conceitos importantes para essa dissertação. Na parte de HCI, também são passados diversos conceitos sobre modelo mental, avaliação de interfaces, modelo de processador humano e teoria da ação. Além disso são descritos estilos de interação, padrões e *guidelines*.

2.2. ENGENHARIA DE SOFTWARE

A Engenharia de Software é um desafio intelectual! Enquanto o hardware é um meio físico, o software é um elemento lógico: o software é desenvolvido mas não é "manufaturado"; tampouco "danifica" ou envelhece com o passar do tempo; é feito, normalmente, sob encomenda, e sua reposição é bastante complicada (quando o hardware apresenta problemas, por exemplo, basta trocar uma peça, o que não é possível com o software) [PRESS94a]. A Engenharia de Software trabalha com idéias, sendo diferente dos tipos clássicos de engenharia, que trabalham com substâncias físicas e objetos. Berry [BERRY92] acrescenta que essas diferenças tornam o software extremamente complexo e a Engenharia de Software intelectualmente desafiante.

Existem diversos tipos de programas e produzir um software é muito mais difícil que se possa pensar. Na verdade, os softwares desenvolvidos para resolver problemas reais são em muitas ordens de magnitude mais difíceis que problemas simples a que os estudantes se propõem a estudar em sala de aula. Os programas de aplicação para o mundo real são indefiníveis, pois eles são ilimitados, contínuos e mudam constantemente para caminhos imprevistos. Se qualquer programa é finito, discreto e, se deixado só, estático e imutável, então existe aí um problema que nunca poderá ser corrigido... [BERRY92]

É inegável que existe a falta de uma assistência metodológica, mas o que fazer quando a maioria dos sistemas que funcionam na indústria são problemas desse tipo? Pressman [PRESS94b] continua essa discussão afirmando que muitas pessoas em posição de responsabilidade têm pouco ou nenhum real entendimento sobre os perigos e oportunidades que o software oferece. A verdade é que sempre que uma tecnologia tem grande impacto, podendo arriscar ou salvar vidas, construir ou destruir negócios, ela deve ser tratada "com mais carinho".

Pressman admite que, embora a Engenharia de Software seja reconhecida como disciplina legítima, muitos profissionais e estudantes ainda não estão conscientes de métodos modernos. Berry, em seu estudo sobre a legitimidade acadêmica da disciplina de Engenharia de Software ajuda a entender o que acontece. Durante sua formação, estudantes da área de informática aprendem a aplicar métodos e técnicas em problemas "infantis". Existe um falso senso de facilidade. Os exercícios de classe são dolorosamente irrealísticos em termos de garantia de qualidade e atividades de manutenção que os softwares irão requerer. Para esses problemas infantis, os métodos modernos não são tão necessários. Entretanto, o hoje estudante e amanhã profissional da área de informática, entrará no mercado de trabalho com uma deficiência muito grande. O resultado disso será sentido diretamente na qualidade do software produzido por ele...

Mas também existem dificuldades pertinentes exclusivamente ao software, os quais Berry [BERRY92] divide em dois grupos: essência (inerente à natureza do software) e acidentes (oriundos da produção do software).

A essência possui algumas propriedades que podem ser problemas, tais como: complexidade (ignorá-la pode gerar um programa incompleto, que não lide com uma situação do mundo real); conformidade (a questão do ajuste do software dentro do sistema, composto ainda por hardware, firmware e peopleware); mutabilidade (uma substituição do software costuma ser mais custo-efetiva que uma mudança); e invisibilidade (o software não é uma realidade física: os diagramas podem capturar no máximo alguns aspectos do sistema, mas não todos).

Os acidentes são as dificuldades na produção em um ponto particular no tempo, de uma particular representação de uma desejada abstração do software, isto é, por exemplo, a dificuldade de programar para uma particular plataforma em uma particular linguagem de programação.

Se o papel do software tem sido crescente nas últimas décadas, sua disseminação em larga escala exige um cuidado muito maior com relação às falhas. A manutenção de software é uma atividade que ocorre após a entrega do mesmo ao cliente, e que consome recursos (pessoal, tempo e dinheiro) da empresa desenvolvedora de software. É importante que essa atividade seja minimizada, tanto quanto possível, para que um projeto seja realmente custo-efetivo. Pode-se ainda afirmar que alguns programas, dada a sua natureza particular, tornam-se não manuteníveis, gerando o que Pressman [PRESS94a] chama de crise do software. Essa crise é afetada pela alta demanda de novos programas no mercado e por um fraco projeto dos mesmos.

2.2.1. DEFINIÇÕES

Berry levanta algumas definições de duas entidades bastante respeitadas na área. Pela definição do SEI (*Software Engineering Institute*), a Engenharia de Software é:

> "a forma da engenharia que aplica os princípios da ciência da computação e matemática para obter soluções custo-efetivas para problemas de software".

Pela definição do IEEE (*Institute of Electrical and Electronic Engineers*), a Engenharia de Software é:

> "a aplicação de uma abordagem sistemática, disciplinada e quantificável para o desenvolvimento, operação e manutenção de software".

Berry sugere uma terceira definição, segundo ele mais completa:

> "**Engenharia de Software é o tipo de engenharia que: aplica uma abordagem sistemática, disciplinada e quantificável; que aplica os princípios da ciência da computação, design, engenharia, gerenciamento, matemática, psicologia e outras disciplinas quando necessárias; e às vezes inventa, para criar, desenvolver, operar e manutenir soluções custo-efetivas, confiáveis, corretas e de alta qualidade para problemas de software**".

Pressman [PRESS94a] parte para uma definição mais prática da Engenharia de Software, argumentando que ela é uma combinação entre métodos para todas as fases do desenvolvimento de software, com as melhores ferramentas para automatizar esses métodos, mais alguns blocos de construção poderosos para a implementação do software, mais as melhores técnicas para assegurar a qualidade de software, somados a uma filosofia para coordenação, controle e gerenciamento.

Em resumo, a disciplina Engenharia de Software é a somatória de métodos (o "como" técnico para construir o software), ferramentas (suporte automatizado ou semi-automatizado para métodos, será aprofundado mais a frente, ainda neste capítulo) e procedimentos (a "cola" que mantém métodos e ferramentas juntos). A somatória, com diferentes combinações, é conhecida como paradigma da Engenharia de Software.

2.2.2. PARADIGMAS

Todo desenvolvimento de software prevê sua evolução em pelo menos três fases genéricas: análise de requisitos, projeto e implementação [CROSS92, p.20]. Essas três fases genéricas são diferentes níveis de abstração que representam diferentes propostas. A análise representa o quê o sistema deverá fazer. O projeto representa como essa funcionalidade será construída. A implementação se responsabiliza pela codificação do que foi decidido nas fases anteriores.

Pressman [PRESS94a] levanta outras três fases genéricas, encontradas em todo desenvolvimento de software, independente da área de aplicação, tamanho do projeto ou complexidade. São elas: definição, desenvolvimento (projeto e implementação) e manutenção. As duas primeiras fases têm respectivamente as mesmas definições oferecidas por Cross. Já a manutenção vem a ser toda mudança que ocorra ao software, seja ela uma correção, adaptação ou melhoramento.

Existem vários paradigmas bastante conhecidos na literatura da Engenharia de Software. Entre eles podemos citar alguns: o modelo clássico de ciclo de vida (waterfall), o modelo espiral, a prototipação, e as técnicas de quarta geração, como pode ser visto na Figura 1, extraída de [PRESS94a]. Cada um deles pode fazer uso de técnicas estruturadas, técnicas orientadas a objetos ou ainda outras. A apresentação de cada paradigma está fora do escopo dessa dissertação.

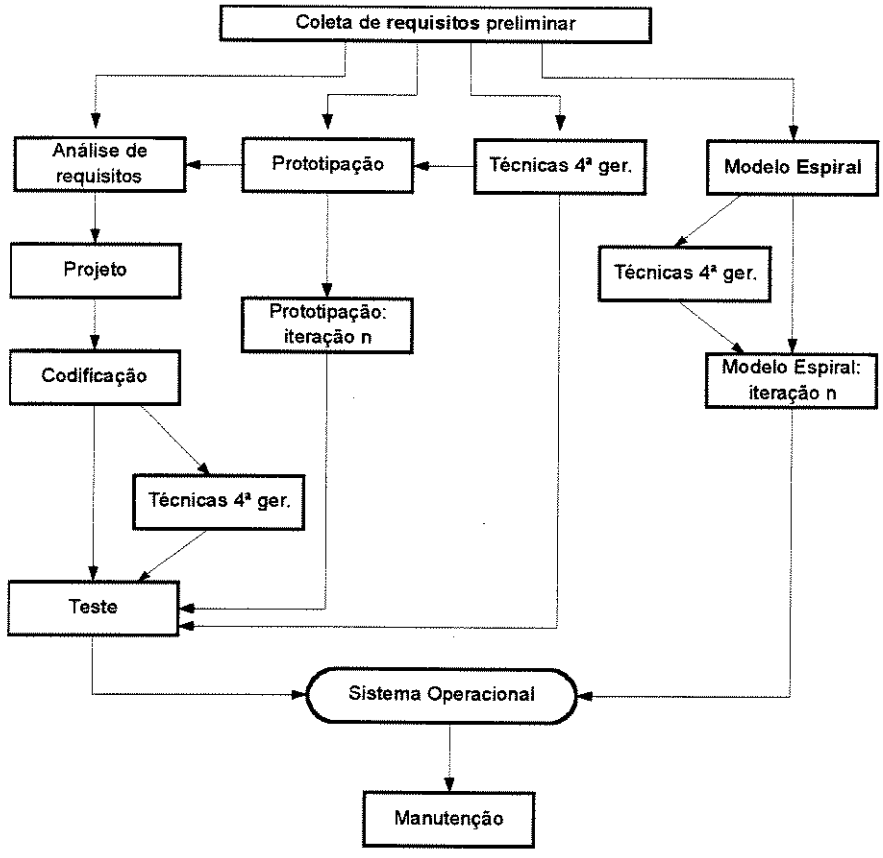

Figura 1: Combinação de Paradigmas.

Pressman defende que não há uma única melhor abordagem, que cada paradigma destes oferece suas vantagens e desvantagens e que a melhor alternativa seria fazer uma combinação de paradigmas.

2.2.3. TAXONOMIA DE TERMOS

Segundo Cross [CROSS92], a Engenharia de Software se perdeu na sua terminologia. Ele tenta definir uma taxonomia para processos que podem ocorrer durante essas três fases genéricas que qualquer paradigma apresenta, como pode ser visto na Figura 2. Dependendo da necessidade que se tenha em relação ao software, pode-se precisar fazer uma:

- engenharia "pra frente" ("*forward engineering*"): segue a sequência normal de análise de requisitos, projeto e implementação;
- engenharia reversa: segue a sequência inversa à engenharia normal. Parte da implementação (e.g., um sistema já implementado e não documentado) e tenta criar uma representação desse sistema, com a proposta de fazer um clone do sistema original. É conduzida por alguém que não o desenvolvedor, sem o benefício dos documentos originais...;
- reestruturação: transforma um sistema de software com uma forma de representação em outra, num mesmo nível de abstração, preservando o comportamento externo do sistema (e.g., alteração de um algoritmo para melhorar o desempenho de uma tarefa);
- reengenharia: inclui modificações com respeito a novos requisitos não encontrados no sistema original (e.g., adição de funcionalidade).

A seguir, a figura que Cross sugere, extraída de [CROSS92].

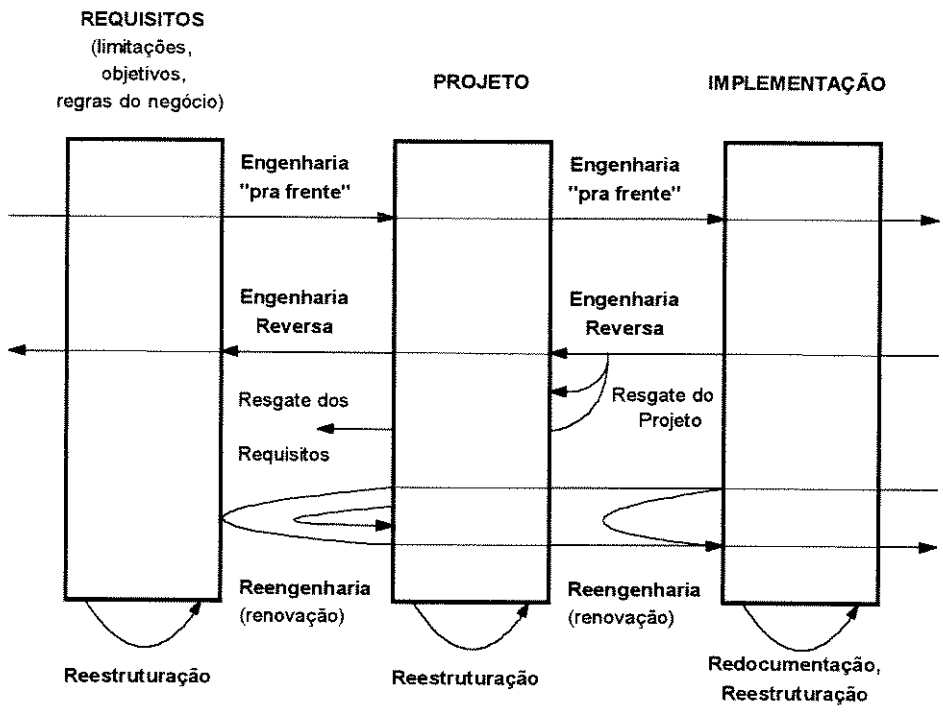

Figura 2: Engenharia "pra frente" e Reversa, Reestruturação e Reengenharia entre as fases do ciclo de vida.

A idéia de mostrar que existem diversos paradigmas (waterfall, espiral, etc) e que, pelas suas fases genéricas pode-se ter processos (engenharia reversa, reengenharia, etc) diferentes, é concentrar a atenção para a questão do *design* de interface. Independentemente da abordagem, paradigma, que se estiver seguindo, haverá um momento no ciclo de vida do desenvolvimento do software em que ocorrerá a preocupação com o *design* dessa interface. Nesse momento, a Engenharia de Software pede ajuda a uma área menos técnica e mais humana, a área de Interação Humano-Computador.

2.3. INTERAÇÃO HUMANO-COMPUTADOR

A Interação Humano-Computador, internacionalmente conhecida como HCI (acrônimo de Human Computer Interaction) é uma área de estudo **multidisciplinar**. É um campo não tão recente, que tem recebido muita atenção nessa última década. Vários autores tentam definir esse campo de estudo, mas poucos são os que conseguem realmente mostrar sua abrangência. Pretende-se mostrar essa abrangência, através de algumas definições e da dificuldade em se estabelecer um currículo básico para HCI, dadas as diferentes perspectivas em relação à área.

2.3.1. DEFINIÇÃO

Ainda hoje não há uma definição para HCI que seja aceita por todos os pesquisadores da área. Várias pesquisas têm sido elaboradas, mas muitas delas de maneira isolada, sob um determinado enfoque, contribuindo para a particularização de conceitos que envolvam HCI.

Foram capturadas definições, oriundas de pesquisas de [ACM92] [JOHNS92] [LUCEN94]. Numa junção das definições apresentadas pelos autores, pode-se afirmar abrangentemente que HCI **é uma disciplina voltada ao projeto, avaliação e implementação de *sistemas interativos* de computação para uso humano e ao estudo dos fenômenos que os cercam. É o estudo da interação entre pessoas, computadores e tarefas**. É uma área que se preocupa em compreender como pessoas e computadores podem executar tarefas interativamente e como tais sistemas interativos são projetados.

É uma área **interdisciplinar** e **multidisciplinar** cujos conhecimentos são derivados de áreas de ciência, engenharia, projeto e arte. Essa área envolve o desenvolvimento de aplicações e princípios, *guidelines*, e métodos para suportar o projeto e avaliação dos sistemas interativos. Compõe-se de diversos pontos de estudo, abordados por áreas de estudo tais como:

- **ciência da computação** (projeto de aplicação e engenharia de interfaces humanas);
- **psicologia** (aplicação de teorias de processos cognitivos e análise empírica do comportamendo do usuário, percepção);
- **linguística** (teoria de formalismos da linguagem, do diálogo);
- **ergonomia** (aspectos físicos da interação: mobiliário, luminosidade, ruídos, teclado, disposição de equipamentos);
- **fatores humanos** (aspectos cognitivos da interação: atividades mentais conscientes e inconscientes);
- **sociologia** e **antropologia** (interação entre tecnologia, trabalho e organização);
- **projeto industrial** (produtos interativos).

Heckel [HECKE94] prefere optar por uma definição um pouco mais "artística", salientando que um sistema interativo também deveria se basear em outras áreas, tão desenvolvidas quanto as citadas anteriormente, tais como: literatura, teatro, publicidade, ensino, fotografia, belas-artes, magia, cinema, vendas, dramaturgia, jornalismo, artes gráficas, música, decoração de ambientes e arquitetura.

Essas definições das subáreas são definições vagas e pouco claras! Por se tratar de uma coleção de diversos pontos de estudo, abordados por diversas áreas de estudo, não foi encontrado nenhum trabalho definindo claramente, ou melhor, consensualmente, cada uma delas e sua importância para a grande área de HCI. Em outras palavras, ainda não está clara a função das partes no todo...

Essa indefinição pode ser vista com o frequente número de publicações voltadas à área de HCI, com os mais variados enfoques. Iniciativas têm sido tomadas por parte de associações e entidades envolvidas no campo, numa tentativa de agrupar essa variedade de pontos abordados. Essas associações e entidades publicam livros, promovem encontros, congressos e *workshops* e trabalham pelo desenvolvimento de uma consensualidade a respeito de HCI. Muito desse trabalho pode ser visto na tentativa de construir um currículo básico para a área.

2.3.2. CURRÍCULO BÁSICO: ALGUNS PROBLEMAS

Muito tem sido discutido a respeito dos conceitos que um profissional ligado à área de HCI (e.g., *designer* de interface) precisa conhecer para exercer sua profissão. As abordagens são diversas, umas mais acadêmicas, outras mais profissionais. A seguir encontram-se as opiniões de diversos autores preocupados com inúmeros problemas pertinentes à questão.

Mantei [MANTE89] afirma que um campo se torna uma ciência válida ao ter um corpo de conhecimento o qual seus membros reconhecem como básico. Novos campos crescem e se desenvolvem de velhos campos, como uma subsérie do corpo de conhecimento que fica tão grande quanto o corpo pai, ou quando requer paradigmas de outros campos para apropriadamente abordar suas questões de pesquisa. Isso ocorreu com HCI.

Segundo a autora, sem uma sólida fundação (uma estrutura que permita passar o conhecimento adiante), resumindo o corpo de conhecimento de HCI, esse campo corre o risco de desaparecer. Essa fundação envolve programas de HCI em universidades, currículos publicados, séries de textos básicos de leitura, etc.

Howard [HOWAR95] trabalha outro ponto interessante (também voltado ao desenvolvimento de interface), que é a aplicação de três recursos de HCI, como pode ser visto na Figura 3: o que *designers* dizem que precisam, o que *designers* realmente precisam e, o que HCI realmente oferece para o desenvolvimento de interfaces. Essas esferas de conhecimento se entrelaçam dando uma clara idéia de como é fácil enxergar a soma de conhecimento relevante ao campo, e de como é difícil extrair dessa figura tão subjetiva algo tão objetivo como um currículo.

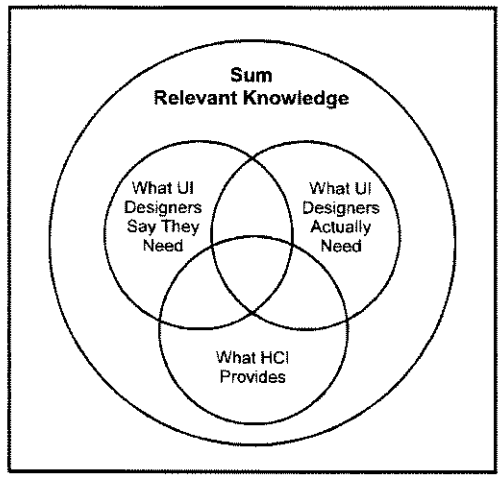

Figura 3: Soma de conhecimento relevante.

Howard propõe que os estudantes devem ser expostos à prática de *design* de *peritos* (professores e profissionais), algo que quase nunca acontece. Complementa que esse processo é essencial aos estudantes, para que eles desenvolvam um senso para o design, em outras palavras, sensibilidade profissional.

Preece [PREEC90] também levanta que a natureza interdisciplinar de HCI é identificada como sendo altamente controversa quando se decide o que e como estudar. Além da falta de um currículo estabelecido, a base de pesquisadores, profissionais e professores tem uma forte influência na perspectiva escolhida e na importância relativa às diferentes questões.

Num levantamento de Programas Educacionais em HCI, existentes ao redor do mundo, realizado por Gasen [GASEN93] [GASEN94a] [GASEN94b], foram encontrados números bastante expressivos mostrando escolas, cursos e professores universitários que ensinam HCI. Suas intenções eram oferecer, a estudantes, informações sobre oportunidades educacionais e, a educadores de HCI, informações sobre outros educadores. Nesses estudos, Gasen argumenta que ainda que ninguém concorde numa abordagem única em como ensinar HCI, há um comprometimento com o mesmo objetivo, desenvolver graduados capazes de projetar sistemas que melhor encontrem as necessidades e o contexto do usuário.

Segundo [ACM92], o rápido desenvolvimento da área deve preparar os estudantes não apenas para o estado atual da tecnologia, mas fornecer fundações para possibilidades futuras. No entanto, conforme Gasen comenta, ao mesmo tempo que HCI é uma área nova e dinâmica, que muda tão rápido quanto a tecnologia, a mudança de um currículo pode levar anos, e a educação de HCI pode ser mais uma das vítimas dos processos burocráticos.

Numa abordagem um pouco mais mercadológica, Gasen [GASEN95], argumenta outro ponto fundamental: em HCI o papel da equipe de especialistas pode variar, os tipos de projetos podem ser vastamente diferentes em tamanho e escopo e as definições práticas podem ser muito diferentes. Cada uma dessas dimensões tem impacto significativo no conhecimento e qualidades necessárias para ser efetivo como um especialista em HCI na definição prática. Ainda assim, o currículo deve ser desenvolvido e algum nível de consenso deve ser alcançado. É necessário um melhor entendimento da natureza prática de HCI e como essa prática pode ser mapeada de volta para uma variedade de experiências práticas. Deve-se determinar o que precisa ser aprendido, quando deve ser aprendido, e qual a melhor estrutura para o ambiente de aprendizado.

2.3.3. HCI E SEUS ESTÁGIOS DE INTERAÇÃO

Para entender claramente seus diferentes níveis, estágios de interação de HCI, Marsh [MARSH90] oferece algumas definições operacionais. Segundo a autora, em HCI existem muitos caminhos separados de comunicação entre o humano e o computador, como o ambiente natural do computador, sua ergonomia, o ambiente de operação e o ambiente de aplicação. Esses **caminhos** são **simultâneos e paralelos**, e têm um diálogo e um componente de interface. O diálogo é a troca observável de símbolos entre o humano e o computador. Ocorre em paralelo desde que os dois participantes estejam interpretando o símbolo no mesmo contexto. Já o componente de interface é o software e hardware de suporte por onde a troca de símbolos ocorre. Em outras palavras, o diálogo é a troca de símbolos e o componente de interface é o meio. Para entender os caminhos simultâneos a autora oferece a Figura 4:

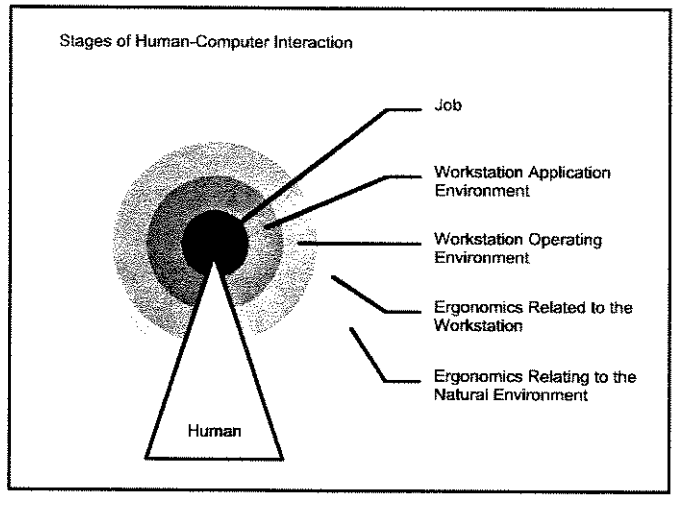

Figura 4: Estágios da Interação Humano-Computador.

Para entender essa figura, a autora define ergonomia como sendo o estudo do relacionamento do humano com máquinas em geral, no caso dessa dissertação o computador. Na ergonomia relacionada ao ambiente natural os ítens a serem observados são: luzes (claridade ou escuridão), espaço (vazio ou cheio, limpo ou sujo), janelas (ventilação ou abafamento), cores (calmas ou 'gritantes'), temperatura (fria ou quente), barulho (pouco ou muito), etc. Na ergonomia relacionada ao computador, altura do chão à cadeira, ajustabilidade, suporte para as costas, conforto, de fácil movimento, suporte para os braços, etc. No ambiente operacional do computador, o sistema operacional. A interação para o ambiente operacional é o diálogo com o computador. No ambiente de aplicação do computador, as ferramentas que a aplicação utiliza, e na tarefa, a aplicação em si.

Todas essas definições mostram que o estudo de HCI pode levar não somente ao *design* de interface, objetivo dessa dissertação, mas a diversos outros de pesquisa.

2.3.4. CICLOS DE VIDA

Não se pretende defender nenhum ciclo de vida como exemplo para o desenvolvimento da interface. Está-se tomando por definição que, em qualquer que seja o ciclo de vida escolhido, haverá um momento para o *design* de interface, e que, nesse momento, deverão ser levados em conta diversos aspectos abordados nessa dissertação, como já disposto no capítulo anterior.

Pretende-se apenas citar que existem ciclos de vida mais voltados à interface, tais como: modelo *snowflake* e modelo de Seeheim, citados por [SILVA95] e o ciclo de vida estrela, proposto por Hix [HIX93a], cuja atividade principal é a avaliação de usabilidade. Esse ciclo minimiza o número de limites de ordenação entre as atividades de desenvolvimento, não sendo necessário, por exemplo, especificar todos os requisitos antes de começar a trabalhar no design.

Segundo Day [DAY93], os ciclos de vida variam em seu escopo, sendo que o modelo estrela contempla o desenvolvimento de interface humano-computador. Esse ciclo também pode ser facilmente "embutido" dentro do modelo espiral, por exemplo, o que leva à figura que Pressman [PRESS94a] sugere, Figura 1, na qual diversas fases do desenvolvimento de um software são contempladas por diferentes modelos, e que as abordagens complementam-se umas às outras [JOHNS92].

2.3.5. MODELO MENTAL

No desenvolvimento de um software, independentemente do ciclo de vida seguido, geralmente haverá diversas pessoas envolvidas na equipe, e cada uma delas desempenhará um "papel" durante o desenvolvimento do software. Na visão dessa dissertação, a melhor definição encontrada para modelos mentais encontra-se nessas figuras abaixo, bastante conhecidas da literatura [HIX93b].

Figura 5: como proposto pelo proponente do projeto

Figura 6: como especificado no pedido do projeto

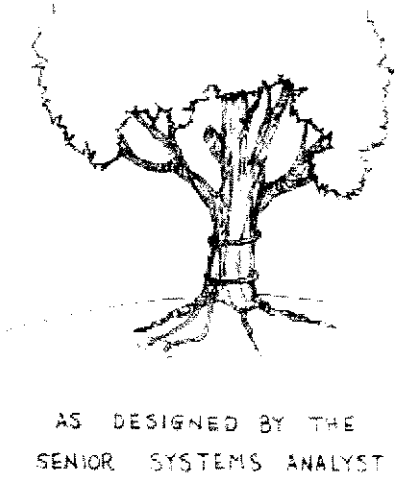

Figura 7: como projetado pelo analista de sistemas sênior

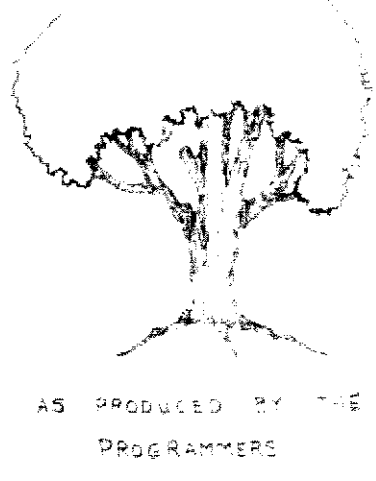

Figura 8: como produzido pelos programadores

Figura 9: como instalado no local de trabalho do usuário

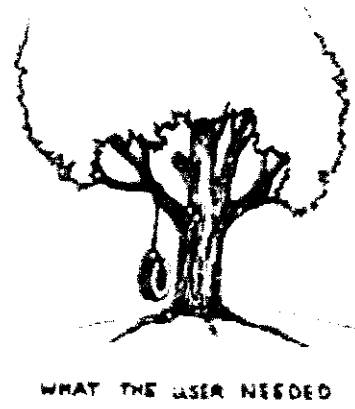

Figura 10: o que o usuário precisava

Através dessas figuras, consegue-se entender o porquê de muitos "desencontros" entre o que se esperava de um software e seu resultado final... O modelo mental do usuário é um ponto fundamental dessa dissertação. É a partir desse modelo mental que a interface precisa ser construída. Em nada adianta desenvolver um software com alta funcionalidade, cuja interface não respeite o modelo mental de interação do usuário final.

Um bom modelo mental deve prevenir erros, além de fornecer *feedback* para o usuário [NEWMA95]. Qualquer modelo mental capaz de "funcionar" terá alguma forma ou estrutura, ligando os vários ítens de conhecimento e relacionamentos causais aprendidos pelo usuário. Como Newman afirma, com relação ao sistema, o modelo mental não é baseado no conhecimento da sua estrutura interna, mas na observação do seu comportamento externo, o que Don Norman [NORMA86] chama de imagem do sistema. Essa imagem do sistema é a interface do usuário.

Em [VERTE90] sugere-se que esses modelos diferentes existem em função do comportamento do *designer* em relação ao usuário:

- *designers* de interface podem considerar-se ou simular usuários potenciais, o que pode apresentar um resultado bastante inesperado, pois depende muito da experiência do designer, além de existir uma enorme diferença entre usuários novatos e *peritos*;
- *designers* de interface podem perguntar a usuários reais, o que aumenta a chance de apresentar um modelo confiável.

Esses comportamentos podem variar dependendo dos critérios e métodos escolhidos para avaliação de interface.

2.3.6. AVALIAÇÃO DE INTERFACE: CRITÉRIOS E MÉTODOS

2.3.6.1 CRITÉRIOS

A **subjetividade** é uma **característica inerente à interface**... tão presente que qualquer tentativa de selecionar critérios para avaliá-la herdará tal característica. Dentre diversos critérios que se aplicam à interface, seguramente "**usabilidade**" é o mais usado.

Como avaliar a usabilidade de uma interface? Os critérios que apresentam respostas a essa pergunta não são bem definidos. A usabilidade sugerida por [HIX93c] possui definição mais abrangente, diferente da definição de outros autores. Para a autora, usabilidade está relacionada à efetividade e eficiência da interface do usuário e da reação do usuário à essa interface, pois, para o usuário, a comunicação é, pelo menos, tão importante quanto a computação.

Jonhson [JOHNS92], por exemplo, define que os critérios de interesse da perspectiva de fatores humanos são: usabilidade, facilidade de aprendizado, eficiência e aceitabilidade, sendo a usabilidade a habilidade de dar suporte às atividades do usuário. Também existem normas internacionais que dão uma outra definição para os critérios, como é o caso da norma ISO/IEC 9126 [ISO9126], que define seis características de qualidade de software, que podem ser perfeitamente aplicadas à interface: funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade. Pela norma, cada uma delas pode se subdividir em subcaracterísticas. O problema dessas características é sua falta de métrica. A norma não fornece métricas para medir cada uma delas. E esses critérios serão, de uma forma ou outra, abordados na escolha dos métodos de estudo do usuário.

2.3.6.2 MÉTODOS

De uma forma geral, pode-se dizer que existem dois métodos genéricos de avaliação: **avaliação analítica** e **avaliação empírica**. As técnicas analíticas não envolvem usuários reais, e oferecem menor necessidade de organização. Já as técnicas empíricas quase sempre envolvem usuários reais, exigindo muito planejamento prévio e uma cuidadosa conduta e análise do experimento [NEWMA95].

A avaliação empírica (ou heurística) é uma inspeção de usabilidade. Pode ser realizada usando um time de avaliadores e uma série de heurísticas de projeto (*guidelines* de proposta geral). Esse tipo de avaliação pode envolver técnicas diferentes, como o uso de *questionários*, *entrevistas* e *observação do usuário*, descritas brevemente em [NEWMA95].

Outra maneira de classificar os métodos de avaliação, proposto nessa dissertação, é dividí-los em **avaliação subjetiva** e **avaliação objetiva**. A avaliação subjetiva é uma avaliação qualitativa de aspectos subjetivos da interface, em que é necessário o julgamento do *designer* para a coleta dos dados. Essa avaliação dependerá de métricas subjetivas, cuja definição se encontra no quarto capítulo. Esse tipo de métrica não será abordado nessa dissertação. Por outro lado, a avaliação objetiva é uma avaliação quantitativa de aspectos objetivos da interface, em que não é necessário o julgamento do *designer* para a coleta dos dados, podendo ser executada automaticamente. Essa avaliação dependerá de métricas objetivas, cuja definição se encontra também no quarto capítulo. É para esse tipo de avaliação objetiva que estarão sendo geradas as métricas dessa dissertação.

2.3.7. MODELO DE PROCESSADOR HUMANO

Como o ser humano processa a informação? Como o ser humano responde às informações que são enviadas ao seu cérebro? Com que rapidez essa resposta é processada? Qual a capacidade da memória do ser humano? Essas são perguntas que até hoje permanecem sem resposta, mas que alguns autores já tentaram responder através de modelos. Um desses modelos é o Modelo de Processador Humano (MPH), desenvolvido por Card, Moran e Newell [CARD83].

O Modelo de Processador Humano funciona como um sistema de processamento de informação. Pode ser descrito como uma série de **memórias** e processadores, seus parâmetros e suas interconexões. É dividido em **três subsistemas**, cada qual com suas memórias e processadores: **sistema perceptual**, **sistema motor** e **sistema cognitivo**. A seguir tem-se suas descrições.

2.3.7.1 MEMÓRIAS E *CHUNKS*

Norman [NORMA72] comenta ser bobagem pensar na memória humana como sendo uma coisa unitária, pois o cérebro permanece sendo um mistério. Segundo ele, existem diferentes tipos de memórias, dentre os quais a memória de curta duração (*short-term memory*, STM) e a de longa duração (*long-term memory*, LTM).

Johnson [JOHNS92] define que a memória de curta duração possui uma capacidade de armazenamento transitória, limitada e sujeita a interferências. A memória de longa duração, por sua vez, retém informações por longos períodos, em estado inativo. A passagem do estado inativo para o ativo é um estado definido para a memória de trabalho (*working memory*). A memória de trabalho é uma definição funcional de um processo da memória.

Um *chunk* é uma unidade de informação, composta de um ou mais ítens de informação, que possuam um significado como conjunto. Existe uma variação da capacidade da memória em armazenar *chunks*. Quanto mais sentido tiver uma informação, mais fácil será armazená-la. Como exemplo, pode-se pedir a uma pessoa que decore, dígito a dígito, o seguinte número: 01925201510194337075. Caso ela consiga, seguramente ela o esquecerá em poucos instantes, pois a memória de trabalho tem um tempo de decaída (*decay time*). Em seguida, pode-se oferecer a informação de que tais números são dois telefones, e a leitura dos mesmos poderá ser da seguinte maneira: (019) 252-0151 e (019) 433-7075. Seguramente a chance dessa pessoa decorar o número aumentará. Isso deve-se ao fato de que a informação passou a ter algum sentido, ou seja, foi lida em *chunks*. Os três primeiros dígitos representam um prefixo de uma cidade, os outros três um prefixo interno da cidade e os outros quatro dígitos um código próprio. Uma vez que o usuário tenha conhecimento dessa estrutura, fica fácil sua memorização.

As informações são registradas na memória através desses *chunks*, e sua recuperação ao estado ativo será tão rápida, quanto mais forte for o seu registro. Segundo Johnson, seu esquecimento será a inabilidade de ativar o registro correto na memória.

2.3.7.2 SUBSISTEMAS

O **sistema perceptual** é responsável pelo armazenamento de imagens visuais e auditivas. Um dos pontos dessa teoria revela que, ainda que os olhos alcancem uma cena visual de grande angulação, detalhes serão observados somente numa curta região, cerca de dois graus, uma região chamada fóvea. O **sistema motor** é responsável pela tradução do pensamento em ação, pela ativação de músculos voluntários. O **sistema cognitivo** é responsável pela conexão das entradas do sistema perceptual para as saídas corretas do sistema motor. Pelo fato de a maioria das tarefas executadas pelo ser humano serem complexas e envolverem aprendizado, recuperação de fatos e soluções de problemas, [CARD83] aponta que esse é o mais complicado dos três subsistemas.

2.3.8. TEORIA DA AÇÃO

Muito do que se pode imaginar em relação à maneira como os usuários executam tarefas pode ser mapeado através da Teoria da Ação, de Norman [NORMA86]. Essa teoria tenta mostrar que a maneira como as pessoas fazem coisas, no caso com o computador, normalmente seguem uma determinada sequência de ação.

2.3.8.1 GOLFO DE EXECUÇÃO X GOLFO DE AVALIAÇÃO

Uma pessoa ao interagir com o sistema tem seus objetivos expressos em termos relevantes à pessoa (em termos psicológicos) e os mecanismos e estados do sistema são expressos em termos relativos ao computador (em termos físicos). Essa discrepância entre variáveis psicológicas e físicas cria grandes problemas que precisam ser endereçados no projeto, análise e uso de sistemas. Segundo Norman, essas discrepâncias representam dois golfos: **golfo de execução** e **golfo de avaliação**. Os golfos são unidirecionais e podem ser vistos na Figura 11, extraída de [NORMA86]:

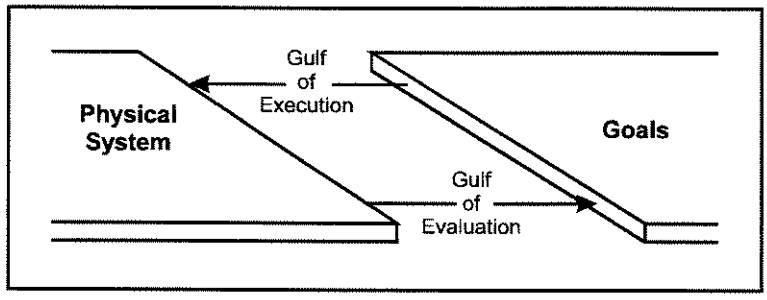

Figura 11: Golfos de Execução e Avaliação.

Para cobrir o golfo de execução são necessários quatro segmentos: formação da intenção, especificação da sequência de ação, execução da ação e contato com os mecanismos de entrada da interface.

Para cobrir o golfo de avaliação são necessários também quatro segmentos: *display* de saída da interface, movimentação para o processamento perceptual desses *displays*, para sua interpretação e para sua avaliação.

Quando, por exemplo, uma mudança necessária na interface do sistema não ocorrer imediatamente após a execução da sequência de ação, a demora resultante pode impedir dramaticamente o processo de avaliação, pois o usuário poderá não lembrar os detalhes das intenções dessa mesma sequência.

O processo de execução e avaliação de uma ação pode então ser aproximado a sete estágios de atividade do usuário: estabelecimento do objetivo, formação da intenção, especificação da sequência de ação, execução da ação, percepção do estado do sistema, interpretação do estado e avaliação do estado do sistema (com relação aos objetivos e intenções).

Esses estágios podem ser melhor entendidos com a Figura 12, extraída de [NORMA86].

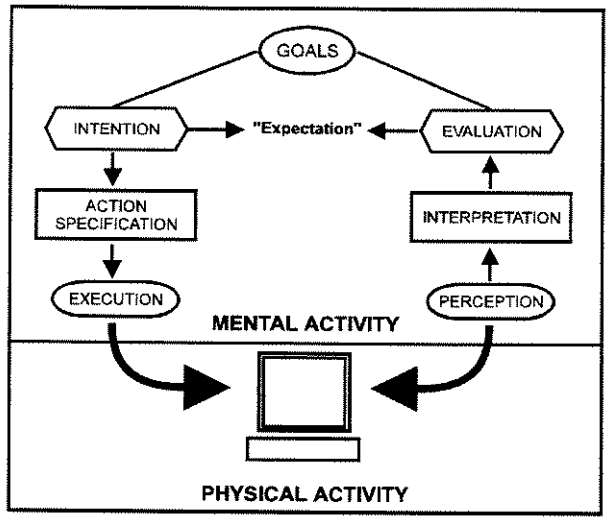

Figura 12: Sete estágios de atividade mental do usuário.

É muito importante ressaltar que uma atividade real não precisa necessariamente progredir como uma simples sequência de estágios. Os mesmos podem aparecer fora de ordem, alguns podem nem aparecer, e outros podem ainda se repetir.

Fica bastante claro que uma das implicações práticas ao cobrir esses golfos é a aproximação do sistema ao usuário e vice-versa.

2.3.8.2 DISTÂNCIA SEMÂNTICA E ARTICULATÓRIA

Sempre que se interage com algum dispositivo, está-se lidando com uma linguagem de interface. A descrição das ações desejadas será uma expressão nessa linguagem de interface. Pode ocorrer o problema da linguagem de entrada não ser a mesma da saída, causando distância semântica e articulatória.

Distância semântica é o relacionamento entre a intenção do usuário e o sentido das expressões e distância articulatória é o relacionamento entre o sentido das expressões e sua forma física. Pode-se entender melhor essa definição através da figura a seguir, extraída de [HUTCH86].

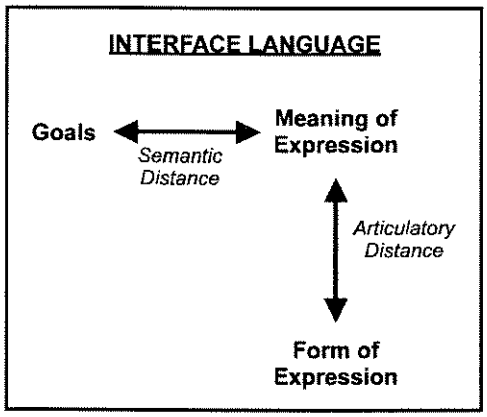

Figura 13: Linguagem da interface.

Um exemplo de distância semântica pode ser claramente identificado entre as plataformas DOS e Unix, analisando seus respectivos comandos "dir" e "ls", ambos com a função de mostrar os arquivos existentes num diretório. Para o usuário, a intenção de ver os arquivos existentes num diretório é melhor expressa e melhor compreendida quando é usado o comando "dir", cujo sentido da expressão é mais claro que o comando "ls". Um exemplo de distância articulatória pode ser claramente identificado entre, por exemplo, as plataformas DOS e *Windows95*. A plataforma DOS exige a digitação de um comando, enquanto que a plataforma *Windows95* exige simplesmente o apontamento de um ícone.

2.3.9. ESTILOS DE INTERAÇÃO

Houve uma grande evolução no estilo de interação, comparando-se os primórdios da computação com os dias de hoje. Essa evolução do estilo de interação pode ser bem entendida através das fases descritas por Pressman [PRESS94a] e Walker [WALKE90].

Segundo Pressman, a evolução passou por quatro fases. Na primeira fase, havia comandos e interfaces de perguntas, e a comunicação era puramente textual. Na segunda, havia menus simples, e havia uma lista de opções a serem selecionadas via digitação. Na terceira, havia janelas e interfaces WIMP (*Windows, Icons, Menus and Pointing devices* - janelas, ícones, menus e dispositivos de apontamento). Na quarta fase, havia hipertexto e multitarefa, tanto em estações de trabalho quanto em micros pessoais.

Segundo Walker, a evolução passou por cinco fases. Na primeira fase havia painéis com plugues, botões, mostradores e o funcionamento era dedicado. O conceito de usuário era diferente do conceito de operador da máquina. Na segunda fase, havia lotes de cartões perfurados e entrada de dados remota. Na terceira fase, havia tempo compartilhado via teletipo. Já havia sistemas operacionais e operação concorrente de múltiplos serviços. A interação era via teletipo. Na quarta fase, entraram os sistemas baseados em menus e, na quinta, controles gráficos e janelas, com o uso do *mouse*.

Cada um desses estilos de interação, disponíveis no software, foram consolidando-se com a evolução do hardware. A maneira como o ser humano interage com a máquina nos dias atuais varia conforme a tecnologia disponível [NEWMA95]. Essas diferentes tecnologias afetam a forma como são apresentadas as funcionalidades do software.

Dos estilos existentes atualmente, Newman [NEWMA95] prefere classificá-los em três categorias gerais: modo de teclas (*key-modal*), manipulação direta (*direct manipulation*), linguística (*linguistic*). Essas três categorias se subdividem em estilos. Essa classificação, entretanto, é pobre, pois não são cobertas todas as possibilidades de interação existentes atualmente. Como o próprio autor prefere afirmar, não é possível cobrir todos os estilos em suas variações. Dos estilos apresentados por Newman, os mais importantes para essa dissertação são os estilos cuja interação é baseada em linha de comando, menu e manipulação direta.

2.3.9.1 LINHA DE COMANDO

Segundo [NEWMA95], esse estilo apresenta uma grande velocidade de operação, sendo normalmente o estilo mais rápido. Suporta uma grande variedade de propriedades funcionais, incluindo entrada e recuperação de dados. De uma maneira geral, exige uma tecnologia simples e, para que tenha um alto grau de usabilidade, requer muito treinamento. Outros estilos mapeiam suas funções para esses comandos e parâmetros, escondendo do usuário sua complexidade.

Segundo [CRAWF90], dependendo da aplicação, esse estilo pode ser uma solução arcaica e inadequada. O usuário precisa conhecer a sintaxe e a semântica do vocabulário usado pela linguagem de comando para conseguir interagir.

2.3.9.2 MENUS

Segundo [NEWMA95], esse estilo exige tecnologia simples. Fornece somente as ações disponíveis à circunstância, prevenindo erros. Um problema que pode existir é o excesso de funções disponíveis e sua organização hierárquica dentro da estrutura do menu. Determinada ação que possui uma frequência de acessos pode ficar em um nível da estrutura cuja navegação se torne cansativa [WALLA95]. Sua seleção, normalmente, pode ser feita por digitação do número da opção, ou uma letra do comando que esteja sublinhada, ou por um dispositivo de apontamento, ou ainda uma *touchscreen*.

Para usar os menus, existem vários detalhes a serem considerados, tais como a redução à memorização de comandos e a redução à necessidade de treinamento [KOVED95]. Outros detalhes são a forma como os menus são apresentados. Um estudo de [CALLA95] faz uma comparação entre menus lineares e menus pizza. Como o menu mais comum é o linear, deve-se ter cuidado por exemplo com o número ótimo de ítens por menu alocados. Um estudo de [WALLA95] compara a preferência do usuário em relação ao *design* de vários menus com pequena profundidade versus poucos menus com grande profundidade. A primeira opção demonstrou ser mais rápida resultando em menos erros.

Segundo [CRAWF90] a vantagem dos menus é declarar as opções disponíveis ao usuário. Ele é auto-explicativo. Seus problemas são os mesmos levantados por [WALLA95], o usuário que precisar de uma função e não souber em que ponto da estrutura dos menus ela se encontra, poderá entediar-se na navegação, prejudicando a usabilidade do sistema.

2.3.9.3 MANIPULAÇÃO DIRETA

Esse será o estilo de interação mais explorado nessa dissertação, visto que é o paradigma mais aceito nos dias de hoje no mercado, e porque faz uso dos estilos citados anteriormente. A popularização do uso de computadores precisou tornar o estilo de interação mais próximo às intenções do usuário. O usuário já não é mais um técnico ou um profissional da informática, mas qualquer pessoa que possua um computador em casa. Não resta dúvidas de que esse estilo foi o melhor assimilado, por traduzir facilmente os objetivos do usuário em ação.

Shneiderman [SHNEI95a] foi o idealizador desse estilo, cuja idéia principal era colocar o usuário no controle de todas as ações, fornecendo uma interface fácil de aprender, rápida em performance, com baixas taxas de erros, fácil de retenção e com alta satisfação subjetiva. Seus princípios eram: apresentação visual de objetos e ações de interesse; ações rápidas, incrementais e reversíveis; seleção por apontamento (ao invés de digitação) e imediato e contínuo *feedback* de resultados de ações. Esse paradigma tornou-se amplamente conhecido em 1983, influenciando largamente o mercado de computadores. Diversos estudos se seguiram, realizados por pesquisadores de diversas partes do mundo, com os mais diferentes interesses, entre estes comparar diversos estilos de interação, dando maior refinamento à manipulação direta. Hoje, quase quinze anos depois do seu começo, muito existe ainda a ser estudado.

Segundo [SHNEI95b], a manipulação direta oferece a satisfatória experiência de operar objetos visíveis, tornando o computador transparente e permitindo que os usuários concentrem-se em suas tarefas. É um estilo que fornece *feedback* ao usuário. Ao trabalhar diretamente com objetos tem-se o conceito *WYSIWYG* (*What You See Is What You Get* - o que você vê é o que você tem).

Segundo [NEWMA95], esse estilo tem alguns problemas que podem atrapalhar o usuário. Um deles é a lentidão ao executar uma tarefa dada a necessidade de apontar e clicar ao invés de simplesmente pressionar uma tecla. Por esta razão, do que se vê hoje, muitos sistemas baseados nesse estilo oferecem a possibilidade de customização da interface, fornecendo combinações de teclas, também conhecidas como *shortcuts*. Outro problema levantado por Newman deve-se a quantidade de objetos disponíveis em tela, podendo contribuir para a distração do usuário, confundindo seus objetivos inicialmente propostos. Maiores problemas desse estilo serão abordados nos capítulos seguintes.

2.3.9.4 QUAL O MELHOR ESTILO?

A escolha de estilos é muito difícil! E não se pretende defender qualquer um desses estilos citados anteriormente. Cada um possui suas vantagens e desvantagens.

A escolha do estilo a ser utilizado irá depender da especificação dos requisitos da interface. E pode ocorrer de nem sempre um único estilo ser suficiente. Para algumas aplicações, que possuem diferentes níveis de complexidade e de usuários interagindo, é interessante oferecer diferentes estilos de interação.

Um exemplo: o *design* de uma interface que estará disponível a partir de um caixa eletrônico, dentro de uma agência bancária, por onde se pode fazer um auto-atendimento. Qual seria o melhor estilo de interação? Qual é a especificação de requisitos para sua interface? Quem são os usuários (novatos ou peritos)? Com que rapidez eles precisam das respostas do sistema? Quantos estão dispostos a aprender a trabalhar com o caixa eletrônico? Qual é o equipamento de hardware mais seguro para que não ocorram problemas frequentes? Essas são perguntas bastante simples mas que já conseguem ilustrar a complexidade da escolha de um estilo. Cada estilo oferece perspectivas diferentes ao usuário no tangente ao modelo mental que ele tem da interação. Como existem muitas diferenças na maneira de executar tarefas entre *peritos* e novatos, por exemplo, nem sempre um estilo é a melhor saída para ambos os usuários... Além dos interesses do usuário, deve-se também levar em conta os interesses econômicos do banco, e mais uma porção de detalhes que são invisíveis aos olhos do usuário final.

Alguns desses bancos oferecem o auto-atendimento via *touchscreens*, no estilo baseado em menus. Para usar uma *touchscreen*, existem vários detalhes que precisam ser levados em conta. Algumas vantagens desse dispositivo são o senso de controle que oferece, a rapidez com que se pode efetuar uma função, a não necessidade de espaço extra para a interação e a sua durabilidade perante um uso contínuo [SHNEI95c]. Alguns detalhes a serem considerados são a precisão oferecida pelas telas [POTTE95] [SEARS95] e a quantidade limitada de dados a serem inseridos [PLAIS95].

É certo oferecer toda essa funcionalidade via menus? Como o software apresenta um número de funções bastante limitado ao usuário, o sistema de menus atende perfeitamente os requisitos mínimos. As chamadas das funções são rápidas e existe um bom direcionamento do usuário em relação à sua intenção de ação. Como a grande maioria das pessoas hoje se depara com esse tipo de interação (dentro de um banco), parece impossível conceber um caixa eletrônico através de linhas de comando, ou com um *mouse* disponível ao lado da tela para que o usuário interaja.

E o que acontece quando esse mesmo sistema passa a ser disponibilizado na forma de *home-banking* (banco em casa)? Ele precisa ser totalmente mapeado para outro estilo de interação, um estilo que se encaixe no que é acessível aos seus usuários: um computador pessoal, que normalmente não possui uma *touchscreen.* A mesma forma de interação não será mais possível. Entretanto, a funcionalidade que o software irá disponibilizar deverá ser a mesma.

Como fazer esse mapeamento? Existem **padrões** de interface conhecidos pelo usuário, disponíveis no mercado. Padrões fazem uso de ***guidelines*** e ambos serão abordados nas próximas seções.

2.3.10. PADRÕES

Pressman [PRESS94a] define padrões de interface como sendo módulos e objetos de HCI que podem já estar pré-empacotados para o *designe*r. A vantagem de se usar padrões está na familiaridade que o usuário encontra. Ao usar uma nova aplicação que faça uso do mesmo padrão, o usuário irá aprender mais rapidamente. Powell [POWEL90] reforça que consistência e familiaridade ajudam a diminuir as curvas de aprendizado. Após um tempo, a interface se tornará intuitiva e, portanto, muito mais produtiva para o usuário final.

Hix [HIX93d] coloca uma diferença entre padrões e guias de estilo comerciais. Para ela, padrões são documentos oficiais, disponíveis publicamente, que fornecem requisitos para o *design* de interface. São palavras muito gerais, gerais a ponto de não se saber onde os padrões estão de fato sendo seguidos. Dentre as organizações existentes, envolvidas com a padronização de interface do usuário, destaca-se a ISO, *International Organization for Standardization*[1].

Reed [REED94] comenta que diversas "forças" têm levado à padronização, entre elas:

- frustração do usuário por usar muitos sistemas inconsistentes e incompatíveis;
- interesse sobre o bem-estar e conforto do operador de computador;
- interesse do empregador em aumentar a produtividade do trabalhador;
- desenvolvedores procurando guias para aumentar eficientemente a qualidade das interfaces do usuário do software;

[1] maiores informações sobre padrões ISO relacionados ao tópico dessa pesquisa podem ser encontrados no Anexo A

- companhias dos EUA interessadas em padrões ergonômicos de sistemas sendo desenvolvidos na ISO e na Comunidade Européia.

Frente a essas "forças", Reed apresenta argumentos a favor e contra a padronização:

A FAVOR	CONTRA
• aumento da facilidade de aprendizado e facilidade de uso	• não se sabe o suficiente de usabilidade para padronizar
• aumento do conforto e bem-estar (estudos apontam que um péssimo *design* de interface pode causar estresse, ainda que essa relação não seja bem entendida)	• padrões inibem a inovação no *design* da interface (a revisão a cada cinco anos pode não fornecer suficiente agilidade para acomodação de novas tecnologias. como já foi comentado)
• auxílio na avaliação de produtos (requerendo produtos em conformidade com um padrão já publicado, organizações podem estar confiantes de que os produtos selecionados fornecem um nível mínimo de usabilidade)	• somente testes podem assegurar a usabilidade (o uso de *guidelines* e padrões podem melhorar a usabilidade de um dado produto, mas não pode lhes garantir um sistema altamente usável, como já foi comentado)
• facilidade de reuso do *design* de interface e código	

Pat [BILL94] comenta ainda que **novas tecnologias e aplicações de interface continuam a emergir**, causando uma pressão adicional aos comitês de padronização, visto que eles precisam completar o trabalho que já estão realizando, além de prestarem atenção a essas evoluções. O resultado dessas pressões é um trabalho muito pesado para o comitê, o que, infelizmente, está acontecendo ao mesmo tempo que muitas companhias estão diminuindo seu nível de suporte para atividades de padronização. Como resultado disso, diversos peritos na área de padrões de HCI estão cessando seu envolvimento com comitês. Tal fato, segundo o autor, não só diminui o progresso dos esforços em padrões como também aumenta o risco do padrão resultante não refletir a total variedade de importantes considerações de *design* de interface.

Os padrões internacionais continuam com seu problema de aplicabilidade. Os padrões não são auto-aplicáveis, pois trabalham, normalmente, o processo de desenvolvimento de algum produto, interferindo portanto na estrutura interna da empresa. Está virando uma obrigação para diversos campos da indústria e comércio ter um certificado de padronização. Para o campo de informática, existem diversas Normas, mas poucas empresas têm 'porte' (estrutura e capital) para implementá-las. Infelizmente, a aplicação de Normas é um processo que, além de custoso, leva tempo. Não se pretende avançar na discussão de padrões, mas mostrar que existe muito material disponível. Foram selecionados da ISO diversas Normas (vide Anexo A) que de alguma maneira poderiam ajudar nessa pesquisa. Não foi possível ter acesso a muitas delas, em função de custo e tempo disponível para seu estudo, além de estarem ainda em estado bastante incipiente.

Não é esse o conceito procurado para essa dissertação. Aqui, procura-se o conceito de guias de estilo comercial [HIX93d], documentos tipicamente produzidos por uma organização, disponíveis comercialmente (e.g., *Microsoft* e *Apple*). Um guia de estilos fornece um *framework* muito mais concreto para *design*. Inclui uma descrição de um estilo ou objeto de interação específico, incluindo seu *look and feel* (aparência e comportamento). Também oferece uma ajuda em quando e como usar um estilo ou objeto de interação particular.

A maioria dos guias de estilos contém seções com descrições de componentes de interação particulares, tais como janelas, menus, controles e caixas de diálogo. Também estão comercialmente disponíveis na forma de *toolkits* (discutido mais a frente). Mas Hix chama a atenção para três pontos importantes a respeito de guias de estilo comerciais: muitos deles (e suas *toolkits* correspondentes), na sua concepção, não tiveram grande participação de profissionais de fatores humanos; sozinhos, eles não são suficientes para garantir usabilidade da interface; tendem a oferecer guias específicas em *o quê* um objeto de interação particular deveria parecer, e *como* deveria se comportar; entretanto, nem sempre definem quando usar aquele objeto de interação particular, que é uma das mais difíceis decisões do *design.*

2.3.11. GUIDELINES

Segundo Newman [NEWMA95] *guidelines* são estratégias de solução possíveis. Cada uma delas possui um contexto ou domínio dentro da qual se aplica. A aplicação de uma *guideline* para um problema é a aplicação de uma heurística, baseada em experiências passadas bem sucedidas.

Guidelines têm diferentes papéis no *design*, bem como limitações, o que indica que sua aplicação precisa ser muito cuidadosa. A maioria das limitações tem a ver mais com a maneira como se selecionam as mesmas, do que com o seu conteúdo. Como elas são descritas com um número mínimo de palavras, pode-se perder o seu significado no todo.

Além desse problema, ainda existe a possibilidade de se aplicar múltiplas *guidelines* para o problema que se está tentando resolver. Seguramente a ordem em que se aplicar essas *guidelines* irá afetar o resultado. Segundo Newman, esses conflitos existem porque as *guidelines* derivam de diversas teorias e experiências e têm seu escopo de aplicação limitado. Um exemplo poderia ser:

- *um menu hierárquico ou cascata é um submenu (também conhecido como menu filho) vinculado ao lado direito de um item de menu...*

Day [DAY93] acrescenta que muitas dessas *guidelines* são compêndios de bons princípios de *design* de interface que foram estabelecidos através de estudos empíricos ou por consenso de *peritos*. Ainda que esses documentos não sejam considerados padrões, uma vez que não foram aprovados por organizações responsáveis por padrões, são usados em larga escala e, portanto, considerados como *padrões de facto*.

Hix [HIX93d] afirma que *guidelines* não são apenas senso comum. *Guidelines* são frequentemente contraditórias, exatamente porque derivam de teorias diferentes, como já foi dito anteriormente. Uma diferença entre padrões e *guidelines* está em que padrões são, ao menos teoricamente, obrigações à interface do usuário, enquanto que *guidelines* não passam de sugestões. Elas são tão gerais na sua aplicabilidade que requerem uma grande quantidade de interpretação para serem úteis.

2.4. *DESIGN* DE INTERAÇÃO VISUAL

Uma das subáreas de HCI é o *design*[2] de interação visual [WADLO93] [WADLO94a] [WADLO94b] [GONZA95] [POWEL90] [TUFTE83]. Sua importância é percebida através da forma como as funções de uma aplicação são apresentadas ao usuário, por exemplo, o modelo mental da interação (e.g., nomes representativos das funções para o usuário, sequência de execução de tarefas e todos os seus passos) e a representação visual dessa interação para o usuário (e.g., uso adequado de objetos, quantidade de objetos na tela, relação entre objetos).

Durante o *design* de interação visual, o mau uso dos objetos visuais (e.g., janelas, ícones, menus) disponíveis para a interação e suas características (e.g., quantidade, tamanho, cor, localização) podem interferir na comunicação entre o humano e o computador, dificultando a compreensão da informação que precisa ser passada ao usuário.

---

[2] Manter o uso da palavra *design* ao invés traduzi-la por "projeto" não é uma mera questão de tradução. Isso deve-se ao fato de que essa palavra, no sentido original, passa um sentido mais amplo, mais artístico, mais humano, algo não tão sistemático como o que a engenharia de software prega. Para entender melhor o conceito de *design*, pode-se tomar por exemplo a arquitetura: quando um arquiteto faz o *design* de uma casa ou escritório, uma estrutura está sendo especificada. Mais significativamente, padrões de vida de seus habitantes estão sendo delineados. Pessoas são tomadas como habitantes, mais que usuários das construções. O objetivo do *designer* é situar seu trabalho no mundo do usuário. Por isso, o *design* precisa ser consciente e manter os interesses humanos como foco, além de ser essencialmente criação (não pode ser completamente reduzida a passos padrões) e comunicação [WINOG96]. Em outras palavras, no *design*, deve-se buscar a sensibilidade ao contexto humano em toda a sua riqueza e variedade.

Como a comunicação é algo que acontece naturalmente entre duas pessoas, se uma não responde não há diálogo. O mesmo deveria acontecer na computação. Mas o que existe nas interfaces atuais é um diálogo 'forçado", ao qual o usuário, como receptor da mensagem, precisa sempre responder, para que a dita interação continue ocorrendo. A aplicação não "raciocina" em cima de dados entrados para ações não programadas, não extrai significado desses dados e portanto não pode manter um verdadeiro diálogo. O próprio nome diz: interface é a interconexão entre dois sistemas. Esse **conceito de interface não necessariamente envolve interação** (inter + ação) entre dois sistemas.

Em HCI, a definição de interação parece impor a coexistência de humanos e computadores [GONZA95]. A idéia de ação recíproca põe um humano e um computador no "mesmo nível de capacidades". Desta perspectiva, a palavra interação pode não refletir a maneira como humanos e computadores trabalham juntos atualmente, mas como se quer trabalhar com o computador, no futuro!

O *design* de interação requer o estudo do humano, computador, interface, linguagem de comunicação e situações nas quais a comunicação pode acontecer. Os *designers* de interação visual devem saber como o sistema humano trabalha, como a informação visual é processada, e como o ser humano reage a diferentes tipos de informação visual. O autor ainda sugere que "não é suficiente ser um bom desenhista ou artista... *design* de interação visual significa entender a mente humana e o processamento do computador, significa produzir condições ótimas para a comunicação humano computador, significa transformar arte em ciência".

O conceito "*design* de interação visual" existe desde o início dessa década, quando nos CHI'90, CHI'91 e CHI'92 (evento internacional voltado para a discussão da área de HCI, vide Anexo A) foram discutidos o tema. No CHI'92 diversos *designers* interferiram na criação de um ambiente maior de discussão. Em 1993, foi publicada a primeira seção homônima ("*Visual Interaction Design*") no *SIGCHI Bulletin*, uma das maiores fontes de HCI em todo o mundo. Também existe uma lista de discussão na Internet com esse fim (vide Anexo A). Tais fatos comprovam a importância que essa subárea de HCI vem adquirindo com o passar dos anos. Nesse trabalho, **toda vez que se escrever o termo interface, quer-se na verdade passar o conceito de interação visual**, uma interação humano-computador que será estimulada através de uma interface consistente.

2.5. ENGENHARIA DE SOFTWARE E INTERAÇÃO HUMANO-COMPUTADOR

Coutaz [COUTA94] descreve que as áreas de ES e HCI almejam desenvolver sistemas de computação úteis e usáveis, mas que atualmente elas o fazem sem dividir uma base científica comum. Como resultado, existe uma diferença gritante de terminologia desnecessária. Segundo o autor, questões fundamentais como usabilidade e processos de desenvolvimento, são endereçadas pelas duas comunidades, com diferentes prioridades, competência e sucesso.

Hefley [HEFLE94] desenvolveu um estudo que destaca a integração das práticas de HCI com as de ES. Nesse estudo, o autor revelou que o projeto e desenvolvimento de HCI têm evoluído para uma disciplina de engenharia plena, para obter usabilidade no sistema. Avanços nessa área são: engenharia de interface do usuário (focando no processo de desenvolvimento do software) e engenharia de usabilidade (focando no produto de software). Ainda assim, muitos métodos e projetos de HCI são mal definidos, não tendo alcançado o nível de disciplina e procedimentos bem definidos, o que é evidente em outras disciplinas de engenharia. Esses métodos são frequentemente *ad hoc*.

Hefley afirma que para desenvolver sistemas interativos de alta qualidade, é necessária a integração de engenharia de HCI e Engenharia de Software, durante todo o ciclo de vida do sistema, como pode ser visto na Figura 14. O autor ainda realça que a prática futura irá abranger projetos para usabilidade (centrado no usuário), projetos para utilidade (centrado na tarefa e organização), e projetos para qualidade de vida no trabalho (*quality of worklife*, centrado em considerações sociais, levando em conta pessoas, grupos, organizações e sociedade).

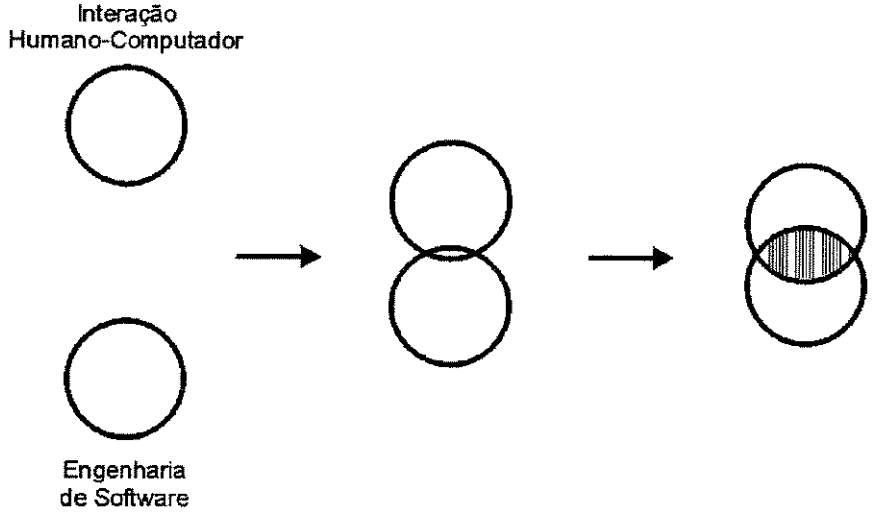

Figura 14: Relação entre as áreas

Chimera [CHIME94a] levanta necessidades para as duas áreas. Segundo ele, HCI precisa de ES para construir interfaces de usuário com maior rapidez. Também para que elas sejam mais confiáveis, mais reusáveis e mais manuteníveis. Por outro lado, a ES precisa de HCI para produzir interfaces altamente usáveis e atraentes. Existe aí uma relação que precisa ser melhor explorada.

Uma das formas de explorar essa relação pode ser através de ferramentas CASE, assunto do próximo capítulo.

3. FERRAMENTAS *CASE* PARA *DESIGN* DE INTERFACE

3.1. INTRODUÇÃO

Antes de falar sobre as ferramentas *CASE* (*computer-aided software engineering*) para *design* de interface, será apresentada uma visão mais ampla das ferramentas CASE, sob o enfoque da engenharia de software, como o próprio nome sugere. As ferramentas têm por objetivo auxiliar os engenheiros de software durante um processo de desenvolvimento. Por se tratar de uma tecnologia bastante recente, apresenta-se como uma solução pontual para um determinado momento do ciclo de desenvolvimento, não permitindo medir "integralmente" seus verdadeiros benefícios. Além dos fatores técnicos existentes, são apresentados alguns problemas referentes à capacidade de uma empresa de desenvolvimento em organizar seu processo de seleção e adoção de uma ferramenta CASE.

Voltando à parte técnica, a seção seguinte aborda realmente o tema pretendido: ferramentas *CASE* para *design* de interface. São apresentados suas vantagens e definições. Dentre as ferramentas, é destacada a importância das ferramentas construtoras de interface e é destacada a importância da existência da característica de independência de plataforma da interface gerada. São apontados alguns critérios para avaliar as ferramentas e, no final, é apresentada a definição de outra categoria de ferramenta, não voltada ao *design* de interface, mas à sua avaliação e crítica.

3.2. FERRAMENTAS *CASE*

Pressman [PRESS94a] introduz o assunto com uma história bastante ilustrativa sobre o uso de ferramentas CASE. É a história do sapateiro, que de tão ocupado fazendo sapatos para os outros, não tinha tempo para fazer os seus próprios. A analogia é direta! Os engenheiros de software têm uma demanda de produtividade tão grande que mal têm tempo de pensar em ferramentas que os ajudem. Essa é a função principal das ferramentas CASE.

Entretanto, segundo Oakes [OAKES92], é impreciso medir os benefícios da tecnologia CASE, devido à sua natureza diversa e ao seu relativo curto tempo de uso. Também existe uma grande diferença nas práticas de adoção das mesmas, e uma falta muito grande de registros relatando ganhos e perdas com CASE.

O autor comenta que os benefícios que se encontram na literatura relatam melhorias na comunicação e na documentação. A comunicação melhora tanto para os engenheiros do projeto entre si quanto para entre os engenheiros do projeto e o comprador, havendo ganhos de produtividade (bastante variáveis) e um reforço no uso de uma metodologia de projeto e padronização.

Pressman sugere uma **taxonomia** para ferramentas *CASE* pelo critério de função: planejamento de sistemas comerciais, gerenciamento de projeto, suporte, análise e projeto, programação, integração e teste, prototipação, manutenção e *framework*. De todas essas funções que uma *CASE* pode ter, tem-se por objetivo ampliar o conceito de ferramenta de prototipação, mais especificamente uma ferramenta voltada ao *design* de interface (Seção 3.3).

3.2.1. SOLUÇÕES PONTUAIS X INTEGRAÇÃO

O problema das ferramentas *CASE* reside em elas serem ferrramentas individuais, soluções pontuais, que cobrem pontos específicos do ciclo de desenvolvimento do software. Tanto Pressman quanto Oakes [PRESS94c] [OAKES92] comentam que se está caminhando em direção a *IPSEs* (*Integrated Project Support Environment*). Mesmo que benefícios possam ser derivados de ferramentas individuais que direcionam atividades separadas da engenharia de software, o poder real do *CASE* só pode ser alcançado pela integração [PRESS94b]. Os benefícios são bem maiores e, dentre eles pode-se citar: transferência "suave" de informação (modelos, programas, documentos, dados), aumento do controle do projeto, alcançado por um melhor planejamento, monitoria e comunicação, além da melhoria da coordenação entre os membros da equipe que estão trabalhando num amplo projeto de software.

É bastante claro que tudo que oferece benefícios também oferece desafios... A integração entre ferramentas demanda representação consistente da informação da engenharia de software, interfaces padronizadas entre as ferramentas e um mecanismo homogêneo para comunicação entre o engenheiro de software e cada ferramenta.

3.2.2. ADOTANDO UMA FERRAMENTA *CASE*

Além dos desafios técnicos levantados anteriormente, existem muitos outros, cobertos por Lucca [LUCCA92] e Oakes [OAKES92]. O primeiro deles é não poder considerar que o simples uso de ferramentas *CASE* ou *IPSE* irá resolver todos os problemas que uma empresa enfrenta ao desenvolver um software. Há pesquisas que apontam para **sucessos e fracassos** no uso dessas ferramentas. O determinante desse resultado é a somatória de diversos fatores que influenciam a introdução da ferramenta na empresa. E como as empresas não são iguais entre si, é fundamental que as ferramentas a serem introduzidas sejam "personalizáveis" dentro da organização, permitindo a incorporação de novas metodologias criadas pelo usuário.

É necessário aplicar uma **metodologia de avaliação e seleção** de ferramentas, de forma a obter um diagnóstico real do problema da empresa e adequar a ferramenta ideal para essa realidade. Toda empresa de desenvolvimento de software possui uma metodologia de desenvolvimento, resultado da somatória de procedimentos gerenciais com métodos técnicos, mais ferramentas automatizadas. Em outras palavras, as ferramentas não são um fim em si mesmas... Elas só têm razão de ser se estiverem inseridas dentro de uma metodologia. A metodologia de desenvolvimento propõe diferentes ciclos de vida, que são compostos de diferentes atividades e fases integradas. As ferramentas *CASE* deveriam acompanhar todas essas fases, sendo maleáveis o suficiente para acompanhar a metodologia de desenvolvimento da empresa.

Lucca complementa seu texto com algumas porcentagens, das quais uma parece ser bastante interessante: enquanto a evolução do hardware em termos de velocidade, poder e capacidade está em torno de 30% ao ano, o aumento da produtividade do desenvolvimento do software não ultrapassa 7% ao ano [LUCCA92]. Essa baixa porcentagem parece ser um bom estimulante para o uso de ferramentas CASE.

Oakes sugere que o sucesso ou falha do esforço de adesão de um *CASE* depende amplamente da habilidade da organização em gerenciar custos a curto e longo prazo. Para tanto, ele acaba por propor, como receita de sucesso, um **processo de adoção** de ferramenta CASE, que envolve seis estágios: consciência, comprometimento, seleção, tentativa de implementação, estratégia de implementação e rotinização.

O estágio de *consciência* tem a ver com tornar-se versado em literatura CASE, participar de *workshops* e seminários, solicitar informação de outras organizações que adotaram essa tecnologia. *Comprometimento* tem a ver com a parceria que precisa existir da parte gerencial para com os que serão diretamente afetados pela ferramenta. *Seleção* envolve o desenvolvimento de uma estratégia que aborda a necessidade de curto e longo prazo da organização. *Tentativa de implementação* é um projeto piloto que deve existir para testar a ferramenta, com alocação real de recursos, pessoal, tempo e dinheiro. O estágio de *Estratégia de implementação* inclui o desafio de migrar para uma *CASE* e a integração desta ferramenta ao ambiente, evitando que ela afete em demasia os elementos do ambiente (pessoal, processos e métodos). Por último, o estágio de *rotinização* tenta prevenir a falta de doutrina dos novos empregados em relação ao sistema, e tenta evitar a subestimação de recursos necessários para suportar o contínuo uso dessa ferramenta *CASE* (e.g., treinamento de pessoal).

3.3. FERRAMENTAS *CASE* PARA *DESIGN* DE INTERFACE

Nessa seção serão abordadas ferramentas *CASE* voltadas ao *design* de interface. Não existe um conjunto de ferramentas padrão que seja completamente aceito no mercado, devido ao fato dessas ferramentas contemplarem diversas funcionalidades, como já foi dito na seção anterior. Para o caso da categoria de ferramentas *CASE* voltada para *design* de interface, faz-se necessário apresentar mais especificamente suas vantagens e definições. Também são abordadas ferramentas de *design* para avaliação e crítica e alguns critérios para selecionar ferramentas de *design* de interface. No final da seção é apresentada a definição de programas de plataforma independente.

3.3.1. VANTAGENS

Myers [MYERS95] [MYERS97] afirma que um software para interface do usuário é frequentemente grande e complexo, sendo difícil de implementar, encontrar seus defeitos (*debug*) e modificar. Algumas porcentagens mostram que num projeto de software estudado, a porcentagem de código usada para interface atingiu entre 48% e 50% do tempo de implementação. Noutro estudo, a utilização de um ambiente para ajudar a construir interfaces, diminuiu em 83% a quantidade de linhas de código escritas, e em algumas aplicações sua construção se deu em 1/10 (um décimo) do tempo esperado.

São números bastante animadores! E essas vantagens não se resumem a esses dados. Outras vantagens de usar ferramentas [MYERS97] [HIX93] [DAY93] [SHNEI92] são: independência da interface do usuário em relação à aplicação, possibilidade de interfaces avançadas, facilidade de modificação, reuso de componentes de interface, consistência entre as múltiplas interfaces e envolvimento de diversos especialistas no desenvolvimento da interação, tais como engenheiros de fatores humanos, artistas gráficos e outros. A interface também será mais estruturada e modular, resultando em maior reusabilidade e confiabilidade.

3.3.2. DEFINIÇÕES

Existe um grande "conflito" de definições sobre essa categoria de ferramentas CASE, como será visto a seguir. Autores como Pressman [PRESS94c] preferem descrever as ferramentas de interface como ferramentas de implementação, definidas apenas como ***toolkits*** ou **UIDS** (*user interface development systems*, sistemas de desenvolvimento de interface do usuário). Essas ferramentas fornecem módulos ou objetos que facilitam a criação de janelas, menus, interação de dispositivos, mensagens de erros, comandos e muitos outros elementos de um ambiente interativo. *Toolkits* estão sendo substituídas por ferramentas de prototipação de interface, que permitem a criação de interfaces sofisticadas voltadas para padrões que foram adotados para o software.

Hix [HIX93e] prefere fazer uma classificação simples e abrangente: **ferramentas de suporte aos estilos de interação**, ***toolkits*** e **sistemas de gerenciamento de interface do usuário**. A autora argumenta que, como existem centenas de ferramentas disponíveis, tanto comerciais como de grupos de pesquisa, e que a evolução das mesmas é prevista, a descrição detalhada de cada uma será provavelmente imprecisa quando chegar às mãos do leitor.

As ferramentas de suporte ao estilo de interação são sistemas interativos que forçam o uso de um particular *look and feel*, baseado num estilo e padrão específico (e.g., OSF/Motif® e OpenLook®). Normalmente não é permitido ao desenvolvedor da interface muita flexibilidade na alteração dos componentes de interação. Sua grande vantagem é a consistência no estilo de interação.

Toolkits são bibliotecas de rotinas (módulos de código) usados por programadores para implementar características de baixo-nível de interface. Essas rotinas são chamadas de dentro de UIMS (*user interface management systems*, sistemas gerenciadores de interface do usuário), ou de qualquer outro programa, oferecendo código para diversas técnicas de interação. Interagem com o código computacional via *callbacks*, que são invocações de procedimentos definidos pelo programador e executados pela aplicação em resposta a alguma ação tomada pelo usuário, em tempo de execução. O problema está em que as interfaces são compostas por centenas de *callbacks*, fazendo do código algo difícil de manter. Segundo a autora, a função das *toolkits* é bastante limitada, comparada a outras ferramentas. Não oferecem suporte para qualquer atividade no processo de desenvolvimento da interface, exceto na implementação. Normalmente são construídas em cima de sistemas de janelas (e.g., X Window System®). Através de um sistema de janelas, um usuário consegue interagir com diversas tarefas e/ou aplicações, cada uma em uma janela diferente.

Sistemas de gerenciamento de interface são uma série integrada de programas interativos para todo o processo de desenvolvimento de interface, incluindo design, representação, prototipação, execução, avaliação e manutenção da interface. Deveria suportar, entre outras coisas, mecanismos de suporte em tempo de execução para sentir as ações do usuário em objeto na tela e fornecer *feedback* na própria tela. Também precisa de um gerador para produzir código da interface a partir de definições da interface, assim como um banco de dados para armazenar tais definições e o código gerado.

Shneiderman [SHNEI92] divide as ferramentas de suporte em **prototipadores**, ***toolkits*** e **UIMS** (*user interface management systems*, sistemas gerenciadores de interface do usuário).

Prototipadores permitem ao *designer* fazer o *layout* da tela, com movimentos do cursor e cliques, marcar regiões para seleção, destaque ou entrada de dados. São ótimos para levar a frente contratos, pois dão ao cliente uma "grosseira" idéia do que o sistema final parecerá. Entretanto, é frustrante ao *designer* implementar as telas construídas durante alguns meses a partir do zero, já que essas ferramentas não geram códigos.

Toolkits são bibliotecas de rotinas destinadas ao controle de alguns widgets, tais como janelas, barras de rolagem, menus *pull-down* e *pop-up*, campos de entrada de dados, botões e caixas de diálogo. São destinadas a programadores *peritos*, dada a sua complexidade...

UIMS possibilitam aos *designers* criar uma interface de usuário completa sem precisar programar numa linguagem de programação tradicional. Os usuários podem usar uma linguagem de programação para implementar funções adicionais como pesquisas em bancos de dados, comunicação de rede, mas a interface pode ser criada, revisada e mantida em linguagem de alto-nível. A principal idéia das UIMS é independência da interface, ou seja, separar o *design* lógico da interface dos seus aspectos de implementação, tal que a modificação em um não influa no outro. Seu uso tende a gerar maior consistência em função do uso de *widgets*, além de proporcionar um controle de terminologia.

Day [DAY93] prefere ainda outra definição: ferramentas **sem geração de códigos** e **com geração de códigos**. Essas ferramentas permitem a ligação de telas através do clicar do *mouse* ou pressionar teclas em diferentes regiões da tela. Através desse mecanismo, pode-se simular interfaces complexas. Esse mecanismo prevê não apenas o *look* da tela, mas também o diálogo usuário-sistema, ou seja, como o usuário interage com o sistema.

Nas ferramentas sem geração de código, normalmente esses pacotes são fáceis de aprender e usar, e o protótipo pode ser modificado rapidamente. Uma desvantagem clara é: o protótipo nunca realmente vira parte do sistema final. O esforço gasto em criar o protótipo não é revertido para o tempo de codificação da interface.

Nas ferramentas com geração de código incluem-se as UIMS, construtores de interface e *toolkits*. A autora não se preocupa em definir cada uma dessas subcategorias. Apenas sugere que elas diferem-se das anteriores principalmente por permitir ao usuário construir a interface de maneira interativa, tornando-a parte do sistema produzido. Essas ferramentas permitem a separação da interface do código funcional, o que possibilita diversas vantagens para o reuso do código. Normalmente elas são mais difíceis de usar que as anteriores, dado o aumento de complexidade nas funcionalidades oferecidas.

Como pode ser visto até o momento, existem diversas classificações a respeito das ferramentas que auxiliam o *designer* na construção de interfaces para o usuário. Além dessas que já foram citadas, existe uma, que para essa dissertação, oferece um campo de visão mais amplo. O trabalho de Myers [MYERS97] oferece a descrição mais completa das categorias e suas definições. Myers define cinco níveis, como pode ser visto abaixo:

Application
Higher-level Tools
Toolkit
Windowing System
Operating System

Figura 15: Componentes de software de interface do usuário.

Sistema de Janelas é o pacote que auxilia o monitor do usuário e controla diferentes contextos separando-os fisicamente em diferentes partes. Já foi definido por Hix e, por não fazer parte do escopo dessa dissertação, não será aprofundado.

Toolkit é uma biblioteca de widgets, que normalmente inclui menus, botões, barras de rolagem, campos de texto de entrada, etc. Inclui a definição de todos os outros autores já citados.

Ferramentas de Alto-Nível tornam o processo de produção de software para interface do usuário mais fácil. Há inúmeros tipos de ferramentas que podem agir em diferentes fases do processo de desenvolvimento da interface. Dividem-se em diversos tipos, baseados no seu formato de especificação, como pode ser visto na tabela a seguir:

Tabela 1: Formas de especificar a interface.

SPECIFICATION FORMAT	EXAMPLES
Language Based	
State Transition Networks	VAPS
Context-Free Grammars	YACC, LEX, Syngraph
Event Languages	Sassafras, EET, HyperTalk
Declarative Languages	Cousin, Open Dialog, Motif UIL
Constraint Languages	Thinglab, C32
Screen Scrapers	Easel
Database Interfaces	Oracle
Visual Programming	LabView, Prograph
Application Frameworks	MacApp, Unidraw, Amulet, Andrew, OLE, OpenDoc
Model-Based Generation	MIKE, UIDE, ITS, Humanoid, MASTERMIND
Interactive Graphical Specification	
Prototypers	Bricklin's Demo, **Director**
Cards	Menulay, HyperCard
Interface Builders	DialogEditor, NeXT Interface Builder, **Visual Basic**, UIM/X
Data Visualization Tools	Data Views
Graphical Editors	Peridot, DEMO, **Marquise**

Desses formatos de especificação, destaca-se a especificação Gráfica Interativa (*Interactive Graphical Specification*), que permite que a interface do usuário seja definida, pelo menos parcialmente, colocando objetos na tela, usando dispositivo de apontamento (e.g., *mouse*). Muitos desses sistemas podem ser usados por não-programadores, facilitando, portanto, seu uso por psicólogos, *designers* gráficos e especialistas de interface no processo de *design* da interface, quando as ferramentas estão sendo utilizadas.

Dividem-se em quatro categorias: ferramentas prototipadoras, que suportam uma sequência de cartões, construtores de interface e editores gráficos para aplicações específicas. Dessas, a que interessa para essa dissertação é a categoria de ferramentas construtoras de interface, tomando por exemplo o VB (*Visual Basic*), utilizado para gerar o estudo de caso (Anexo D). Como parâmetro de comparação, são apresentadas as suas duas categorias anteriores.

3.3.2.1 FERRAMENTAS PROTOTIPADORAS

Permitem ao *designer* construir rapidamente alguns exemplos de como as telas irão parecer no programa. Frequentemente essas ferramentas não podem ser usadas para criar as reais interfaces do programa, uma vez que elas não geram o código da interface. Na maioria dos prototipadores, nenhuma *toolkit* é utiliizada, significando que seu uso normal envolve a criação prévia de diferentes *designs* da interface para sua total reimplementação num outro sistema.

3.3.2.2 CARTÕES[3]

Programas gráficos limitados a interfaces que são apresentadas na maioria das vezes como uma sequência de páginas estáticas, às vezes chamadas de *frames* ou *forms*. Cada página contém uma série de *widgets* que podem causar a transferência para outros cartões.

3.3.2.3 CONSTRUTORES DE INTERFACE

Permite ao *designer* criar caixas de diálogo, menus e janelas que são parte de uma "extensa" interface de usuário, selecionados de uma biblioteca de *widgets* pré-definida, e colocá-los na tela usando um *mouse*. Geralmente também há um suporte para o sequenciamento, tal como o tratamento de sub-diálogos, quando um botão em particular é pressionado. Como os construtores de interface fazem uso de uma biblioteca de *widgets* pré-definida, podem ser usados para construir aplicações reais. A grande maioria gera códigos em C, que podem ser compilados com o código da aplicação.

Ainda que tornem mais fácil o *layout* de caixas de diálogo e menus, essa é só uma parte do problema de *design* de interface. Essas ferramentas não fornecem nenhuma guia em relação a como criar boas interfaces, oferecendo ao *designer* bastante liberdade. Isso permite que interfaces ruins sejam construídas facilmente, reforçando a crença errada de que a interface do usuário é meramente a superfície perceptível do sistema [COUTA94].

Outra desvantagem é não conseguir manipular *widgets* que mudam dinamicamente. Um exemplo disso é a mudança de *layout* de um menu (ativação/desativação de funções disponíveis ao usuário frente à seleção de um objeto específico da aplicação), que precisa ser programada diretamente em código.

Pelo fato de gerar código, uma característica muito valiosa dessa ferramenta seria a independência de plataforma da interface resultante, como pode ser visto a seguir.

3.3.3. PROGRAMAS DE PLATAFORMA INDEPENDENTE

Myers [MYERS97], em seu estudo já levantou a questão das ***virtual toolkits*** (*toolkits* virtuais), desenvolvidas para esconder as diferenças entre várias *toolkits*, fornecendo *widgets* virtuais que podem ser mapeadas nas *widgets* de cada *toolkit*. Outro nome para essas ferramentas é sistemas de desenvolvimento entre-plataformas (*cross-platform*).

Chimera [CHIME94b] trabalha mais esse conceito e define que o objetivo de se criar programas de plataforma independente é permitir que **o esforço total seja menor que a soma dos esforços necessários para as plataformas individuais**. A idéia é que a interface se comporte como um camaleão, variando seu *look and feel*, dependendo do ambiente em que se encontrar. Tal comportamento é complicado em função de haver funcionalidades em um *look and feel* que não existem em outro. Para esse problema são sugeridas duas soluções: não oferecer a funcionalidade em nenhum dos *look and feel* suportados ou fazer um conjunto de abstrações de nível mais alto para os casos. Tais abstrações podem adicionar funcionalidade para um *look and feel*, mas nunca violá-lo.

[3] o nome "cartão" origina-se no *Hypercard* da *Macintosh*

A principal vantagem para o uso de programas de plataforma independente é clara: existe apenas uma API (*application program interface*, interface do programa de aplicação) e, portanto, somente um modelo de alocação de memória, um modelo gráfico e um modelo de evento para aprender.

3.3.4. CRITÉRIOS PARA AVALIAÇÃO DE FERRAMENTAS

É muito difícil avaliar e comparar ferramentas de desenvolvimento de interface [HIX93e], pois elas são aplicações de software bastante complexas e relativamente novas (menos de uma década). Além disso, diferentes tipos de aplicações de software chamam a si mesmas de ferramentas de desenvolvimento de interface.

Existem **critérios gerais para se avaliar as ferramentas** [MYERS97] [HIX93e] [BASS94] [KELLER94]. Podem ser divididos em critérios para avaliação da própria ferramenta e da interface resultante do uso da ferramenta. Além desses, pode-se levantar ainda outros aspectos.

Com relação à ferramenta, destaca-se:

- funcionalidade (capacidades da ferramenta)
- usabilidade (as ferramentas precisam ter interfaces de alta usabilidade)
- habilidade de produzir interfaces de manipulação direta via manipulação direta
- estilos suportados (relacionado à funcionalidade; disponibilidade de *widgets*)
- customização, incluindo adição ou modificação, das *widgets*
- criação de objetos dinâmicos (suporte ao desenvolvimento de objetos de interação dinâmicos, em tempo de execução; normalmente as ferramentas não suportam)
- suporte à avaliação formativa (ao longo do desenvolvimento) e refinamento iterativo
- tipo de estrutura de controle e *callbacks* (importante no acoplamento a outro software)
- custo e documentação (variação muito grande)
- suporte ao comprador (dependendo da dificuldade de uso da ferramenta, pode haver cursos de treinamento)
- profundidade (quanto da interface a ferramenta cobre)
- abrangência (quantos estilos de interação são suportados)
- eficiência para *designers* (rapidez com que se criam as interfaces)
- robustez (minimização dos *bugs* na ferramenta)

Com relação à interface resultante do uso da ferramenta, destaca-se:

- portabilidade (interface resultante operando em múltiplas plataformas)
- qualidade das interfaces resultantes (auxílio da ferramenta ao *designer* em avaliar a interface e melhorar sua qualidade)
- performance em tempo de execução da interface resultante
- customização da interface do usuário

Outros aspectos:

- falta de desenvolvedores experientes no mercado
- falta de conhecimento
- falta de padronização
- falta de tempo para investigar ferramentas
- medo de obsolescência das ferramentas
- custo extra envolvido no aprendizado
- medo de adotar uma abordagem não padronizada

Esses critérios têm a função de mostrar ao leitor a quantidade de informações necessárias para avaliar uma ferramenta *CASE* para desenvolvimento de interface.

3.4. FERRAMENTAS DE *DESIGN* PARA AVALIAÇÃO E CRÍTICA

Todas as definições de ferramentas até o momento envolvem a parte de desenvolvimento da interface, inserida em algum processo de desenvolvimento. Entretanto, pode-se trabalhar o conceito de ferramentas que sejam capazes de avaliar a interface criada após seu desenvolvimento, fornecendo ao designer, sugestões para sua melhoria. Shneiderman [SHNEI92] apresenta pacotes de avaliação como uma alternativa a ser trabalhada por desenvolvedores de interface. Tais ferramentas deveriam ser capazes de avaliar a própria interface, baseadas em padrões e *guidelines* propostos e aceitos amplamente. Um exemplo que Shneiderman oferece é *Tulli's Display Analysis Program*, um programa de análise de tela (alfanumérica, não gráfica). Segundo o autor, essas ferramentas serão mais atrativas à medida que as ferramentas voltadas para interface se tornem mais difundidas.

Esse conceito de ferramentas de avaliação de interface é extremamente importante! O foco dessa dissertação, métricas objetivas para consistência de interface, tem por objetivo fornecer uma base de métricas que, ao serem implementadas, possam fornecer o conceito de uma ferramenta para avaliação e crítica da interface.

4. MÉTRICAS OBJETIVAS PARA CONSISTÊNCIA DE INTERFACE

4.1. INTRODUÇÃO

O título desse capítulo aborda o foco dessa dissertação: métricas objetivas para consistência de interface. Para compreendê-las, é necessário assimilar outros conceitos. Para tanto, apresenta-se a motivação e a diferença entre métricas objetivas e subjetivas. Depois são apresentados os conceitos de interface, objetos da interface, contextos e categorias, além da estrutura das métricas. Após essa parte, são apresentadas as métricas objetivas (em diversas tabelas), resultado do cruzamento entre contextos e categorias.

4.2. MOTIVAÇÃO

O levantamento de métricas foi inspirado no trabalho de [MAHAJ97], cujas métricas foram coletadas a partir de padrões, *guidelines* e guias de estilo comercial (conceitos já trabalhados no capítulo anterior), provenientes de pesquisadores do HCIL (Laboratório de HCI da Universidade de Maryland) e de pesquisadores da GE (General Electric). Essas métricas encontram-se em anexo [Anexo A].

O principal motivo de trabalhar com as métricas propostas por Mahajan foi mostrar uma forma diferente de estruturar métricas, baseadas em objetos reais de interface, na forma como eles são concebidos por um *designer* de interface.

Essa abordagem é bastante recente, visto que não foram encontrados trabalhos semelhantes, e tem uma vantagem bastante clara: garantir a consistência da interface em um estágio prévio à sua conclusão, auxiliando o *designer* durante a fase em que a interface está sendo desenvolvida, o que pode diminuir três fatores críticos num processo de desenvolvimento de software que envolva avaliação de interface: tempo, custo e pessoal capacitado.

Outro motivo para essa dissertação foi a necessidade de compreender diversos pontos que ficaram incompletos no trabalho de [MAHAJ97], tais como:

- categorias: as categorias em que as métricas estão divididas estão presentes nesse artigo, mas não foram definidas, somente citadas (layout espacial, alinhamento, uso de cores, fonte, obtenção de atenção, etc);
- categorias x métricas: as métricas não foram apresentadas sob as categorias, o que não deixa claro ao leitor quais métricas pertencem a quais categorias;
- métricas x "submétricas": o autor comenta que algumas métricas foram divididas, mas não deixa claro quais foram, sendo difícil inferir quais métricas são "submétricas";
- contextos: as métricas não definem contextos, o que causa dúvidas em relação à sua aplicação, ou seja, não fica claro onde exatamente as métricas podem ser aplicadas; por exemplo, em um único objeto ou entre dois objetos do mesmo tipo, ou ainda entre objetos de tipos diferentes.

Portanto, essa dissertação tem o fim de auxiliar a complementação desse trabalho, propondo outros conceitos e uma estrutura diferente de apresentação das métricas, além de diversas outras métricas.

4.3. MÉTRICAS OBJETIVAS X SUBJETIVAS

Todos os objetos de interface possuem características específicas que são evidenciadas por seus atributos. As métricas são coletadas a partir desses atributos presentes na interface. No contexto dessa dissertação, propõe-se a divisão das métricas em dois grupos: métricas objetivas e métricas subjetivas.

As **métricas objetivas** podem ser parametrizadas porque são **quantitativas** e podem possuir valores padrão pré-estabelecidos pelo designer. São independentes do julgamento do *designer* no momento da coleta dos dados, o que permite **coleta automática**. São métricas voltadas para avaliações objetivas (conceito oferecido no segundo capítulo).

As **métricas subjetivas** não podem ser parametrizadas porque são **qualitativas**, ou seja, não podem possuir valores pré-estabelecidos pelo designer. São dependentes do julgamento do *designer* no momento da coleta dos dados, o que não permite coleta automática. São métricas voltadas para avaliações subjetivas (conceito oferecido no segundo capítulo).

Essa divisão pode ser percebida em qualquer objeto, como por exemplo no objeto barra de ferramentas de uma aplicação desenvolvida para o ambiente Windows. Nesse objeto, suponha, por exemplo, que existe um item da barra de ferramentas que chama a função "*Open*", uma função bastante conhecida no ambiente Windows. Desse item da barra, pode ser extraída a seguinte métrica: representação icônica. Essa métrica poderia ter uma escala qualquer cuja pontuação estaria relacionada ao grau de compreensão do usuário em relação ao ícone, a representação semântica necessária para o usuário, frente à função "*Open*" dessa aplicação. Outra métrica poderia ser: ícone padrão. Essa métrica seria uma verificação, baseada numa lista padrão, que checasse o ícone utilizado no item da barra que chama a função "*Open*" com o mesmo utilizado em "todas" as aplicações destinadas ao ambiente Windows.

Na primeira métrica é óbvia a dependência de um julgamento para a coleta do dado, o que impossibilita sua coleta automática. É uma métrica subjetiva. Na segunda métrica, a conferência do ícone com uma lista padrão pode ser feita automaticamente, independente de julgamento no momento da coleta do dado. É uma métrica objetiva.

Nessa dissertação pretende-se trabalhar **somente as métricas objetivas**!

As métricas estarão voltadas para partes específicas de objetos específicos da Interface. A seção a seguir apresenta o conceito de Interface.

4.4. INTERFACE

A interação visual entre o usuário e uma aplicação pode ocorrer através dos mais diversos tipos de objetos presentes numa interface, como pode ser visto na Figura 16, capturada do *Word97*, da *Microsoft*:

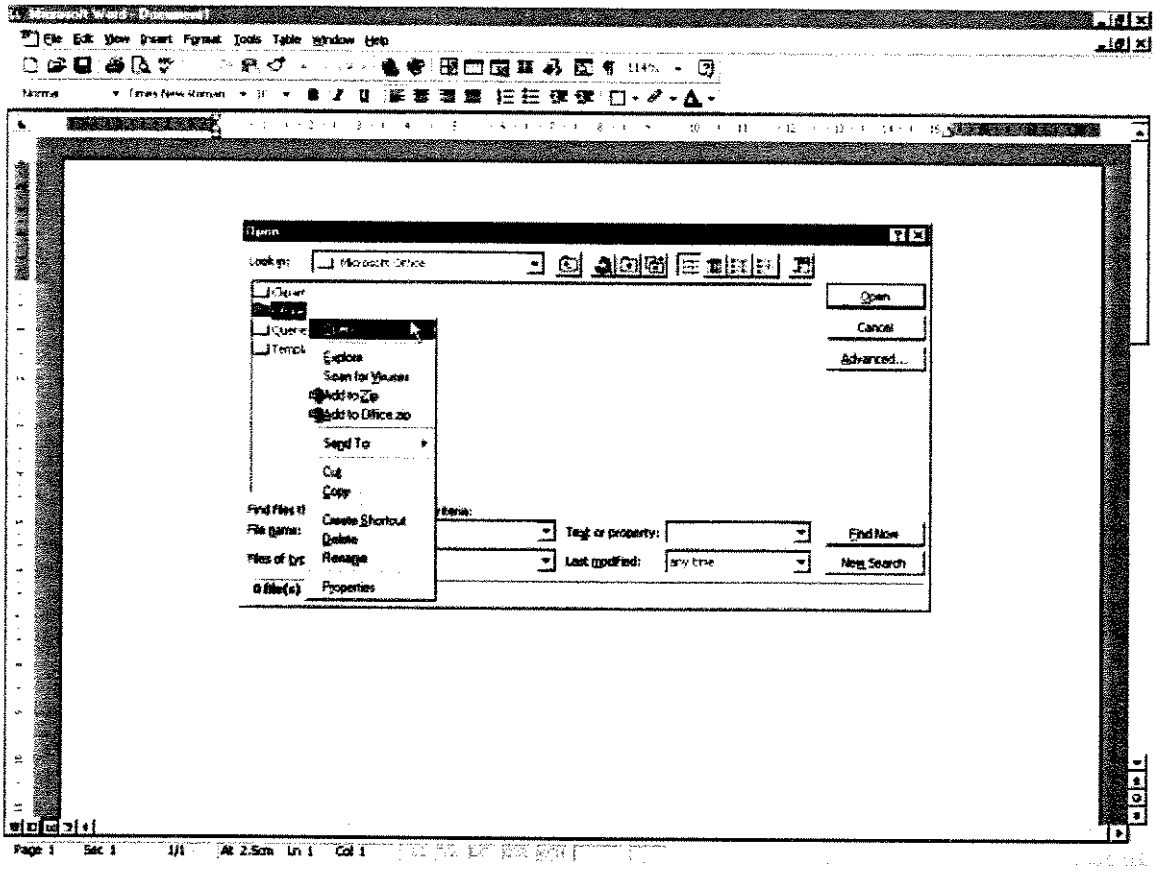

Figura 16: Interface do *Word97*

Essa figura contém diversos tipos de objetos, tais como janela, botão de comando, barra de ferramentas, barra de rolagem, barra de estado, etc. Para limitar o escopo dessa dissertação, os quatro tipos de objetos analisados são:

- janela (Ja)
- botão de comando (BC)
- menu (Me)
- barra de ferramentas (BF)

Essa figura também permite uma abstração do conceito de objetos numa interface gráfica. Para essa dissertação a Interface é composta por uma grande quantidade de **objetos**, dos quais estão sendo cobertos apenas os citados acima. O agrupamento de determinados objetos dentro da área de trabalho do objeto janela dá origem a uma **caixa de diálogo**, uma janela padrão que requer do usuário uma informação, para que a aplicação execute uma ação. O conjunto de caixas de diálogo forma a **aplicação**, como pode ser visto na Figura 17.

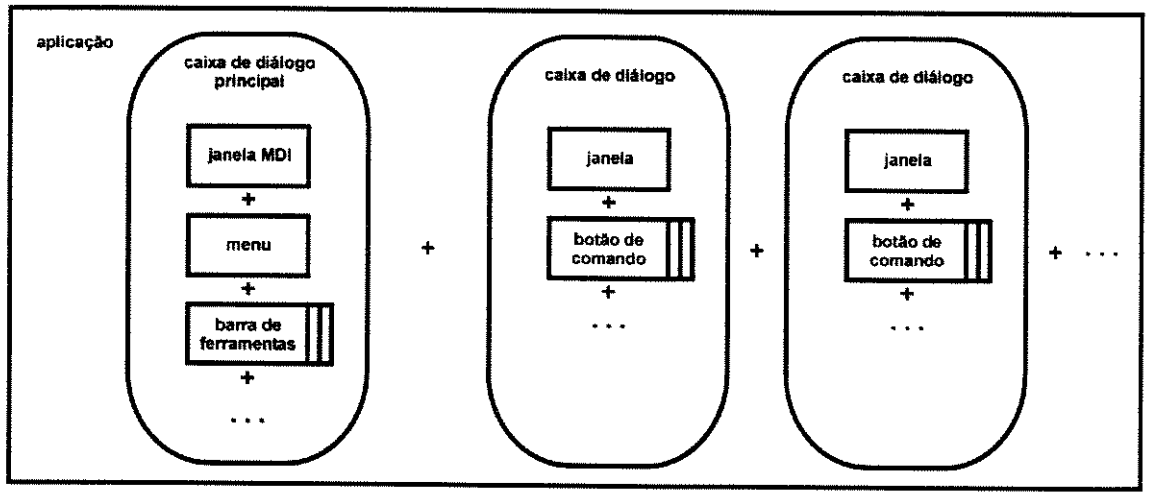

Figura 17: Relação de Agrupamento de Objetos da Interface

A seguir encontra-se uma definição mais detalhada de cada um dos objetos que serão trabalhados na dissertação. Sua nomenclatura e definição foi baseada no *Books Online* [VBonline] do Visual Basic 5.0, uma ferramenta construtora de interface [MYERS97] que oferece uma *toolkit* de objetos *Microsoft*.

4.5. OBJETOS DA INTERFACE

Para cada objeto da interface a ser trabalhado nas métricas, será apresentada uma definição através de uma figura, que contempla suas partes e sua explicação. Essas partes que o objeto possui obrigatoriamente ou não, serão os "focos" das métricas, como pode ser visto a seguir, na seção de estrutura das métricas. Para facilitar a compreensão as figuras dos objetos seguem um padrão, como o mostrado abaixo, na Figura 18.

O objeto possui duas partes, uma responsável pelo contato com a aplicação (parte superior), outra responsável pelo contato com o usuário (parte inferior). Com exceção do objeto janela, os demais objetos fazem contato com a aplicação através da ***callback***. A *callback* representa a ligação entre o objeto da interface e a função da aplicação que o mesmo irá ativar, como pode ser visto na Figura 19. Uma *callback*, deve obrigatoriamente ativar uma chamada de **ação direta** ou de **caixa de diálogo**. Como a *callback* relaciona-se à uma função da aplicação, podem existir 2 problemas previstos: função inexistente e função nula. O primeiro problema refere-se a uma *callback* que ative uma função que não existe. O segundo, uma função que não execute nada, por exemplo, esteja vazia. Embora, na visão desse autor, esses sejam problemas da aplicação e não da interface, serão abordados porque têm forte impacto na compreensão do sistema pelo usuário.

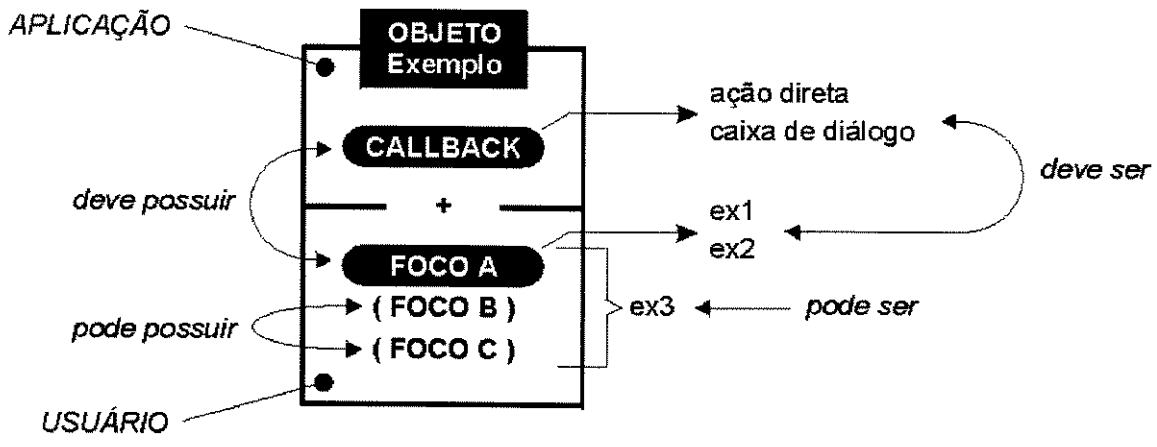

Figura 18: Padrão para as figuras dos objetos

Da Figura 18, os focos que estiverem com fundo preto são obrigatórios, ou seja, quando houver um objeto, essa parte do objeto precisa necessariamente existir. Os focos sem fundo preto, entre parênteses, são opcionais, ou seja, o objeto pode possuir, mas não é uma parte obrigatória. Os focos quando seguidos de uma seta, devem ser o que estiver descrito na frente da mesma. Por exemplo, a *callback* deve ser uma chamada de ação direta ou de caixa de diálogo e o foco A deve ser ex1 ou ex2. Mas o foco A também pode ser ex3, assim como os focos B e C, porque são seguidos de colchetes.

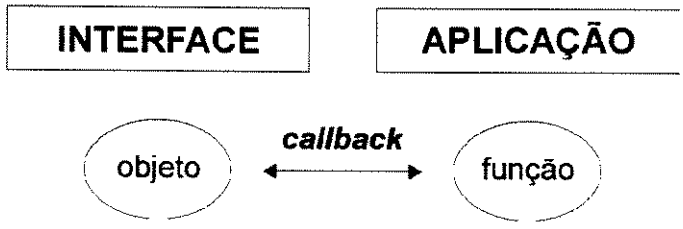

Figura 19: Conceito de *callback*

A seguir a descrição dos objetos da interface a serem trabalhados.

4.5.1. JANELA

Existem dois tipos diferentes de janelas disponíveis: MDI (*Multiple Document Interface*) e padrão. A janela MDI permite criar uma aplicação que mantém múltiplas janelas padrão dentro de uma única (e.g., o *Word97* utiliza esse recurso). Na Figura 16 temos ambos os exemplos.

A janela MDI é a janela principal do software, que contém os objetos menu e barra de ferramentas. Uma janela padrão possue uma propriedade chamada *MDIchild*, que se estiver definida como verdadeira, só lhe permitirá permanecer dentro do espaço de trabalho da janela MDI. Um bom exemplo disso é uma janela de um documento do Word97. Jamais uma janela de um documento ocupará outro espaço que não o espaço de trabalho definido pela janela MDI. Por outro lado, se a a propriedade *MDIchild* estiver definida como falsa, será possível posicionar a janela fora do espaço de trabalho da janela MDI. Nesse caso, o exemplo é a janela "*Open*".

Uma janela não tem *callback* porque não executa uma função da aplicação. Seu objetivo é ser um objeto "hospedeiro", em cuja área de trabalho possam ser inseridos outros objetos., A parte que faz contato com o usuário está dividida em barra de título (barra) e área de trabalho (área). Pela Figura 20, a barra de título deve possuir obrigatoriamente uma legenda e pode possuir um ícone e três botões de controle (minimização, restauração/maximização e saída). O ícone e os botões de controle podem ser padrão.

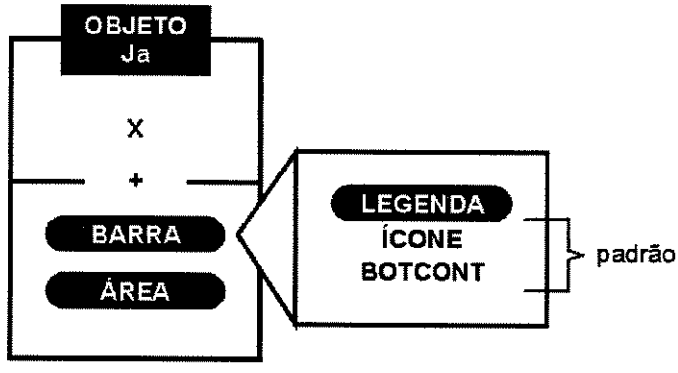

Figura 20: Objeto janela

4.5.2. BOTÃO DE COMANDO

Um botão de comando é usado para começar, interromper ou finalizar um processo. Na Figura 16, na janela "*Open*", pode-se ver cinco botões de comando ("*Open*", "*Cancel*", "*Advanced*", "*Find Now*" e "*New Search*". Pela Figura 21, esse objeto deve possuir obrigatoriamente uma *callback* e uma legenda. Toda *callback* obrigatoriamente chamará uma ação direta ou uma caixa de diálogo. Tanto a legenda como a *tooltip* e o ícone podem ser padrão.

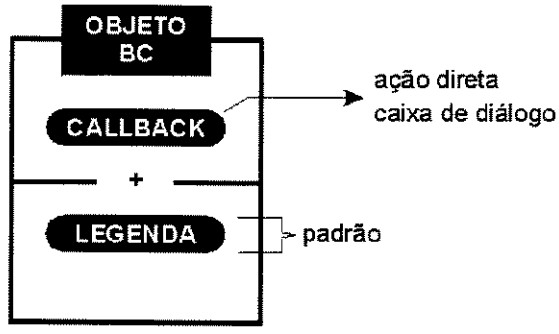

Figura 21: Objeto botão de comando

4.5.3. MENU

O conjunto de comandos disponíveis ao usuário, normalmente situado imediatamente abaixo da barra de título da janela MDI, como pode ser visto na Figura 16. Pela Figura 22, esse objeto pode possuir n ocorrências de combinações entre suas partes. Cada combinação deve possuir obrigatoriamente uma *callback* e uma legenda e pode possuir um *shortcut* (combinação de teclas) e um ícone. Toda *callback* obrigatoriamente chamará uma ação direta ou uma caixa de diálogo e toda legenda obrigatoriamente será, conforme definição da nomenclatura do VB [VBOnLine], dependendo do nível na hierarquia do menu, ou título, subtítulo, subsubtítulo, ou item, subitem, subsubitem[4] (para fins de facilidade, limitou-se a hierarquia no menu a três níveis). Tanto a legenda como o *shortcut* e o ícone podem ser padrão.

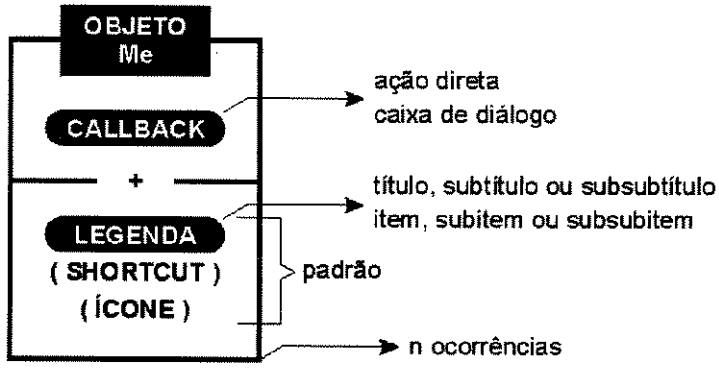

Figura 22: Objeto menu

[4] por definição, a hierarquia do menu prevê que um título pode possuir itens e subtítulos (e.g., no objeto menu do *Word97*, "*File*" é um título e, "*Open*" e "*Send To*" são, respectivamente, item e subtítulo de "*File*"); um subtítulo pode possuir subítens e subsubtítulos; e um subsubtítulo pode possuir subsubítens.

4.5.4. BARRA DE FERRAMENTAS

O conjunto de botões que correspondem a ítens, subítens e subsubítens de um menu, oferecem uma interface gráfica aos comandos da aplicação mais frequentemente usados pelo usuário, como pode ser visto na Figura 16, logo abaixo do menu. Pela Figura 23, esse objeto pode possuir n ocorrências de combinações entre suas partes. Cada combinação deve possuir obrigatoriamente uma *callback* e um ícone, e pode possuir uma *tooltip* e uma legenda. Toda *callback* obrigatoriamente chamará uma ação direta ou uma caixa de diálogo. Todo ícone faz parte de um grupo de ícones. Tanto o ícone como a *tooltip* podem ser padrão.

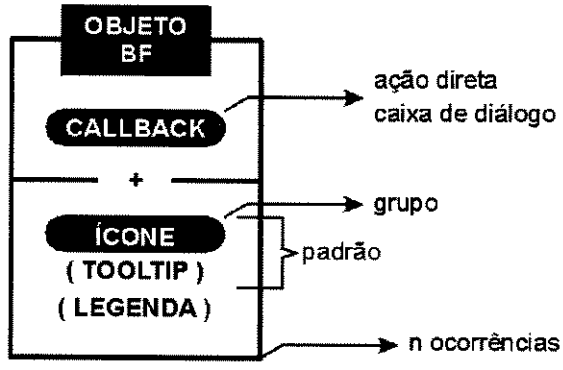

Figura 23: Objeto barra de ferramentas

Uma vez que os objetos foram descritos, divididos em partes, que serão os focos das métricas, pode-se partir para o conceito de contextos.

4.6. CONTEXTOS

A partir da relação apresentada na Figura 17 é proposto um esquema que permite ao *designer* "enxergar" a estrutura de outra maneira, sob o ponto de vista de **contextos**, como pode ser visto na Figura 24.

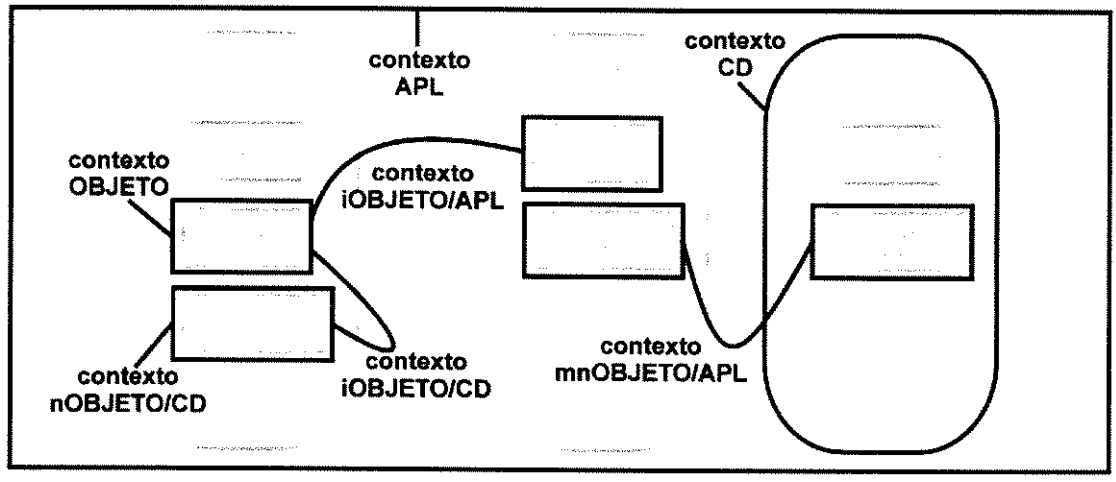

Figura 24: Relação de Contextos

Os **contextos facilitam a visão do *designer*** porque possibilitam diversos tipos de relações entre objetos! Permitem visualizar essas relações de forma nivelada e isolada, tendo por único objetivo favorecer o reuso de métricas (Seção 4.7). Os contextos estão descritos a seguir e já estão apresentados com suas relações previstas.

O **contexto OBJETO** trata de uma única ocorrência de um tipo de objeto. É o contexto base para os demais! São analisados:

- objeto janela
- objeto botão de comando
- objeto menu
- objeto barra de ferramentas

Como observação, a janela que está sendo analisada é a janela padrão, não a janela MDI.

O **contexto nOBJETO/CD** trata n ocorrências de um tipo de objeto (onde n>1, e uma ocorrência é diferente da outra), presentes numa única caixa de diálogo. São analisados:

- n ocorrências do objeto botão de comando (nBC)
- n ocorrências de objeto barra de ferramentas (nBF)

Como observação, não se enquadram nesse contexto: objeto menu, que só possui uma ocorrência em toda a aplicação e o objeto janela, que, por definição, só tem uma ocorrência para cada caixa de diálogo.

O **contexto iOBJETO/CD** trata a relação entre diferentes tipos de objetos presentes numa única caixa de diálogo. É analisada:

- uma ocorrência do objeto menu com uma ou mais ocorrências do objeto barra de ferramentas (Me x nBF).

O **contexto CD** (caixa de diálogo) trata uma única ocorrência da relação entre um objeto janela e os demais objetos contidos na sua área de trabalho. Nessa dissertação está sendo analisada a relação entre o objeto janela e as n ocorrências do objeto botão de comando.

O contexto **mnOBJETO/APL** trata mn ocorrências de um tipo de objeto (onde n>1, e uma ocorrência é diferente da outra), presentes em uma aplicação, mas em diferentes caixas de diálogo. São analisados:

- mn ocorrências do objeto janela (mnJa/APL)
- mn ocorrências do objeto botão de comando (mnBC/APL)

Como observação, o objeto menu só possui uma ocorrência, que o impede de estar nesse contexto. O objeto barra de ferramentas só aparece na caixa de diálogo principal, o que também o impede de estar nesse contexto.

O **contexto iOBJETO/APL** trata a relação entre diferentes tipos de objetos presentes em toda a aplicação. É analisada:

- uma ocorrência do objeto menu com uma ou mais ocorrências do objeto janela (Me x mnJa/APL)

O **contexto APL** trata todas as relações possíveis entre todos os contextos anteriores. Suas métricas são a somatória de todas as métricas de todos os contextos.

4.7. REUSO DE MÉTRICAS

Quanto mais abrangente fica o contexto, mais ele pode fazer uso de métricas já estabelecidas em contextos anteriores. Um exemplo bastante simples dessa afirmação é: no contexto OBJETO BC (Botão de Comando), existe uma métrica que questiona logicamente (valor sim ou não) se sua legenda é padrão. Ao passar para o contexto nOBJETO/CD nBC (dois ou mais Botões de Comando por Caixa de Diálogo), pode-se verificar cada legenda de cada Botão de Comando presente na Caixa de Diálogo, de modo a se obter a quantidade de legendas padrão. Dessa maneira, através do reuso das métricas, evita-se a repetição de definições (e.g., o contexto OBJETO é base para todos os demais contextos).

O reuso de métricas nos contextos pode ser visto genericamente através da Figura 25, e é explicado logo a seguir.

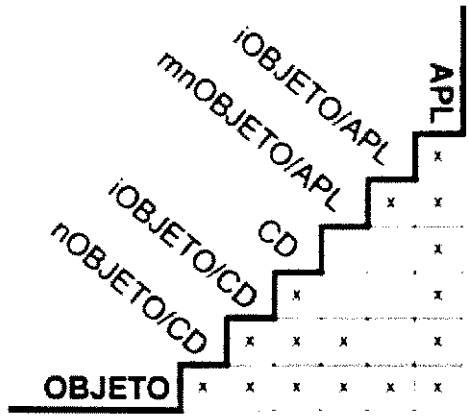

Figura 25: Relação de dependência entre contextos.

Pela figura, pode-se ver quais contextos dependem uns dos outros, através da relação "vertical". O contexto objeto não depende de ninguém, pois é a base e está todo na horizontal. O contexto APLICAÇÃO depende de todos os outros contextos, pois está todo na vertical. Os contextos intermediários têm suas relações de dependência como pode ser visto a seguir:

- as métricas do contexto nOBJETO/CD (nBC e nBF) dependem do contexto OBJETO
- as métricas do contexto iOBJETO/CD (Me x nBF) dependem do contexto OBJETO (Me e BF) e contexto nOBJETO/CD (nBF)
- as métricas do contexto CD dependem do contexto OBJETO (BC), contexto nOBJETO/CD (nBC e nBF) e contexto iOBJETO/CD (Me x nBF)
- as métricas do contexto mnOBJETO/APL (mnJa/APL e mnBC/APL) dependem do contexto OBJETO (Ja e BC) e contexto nOBJETO/CD (nBC)
- as métricas do contexto iOBJETO/APL (Me x mnJa/APL) dependem do contexto OBJETO (Me e Ja) e contexto mnOBJETO/APL (mnJa/APL)

Após o conceito de contextos, é necessário entender o conceito de categorias.

4.8. CATEGORIAS

As categorias são responsáveis por agrupar características da Interface. Nessa dissertação, são definidos três grandes grupos de categorias: gráfica, textual e lógica. Algumas características são bastante genéricas, podendo ser aplicadas em diversos contextos, outras são mais específicas. A seguir encontram-se melhor descritas as categorias.

A **categoria GRÁFICA** (G) aborda diversas características relacionadas a altura, largura, localização e cor. Foram consideradas gráficas algumas características de texto (legenda ou *tooltip*) dos objetos que impactam mais visualmente que textualmente. São elas:

- quantidade de caracteres
- tipo de fonte (e.g., Times, Times New Roman, Arial, Helvética etc)
- estilo de fonte (e.g., normal, negrito, itálico, sublinhado etc)
- tamanho de fonte (e.g., 8 pt, 10 pt, 12 pt, 14 pt etc)
- cor de fonte (e.g., preto, branco, vermelho etc)

A **categoria TEXTUAL** (T) aborda diversas características relacionadas à correção textual (legenda ou *tooltip*). São elas:

- ortografia (e.g., "*Opem*" ao invés de "*Open*")
- gramática (concordância gramatical, regência verbal, voz passiva/ativa, e.g., "*Contents end Index*" ao invés de "*Contents and Index*"; "*Open*" e "*Insertion of Table*" ao invés de "*Open*" e "*Insert Table*")
- capitalização (e.g., "*OpEn*" ao invés de "*Open*")
- abreviação (e.g., "*Bullets and Numb.*" ao invés de "*Bullets and Numbering*")
- sinônimos (e.g., "*End*", "*Close*", "*Ok*")

Todas essa características dependem de padrões pré-estabelecidos. Por exemplo, mesmo que não haja sentido nesse contexto, pode ser que "*Opem*" seja uma palavra que tenha algum sentido em outro contexto. Mas isso dependerá do que estiver sendo avaliado. Provavelmente se "*Opem*" for um item do objeto menu cujo título seja "*File*", ele estará provavelmente errado ortograficamente. Entretanto, se esse texto estiver num objeto botão de comando que se encontra dentro de uma caixa de diálogo específica, em que todos os botões de comando apresentam siglas em seus textos, provavelmente ele estará correto ortograficamente, pois "*Opem*" terá um significado específico. Isso é válido para todas essas características.

A **categoria LÓGICA** (L) aborda diversas características relacionadas à capacidade de execução da tarefa e controle dos objetos, bem como a existência ou não de determinadas partes de objetos. É a mais específica das características, por isso não pode ser generalizada como as categorias anteriores. Para citar um exemplo específico do objeto menu, alguns dos seus ítens podem apresentar em sua legenda os "3 pontos" ("..."). Esses 3 pontos servem para indicar a chamada de uma caixa de diálogo ao invés de uma ação direta. O *designer* pode se esquecer desse fato e apresentar ao usuário uma legenda com os 3 pontos chamando uma ação direta, o que seria errado.

4.9. ESTRUTURA DA MÉTRICA

Cada métrica é dividida em 4 partes, como pode ser visto na Figura 26. Essas partes representam:

- primeira (entre colchetes): cruzamento entre contexto e categoria
- segunda (entre dois pontos e reticências): foco
- terceira (entre reticências e um ponto): métrica *de facto*
- quarta (após um ponto): sufixo da métrica

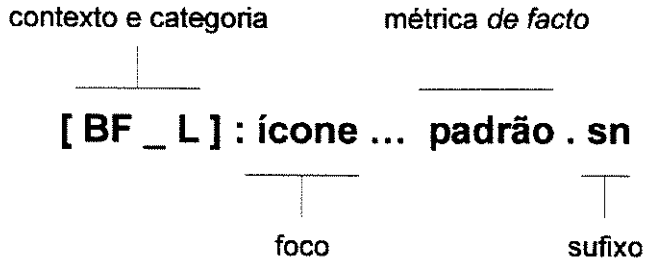

Figura 26: Estrutura da métrica

Das quatro partes, a única que precisa uma explicação a mais é a quarta, sufixo. Uma métrica pode não precisar necessariamente possuir um sufixo. Sua vantagem é indicar mais claramente o objetivo da métrica e o tipo de dado que se pretende coletar. Quando aparecem, os dados a serem coletados limitam-se a valores lógicos e numéricos, podendo ser:

- qtd: valor (numérico)
- min: valor mínimo (numérico) entre as métricas; depende de "qtd"
- max: valor máximo (numérico); depende de "qtd"
- med: valor médio (numérico); depende de "qtd"
- sn: valor sim ou não (lógico)

Um sufixo sempre aparece após a combinação entre suas 3 partes anteriores ("contexto e categoria", "foco" e "métrica *de facto*"). Também podem aparecer combinações de sufixos, como por exemplo "qtd.min", que refere-se ao valor mínimo da quantidade de alguma combinação entre partes da métrica.

Normalmente, após usar o sufixo ".sn" para obter um valor lógico, será levantada a quantidade de ocorrências de determinado valor lógico, usando o sufixo "qtd". Quando isso ocorrer dentro de um mesmo contexto, a descrição da métrica com o sufixo ".qtd" seguramente conterá o texto padrão "quantidade de * = s", em que o sinal " * " refere-se à combinação das partes da métrica anterior ao sufixo. Isso será uma constante na descrição das métricas.

4.10. CRUZAMENTO ENTRE CONTEXTOS E CATEGORIAS

Essa seção mostra o cruzamento entre os contextos e as categorias, abordados anteriormente. As métricas objetivas de consistência de interface são fruto desse cruzamento, e estão agrupadas em tabelas, representadas por um número (e.g., 8G), como pode ser visto na Figura 27. Essa figura é um "resumo" dos conceitos que foram passados até o momento. Cada uma dessas tabelas encontra-se na próxima seção.

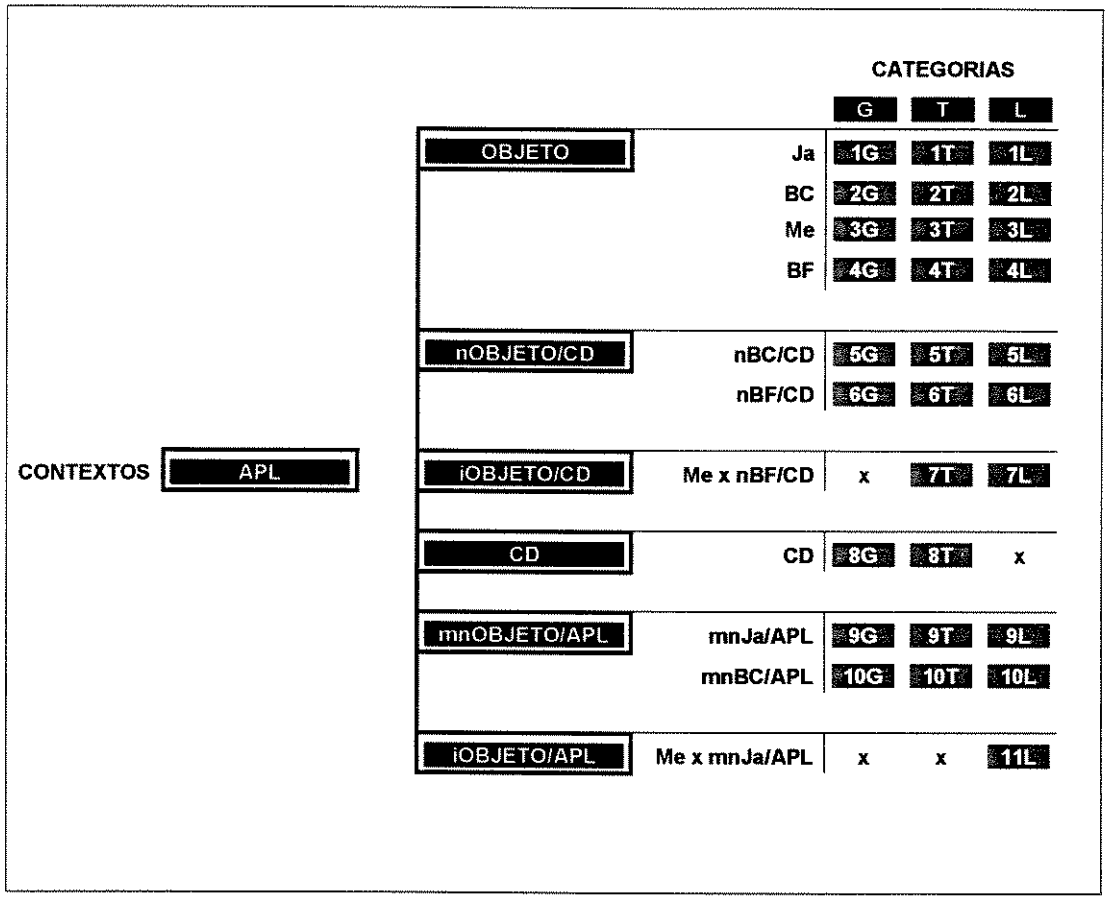

Figura 27: Cruzamento entre Contextos e Categorias

Para facilitar a leitura das métricas, todos os contextos são reintroduzidos e complementados com alguns exemplos. Dentre essas combinações, não existe cruzamento em 7G porque o contexto inter-objetos de caixa de diálogo não apresenta relação gráfica entre as métricas dispostas em Me e nBF. O mesmo ocorre para o cruzamento 8L, contexto caixa de diálogo, em que não foi encontrada relação lógica entre as métricas dispostas em Ja e nBC. Situação semelhante ocorre para os cruzamentos 11G e 11T, contexto inter-objetos da aplicação, em que não foram encontradas relações gráfica e textual entre as métricas dispostas em Me e mnJa/APL. A seguir, as métricas.

4.10.1. CONTEXTO OBJETO

O contexto OBJETO é uma generalização dos contextos referentes aos objetos janela, botão de comando, menu e barra de ferramentas. A seguir encontra-se a descrição de cada um desses contextos específicos e suas respectivas métricas.

4.10.1.1 Contexto Ja

O contexto Janela apresenta métricas relacionadas ao objeto janela, divididas nas categorias Gráfica, Textual e Lógica, cada qual com seus respectivos focos. A categoria Gráfica (Tabela 2) aborda 3 focos: área de trabalho (e.g., razão de aspecto), barra de título (e.g., cor) e legenda (e.g., tamanho da fonte). A categoria Textual (Tabela 3) aborda 1 foco: legenda (e.g., ortografia). A categoria Lógica (Tabela 4) aborda 2 focos: geral (e.g., mobilidade) e legenda (e.g., ícone associado à legenda).

Tabela 2: 1G - contexto Janela e categoria Gráfica

ÁREA DE TRABALHO	
[Ja_G]:área...altura	altura da área de trabalho do objeto janela
[Ja_G]:área...largura	largura da área de trabalho do objeto janela
[Ja_G]:área...razão_aspecto	altura / largura da área de trabalho do objeto janela
[Ja_G]:área...quadrado	altura x largura da área de trabalho do objeto janela
[Ja_G]:área...cor	cor da área de trabalho do objeto janela
[Ja_G]:área...localização	valor da opção da área de trabalho do objeto janela (e.g., centralizado na tela, na janela pai, etc.)
BARRA DE TÍTULO	
[Ja_G]:barra...cor	cor da barra de título do objeto janela
LEGENDA G1: fonte	
[Ja_G]:legenda...fonte(tipo)	tipo de fonte aplicado à legenda da barra de títulos do objeto janela
[Ja_G]:legenda...fonte(estilo)	estilo de fonte aplicado à legenda da barra de títulos do objeto janela
[Ja_G]:legenda...fonte(tamanho)	tamanho de fonte aplicado à legenda da barra de títulos do objeto janela
[Ja_G]:legenda...fonte(cor)	cor de fonte aplicado à legenda da barra de títulos do objeto janela
LEGENDA G2: caracteres	
[Ja_G]:legenda...caracteres.qtd	quantidade de caracteres usada na legenda da barra de títulos do objeto janela

Tabela 3: 1T - contexto Janela e categoria Textual

LEGENDA T1: ortografia, gramática, capitalização e abreviação	
[Ja_T]:legenda...conteúdo	conteúdo da legenda da barra de títulos do objeto janela
[Ja_T]:legenda...ortografia.sn	valor sim ou não, legenda da barra de títulos do objeto janela com erro de ortografia
[Ja_T]:legenda...gramática1.sn	valor sim ou não, legenda da barra de títulos do objeto janela com erro de gramática1
[Ja_T]:legenda...capitalização.sn	valor sim ou não, legenda da barra de títulos do objeto janela com erro de capitalização
[Ja_T]:legenda...abreviação.sn	valor sim ou não, legenda da barra de títulos do objeto janela com erro de abreviação

Tabela 4: 1L - contexto Janela e categoria Lógica

GERAL	
[Ja_L]:...mobilidade.sn	valor sim ou não, objeto janela tem condição de mobilidade
[Ja_L]:...redimensão.sn	valor sim ou não, objeto janela tem condição de redimensionamento
LEGENDA L1 (ÍCONE)	
[Ja_L]:legenda(ícone)...ícone.sn	valor sim ou não, ícone associado à legenda da barra de título do objeto janela existe

LEGENDA L2 (BOTÃO DE CONTROLE)	
[Ja_L]:legenda(botcont)...botcont.sn	valor sim ou não, existem botões de controle na barra de título do objeto janela
[Ja_L]:legenda(botcont)...botcont.sn.opc	valor de opção, resultado da combinação de botões de controle (e.g., (minimizar, maximizar/restaurar, fechar), (minimizar e fechar), (interrogação e fechar)); depende de [Ja_L]:legenda(botcont)...botcont.sn = s

4.10.1.2 Contexto BC

O contexto Botão de Comando apresenta métricas relacionadas ao objeto botão de comando, divididas nas categorias Gráfica, Textual e Lógica, cada qual com seus respectivos focos. A categoria Gráfica (Tabela 5) aborda 2 focos: geral (e.g., localização) e legenda (e.g., tamanho da fonte). A categoria Textual (Tabela 6) aborda 1 foco: legenda (e.g., ortografia). A categoria Lógica (Tabela 7) aborda 2 focos: *callback* (e.g., função inexistente) e legenda (e.g., legenda padrão), bem como o cruzamento entre esses 2 focos.

Tabela 5: 2G - contexto Botão de Comando e categoria Gráfica

GERAL	
[BC_G]:...altura	altura do objeto botão de comando
[BC_G]:...largura	largura do objeto botão de comando
[BC_G]:...razão_aspecto	altura / largura do objeto botão de comando
[BC_G]:...quadrado	altura x largura do objeto botão de comando
[BC_G]:...cor	cor do objeto botão de comando
[BC_G]:...localização	coordenadas xy dos quatro vértices do objeto botão de comando (em relação à sua caixa de diálogo)
LEGENDA G1: fonte	
[BC_G]:legenda...fonte(tipo)	tipo de fonte aplicado à legenda do objeto botão de comando
[BC_G]:legenda...fonte(estilo)	estilo de fonte aplicado à legenda do objeto botão de comando
[BC_G]:legenda...fonte(tamanho)	tamanho de fonte aplicado à legenda do objeto botão de comando
[BC_G]:legenda...fonte(cor)	cor de fonte aplicado à legenda do objeto botão de comando
LEGENDA G2: caracteres	
[BC_G]:legenda...caracteres.qtd	quantidade de caracteres da legenda do objeto botão de comando

Tabela 6: 2T - contexto Botão de Comando e categoria Textual

LEGENDA T1: ortografia, gramática, capitalização e abreviação	
[BC_T]:legenda...conteúdo	conteúdo da legenda do objeto botão de comando
[BC_T]:legenda...ortografia.sn	valor sim ou não, erro de ortografia na legenda do objeto botão de comando existe
[BC_T]:legenda...gramática1.sn	valor sim ou não, erro de gramática na legenda do objeto botão de comando existe
[BC_T]:legenda...capitalização.sn	valor sim ou não, erro de capitalização na legenda do objeto botão de comando existe
[BC_T]:legenda...abreviação.sn	valor sim ou não, erro de abreviação na legenda do objeto botão de comando existe

Tabela 7: 2L - contexto Botão de Comando e categoria Lógica

CALLBACK L1: ação direta e caixa de diálogo	
[BC_L]:*callback*...ação.sn	valor sim ou não, *callback* do objeto botão de comando chama ação direta
[BC_L]:*callback*...cd.sn	valor sim ou não, *callback* do objeto botão de comando chama caixa de diálogo
***CALLBACK* L2: função inexistente e nula**	
[BC_L]:*callback*...funcinex.sn	valor sim ou não, *callback* do objeto botão de comando com função inexistente
[BC_L]:*callback*...funcnula.sn	valor sim ou não, *callback* do objeto botão de comando com função nula
LEGENDA L1: 3pontos, padrão, sublinhado, sublinhado padrão e combinação padrão	
[BC_L]:legenda...3pontos.sn	valor sim ou não, legenda do objeto botão de comando tem três pontos
[BC_L]:legenda...padrão.sn	valor sim ou não, legenda do objeto botão de comando é padrão; depende de lista padrão (e.g., "*ok*", "*cancel*", "*yes*")

[BC_L]:legenda...sublinhado.sn	valor sim ou não, legenda do objeto botão de comando tem caracter sublinhado
[BC_L]:legenda...sublinhado_padrão.sn	valor sim ou não, legenda do objeto botão de comando tem caracter sublinhado padrão; depende de lista padrão (e.g., "*Open*" tem caracter "O" sublinhado)
[BC_L]:legenda...comb_padrão.sn	valor sim ou não, combinação padrão entre legenda padrão e caracter sublinhado padrão do objeto botão de comando existe

***CALLBACK* L3: *CALLBACK* L2 e *CALLBACK* L1 (função inexistente e nula) e (ação direta e caixa de diálogo)**	
[BC_L]:*callback*...ação_funcinex.sn	valor sim ou não, *callback* do objeto botão de comando chama ação direta com função inexistente
[BC_L]:*callback*...ação_funcnula.sn	valor sim ou não, *callback* do objeto botão de comando chama ação direta com função nula
[BC_L]:*callback*...cd_funcinex.sn	valor sim ou não, *callback* do objeto botão de comando chama caixa de diálogo com função inexistente
[BC_L]:*callback*...cd_funcnula.sn	valor sim ou não, *callback* do objeto botão de comando chama caixa de diálogo com função nula
CALLBACK L1 e LEGENDA L1 (caixa de diálogo) e (3pontos)	
[BC_L]:*callback*/legenda...cd_3pontos.sn	valor sim ou não, *callback* do objeto botão de comando que chama caixa de diálogo tem legenda com 3 pontos; depende de [BC_L]:*callback*...cd.sn = s e [BC_L]:legenda...3pontos.sn = s

4.10.1.3 Contexto Me

O contexto Menu apresenta métricas relacionadas ao objeto menu, divididas nas categorias Gráfica, Textual e Lógica, cada qual com seus respectivos focos. A categoria Gráfica (Tabela 8) aborda 2 focos: geral (e.g., cor) e legenda (e.g., tamanho da fonte). A categoria Textual (Tabela 9) aborda 1 foco: legenda (e.g., ortografia). A categoria Lógica (Tabela 10) aborda 2 focos: *callback* (e.g., função inexistente) e legenda (e.g., legenda padrão), bem como os diversos cruzamentos possíveis entre eles. Tais cruzamentos oferecem uma rica quantidade de métricas, de média complexidade (e.g., legendas subitem para cada legenda título) a grande complexidade (e.g., combinação padrão entre legenda item, sublinhado da legenda item, *shortcut* associado a legenda e ícone associado à legenda).

Tabela 8: 3G - contexto Menu e categoria Gráfica

GERAL	
[Me_G]:...cor	cor do objeto menu
LEGENDA G1: fonte	
[Me_G]:legenda...fonte(tipo)	tipo de fonte aplicado à legenda do objeto menu
[Me_G]:legenda...fonte(estilo)	estilo de fonte aplicado à legenda do objeto menu
[Me_G]:legenda...fonte(tamanho)	tamanho de fonte aplicado à legenda do objeto menu
[Me_G]:legenda...fonte(cor)	cor de fonte aplicado à legenda do objeto menu
LEGENDA G2: caracteres	
[Me_G]:legenda...caracteres.qtd	quantidade de caracteres, por cada legenda do objeto menu (todo tipo de legenda: título, subtítulo, subsubtítulo, item, subitem e subsubitem)
[Me_G]:legenda...caracteres.qtd.min	valor mínimo entre as n ocorrências de *[5]
[Me_G]:legenda...caracteres.qtd.max	valor máximo entre as n ocorrências de *
[Me_G]:legenda...caracteres.qtd.med	valor médio entre as n ocorrências de *

[5] o símbolo "*" refere-se à métrica anterior. Quando usado, tem a função de evitar reescrever o texto da métrica anterior.

Tabela 9: 3T - contexto Menu e categoria Textual

LEGENDA T1: ortografia, gramática, capitalização, abreviação e sinônimos	
[Me_T]:legenda...conteúdo	conteúdo de cada uma das legendas do objeto menu
[Me_T]:legenda...ortografia.sn	valor sim ou não, legenda do objeto menu com erro de ortografia
[Me_T]:legenda...ortografia.sn.qtd	quantidade de * = s[6]
[Me_T]:legenda...gramática1.sn	valor sim ou não, legenda do objeto menu com erro de gramática
[Me_T]:legenda...gramática1.sn.qtd	quantidade de * = s
[Me_T]:legenda...capitalização.sn	valor sim ou não, legenda do objeto menu com erro de capitalização
[Me_T]:legenda...capitalização.sn.qtd	quantidade de * = s
[Me_T]:legenda...abreviação.sn	valor sim ou não, legenda do objeto menu com erro de abreviação
[Me_T]:legenda...abreviação.sn.qtd	quantidade de * = s
[Me_T]:legenda...gramática2.sn	valor sim ou não, legendas do objeto menu com erro de gramática entre si
[Me_T]:legenda...gramática2.sn.qtd	quantidade de * = s
[Me_T]:legenda...sinônimos.sn	valor sim ou não, legendas do objeto menu são sinônimos
[Me_T]:legenda...sinônimos.sn.qtd	quantidade de * = s

Tabela 10: 3L - contexto Menu e categoria Lógica

***CALLBACK* L1: ação direta e caixa de diálogo**	
[Me_L]:*callback*...ação.sn	valor sim ou não, *callback* do objeto menu chama ação direta
[Me_L]:*callback*...ação.sn.qtd	quantidade de * = s
[Me_L]:*callback*...cd.sn	valor sim ou não, *callback* do objeto menu chama caixa de diálogo
[Me_L]:*callback*...cd.dn.qtd	quantidade de * = s
***CALLBACK* L2: função inexistente, nula e repetida**	
[Me_L]:*callback*...funcinex.sn	valor sim ou não, *callback* do objeto menu com função inexistente
[Me_L]:*callback*...funcinex.sn.qtd	quantidade de * = s
[Me_L]:*callback*...funcnula.sn	valor sim ou não, *callback* do objeto menu com função nula
[Me_L]:*callback*...funcnula.sn.qtd	quantidade de * = s
[Me_L]:*callback*...funcrepetida.sn	valor sim ou não, *callback* do objeto menu com função repetida
[Me_L]:*callback*...funcrepetida.sn.qtd	quantidade de * = s
[Me_L]:*callback*...funcrepetição.sn.qtd	quantidade de * = s, por cada função[7]
LEGENDA L1: título, subtítulo e subsubtítulo	
[Me_L]:legenda...leg(t).qtd	quantidade de legendas título do objeto menu
[Me_L]:legenda...leg(st).qtd	quantidade de legendas subtítulo do objeto menu
[Me_L]:legenda...leg(sst).qtd	quantidade de legendas subsubtítulo do objeto menu
LEGENDA L2: item, subitem e subsubitem	
[Me_L]:legenda...leg(i).qtd	quantidade de legendas item do objeto menu
[Me_L]:legenda...leg(si).qtd	quantidade de legendas subitem do objeto menu
[Me_L]:legenda...leg(ssi).qtd	quantidade de legendas subsubitem do objeto menu
LEGENDA L3: padrão, sublinhado, repetida, sublinhado padrão e sublinhado repetido	
[Me_L]:legenda...qtd	quantidade de legendas do objeto menu
[Me_L]:legenda...padrão.sn	valor sim ou não, legenda do objeto menu é padrão; depende de lista padrão
[Me_L]:legenda...padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...sublinhado.sn	valor sim ou não, legenda do objeto menu tem caracter sublinhado
[Me_L]:legenda...sublinhado.sn.qtd	quantidade de * = s
[Me_L]:legenda...repetida.sn	valor sim ou não, legenda do objeto menu é repetida
[Me_L]:legenda...repetida.sn.qtd	quantidade de * = s
[Me_L]:legenda...repetição.sn.qtd	quantidade de * = s, por cada legenda repetida
[Me_L]:legenda...sublinhado_padrão.sn	valor sim ou não, sublinhado da legenda do objeto menu é padrão; depende de lista padrão; depende de [Me_L]:legenda...sublinhado.sn = s

[6] o símbolo "* = s" refere-se à métrica anterior que apresenta valor lógico igual a sim. Quando usado, tem a função de evitar reescrever o texto da métrica anterior.

[7] Toda métrica que for verificar se algo é repetido, precisa consistir dois pontos. Suponha, por exemplo, que uma legenda título possui 7 legendas item. Dessas 7 legendas, por algum erro do designer, 4 delas chamam *callback*s com a mesma função x e as outras 3 chamam *callback*s com a mesma função y.
A métrica **[Me_L]:*callback*...funcrepetida.sn** checará se existem funções repetidas (seu valor lógico será sim).
A métrica **[Me_L]:*callback*...funcrepetida.sn.qtd** irá apontar a existência de 2 funções repetidas, x e y.
A métrica **[Me_L]:*callback*...funcrepetição.sn.qtd** irá apontar o valor 4 para a função x e 3 para a função y.

[Me_L]:legenda...sublinhado_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...sublinhado_repetido.sn	valor sim ou não, legenda do objeto menu tem caracter sublinhado repetido; depende de [Me_L]:legenda...sublinhado.sn = s
[Me_L]:legenda...sublinhado_repetido.sn.qtd	quantidade de * = s
[Me_L]:legenda...sublinhado_repetição.sn.qtd	quantidade de * = s, por cada caracter sublinhado repetido
LEGENDA L4 (*SHORTCUT*) (só legenda item, subitem e subsubitem)	
[Me_L]:legenda...*shortcut*.sn	valor sim ou não, *shortcut* associado à legenda do objeto menu existe
[Me_L]:legenda...*shortcut*.sn.qtd	quantidade de * = s
[Me_L]:legenda...*shortcut*_conteúdo	conteúdo do *shortcut* do objeto menu
[Me_L]:legenda...*shortcut*_padrão.sn	valor sim ou não, *shortcut* associado à legenda do objeto menu é padrão; depende de lista padrão; depende de [Me_L]:legenda...*shortcut*.sn = s
[Me_L]:legenda...*shortcut*_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...*shortcut*_repetido.sn	valor sim ou não, *shortcut* associado à legenda do objeto menu é repetido; depende de [Me_L]:legenda...*shortcut*.sn = s
[Me_L]:legenda...*shortcut*_repetido.sn.qtd	quantidade de * = s
[Me_L]:legenda...*shortcut*_repetição.sn.qtd	quantidade de * = s, por cada *shortcut* repetido
LEGENDA L5 (ÍCONE) (só legenda item, subitem e subsubitem)	
[Me_L]:legenda...ícone.sn	valor sim ou não, ícone associado à legenda do objeto menu existe
[Me_L]:legenda...ícone.sn.qtd	quantidade de * = s
[Me_L]:legenda...ícone_conteúdo	conteúdo do ícone do objeto menu
[Me_L]:legenda...ícone_padrão.sn	valor sim ou não, ícone associado à legenda do objeto menu é padrão; depende de lista padrão; depende de [Me_L]:legenda...ícone.sn = s
[Me_L]:legenda...ícone_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...ícone_repetido.sn	valor sim ou não, ícone associado à legenda do objeto menu é repetido; depende de [Me_L]:legenda...ícone.sn = s
[Me_L]:legenda...ícone_repetido.sn.qtd	quantidade de * = s
[Me_L]:legenda...ícone_repetição.sn.qtd	quantidade de * = s, por cada ícone repetido
LEGENDA L6: 3pontos (só legenda item, subitem e subsubitem)	
[Me_L]:legenda...3pontos.sn	valor sim ou não, legenda do objeto menu tem 3 pontos
[Me_L]:legenda...3pontos.sn.qtd	quantidade de * = s

***CALLBACK* L3: *CALLBACK* L1 e *CALLBACK* L2** **(ação direta e caixa de diálogo) e (função inexistente, nula e repetida)**	
[Me_L]:*callback*...ação_funcinex.sn	valor sim ou não, *callback* do objeto menu chama ação direta com função inexistente
[Me_L]:*callback*...ação_funcinex.sn.qtd	quantidade de * = s
[Me_L]:*callback*...ação_funcnula.sn	valor sim ou não, *callback* do objeto menu chama ação direta com função nula
[Me_L]:*callback*...ação_funcnula.sn.qtd	quantidade de * = s
[Me_L]:*callback*...ação_funcrepetida.sn	valor sim ou não, *callback* do objeto menu chama ação direta com função repetida
[Me_L]:*callback*...ação_funcrepetida.sn.qtd	quantidade de * = s
[Me_L]:*callback*...ação_funcrepetição.sn.qtd	quantidade de * = s, por cada função repetida
[Me_L]:*callback*...cd_funcinex.sn	valor sim ou não, *callback* do objeto menu chama caixa de diálogo com função inexistente
[Me_L]:*callback*...cd_funcinex.sn.qtd	quantidade de * = s
[Me_L]:*callback*...cd_funcnula.sn	valor sim ou não, *callback* do objeto menu chama caixa de diálogo com função nula
[Me_L]:*callback*...cd_funcnula.sn.qtd	quantidade de * = s
[Me_L]:*callback*...cd_funcrepetida.sn	valor sim ou não, *callback* do objeto menu chama caixa de diálogo com função repetida
[Me_L]:*callback*...cd_funcrepetida.sn.qtd	quantidade de * = s
[Me_L]:*callback*...cd_funcrepetição.sn.qtd	quantidade de * = s, por cada função repetida
LEGENDA L7: LEGENDA L1 e LEGENDA L1 **(subtítulo e subsubtítulo) e (título e subtítulo)**	
[Me_L]:legenda...leg(t)_leg(st).qtd	quantidade de legendas subtítulo em cada legenda título do objeto menu
[Me_L]:legenda...leg(t)_leg(st).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(st).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(st).qtd.med	valor médio entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(sst).qtd	quantidade de legendas subsubtítulo em cada legenda título do objeto menu
[Me_L]:legenda...leg(t)_leg(sst).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(sst).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(sst).qtd.med	valor médio entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(sst).qtd	quantidade de legendas subsubtítulo em cada legenda subtítulo do objeto menu
[Me_L]:legenda...leg(st)_leg(sst).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(sst).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(sst).qtd.med	valor médio entre as n ocorrências de *
LEGENDA L8: LEGENDA L2 e LEGENDA L1 **(item, subitem e subsubitem) e (título, subtítulo e subsubtítulo)**	
[Me_L]:legenda...leg(t)_leg(i).qtd	quantidade de legendas item em cada legenda título do objeto menu
[Me_L]:legenda...leg(t)_leg(i).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(i).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(i).qtd.med	valor médio entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(si).qtd	quantidade de legendas subitem em cada legenda título do objeto menu
[Me_L]:legenda...leg(t)_leg(si).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(si).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(si).qtd.med	valor médio entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(ssi).qtd	quantidade de legendas subsubitem em cada legenda título do objeto menu
[Me_L]:legenda...leg(t)_leg(ssi).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(ssi).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(t)_leg(ssi).qtd.med	valor médio entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(si).qtd	quantidade de legendas subitem em cada legenda subtítulo do objeto menu
[Me_L]:legenda...leg(st)_leg(si).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(si).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(si).qtd.med	valor médio entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(ssi).qtd	quantidade de legendas subsubitem em cada legenda subtítulo do objeto menu
[Me_L]:legenda...leg(st)_leg(ssi).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(ssi).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(st)_leg(ssi).qtd.med	valor médio entre as n ocorrências de *
[Me_L]:legenda...leg(sst)_leg(ssi).qtd	quantidade de legendas subsubitem em cada legenda subsubtítulo do objeto menu
[Me_L]:legenda...leg(sst)_leg(ssi).qtd.min	valor mínimo entre as n ocorrências de *
[Me_L]:legenda...leg(sst)_leg(ssi).qtd.max	valor máximo entre as n ocorrências de *
[Me_L]:legenda...leg(sst)_leg(ssi).qtd.med	valor médio entre as n ocorrências de *

LEGENDA L9: LEGENDA L3 e LEGENDA L1 (padrão, sublinhado, repetida, sublinhado padrão e sublinhado repetido) e (título)	
[Me_L]:legenda...leg(t)_padrão.sn	valor sim ou não, legenda título do objeto menu é padrão; depende de lista padrão (e.g., "*File*" é legenda título padrão)
[Me_L]:legenda...leg(t)_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(t)_sublinhado.sn	valor sim ou não, legenda título do objeto menu tem caracter sublinhado; depende de lista padrão (e.g., "*File*" tem caracter sublinhado)
[Me_L]:legenda...leg(t)_sublinhado.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(t)_repetida.sn	valor sim ou não, legenda título do objeto menu é repetida
[Me_L]:legenda...leg(t)_repetida.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(t)_repetição.sn.qtd	quantidade de * = s, por cada legenda título
[Me_L]:legenda...leg(t)_sublinhado_padrão.sn	valor sim ou não, legenda título do objeto menu tem caracter sublinhado padrão; depende de lista padrão (e.g., "*File*" tem caracter "F" sublinhado); depende de [Me_L]:legenda...leg(t)_sublinhado.sn = s
[Me_L]:legenda...leg(t)_sublinhado_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(t)_sublinhado_repetido.sn	valor sim ou não, legenda título do objeto menu tem caracter sublinhado repetido
[Me_L]:legenda...leg(t)_sublinhado_repetido.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(t)_sublinhado_repetição.sn.qtd	quantidade de * = s, por cada legenda título
[Me_L]:legenda...leg(t)_comb_padrão.sn	valor sim ou não, combinação padrão do objeto menu entre legenda título padrão e sublinhado padrão da legenda título existe; depende de [Me_L]:legenda...leg(t)_padrão.sn = s e [Me_L]:legenda...leg(t)_sublinhado_padrão.sn = s
[Me_L]:legenda...leg(t)_comb_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(t)_seq_comb_padrão.sn	valor sim ou não, sequência entre combinações padrão do objeto menu existe; depende de lista padrão; depende de [Me_L]:legenda...leg(t)_comb_padrão.sn = s

LEGENDA L10: LEGENDA L3 e LEGENDA L2 (padrão, sublinhado, repetida, sublinhado padrão e sublinhado repetido) e (item)	
[Me_L]:legenda...leg(i)_padrão.sn	valor sim ou não, legenda item do objeto menu é padrão; depende de lista padrão (e.g., "*Open*" é legenda item padrão)
[Me_L]:legenda...leg(i)_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_sublinhado.sn	valor sim ou não, legenda item do objeto menu tem caracter sublinhado (e.g., "*Open*" tem caracter sublinhado)
[Me_L]:legenda...leg(i)_sublinhado.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_repetida.sn	valor sim ou não, legenda item do objeto menu é repetida
[Me_L]:legenda...leg(i)_repetida.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_repetição.sn.qtd	quantidade de * = s, por cada legenda item
[Me_L]:legenda...leg(i)_sublinhado_padrão.sn	valor sim ou não, legenda item do objeto menu tem caracter sublinhado padrão; depende de lista padrão (e.g., "*Open*" tem caracter "O" sublinhado); depende de [Me_L]:legenda...leg(i)_sublinhado.sn = s
[Me_L]:legenda...leg(i)_sublinhado_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_sublinhado_repetido.sn	valor sim ou não, legenda item do objeto menu tem caracter sublinhado repetido
[Me_L]:legenda...leg(i)_sublinhado_repetido.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_sublinhado_repetição.sn.qtd	quantidade de * = s, por cada legenda item
[Me_L]:legenda...leg(i)_comb_padrão1.sn	valor sim ou não, combinação padrão entre legenda item e sublinhado da legenda item do objeto menu existe (e.g., "*Save As*")
[Me_L]:legenda...leg(i)_comb_padrão1.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_comb_padrão2.sn	valor sim ou não, combinação padrão entre legenda item, sublinhado da legenda item e *shortcut* associado à legenda item do objeto menu existe (e.g., "*Clear*")
[Me_L]:legenda...leg(i)_comb_padrão2.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_comb_padrão3.sn	valor sim ou não, combinação padrão entre legenda item, sublinhado da legenda item e ícone associado à legenda item do objeto menu existe (e.g., "*Print Preview*")
[Me_L]:legenda...leg(i)_comb_padrão3.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_comb_padrão4.sn	valor sim ou não, combinação padrão entre legenda item, sublinhado da legenda item, *shortcut* associado à legenda item e ícone associado à legenda item do objeto menu existe (e.g., "*Open*")
[Me_L]:legenda...leg(i)_comb_padrão4.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_seq_comb_padrão.sn	valor sim ou não, sequência entre combinações padrão do objeto menu existe; depende de lista padrão; depende de [Me_L]:legenda...leg(t)_combinação_padrão.sn = s

LEGENDA L11: LEGENDA L4 e LEGENDA L2 (*shortcut*) e (item, subitem e subsubitem)	
[Me_L]:legenda...leg(i)_*shortcut*.sn	valor sim ou não, *shortcut* associado à legenda item do objeto menu existe
[Me_L]:legenda...leg(i)_*shortcut*.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_*shortcut*_padrão.sn	valor sim ou não, *shortcut* associado à legenda item do objeto menu é padrão; depende de lista padrão; depende de [Me_L]:legenda...leg(i)_*shortcut*.sn = s
[Me_L]:legenda...leg(i)_*shortcut*_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_*shortcut*_repetido.sn	valor sim ou não, *shortcut* associado à legenda item do objeto menu é repetido; depende de [Me_L]:legenda...leg(i)_*shortcut*.sn = s
[Me_L]:legenda...leg(i)_*shortcut*_repetido.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_*shortcut*_repetição.sn.qtd	quantidade de * = s, por cada *shortcut* repetido
[Me_L]:legenda...leg(si)_*shortcut*.sn	valor sim ou não, *shortcut* associado à legenda subitem do objeto menu existe
[Me_L]:legenda...leg(si)_*shortcut*.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(ssi)_*shortcut*.sn	valor sim ou não, *shortcut* associado à legenda subsubitem do objeto menu existe
[Me_L]:legenda...leg(ssi)_*shortcut*.sn.qtd	quantidade de * = s
LEGENDA L12: LEGENDA L5 e LEGENDA L2 (ícone) e (item, subitem e subsubitem)	
[Me_L]:legenda...leg(i)_ícone.sn	valor sim ou não, ícone associado à legenda item do objeto menu existe
[Me_L]:legenda...leg(i)_ícone.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_ícone_padrão.sn	valor sim ou não, ícone associado à legenda item do objeto menu é padrão; depende de lista padrão; depende de [Me_L]:legenda...leg(i)_ícone.sn = s
[Me_L]:legenda...leg(i)_ícone_padrão.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_ícone_repetido.sn	valor sim ou não, ícone associado à legenda item do objeto menu é repetido; depende de [Me_L]:legenda...leg(i)_ícone.sn = s
[Me_L]:legenda...leg(i)_ícone_repetido.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(i)_ícone_repetição.sn.qtd	quantidade de * = s, por cada ícone repetido
[Me_L]:legenda...leg(si)_ícone.sn	valor sim ou não, ícone associado à legenda subitem do objeto menu existe
[Me_L]:legenda...leg(si)_ícone.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(ssi)_ícone.sn	valor sim ou não, ícone associado à legenda subsubitem do objeto menu existe
[Me_L]:legenda...leg(ssi)_ícone.sn.qtd	quantidade de * = s
LEGENDA L13: LEGENDA L6 e LEGENDA L2 (3pontos) e (item, subitem e subsubitem)	
[Me_L]:legenda...leg(i)_3pontos.sn	valor sim ou não, legenda item do objeto menu tem 3 pontos
[Me_L]:legenda...leg(i)_3pontos.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(si)_3pontos.sn	valor sim ou não, legenda subitem do objeto menu tem 3 pontos
[Me_L]:legenda...leg(si)_3pontos.sn.qtd	quantidade de * = s
[Me_L]:legenda...leg(ssi)_3pontos.sn	valor sim ou não, legenda subsubitem do objeto menu tem 3 pontos
[Me_L]:legenda...leg(ssi)_3pontos.sn.qtd	quantidade de * = s

LEGENDA L2 e *CALLBACK* L1 **(item, subitem e subsubitem) e (ação direta e caixa de diálogo)**	
[Me_L]:legenda/*callback*...leg(i)_ação.qtd	quantidade de legendas item do objeto menu com *callback* que chama ação direta
[Me_L]:legenda/*callback*...leg(i)_cd.qtd	quantidade de legendas item do objeto menu com *callback* que chama caixa de diálogo
[Me_L]:legenda/*callback*...leg(si)_ação.qtd	quantidade de legendas subitem do objeto menu com *callback* que chama ação direta
[Me_L]:legenda/*callback*...leg(si)_cd.qtd	quantidade de legendas subitem do objeto menu com *callback* que chama caixa de diálogo
[Me_L]:legenda/*callback*...leg(ssi)_ação.qtd	quantidade de legendas subsubitem do objeto menu com *callback* que chama ação direta
[Me_L]:legenda/*callback*...leg(ssi)_cd.qtd	quantidade de legendas subsubitem do objeto menu com *callback* que chama caixa de diálogo

LEGENDA L2 e *CALLBACK* L2 **(item, subitem e subsubitem) e (função inexistente, nula e repetida)**	
[Me_L]:legenda/*callback*...leg(i)_funcinex.qtd	quantidade de legendas item do objeto menu com *callback* com função inexistente
[Me_L]:legenda/*callback*...leg(i)_funcnula.qtd	quantidade de legendas item do objeto menu com *callback* com função nula
[Me_L]:legenda/*callback*...leg(i)_funcrepetida.qtd	quantidade de legendas item do objeto menu com *callback* com função repetida
[Me_L]:legenda/*callback*...leg(si)_funcinex.qtd	quantidade de legendas subitem do objeto menu com *callback* com função inexistente
[Me_L]:legenda/*callback*...leg(si)_funcnula.qtd	quantidade de legendas subitem do objeto menu com *callback* com função nula
[Me_L]:legenda/*callback*...leg(si)_funcrepetida.qtd	quantidade de legendas subitem do objeto menu com *callback* com função repetida
[Me_L]:legenda/*callback*...leg(ssi)_funcinex.qtd	quantidade de legendas subsubitem do objeto menu com *callback* com função inexistente
[Me_L]:legenda/*callback*...leg(ssi)_funcnula.qtd	quantidade de legendas subsubitem do objeto menu com *callback* com função nula
[Me_L]:legenda/*callback*...leg(ssi)_funcrepetida.qtd	quantidade de legendas subsubitem do objeto menu com *callback* com função repetida

LEGENDA L2 e *CALLBACK* L3 **(item, subitem e subsubitem), (ação direta e caixa de diálogo) e (função inexistente, nula e repetida)**	
[Me_L]:legenda/*callback*...leg(i)_ação_funcinex.qtd	quantidade de legendas item do objeto menu com *callback* que chama ação direta com função inexistente
[Me_L]:legenda/*callback*...leg(i)_ação_funcnula.qtd	quantidade de legendas item do objeto menu com *callback* que chama ação direta com função nula
[Me_L]:legenda/*callback*...leg(i)_ação_funcrepetida.qtd	quantidade de legendas item do objeto menu com *callback* que chama ação direta com função repetida
[Me_L]:legenda/*callback*...leg(si)_ação_funcinex.qtd	quantidade de legendas subitem do objeto menu com *callback* que chama ação direta com função inexistente
[Me_L]:legenda/*callback*...leg(si)_ação_funcnula.qtd	quantidade de legendas subitem do objeto menu com *callback* que chama ação direta com função nula
[Me_L]:legenda/*callback*...leg(si)_ação_funcrepetida.qtd	quantidade de legendas subitem do objeto menu com *callback* que chama ação direta com função repetida
[Me_L]:legenda/*callback*...leg(ssi)_ação_funcinex.qtd	quantidade de legendas subsubitem do objeto menu com *callback* que chama ação direta com função inexistente
[Me_L]:legenda/*callback*...leg(ssi)_ação_funcnula.qtd	quantidade de legendas subsubitem do objeto menu com *callback* que chama ação direta com função nula
[Me_L]:legenda/*callback*...leg(ssi)_ação_funcrepetida.qtd	quantidade de legendas subsubitem do objeto menu com *callback* que chama ação direta com função repetida
[Me_L]:legenda/*callback*...leg(i)_cd_funcinex.qtd	quantidade de legendas item do objeto menu com *callback* que chama caixa de diálogo com função inexistente
[Me_L]:legenda/*callback*...leg(i)_cd_funcnula.qtd	quantidade de legendas item do objeto menu com *callback* que chama caixa de diálogo com função nula
[Me_L]:legenda/*callback*...leg(i)_cd_funcrepetida.qtd	quantidade de legendas item do objeto menu com *callback* que chama caixa de diálogo com função repetida
[Me_L]:legenda/*callback*...leg(si)_cd_funcinex.qtd	quantidade de legendas subitem do objeto menu com *callback* que chama caixa de diálogo com função inexistente
[Me_L]:legenda/*callback*...leg(si)_cd_funcnula.qtd	quantidade de legendas subitem do objeto menu com *callback* que chama caixa de diálogo com função nula
[Me_L]:legenda/*callback*...leg(si)_cd_funcrepetida.qtd	quantidade de legendas subitem do objeto menu com *callback* que chama caixa de diálogo com função repetida
[Me_L]:legenda/*callback*...leg(ssi)_cd_funcinex.qtd	quantidade de legendas subsubitem do objeto menu com *callback* que chama caixa de diálogo com função inexistente
[Me_L]:legenda/*callback*...leg(ssi)_cd_funcnula.qtd	quantidade de legendas subsubitem do objeto menu com *callback* que chama caixa de diálogo com função nula
[Me_L]:legenda/*callback*...leg(ssi)_cd_funcrepetida.qtd	quantidade de legendas subsubitem do objeto menu com *callback* que chama caixa de diálogo com função repetida

LEGENDA G2 e LEGENDA L1 (caracteres) e (título, subtítulo e subsubtítulo)	
[Me_G]:legenda...leg(t)_caracteres.qtd	quantidade de caracteres, por cada legenda título do objeto menu
[Me_G]:legenda...leg(t)_caracteres.qtd.min	valor mínimo entre as n ocorrências de *
[Me_G]:legenda...leg(t)_caracteres.qtd.max	valor máximo entre as n ocorrências de *
[Me_G]:legenda...leg(t)_caracteres.qtd.med	valor médio entre as n ocorrências de *
[Me_G]:legenda...leg(st)_caracteres.qtd	quantidade de caracteres, por cada legenda subtítulo do objeto menu
[Me_G]:legenda...leg(st)_caracteres.qtd.min	valor mínimo entre as n ocorrências de *
[Me_G]:legenda...leg(st)_caracteres.qtd.max	valor máximo entre as n ocorrências de *
[Me_G]:legenda...leg(st)_caracteres.qtd.med	valor médio entre as n ocorrências de *
[Me_G]:legenda...leg(sst)_caracteres.qtd	quantidade de caracteres, por cada legenda subsubtítulo do objeto menu
[Me_G]:legenda...leg(sst)_caracteres.qtd.min	valor mínimo entre as n ocorrências de *
[Me_G]:legenda...leg(sst)_caracteres.qtd.max	valor máximo entre as n ocorrências de *
[Me_G]:legenda...leg(sst)_caracteres.qtd.med	valor médio entre as n ocorrências de *
LEGENDA G2 e LEGENDA L2 (caracteres) e (item, subitem e subsubitem)	
[Me_G]:legenda...leg(i)_caracteres.qtd	quantidade de caracteres, por cada legenda item do objeto menu
[Me_G]:legenda...leg(i)_caracteres.qtd.min	valor mínimo entre as n ocorrências de *
[Me_G]:legenda...leg(i)_caracteres.qtd.max	valor máximo entre as n ocorrências de *
[Me_G]:legenda...leg(i)_caracteres.qtd.med	valor médio entre as n ocorrências de *
[Me_G]:legenda...leg(si)_caracteres.qtd	quantidade de caracteres, por cada legenda subitem do objeto menu
[Me_G]:legenda...leg(si)_caracteres.qtd.min	valor mínimo entre as n ocorrências de *
[Me_G]:legenda...leg(si)_caracteres.qtd.max	valor máximo entre as n ocorrências de *
[Me_G]:legenda...leg(si)_caracteres.qtd.med	valor médio entre as n ocorrências de *
[Me_G]:legenda...leg(ssi)_caracteres.qtd	quantidade de caracteres, por cada legenda subsubitem do objeto menu
[Me_G]:legenda...leg(ssi)_caracteres.qtd.min	valor mínimo entre as n ocorrências de *
[Me_G]:legenda...leg(ssi)_caracteres.qtd.max	valor máximo entre as n ocorrências de *
[Me_G]:legenda...leg(ssi)_caracteres.qtd.med	valor médio entre as n ocorrências de *

4.10.1.4 Contexto BF

O contexto Barra de Ferramentas apresenta métricas relacionadas ao objeto Barra de Ferramentas, divididas nas categorias Gráfica, Textual e Lógica, cada qual com seus respectivos focos. A categoria Gráfica (Tabela 11) aborda 1 foco: ícone (e.g., tamanho da fonte). A categoria Textual (Tabela 12) aborda 1 foco: ícone (e.g., ortografia da legenda do ícone). A categoria Lógica (Tabela 13) aborda 2 focos: *callback* (e.g., função inexistente) e ícone (e.g., *tooltip* associada ao ícone). Tais cruzamentos oferecem métricas de média complexidade (e.g., combinação padrão entre ícone e *tooltip* associada ao ícone, por cada grupo de ícones) a grande complexidade (e.g., sequências de combinação padrão, por cada grupo de ícones).

Tabela 11: 4G - contexto Barra de Ferramentas e categoria Gráfica

ÍCONE G1 (*TOOLTIP*): fonte	
[BF_G]:ícone...*tooltip*_fonte(tipo)	tipo de fonte aplicado à cada *tooltip* associada ao ícone do objeto barra de ferramentas
[BF_G]:ícone...*tooltip*_fonte(estilo)	estilo de fonte aplicado à cada *tooltip* associada ao ícone do objeto barra de ferramentas
[BF_G]:ícone...*tooltip*_fonte(tamanho)	tamanho de fonte aplicado à cada *tooltip* associada ao ícone do objeto barra de ferramentas
[BF_G]:ícone...*tooltip*_fonte(cor)	cor de fonte aplicado à cada *tooltip* associada ao ícone do objeto barra de ferramentas
ÍCONE G2 (*TOOLTIP*): caracteres	
[BF_G]:ícone...*tooltip*_caracteres.qtd	quantidade de caracteres, por cada *tooltip* associada ao ícone do objeto barra de ferramentas
[BF_G]:ícone...*tooltip*_caracteres.qtd.min	valor mínimo entre as n ocorrências de *
[BF_G]:ícone...*tooltip*_caracteres.qtd.max	valor máximo entre as n ocorrências de *
[BF_G]:ícone...*tooltip*_caracteres.qtd.med	valor médio entre as n ocorrências de *

ÍCONE G3 (LEGENDA): fonte	
[BF_G]:ícone...legenda_fonte(tipo)	tipo de fonte aplicado à legenda associada ao ícone do objeto barra de ferramentas
[BF_G]:ícone...legenda_fonte(estilo)	estilo de fonte aplicado à legenda associada ao ícone do objeto barra de ferramentas
[BF_G]:ícone...legenda_fonte(tamanho)	tamanho de fonte aplicado à legenda associada ao ícone do objeto barra de ferramentas
[BF_G]:ícone...legenda_fonte(cor)	cor de fonte aplicado à legenda associada ao ícone do objeto barra de ferramentas
ÍCONE G4 (LEGENDA): caracteres	
[BF_G]:ícone...legenda_caracteres.qtd	quantidade de caracteres, por cada legenda associada ao ícone do objeto barra de ferramentas
[BF_G]:ícone...legenda_caracteres.qtd.min	valor mínimo entre as n ocorrências de *
[BF_G]:ícone...legenda_caracteres.qtd.max	valor máximo entre as n ocorrências de *
[BF_G]:ícone...legenda_caracteres.qtd.med	valor médio entre as n ocorrências de *

Tabela 12: 4T - contexto Barra de Ferramentas e categoria Textual

ÍCONE T1 (*TOOLTIP*): ortografia, gramática, capitalização, abreviação e sinônimos	
[BF_T]:ícone...*tooltip*_conteúdo	conteúdo de cada uma das *tooltips* associadas aos ícones do objeto barra de ferramentas
[BF_T]:ícone...*tooltip*_ortografia.sn	valor sim ou não, *tooltip* associada ao ícone do objeto barra de ferramentas com erro de ortografia
[BF_T]:ícone...*tooltip*_ortografia.sn.qtd	quantidade de * = s
[BF_T]:ícone...*tooltip*_gramática1.sn	valor sim ou não, *tooltip* associada ao ícone do objeto barra de ferramentas com erro de gramática
[BF_T]:ícone...*tooltip*_gramática1.sn.qtd	quantidade de * = s
[BF_T]:ícone...*tooltip*_capitalização.sn	valor sim ou não, *tooltip* associada ao ícone do objeto barra de ferramentas com erro de capitalização
[BF_T]:ícone...*tooltip*_capitalização.sn.qtd	quantidade de * = s
[BF_T]:ícone...*tooltip*_abreviação.sn	valor sim ou não, *tooltip* associada ao ícone do objeto barra de ferramentas com erro de abreviação
[BF_T]:ícone...*tooltip*_abreviação.sn.qtd	quantidade de * = s
[BF_T]:ícone...*tooltip*_gramática2.sn	valor sim ou não, *tooltips* associadas aos ícones do objeto barra de ferramentas com erro de gramática entre si
[BF_T]:ícone...*tooltip*_gramática2.sn.qtd	quantidade de * = s
[BF_T]:ícone...*tooltip*_sinônimos.sn	valor sim ou não, *tooltips* associadas aos ícones do objeto barra de ferramentas são sinônimos
[BF_T]:ícone...*tooltip*_sinônimos.sn.qtd	quantidade de * = s
ÍCONE T2 (LEGENDA): ortografia, gramática, capitalização, abreviação e sinônimos	
[BF_T]:ícone...legenda_conteúdo	conteúdo de cada uma das legendas associadas aos ícones do objeto barra de ferramentas
[BF_T]:ícone...legenda_ortografia.sn	valor sim ou não, legenda associada ao ícone do objeto barra de ferramentas com erro de ortografia
[BF_T]:ícone...legenda_ortografia.sn.qtd	quantidade de * = s
[BF_T]:ícone...legenda_gramática1.sn	valor sim ou não, legenda associada ao ícone do objeto barra de ferramentas com erro de gramática
[BF_T]:ícone...legenda_gramática1.sn.qtd	quantidade de * = s
[BF_T]:ícone...legenda_capitalização.sn	valor sim ou não, legenda associada ao ícone do objeto barra de ferramentas com erro de capitalização
[BF_T]:ícone...legenda_capitalização.sn.qtd	quantidade de * = s
[BF_T]:ícone...legenda_abreviação.sn	valor sim ou não, legenda associada ao ícone do objeto barra de ferramentas com erro de abreviação
[BF_T]:ícone...legenda_abreviação.sn.qtd	quantidade de * = s
[BF_T]:ícone...legenda_gramática2.sn	valor sim ou não, legendas associadas aos ícones do objeto barra de ferramentas com erro de gramática entre si
[BF_T]:ícone...legenda_gramática2.sn.qtd	quantidade de * = s
[BF_T]:ícone...legenda_sinônimos.sn	valor sim ou não, legendas associadas aos ícones do objeto barra de ferramentas são sinônimos
[BF_T]:ícone...legenda_sinônimos.sn.qtd	quantidade de * = s

Tabela 13: 4L - contexto Barra de Ferramentas e categoria Lógica

CALLBACK L1: ação direta e caixa de diálogo	
[BF_L]:*callback*...ação.sn	valor sim ou não, *callback* do objeto barra de ferramentas chama ação direta
[BF_L]:*callback*...ação.sn.qtd	quantidade de * = s
[BF_L]:*callback*...cd.sn	valor sim ou não, *callback* do objeto barra de ferramentas chama caixa de diálogo
[BF_L]:*callback*...cd.sn.qtd	quantidade de * = s
CALLBACK L2: função inexistente, nula e repetida	
[BF_L]:*callback*...funcinex.sn	valor sim ou não, *callback* do objeto barra de ferramentas com função inexistente
[BF_L]:*callback*...funcinex.sn.qtd	quantidade de * = s
[BF_L]:*callback*...funcnula.sn	valor sim ou não, *callback* do objeto barra de ferramentas com função nula
[BF_L]:*callback*...funcnula.sn.qtd	quantidade de * = s
[BF_L]:*callback*...repetida.sn	valor sim ou não, *callback* do objeto barra de ferramentas com função repetida
[BF_L]:*callback*...repetida.sn.qtd	quantidade de * = s
[BF_L]:*callback*...repetição.sn.qtd	quantidade de * = s, por cada função
ÍCONE L1: padrão e repetido	
[BF_L]:ícone...conteúdo	conteúdo de cada um dos ícones do objeto barra de ferramentas
[BF_L]:ícone...qtd	quantidade de ícones do objeto barra de ferramentas
[BF_L]:ícone...padrão.sn	valor sim ou não, ícone do objeto barra de ferramentas é padrão; depende de lista padrão
[BF_L]:ícone...padrão.sn.qtd	quantidade de * = s
[BF_L]:ícone...repetido.sn	valor sim ou não, ícone do objeto barra de ferramentas é repetido
[BF_L]:ícone...repetido.sn.qtd	quantidade de * = s
[BF_L]:ícone...repetição.sn.qtd	quantidade de * = s, por cada ícone repetido
[BF_L]:ícone...comb_padrão.sn	valor sim ou não, combinação padrão entre ícone e *tooltip* associada ao ícone do objeto barra de ferramentas existe; depende de lista padrão; depende de [BF_L]:ícone...padrão.sn = s e [BF_L]:ícone...*tooltip*_padrão.sn = s
[BF_L]:ícone...comb_padrão.sn.qtd	quantidade de * = s
[BF_L]:ícone...seq_comb_padrão.sn	valor sim ou não, sequência entre combinações padrão do objeto barra de ferramentas existe; depende de lista padrão; depende de [BF_L]:ícone...comb_padrão.sn = s
[BF_L]:ícone...seq_comb_padrão.sn.qtd	quantidade de * = s
ÍCONE L2: grupo	
[BF_L]:ícone...grupo.sn	valor sim ou não, o ícone do objeto barra de ferramentas pertence a um grupo de ícones
[BF_L]:ícone...grupo.sn.qtd	quantidade de grupos de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...grupo.sn = s
ÍCONE L3 (_TOOLTIP_): padrão e repetida	
[BF_L]:ícone...*tooltip*.sn	valor sim ou não, *tooltip* associada ao ícone do objeto barra de ferramentas existe
[BF_L]:ícone...*tooltip*.sn.qtd	quantidade de * = s
[BF_L]:ícone...*tooltip*_conteúdo	conteúdo de cada uma das *tooltips* associadas aos ícones do objeto barra de ferramentas
[BF_L]:ícone...*tooltip*_padrão.sn	valor sim ou não, *tooltip* associada aos ícone do objeto barra de ferramentas é padrão; depende de lista padrão; depende de [BF_L]:ícone...*tooltip*.sn = s
[BF_L]:ícone...*tooltip*_padrão.sn.qtd	quantidade de * = s
[BF_L]:ícone...*tooltip*_repetida.sn	valor sim ou não, *tooltip* associada ao ícone do objeto barra de ferramentas é repetida
[BF_L]:ícone...*tooltip*_repetida.sn.qtd	quantidade de * = s
[BF_L]:ícone...*tooltip*_repetição.sn.qtd	quantidade de * = s, por cada ícone repetida
ÍCONE L4 (LEGENDA): padrão e repetida	
[BF_L]:ícone...legenda.sn	valor sim ou não, legenda associada ao ícone do objeto barra de ferramentas existe
[BF_L]:ícone...legenda.sn.qtd	quantidade de * = s
[BF_L]:ícone...legenda_conteúdo	conteúdo de cada uma das legendas associadas aos ícones do objeto barra de ferramentas
[BF_L]:ícone...legenda_padrão.sn	valor sim ou não, legenda associada aos ícone do objeto barra de ferramentas é padrão; depende de lista padrão; depende de [BF_L]:ícone...legenda.sn = s
[BF_L]:ícone...legenda_padrão.sn.qtd	quantidade de * = s
[BF_L]:ícone...legenda_repetida.sn	valor sim ou não, legenda associada ao ícone do objeto barra de ferramentas é repetida
[BF_L]:ícone...legenda_repetida.sn.qtd	quantidade de * = s
[BF_L]:ícone...legenda_repetição.sn.qtd	quantidade de * = s, por cada legenda repetida

***CALLBACK* L3: *CALLBACK* L1 e *CALLBACK* L2** **(ação direta e caixa de diálogo) e (função inexistente, nula e repetida)**	
[BF_L]:*callback*...ação_funcinex.sn	valor sim ou não, *callback* do objeto barra de ferramentas chama ação direta com função inexistente
[BF_L]:*callback*...ação_funcinex.sn.qtd	quantidade de * = s
[BF_L]:*callback*...ação_funcnula.sn	valor sim ou não, *callback* do objeto barra de ferramentas chama ação direta com função nula
[BF_L]:*callback*...ação_funcnula.sn.qtd	quantidade de * = s
[BF_L]:*callback*...ação_funcrepetida.sn	valor sim ou não, *callback* do objeto barra de ferramentas chama ação direta com função repetida
[BF_L]:*callback*...ação_funcrepetida.sn.qtd	quantidade de * = s
[BF_L]:*callback*...ação_funcrepetição.sn.qtd	quantidade de * = s, por cada função repetida
[BF_L]:*callback*...cd_funcinex.sn	valor sim ou não, *callback* do objeto barra de ferramentas chama caixa de diálogo com função inexistente
[BF_L]:*callback*...cd_funcinex.sn.qtd	quantidade de * = s
[BF_L]:*callback*...cd_funcnula.sn	valor sim ou não, *callback* do objeto barra de ferramentas chama caixa de diálogo com função nula
[BF_L]:*callback*...cd_funcnula.sn.qtd	quantidade de * = s
[BF_L]:*callback*...cd_funcrepetida.sn	valor sim ou não, *callback* do objeto barra de ferramentas chama caixa de diálogo com função repetida
[BF_L]:*callback*...cd_funcrepetida.sn.qtd	quantidade de * = s
[BF_L]:*callback*...cd_funcrepetição.sn.qtd	quantidade de * = s, por cada função repetida

ÍCONE L5: ÍCONE L1 e ÍCONE L2 **(padrão e repetido) e (grupo)**	
[BF_L]:ícone...ícone_grupo.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...ícone_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...ícone_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...ícone_grupo.qtd.med	valor médio entre as n ocorrências de *
[BF_L]:ícone...padrão_grupo.qtd	quantidade de ícones padrão, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...padrão.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...padrão_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...padrão_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...padrão_grupo.qtd.med	valor médio entre as n ocorrências de *
[BF_L]:ícone...repetido_grupo.qtd	quantidade de ícones repetidos, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...repetido.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...repetido_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...repetido_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...repetido_grupo.qtd.med	valor médio entre as n ocorrências de *
[BF_L]:ícone...comb_padrão_grupo.qtd	quantidade de combinações padrão entre ícone e *tooltip* associada ao ícone, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...comb_padrão.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...comb_padrão_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...comb_padrão_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...comb_padrão_grupo.qtd.med	valor médio entre as n ocorrências de *
[BF_L]:ícone...seq_comb_padrão_grupo.qtd	quantidade de sequências de combinação padrão, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:seq_comb_padrão.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...seq_comb_padrão_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...seq_comb_padrão_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...seq_comb_padrão_grupo.qtd.med	valor médio entre as n ocorrências de *

ÍCONE L6: ÍCONE L3 e ÍCONE L2 **(*tooltip*) e (grupo)**	
[BF_L]:ícone...*tooltip*_grupo.qtd	quantidade de *tooltips* associadas aos ícones, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...*tooltip*.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...*tooltip*_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...*tooltip*_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...*tooltip*_grupo.qtd.med	valor médio entre as n ocorrências de *
[BF_L]:ícone...*tooltip*_padrão_grupo.qtd	quantidade de *tooltips* padrão associadas aos ícones, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...*tooltip*_padrão.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...*tooltip*_padrão_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...*tooltip*_padrão_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...*tooltip*_padrão_grupo.qtd.med	valor médio entre as n ocorrências de *
[BF_L]:ícone...*tooltip*_repetida_grupo.sn	quantidade de *tooltips* repetidas associadas aos ícones, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...*tooltip*_repetida.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...*tooltip*_repetida_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...*tooltip*_repetida_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...*tooltip*_repetida_grupo.qtd.med	valor médio entre as n ocorrências de *

ÍCONE L7: ÍCONE L4 e ÍCONE L2 (legenda) e (grupo)	
[BF_L]:ícone...legenda_grupo.qtd	quantidade de legendas associadas aos ícones, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...legenda.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...legenda_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...legenda_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...legenda_grupo.qtd.med	valor médio entre as n ocorrências de *
[BF_L]:ícone...legenda_padrão_grupo.qtd	quantidade de legendas padrão associadas aos ícones, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...legenda_padrão.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...legenda_padrão_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...legenda_padrão_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...legenda_padrão_grupo.qtd.med	valor médio entre as n ocorrências de *
[BF_L]:ícone...legenda_repetida_grupo.sn	quantidade de legendas repetidas associadas aos ícones, por cada grupo de ícones do objeto barra de ferramentas; depende de [BF_L]:ícone...legenda_repetida.sn = s e [BF_L]:ícone...grupo.sn = s
[BF_L]:ícone...legenda_repetida_grupo.qtd.min	valor mínimo entre as n ocorrências de *
[BF_L]:ícone...legenda_repetida_grupo.qtd.max	valor máximo entre as n ocorrências de *
[BF_L]:ícone...legenda_repetida_grupo.qtd.med	valor médio entre as n ocorrências de *

ÍCONE L2 e *CALLBACK* L1 (grupo) e (ação direta e caixa de diálogo)	
[BF_L]:ícone/*callback*...grupo_ação.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama ação direta
[BF_L]:ícone/*callback*...grupo_cd.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama caixa de diálogo
ÍCONE L2 e *CALLBACK* L2 (grupo) e (função inexistente, nula e repetida)	
[BF_L]:ícone/*callback*...grupo_funcinex.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas com função inexistente
[BF_L]:ícone/*callback*...grupo_funcnula.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas com função nula
[BF_L]:ícone/*callback*...grupo_funcrepetida.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas com função repetida
ÍCONE L2 e *CALLBACK* L3 (grupo), (ação direta e caixa de diálogo) e (função inexistente, nula e repetida)	
[BF_L]:ícone/*callback*...grupo_ação_funcinex.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama ação direta com função inexistente
[BF_L]:ícone/*callback*...grupo_ação_funcnula.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama ação direta com função nula
[BF_L]:ícone/*callback*...grupo_ação_funcrepetida.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama ação direta com função repetida
[BF_L]:ícone/*callback*...grupo_cd_funcinex.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama caixa de diálogo com função inexistente
[BF_L]:ícone/*callback*...grupo_cd_funcnula.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama caixa de diálogo com função nula
[BF_L]:ícone/*callback*...grupo_cd_funcrepetida.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama caixa de diálogo com função repetida

4.10.2. Contexto nOBJETO/CD

O contexto nOBJETO/CD é uma generalização dos contextos referentes a n ocorrências de um tipo de objeto presentes numa caixa de diálogo. Tal contexto aborda n ocorrências do objeto botão de comando (nBC/CD) e n ocorrências do objeto barra de ferramentas (nBF/CD). A seguir encontra-se a descrição de cada um desses contextos específicos e suas respectivas métricas.

4.10.2.1 Contexto nBC/CD

Para as tabelas seguintes, referentes ao contexto nBC/CD (dois ou mais Botões de Comando por caixa de diálogo), prevê-se a relação entre todos os botões que estejam dentro de uma única caixa de diálogo. Como já foi explicado anteriormente, esse contexto fará uso de métricas já estabelecidas no contexto BC.

A categoria Gráfica (Tabela 14) aborda 2 focos: geral (e.g., razão de aspecto) e legenda (e.g., tamanho da fonte). Para exemplificar o foco geral, razão de aspecto e quadrado entre os n botões, apresenta-se a Figura 28, em que existem diferenças claras entre os botões de comando "*Add*" e "*Delete*" quando comparados com "*OK*" e "*Cancel*".

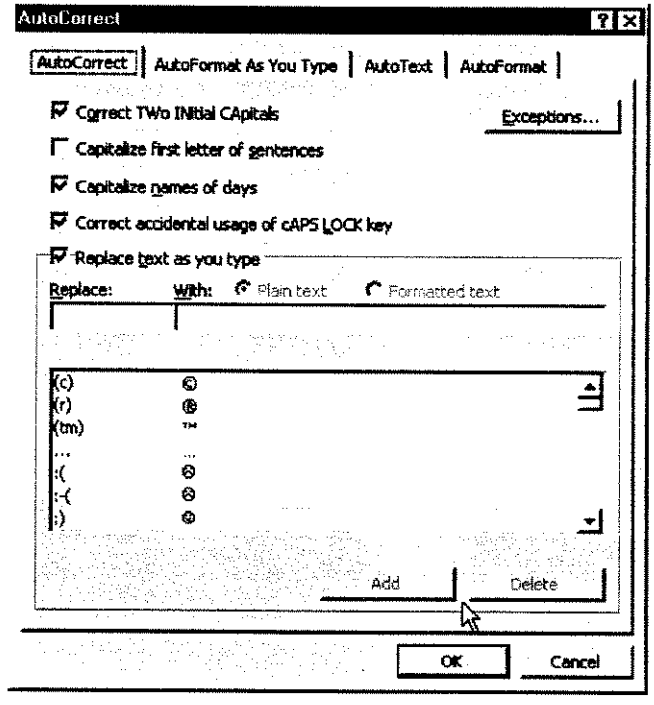

Figura 28: Exemplo de n botões de comando presentes numa mesma caixa de diálogo

A categoria Textual (Tabela 15) aborda 1 foco: legenda (e.g., ortografia). A categoria Lógica (Tabela 16) aborda 3 focos: geral (e.g., localização padrão do grupo de botões de comando), *callback* (e.g., função inexistente) e legenda (e.g., legenda padrão e sequência padrão). Para exemplificar o foco legenda, legenda padrão e sequência padrão, a Figura 28 mostra os botões "*OK*" e "*Cancel*". Além disso, os botões "*OK*" e "*Cancel*" fazem parte de um grupo padrão, que possue localização padrão (caso o *designer* tenha definido a posição inferior direita como a localização ideal para esse grupo padrão). Os botões "*Add*" e "*Delete*" poderão vir a ser considerados legenda padrão, desde que estejam inseridos na lista padrão que a métrica consiste no contexto BC.

Tabela 14: 5G - contexto dois ou mais Botões de Comando por caixa de diálogo e categoria Gráfica

GERAL	
[nBC_G]:...qtd	quantidade de botões de comando presentes na caixa de diálogo
[nBC_G]:...razão_aspecto.sn	valor sim ou não, existe diferença na razão de aspecto entre as n ocorrências da métrica [BC_G]:...razão_aspecto
[nBC_G]:...razão_aspecto.sn.qtd	quantidade de * = s
[nBC_G]:...razão_aspecto.min	valor mínimo entre as n ocorrências da métrica [BC_G]:...razão_aspecto
[nBC_G]:...razão_aspecto.max	valor máximo entre as n ocorrências da métrica [BC_G]:...razão_aspecto
[nBC_G]:...razão_aspecto.med	valor médio entre as n ocorrências da métrica [BC_G]:...razão_aspecto
[nBC_G]:...quadrado.sn	valor sim ou não, existe diferença no quadrado entre as n ocorrências da métrica [BC_G]:...quadrado
[nBC_G]:...quadrado.sn.qtd	quantidade de * = s
[nBC_G]:...quadrado.min	valor mínimo entre as n ocorrências da métrica [BC_G]:...quadrado
[nBC_G]:...quadrado.max	valor máximo entre as n ocorrências da métrica [BC_G]:...quadrado
[nBC_G]:...quadrado.med	valor médio entre as n ocorrências da métrica [BC_G]:...quadrado
[nBC_G]:...cor_dif.sn	valor sim ou não, existe diferença de cores entre as n ocorrências da métrica [BC_G]:...cor
[nBC_G]:...cor_dif.sn.qtd	quantidade de * = s
[nBC_G]:...sobreposição.sn	valor sim ou não, existe sobreposição entre as n ocorrências da métrica [BC_G]:...localização
[nBC_G]:...sobreposição.sn.qtd	quantidade de * = s
[nBC_G]:...desalinh_h.sn	valor sim ou não, existe desalinhamento horizontal entre as n ocorrências da métrica [BC_G]:...localização, quando os n botões de comando estão um ao lado do outro na caixa de diálogo
[nBC_G]:...desalinh_h.sn.qtd	quantidade de * = s
[nBC_G]:...desalinh_v.sn	valor sim ou não, existe desalinhamento vertical entre as n ocorrências da métrica [BC_G]:...localização, quando os n botões de comando estão um acima do outro na caixa de diálogo
[nBC_G]:...desalinh_v.sn.qtd	quantidade de * = s
LEGENDA G1: fonte	
[nBC_G]:legenda...fonte(tipo_dif).sn	valor sim ou não, existe diferença de tipos de fonte entre os n objetos botão de comando presentes na caixa de diálogo
[nBC_G]:legenda...fonte(tipo_dif).sn.qtd	quantidade de * = s
[nBC_G]:legenda...fonte(estilo_dif).sn	valor sim ou não, existe diferença de estilos de fonte entre os n objetos botão de comando presentes na caixa de diálogo
[nBC_G]:legenda...fonte(estilo_dif).sn.qtd	quantidade de * = s
[nBC_G]:legenda...fonte(tamanho_dif).sn	valor sim ou não, existe diferença de tamanhos de fonte entre os n objetos botão de comando presentes na caixa de diálogo
[nBC_G]:legenda...fonte(tamanho_dif).sn.qtd	quantidade de * = s
[nBC_G]:legenda...fonte(cor_dif).sn	valor sim ou não, existe diferença de cores de fonte entre os n objetos botão de comando presentes na caixa de diálogo
[nBC_G]:legenda...fonte(cor_dif).sn.qtd	quantidade de * = s
LEGENDA G2: caracteres	
[nBC_G]:legenda...caracteres.qtd.min	valor mínimo entre as n ocorrências da métrica [BC_G]:legenda...caracteres.qtd
[nBC_G]:legenda...caracteres.qtd.max	valor máximo entre as n ocorrências da métrica [BC_G]:legenda...caracteres.qtd
[nBC_G]:legenda...caracteres.qtd.med	valor médio entre as n ocorrências da métrica [BC_G]:legenda...caracteres.qtd

Tabela 15: 5T - contexto dois ou mais Botões de Comando por caixa de diálogo e categoria Textual

LEGENDA	
[nBC_T]:legenda...ortografia.sn.qtd	quantidade de erros de ortografia entre as n ocorrências da métrica [BC_T]:legenda...ortografia.sn
[nBC_T]:legenda...gramática1.sn.qtd	quantidade de erros de gramática entre as n ocorrências da métrica [BC_T]:legenda...gramática1.sn
[nBC_T]:legenda...capitalização.sn.qtd	quantidade de erros de capitalização entre as n ocorrências da métrica [BC_T]:legenda...capitalização.sn
[nBC_T]:legenda...abreviação.sn.qtd	quantidade de erros de abreviação entre as n ocorrências da métrica [BC_T]:legenda...abreviação.sn
[nBC_T]:legenda...gramática2.sn	valor sim ou não, existem erros de concordância gramatical entre as legendas dos n objetos botão de comando
[nBC_T]:legenda...gramática2.sn.qtd	quantidade de * = s
[nBC_T]:legenda...sinônimos.sn	valor sim ou não, existem sinônimos entre as legendas dos n objetos botão de comando
[nBC_T]:legenda...sinônimos.sn.qtd	quantidade de * = s

Tabela 16: 5L - contexto dois ou mais Botões de Comando por caixa de diálogo e categoria Lógica

GERAL	
[nBC_L]:...grupo_padrão.sn	valor sim ou não, grupo padrão de botões de comando existe (dentro da caixa de diálogo); depende de lista padrão (e.g., *ok*, *cancel*)
[nBC_L]:...grupo_padrão.sn.qtd	quantidade de * = s
[nBC_L]:...grupo_padrão_localização	coordenadas xy dos quatro vértices do grupo padrão de botões de comando (em relação à sua caixa de diálogo); depende de [nBC_L]:...grupo_padrão.sn = s
***CALLBACK* L1: ação direta e caixa de diálogo**	
[nBC_L]:*callback*...ação.sn.qtd	quantidade de *callback*s com chamada de ação direta entre as n ocorrências da métrica [BC_L]:*callback*...ação.sn
[nBC_L]:*callback*...cd.sn.qtd	quantidade de *callback*s com chamada de caixa de diálogo entre as n ocorrências da métrica [BC_L]:*callback*...cd.sn
***CALLBACK* L2: função inexistente e nula**	
[nBC_L]:*callback*...funcinex.sn.qtd	quantidade de *callback*s com função inexistente entre as n ocorrências da métrica [BC_L]:*callback*...funcinex.sn
[nBC_L]:*callback*...funcnula.sn.qtd	quantidade de *callback*s com função nula entre as n ocorrências da métrica [BC_L]:*callback*...funcnula.sn
LEGENDA L1: 3pontos, padrão, sublinhado, repetida, sublinhado padrão, sublinhado repetido, combinação padrão e sequência padrão	
[nBC_L]:legenda...3pontos.sn.qtd	quantidade de legendas com três pontos entre as n ocorrências da métrica [BC_L]:legenda...3pontos.sn
[nBC_L]:legenda...padrão.sn.qtd	quantidade de legendas padrão entre as n ocorrências da métrica [BC_L]:legenda...padrão.sn
[nBC_L]:legenda...sublinhado.sn.qtd	quantidade de legendas com caracter sublinhado entre as n ocorrências da métrica [BC_L]:legenda...sublinhado.sn
[nBC_L]:legenda...repetida.sn	valor sim ou não, existem legendas repetidas entre os n objetos botão de comando
[nBC_L]:legenda...repetida.sn.qtd	quantidade de * = s; depende de [nBC_L]:legenda...repetida.sn = s
[nBC_L]:legenda...repetição.sn.qtd	quantidade de * = s, por cada legenda; depende de [nBC_L]:legenda...repetida.sn = s
[nBC_L]:legenda...sublinhado_padrão.sn.qtd	quantidade de legendas com caracter sublinhado padrão entre as n ocorrências da métrica [BC_L]:legenda...sublinhado_padrão.sn
[nBC_L]:legenda...sublinhado_repetido.sn	valor sim ou não, existem legendas com caracter sublinhado repetido entre os n objetos botão de comando
[nBC_L]:legenda...sublinhado_repetido.sn.qtd	quantidade de * = s
[nBC_L]:legenda...sublinhado_repetição.sn.qtd	quantidade de * = s, por cada caracter sublinhado repetido
[nBC_L]:legenda...comb_padrão.sn.qtd	quantidade de combinações padrão entre as n ocorrências da métrica [nBC_L]:legenda...comb_padrão.sn
[nBC_L]:legenda...seqpadrão.sn	valor sim ou não, sequência padrão entre as combinações padrão existe; depende de lista padrão (e.g., *ok*, *cancel*)
[nBC_L]:legenda...seqpadrão.sn.qtd	quantidade de * = s

***CALLBACK* L3: *CALLBACK* L2 e *CALLBACK* L1 (função inexistente e nula) e (ação direta e caixa de diálogo)**	
[nBC_L]:*callback*...ação_funcinex.sn.qtd	quantidade de *callback*s que chamam ação direta com função inexistente entre as n ocorrências da métrica [nBC_L]:*callback*...ação_funcinex.sn
[nBC_L]:*callback*...ação_funcnula.sn.qtd	quantidade de *callback*s que chamam ação direta com função nula entre as n ocorrências da métrica [nBC_L]:*callback*...ação_funcnula.sn
[nBC_L]:*callback*...cd_funcinex.sn.qtd	quantidade de *callback*s que chamam caixa de diálogo com função inexistente entre as n ocorrências da métrica [nBC_L]:*callback*...cd_funcinex.sn
[nBC_L]:*callback*...cd_funcnula.sn.qtd	quantidade de *callback*s que chamam caixa de diálogo com função nula entre as n ocorrências da métrica [nBC_L]:*callback*...cd_funcnula.sn
***CALLBACK* L1 e LEGENDA L1 (caixa de diálogo) e (3pontos)**	
[nBC_L]:*callback*/legenda...cd_3pontos.sn.qtd	quantidade de *callback*s que chamam caixa de diálogo e tem legenda com 3 pontos

4.10.2.2 Contexto nBF/CD

Para as tabelas seguintes, referentes ao contexto nBF/CD (duas ou mais Barras de Ferramentas por Caixa de Diálogo), prevê-se a relação entre todos as barras de ferramentas que estejam dentro da caixa de diálogo principal (no Word97, cada barra de ferramentas é identidicada pelas 2 colunas presentes no seu lado esquerdo). Como já foi explicado anteriormente, esse contexto fará uso de métricas já estabelecidas no contexto BF.

A categoria Gráfica (Tabela 17) aborda 1 foco: ícone (e.g., diferença de tamanho de fontes entre as *tooltips* dos ícones). A categoria Textual (Tabela 18) aborda 1 foco: ícone (e.g., ortografia da *tooltip* associada ao ícone). A categoria Lógica (Tabela 19) aborda 2 focos: *callback* (e.g., função inexistente) e ícone (e.g., padrão e quantidade de grupos). Para exemplificar o foco ícone, padrão e quantidade de grupos, pode-se ver pela Figura 29 que existem duas barras de ferramentas, respectivamente "*Standard*" e "*Formatting*", que normalmente estão dispostas para o usuário. Diversos ícones são padrão (e.g., "*New*", "*Open*", "*Save*", "*Print*"). Como observação, ressalta-se que entre essas barras não existem ícones com legenda e também não existe nenhum ícone repetido.

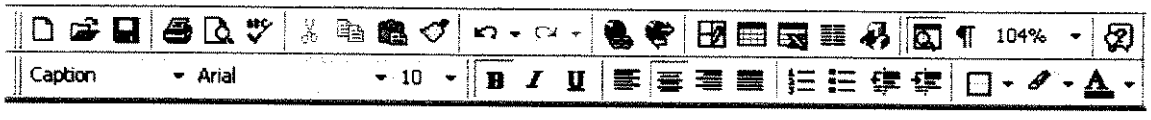

Figura 29: Exemplo de 2 barras de ferramentas

Mas também existem outras barras de ferramentas que podem eventualmente estar dispostas em conjunto com essas duas anteriores, entre elas, como pode ser visto na Figura 30, as barras de ferramentas "*Tables and Borders*", "*Forms*" e "*Reviewing*". Nessa figura, pode-se perceber a repetição do primeiro ícone ("*Draw Table*") da primeira barra no quinto ícone da segunda.

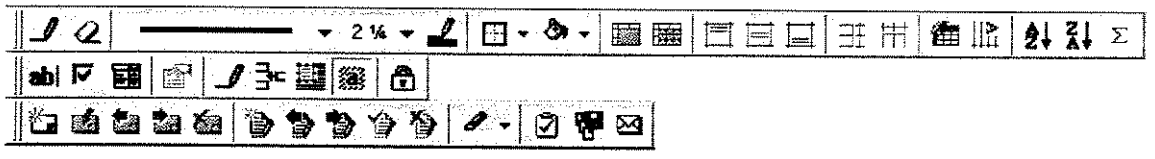

Figura 30: Exemplo de outras 3 barras de ferramentas

Através da Figura 31, pode-se perceber a quantidade de grupos, bem como a quantidade mínima de ícones por cada grupo de ícones entre todas as barras. A quantidade varia de dois (e.g., "*Undo*" e "*Redo*"), na primeira barra ("*Standard*"), a cinco (e.g., "*Draw Table*", "*Insert Table*", "*Insert Frame*", "*Form Field Shading*" e "*Protect Form*"), na quarta barra ("*Forms*"). Além disso, todos os ícones possuem *tooltip* e nenhum ícone possui legenda associada.

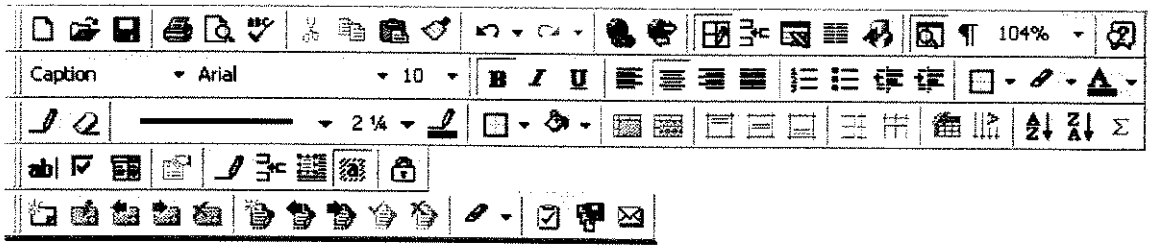

Figura 31: Exemplo com 5 barras de ferramentas

A seguir, as tabelas:

Tabela 17: 6G - contexto duas ou mais Barras de Ferramentas por caixa de diálogo e categoria Gráfica

ÍCONE G1 (*TOOLTIP*): fonte	
[nBF_G]:ícone...*tooltip*_fonte(tipo_dif).sn	valor sim ou não, existe diferença de tipos de fonte entre os n objetos barra de ferramentas presentes na caixa de diálogo
[nBF_G]:ícone...*tooltip*_fonte(tipo_dif).sn.qtd	quantidade de * = s
[nBF_G]:ícone...*tooltip*_fonte(estilo_dif).sn	valor sim ou não, existe diferença de estilos de fonte entre os n objetos barra de ferramentas presentes na caixa de diálogo
[nBF_G]:ícone...*tooltip*_fonte(estilo_dif).sn.qtd	quantidade de * = s
[nBF_G]:ícone...*tooltip*_fonte(tamanho_dif).sn	valor sim ou não, existe diferença de tamanhos de fonte entre os n objetos barra de ferramentas presentes na caixa de diálogo
[nBF_G]:ícone...*tooltip*_fonte(tamanho_dif).sn.qtd	quantidade de * = s
[nBF_G]:ícone...*tooltip*_fonte(cor_dif).sn	valor sim ou não, existe diferença de cores de fonte entre os n objetos barra de ferramentas presentes na caixa de diálogo
[nBF_G]:ícone...*tooltip*_fonte(cor_dif).sn.qtd	quantidade de * = s
ÍCONE G2 (*TOOLTIP*): caracteres	
[nBF_G]:ícone...*tooltip*_caracteres.qtd.min.min	valor mínimo da quantidade de caracteres entre as n métricas [BF_G]:ícone...*tooltip*_caracteres.qtd.min
[nBF_G]:ícone...*tooltip*_caracteres.qtd.max.max	valor máximo da quantidade de caracteres entre as n métricas [BF_G]:ícone...*tooltip*_caracteres.qtd.max
[nBF_G]:ícone...*tooltip*_caracteres.qtd.med.med	valor médio da quantidade de caracteres entre as n métricas [BF_G]:ícone...*tooltip*_caracteres.qtd.med
ÍCONE G3 (LEGENDA): fonte	
[nBF_G]:ícone...legenda_fonte(tipo_dif).sn	valor sim ou não, existe diferença de tipos de fonte entre os n objetos barra de ferramentas presentes na caixa de diálogo
[nBF_G]:ícone...legenda_fonte(tipo_dif).sn.qtd	quantidade de * = s
[nBF_G]:ícone...legenda_fonte(estilo_dif).sn	valor sim ou não, existe diferença de estilos de fonte entre os n objetos barra de ferramentas presentes na caixa de diálogo
[nBF_G]:ícone...legenda_fonte(estilo_dif).sn.qtd	quantidade de * = s
[nBF_G]:ícone...legenda_fonte(tamanho_dif).sn	valor sim ou não, existe diferença de tamanhos de fonte entre os n objetos barra de ferramentas presentes na caixa de diálogo
[nBF_G]:ícone...legenda_fonte(tamanho_dif).sn.qtd	quantidade de * = s
[nBF_G]:ícone...legenda_fonte(cor_dif).sn	valor sim ou não, existe diferença de cores de fonte entre os n objetos barra de ferramentas presentes na caixa de diálogo
[nBF_G]:ícone...legenda_fonte(cor_dif).sn.qtd	quantidade de * = s

ÍCONE G4 (LEGENDA): caracteres	
[nBF_G]:ícone...legenda_caracteres.qtd.min	valor mínimo da quantidade de caracteres entre as n métricas [BF_G]:ícone...legenda_caracteres.qtd.min
[nBF_G]:ícone...legenda_caracteres.qtd.max	valor máximo da quantidade de caracteres entre as n métricas [BF_G]:ícone...legenda_caracteres.qtd.max
[nBF_G]:ícone...legenda_caracteres.qtd.med	valor médio da quantidade de caracteres entre as n métricas [BF_G]:ícone...legenda_caracteres.qtd.med

Tabela 18: 6T - contexto duas ou mais Barras de Ferramentas por caixa de diálogo e categoria Textual

ÍCONE T1 (*TOOLTIP*): ortografia, gramática, capitalização, abreviação e sinônimos	
[nBF_T]:ícone...*tooltip*_ortografia.sn.qtd	quantidade de erros de ortografia entre as n métricas [BF_T]: ícone...*tooltip*_ortografia.sn.qtd
[nBF_T]:ícone...*tooltip*_gramática1.sn.qtd	quantidade de erros de gramática entre as n métricas [BF_T]: ícone...*tooltip*_gramática1.sn.qtd
[nBF_T]:ícone...*tooltip*_capitalização.sn.qtd	quantidade de erros de capitalização entre as n métricas [BF_T]: ícone...*tooltip*_capitalização.sn.qtd
[nBF_T]:ícone...*tooltip*_abreviação.sn.qtd	quantidade de erros de abreviação entre as n métricas [BF_T]:ícone...*tooltip*_abreviação.sn.qtd
[nBF_T]:ícone...*tooltip*_gramática2	valor sim ou não, existe concordância gramatical entre as *tooltips* dos n objetos barra de ferramentas
[nBF_T]:ícone...*tooltip*_gramática2.qtd	quantidade de * = s
[nBF_T]:ícone...*tooltip*_sinônimos	valor sim ou não, existem sinônimos entre as *tooltips* dos n objetos barra de ferramentas
[nBF_T]:ícone...*tooltip*_sinônimos.qtd	quantidade de * = s
ÍCONE T2 (LEGENDA): ortografia, gramática, capitalização, abreviação e sinônimos	
[nBF_T]:ícone...legenda_ortografia.sn.qtd	quantidade de erros de ortografia entre as n ocorrências da métrica [BF_T]:ícone...legenda_ortografia.sn.qtd
[nBF_T]:ícone...legenda_gramática1.sn.qtd	quantidade de erros de gramática entre as n ocorrências da métrica [BF_T]:ícone...legenda_gramática1.sn.qtd
[nBF_T]:ícone...legenda_capitalização.sn.qtd	quantidade de erros de capitalização entre as n ocorrências da métrica [BF_T]:ícone...legenda_capitalização.sn.qtd
[nBF_T]:ícone...legenda_abreviação.sn.qtd	quantidade de erros de abreviação entre as n ocorrências da métrica [BF_T]:ícone...legenda_abreviação.sn.qtd
[nBF_T]:ícone...legenda_gramática2	valor sim ou não, existe concordância gramatical entre as legendas dos n objetos barra de ferramentas
[nBF_T]:ícone...legenda_gramática2.qtd	quantidade de * = s
[nBF_T]:ícone...legenda_sinônimos	valor sim ou não, existem sinônimos entre as legendas dos n objetos barra de ferramentas
[nBF_T]:ícone...legenda_sinônimos.qtd	quantidade de * = s

Tabela 19: 6L - contexto duas ou mais Barras de Ferramentas por caixa de diálogo e categoria Lógica

CALLBACK L1: ação direta e caixa de diálogo	
[nBF_L]:*callback*...ação.sn.qtd	quantidade de *callback*s com chamada de ação direta entre as n ocorrências da métrica [BF_L]:*callback*...ação.sn.qtd
[nBF_L]:*callback*...cd.sn.qtd	quantidade de *callback*s com chamada de caixa de diálogo entre as n ocorrências da métrica [BF_L]:*callback*...cd.sn.qtd
CALLBACK L2: função inexistente, nula e repetida	
[nBF_L]:*callback*...funcinex.sn.qtd	quantidade de *callback*s com função inexistente entre as n ocorrências da métrica [BF_L]:*callback*...funcinex.sn.qtd
[nBF_L]:*callback*...funcnula.sn.qtd	quantidade de *callback*s com função nula entre as n ocorrências da métrica [BF_L]:*callback*...funcnula.sn.qtd
[nBF_L]:*callback*...funcrepetida.sn.qtd	quantidade de *callback*s com função repetida entre as n ocorrências da métrica [BF_L]:*callback*...funcrepetida.sn.qtd
[nBF_L]:*callback*...funcrepetição.sn.qtd	quantidade de repetições, por cada *callback* entre as n ocorrências da métrica [BF_L]:*callback*...funcrepetição.sn.qtd
ÍCONE L1: padrão e repetido	
[nBF_L]:ícone...qtd	quantidade de ícones entre as n ocorrências da métrica [BF_L]:ícone...qtd
[nBF_L]:ícone...padrão.sn.qtd	quantidade de ícones padrão entre as n ocorrências da métrica [BF_L]:ícone...padrão.sn.qtd
[nBF_L]:ícone...repetido.qtd	quantidade de ícones repetidos entre as n ocorrências da métrica [BF_L]:ícone...repetido.sn.qtd
[nBF_L]:ícone...repetição.qtd	quantidade de repetições, por cada ícone entre as n ocorrências da métrica [BF_L]:ícone...repetição.sn.qtd
[nBF_L]:ícone...comb_padrão.sn.qtd	quantidade de combinações padrão entre as n ocorrências da métrica [BF_L]:ícone...comb_padrão.sn.qtd
[nBF_L]:ícone...seq_comb_padrão.sn.qtd	quantidade de sequências de combinação padrão entre as n ocorrências da métrica [BF_L]:ícone...seq_comb_padrão.sn.qtd
ÍCONE L2: grupo	
[nBF_L]:ícone...grupo.sn.qtd.min	valor mínimo das n ocorrências da métrica [BF_L]:ícone...grupo.sn.qtd
[nBF_L]:ícone...grupo.sn.qtd.max	valor máximo das n ocorrências da métrica [BF_L]:ícone...grupo.sn.qtd
[nBF_L]:ícone...grupo.sn.qtd.med	valor médio das n ocorrências da métrica [BF_L]:ícone...grupo.sn.qtd
ÍCONE L3 (_TOOLTIP_): padrão e repetida	
[nBF_L]:ícone...*tooltip*.sn.qtd	quantidade *tooltips* associados aos ícones existentes entre as n métricas [BF_L]:ícone...legenda.sn.qtd
[nBF_L]:ícone...*tooltip*_padrão.sn.qtd	quantidade de *tooltips* padrão entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_padrão.sn.qtd
[nBF_L]:ícone...*tooltip*_repetido.sn.qtd	quantidade de *tooltips* repetidos entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_repetido.sn.qtd
[nBF_L]:ícone...*tooltip*_repetição.sn.qtd	quantidade de repetições, por cada *tooltip* entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_repetição.sn.qtd
ÍCONE L4 (LEGENDA): padrão e repetida	
[nBF_L]:ícone...legenda.sn.qtd	quantidade legendas associadas aos ícones existentes entre as n métricas [BF_L]:ícone...legenda.sn.qtd
[nBF_L]:ícone...legenda_padrão.sn.qtd	quantidade de legendas padrão entre as n ocorrências da métrica [BF_L]:ícone...legenda_padrão.sn.qtd
[nBF_L]:ícone...legenda_repetido.sn.qtd	quantidade de legendas repetidas entre as n ocorrências da métrica [BF_L]:ícone...legenda_repetido.sn.qtd
[nBF_L]:ícone...legenda_repetição.sn.qtd	quantidade de repetições, por cada legenda entre as n ocorrências da métrica [BF_L]:ícone...legenda_repetição.sn.qtd

***CALLBACK* L3: *CALLBACK* L1 e *CALLBACK* L2** **(ação direta e caixa de diálogo) e (função inexistente, nula e repetida)**	
[nBF_L]:*callback*...ação_funcinex.sn.qtd	quantidade de *callbacks* que chamam ação direta com função inexistente entre as n ocorrências da métrica [BF_L]:*callback*...ação_funcinex.sn.qtd
[nBF_L]:*callback*...ação_funcnula.sn.qtd	quantidade de *callbacks* que chamam ação direta com função nula entre as n ocorrências da métrica [BF_L]:*callback*...ação_funcnula.sn.qtd
[nBF_L]:*callback*...ação_funcrepetida.sn.qtd	quantidade de *callbacks* que chamam ação direta com função repetida entre as n ocorrências da métrica [BF_L]:*callback*...ação_funcrepetida.sn.qtd
[nBF_L]:*callback*...ação_funcrepetição.sn.qtd	quantidade de *callbacks* que chamam ação direta com função repetida entre as n ocorrências da métrica [BF_L]:*callback*...ação_funcrepetição.sn.qtd
[nBF_L]:*callback*...cd_funcinex.sn.qtd	quantidade de *callbacks* que chamam caixa de diálogo com função inexistente entre as n ocorrências da métrica [BF_L]:*callback*...cd_funcinex.sn.qtd
[nBF_L]:*callback*...cd_funcnula.sn.qtd	quantidade de *callbacks* que chamam caixa de diálogo com função nula entre as n ocorrências da métrica [BF_L]:*callback*...cd_funcnula.sn.qtd
[nBF_L]:*callback*...cd_funcrepetida.sn.qtd	quantidade de *callbacks* que chamam caixa de diálogo com função repetida entre as n ocorrências da métrica [BF_L]:*callback*...cd_funcrepetida.sn.qtd
[nBF_L]:*callback*...cd_funcrepetição.sn.qtd	quantidade de *callbacks* que chamam caixa de diálogo com função repetida entre as n ocorrências da métrica [BF_L]:*callback*...cd_funcrepetição.sn.qtd
ÍCONE L5: ÍCONE L1 e ÍCONE L2 **(padrão e repetido) e (grupo)**	
[nBF_L]:ícone...ícone_grupo.qtd.min	valor mínimo da quantidade de ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...ícone_grupo.qtd.min
[nBF_L]:ícone...ícone_grupo.qtd.max	valor máximo da quantidade de ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...ícone_grupo.qtd.max
[nBF_L]:ícone...ícone_grupo.qtd.med	valor médio da quantidade de ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...ícone_grupo.qtd.med
[nBF_L]:ícone...padrão_grupo.qtd.min	valor mínimo da quantidade de ícones padrão, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...padrão_grupo.qtd.min
[nBF_L]:ícone...padrão_grupo.qtd.max	valor máximo da quantidade de ícones padrão, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...padrão_grupo.qtd.max
[nBF_L]:ícone...padrão_grupo.qtd.med	valor médio da quantidade de ícones padrão, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...padrão_grupo.qtd.med
[nBF_L]:ícone...repetido_grupo.qtd.min	valor mínimo da quantidade de ícones repetidos, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...ícone_repetido_grupo.qtd.min
[nBF_L]:ícone...repetido_grupo.qtd.max	valor máximo da quantidade de ícones repetidos, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...ícone_repetido_grupo.qtd.max
[nBF_L]:ícone...repetido_grupo.qtd.med	valor médio da quantidade de ícones repetidos, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...ícone_repetido_grupo.qtd.med
[nBF_L]:ícone...comb_padrão_grupo.qtd.min	valor mínimo da quantidade de combinações padrão entre o ícone e *tooltip* associada, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...comb_padrão_grupo.qtd.min
[nBF_L]:ícone...comb_padrão_grupo.qtd.max	valor máximo da quantidade de combinações padrão entre o ícone e *tooltip* associada, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...comb_padrão_grupo.qtd.max
[nBF_L]:ícone...comb_padrão_grupo.qtd.med	valor médio da quantidade de combinações padrão entre o ícone e *tooltip* associada, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...comb_padrão_grupo.qtd.med
[nBF_L]:ícone...seq_comb_padrão_grupo.qtd.min	valor mínimo da quantidade de sequências de combinação padrão, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...seq_comb_padrão_grupo.qtd.min
[nBF_L]:ícone...seq_comb_padrão_grupo.qtd.max	valor máximo da quantidade de sequências de combinação padrão, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...seq_comb_padrão_grupo.qtd.max
[nBF_L]:ícone...seq_comb_padrão_grupo.qtd.med	valor médio da quantidade de sequências de combinação padrão, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...seq_comb_padrão_grupo.qtd.med

ÍCONE L6: ÍCONE L3 e ÍCONE L2 (*tooltip*) e (grupo)	
[nBF_L]:ícone...*tooltip*_grupo.qtd.min	valor mínimo da quantidade de *tooltips* associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_grupo.qtd.min
[nBF_L]:ícone...*tooltip*_grupo.qtd.max	valor máximo da quantidade de *tooltips* associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_grupo.qtd.max
[nBF_L]:ícone...*tooltip*_grupo.qtd.med	valor médio da quantidade de *tooltips* associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_grupo.qtd.med
[nBF_L]:ícone...*tooltip*_padrão_grupo.qtd.min	valor mínimo da quantidade de *tooltips* padrão associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_padrão_grupo.qtd.min
[nBF_L]:ícone...*tooltip*_padrão_grupo.qtd.max	valor máximo da quantidade de *tooltips* padrão associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_padrão_grupo.qtd.max
[nBF_L]:ícone...*tooltip*_padrão_grupo.qtd.med	valor médio da quantidade de *tooltips* padrão associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_padrão_grupo.qtd.med
[nBF_L]:ícone...*tooltip*_repetida_grupo.qtd.min	valor mínimo da quantidade de *tooltips* repetidas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_repetida_grupo.qtd.min
[nBF_L]:ícone...*tooltip*_repetida_grupo.qtd.max	valor máximo da quantidade de *tooltips* repetidas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_repetida_grupo.qtd.max
[nBF_L]:ícone...*tooltip*_repetida_grupo.qtd.med	valor médio da quantidade de *tooltips* repetidas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...*tooltip*_repetida_grupo.qtd.med
ÍCONE L7: ÍCONE L4 e ÍCONE L2 (legenda) e (grupo)	
[nBF_L]:ícone...legenda_grupo.qtd.min	valor mínimo da quantidade de legendas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_grupo.qtd.min
[nBF_L]:ícone...legenda_grupo.qtd.max	valor máximo da quantidade de legendas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_grupo.qtd.max
[nBF_L]:ícone...legenda_grupo.qtd.med	valor médio da quantidade de legendas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_grupo.qtd.med
[nBF_L]:ícone...legenda_padrão_grupo.qtd.min	valor mínimo da quantidade de legendas padrão associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_padrão_grupo.qtd.min
[nBF_L]:ícone...legenda_padrão_grupo.qtd.max	valor máximo da quantidade de legendas padrão associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_padrão_grupo.qtd.max
[nBF_L]:ícone...legenda_padrão_grupo.qtd.med	valor médio da quantidade de legendas padrão associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_padrão_grupo.qtd.med
[nBF_L]:ícone...legenda_repetida_grupo.qtd.min	valor mínimo da quantidade de legendas repetidas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_repetida_grupo.qtd.min
[nBF_L]:ícone...legenda_repetida_grupo.qtd.max	valor mínimo da quantidade de legendas repetidas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_repetida_grupo.qtd.max
[nBF_L]:ícone...legenda_repetida_grupo.qtd.med	valor mínimo da quantidade de legendas repetidas associadas aos ícones, por cada grupo de ícones entre as n ocorrências da métrica [BF_L]:ícone...legenda_repetida_grupo.qtd.med

ÍCONE L2 e *CALLBACK* L1 **(grupo) e (ação direta e caixa de diálogo)**	
[nBF_L]:ícone/*callback*...grupo_ação.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama ação direta entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_ação.qtd
[nBF_L]:ícone/*callback*...grupo_cd.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama caixa de diálogo entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_cd.qtd
ÍCONE L2 e *CALLBACK* L2 **(grupo) e (função inexistente, nula e repetida)**	
[nBF_L]:ícone/*callback*...grupo_funcinex.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas com função inexistente entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_funcinex.qtd
[nBF_L]:ícone/*callback*...grupo_funcnula.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas com função nula entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_funcnula.qtd
[nBF_L]:ícone/*callback*...grupo_funcrepetida.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas com função repetida entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_funcrepetida.qtd
ÍCONE L2 e *CALLBACK* L3 **(grupo), (ação direta e caixa de diálogo) e (função inexistente, nula e repetida)**	
[nBF_L]:ícone/*callback*...grupo_ação_funcinex.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama ação direta com função inexistente entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_ação_funcinex.qtd
[nBF_L]:ícone/*callback*...grupo_ação_funcnula.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama ação direta com função nula entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_ação_funcnula.qtd
[nBF_L]:ícone/*callback*...grupo_ação_funcrepetida.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama ação direta com função repetida entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_ação_funcrepetida.qtd
[nBF_L]:ícone/*callback*...grupo_cd_funcinex.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama caixa de diálogo com função inexistente entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_cd_funcinex.qtd
[nBF_L]:ícone/*callback*...grupo_cd_funcnula.qtd	quantidade de ícones, por cada grupo de ícones do objeto barra de ferramentas que chama caixa de diálogo com função nula entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_cd_funcnula.qtd
[nBF_L]:ícone/*callback*...grupo_cd_funcrepetida.qtd	quantidade de ícones por cada grupo de ícones do objeto barra de ferramentas que chama caixa de diálogo com função repetida entre as n ocorrências da métrica [BF_L]:ícone/*callback*...grupo_cd_funcrepetida.qtd

4.10.3. Contexto iOBJETO/CD

O contexto iOBJETO/CD é uma generalização dos contextos referentes a relação entre diferentes objetos presentes numa caixa de diálogo. Entre os objetos trabalhados, a única relação tratada refere-se ao objeto menu (Me) e as n ocorrências do objeto barra de ferramentas (nBF/CD). A seguir encontra-se a descrição desse contexto e suas respectivas métricas.

4.10.3.1 Contexto Me x nBF/CD

Para as tabelas seguintes, referentes ao contexto Me x nBF/CD (Menu versus duas ou mais Barras de Ferramentas), prevê-se a relação entre o menu e todas as barras de ferramentas que estejam dentro da caixa de diálogo principal. Como já foi explicado anteriormente, esse contexto fará uso de métricas já estabelecidas nos contextos Me e nBF/CD. Também, não existe a categoria gráfica nesse contexto, visto que esses objetos não compartilham semelhança gráfica alguma. Essa relação pode ser vista através da Figura 32.

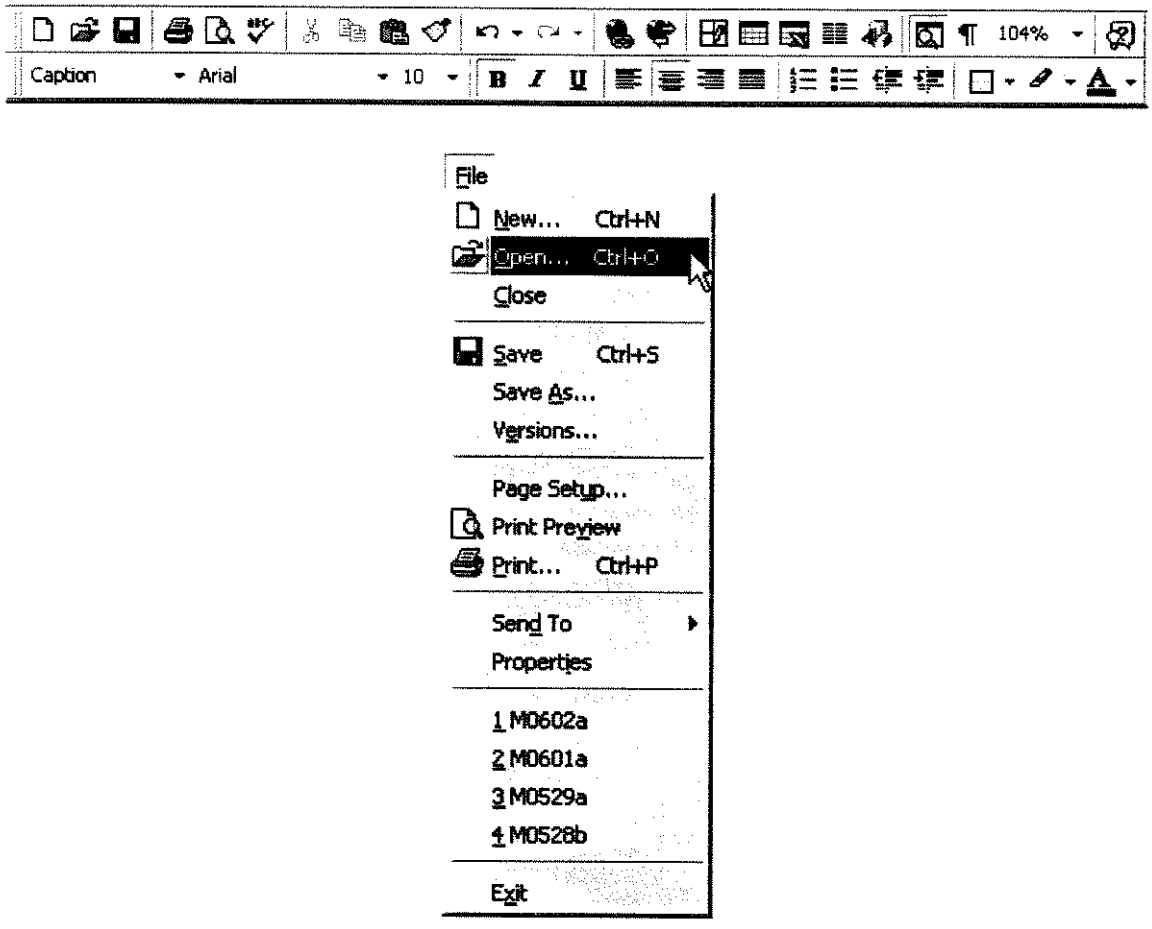

Figura 32: Exemplo de menu com 2 barras de ferramentas

A categoria Textual (Tabela 20) aborda 1 foco: legenda (e.g., gramática e sinônimos). A categoria Lógica (Tabela 21) aborda 3 focos: legenda/ícone (e.g., igualdade da sequência padrão), grupos (e.g., sequência padrão) e *callback* (e.g., caixas de diálogo). Para exemplificar o foco *callback*, pode-se verificar, por exemplo, a quantidade legendas item do menu com *callback* que chama caixa de diálogo (sem os três pontos) que está relacionada a ícones da barra de ferramentas com *callback* que chama caixa de diálogo, por cada legenda título (e.g., na legenda título "*File*", está representada na barra de ferramentas somente a legenda item "*Open*"; as demais legendas item do menu que chamam caixa de diálogo, como por exemplo a legenda "*Print*", na barra de ferramentas chamam ação direta).

Outra métrica que pode ser verificada é a quantidade de ícones da barra de ferramentas, que chamam *callback* com ação direta que não se relacionam a legendas item que chamam *callback* com ação direta (e.g., o ícone "*Format Painter*", o décimo ícone da barra de ferramentas "*Standard*" não se encontra representada por nenhuma legenda item no objeto menu).

Ainda outro exemplo de métrica que pode ser visto nesse contexto é a relação de legendas item com *callback* que chamam caixa de diálogo que está relacionada a ícones com *callback* que chama ação direta, numa relação de 1 para n (e.g., na legenda título "*Format*", a legenda item "*Paragraph*" está representada na barra de ferramentas "*Formatting*" através de quatro ícones, da esquerda para a direita, respectivamente, "*Align Left*", "*Center*", *Align Right*" e "*Justify*").

Uma outra métrica bastante interessante é a métrica de sequência padrão, de igualdade de sequência padrão entre as combinações padrão do objeto menu e as combinações padrão dos n objetos barra de ferramentas (e.g., no objeto menu, "*Print Preview*" e "*Print*", na barra de ferramentas, "*Print*" e "*Print Preview*").

Tabela 20: 7T - contexto Menu versus uma ou mais Barra de Ferramentas e categoria Textual

LEGENDA T1: gramática e sinônimos	
[MexnBF_T]:legenda...gramática3	valor sim ou não, existe erro de concordância gramatical entre as legendas do objeto menu e as legendas dos n objetos barra de ferramentas
[MexnBF_T]:legenda...gramática3.qtd	quantidade de * = s
[MexnBF_T]:legenda...sinônimos	valor sim ou não, existem sinônimos entre as legendas do objeto menu e as legendas dos n objetos barra de ferramentas
[MexnBF_T]:legenda...sinônimos.qtd	quantidade de * = s

Tabela 21: 7L - contexto Menu versus uma ou mais Barra de Ferramentas e categoria Lógica

LEGENDA DO OBJETO MENU e ÍCONE DOS n OBJETOS BARRA DE FERRAMENTAS (combinação padrão) e (grupo)	
[MexnBF_L]:legenda/ícone...grupo_comb_padrão_seqpadrão.sn	valor sim ou não, existe igualdade de sequência padrão entre as combinações padrão do objeto menu e as combinações padrão dos n objetos barra de ferramentas, por cada grupo de ícones dos n objetos barra de ferramentas (e.g., *Cut*, *Copy*, *Paste*)
[MexnBF_L]:legenda/ícone...grupo_combinação_seqpadrão.sn.qtd	quantidade de * = s
SEQPADRÃO dos GRUPOS para cada um dos n objetos barra de ferramentas em função do objeto menu	
[MexnBF_L]:grupos...seqpadrão.sn	valor sim ou não, checagem da sequência padrão dos grupos para cada um dos n objetos barra de ferramentas em função do objeto menu (e.g., (*Cut*, *Copy*, *Paste*), (*Undo*, *Redo*))
[MexnBF_L]:grupos...seqpadrão.sn.qtd	quantidade de * = s
***CALLBACK* L1: ações diretas**	
[MexnBF_L]:*callback*...(ação→ação).qtd	quantidade de legendas item com *callback* que chama ação direta do objeto menu que está relacionada a ícones com *callback* que chama ação direta dos n objetos barra de ferramentas; depende de [Me_L]:legenda/*callback*...leg(i)_ação.qtd e [nBF_L]:*callback*...ação.sn.qtd; (e.g., a legenda item "*Save*" do objeto menu e o ícone "*Save*" do objeto barra de ferramentas)
[MexnBF_L]:*callback*...(ação→ação)!.qtd	quantidade de legendas item com *callback* que chama ação direta do objeto menu que não está relacionada a ícones com *callback* que chama ação direta dos n objetos barra de ferramentas; depende de [Me_L]:legenda/*callback*...leg(i)_ação.qtd e [nBF_L]:*callback*...cd.sn.qtd; (e.g., a legenda item "Select All" do objeto menu)
[MexnBF_L]:*callback*...(ação←ação)!.qtd	quantidade de ícones com *callback* que chama ação direta dos n objetos barra de ferramentas que não está relacionada a legendas item com *callback* que chama ação direta do objeto menu; depende de [Me_L]:legenda/*callback*...leg(i)_cd.qtd e [nBF_L]:*callback*...ação.sn.qtd; (e.g., o ícone "*Format Painter*" do objeto barra de ferramentas)
***CALLBACK* L2: caixas de diálogo**	
[MexnBF_L]:*callback*...(cd→cd).qtd	quantidade de legendas item com *callback* que chama caixa de diálogo do objeto menu que está relacionada a ícones com *callback* que chama caixa de diálogo dos n objetos barra de ferramentas; depende de [Me_L]:legenda/*callback*...leg(i)_cd.qtd e [nBF_L]:*callback*...cd.sn.qtd; (e.g., a legenda item "*Open*" do objeto menu e o ícone "*Open*" do objeto barra de ferramentas)
[MexnBF_L]:*callback*...(cd→cd)!.qtd	quantidade de legendas item com *callback* que chama caixa de diálogo do objeto menu que não está

	relacionada a ícones com *callback* que chama caixa de diálogo dos n objetos barra de ferramentas; depende de [Me_L]:legenda/*callback*...leg(i)_cd.qtd e [nBF_L]:*callback*...ação.sn.qtd; (e.g., a legenda item "PageSetup" do objeto menu)
[MexnBF_L]:*callback*...(cd←cd)!.qtd	quantidade de ícones com *callback* que chama caixa de diálogo dos n objetos barra de ferramentas que não está relacionada a legendas item com *callback* que chama caixa de diálogo do objeto menu; depende de [Me_L]:legenda/*callback*...leg(i)_ação.qtd e [nBF_L]:*callback*...cd.sn.qtd; (e.g., o ícone "Manage Fields" do objeto barra de ferramentas)
***CALLBACK* L3: caixa de diálogo do objeto menu e ação direta dos n objetos barra de ferramentas**	
[MexnBF_L]:*callback*...(cd→ação).qtd	quantidade de legendas item com *callback* que chama caixa de diálogo do objeto menu que está relacionada a ícones com *callback* que chama ação direta dos n objetos barra de ferramentas, numa relação de 1 pai para n filhos; (e.g., a legenda item "Font" do objeto menu e os ícones "bold", "italic" e "underline" dos n objetos barra de ferramentas)
***CALLBACK* L1 e [Me_L]LEGENDA L8** **(ações diretas) e (item por título)**	
[MexnBF_L]:*callback*...leg(t)_(ação→ação).qtd	quantidade de legendas item com *callback* que chama ação direta do objeto menu que está relacionada a ícones com *callback* que chama ação direta dos n objetos barra de ferramentas, por cada legenda titulo do objeto menu
[MexnBF_L]:*callback*...leg(t)_(ação→ação)!.qtd	quantidade de legendas item com *callback* que chama ação direta do objeto menu que não está relacionada a ícones com *callback* que chama ação direta dos n objetos barra de ferramentas, por cada legenda titulo do objeto menu
[MexnBF_L]:*callback*...leg(t)_(ação←ação)!.qtd	quantidade de ícones com *callback* que chama ação direta dos n objetos barra de ferramentas que não está relacionada a legendas item com *callback* que chama ação direta do objeto menu, por cada legenda titulo do objeto menu
***CALLBACK* L2 e [Me_L]LEGENDA L8** **(caixas de diálogo) e (ítem por título)**	
[MexnBF_L]:*callback*...leg(t)_(cd→cd).qtd	quantidade de legendas item com *callback* que chama caixa de diálogo do objeto menu que está relacionada a ícones com *callback* que chama caixa de diálogo dos n objetos barra de ferramentas, por cada legenda titulo do objeto menu
[MexnBF_L]:*callback*...leg(t)_(cd→cd)!.qtd	quantidade de legendas item com *callback* que chama caixa de diálogo do objeto menu que não está relacionada a ícones com *callback* que chama caixa de diálogo dos n objetos barra de ferramentas, por cada legenda titulo do objeto menu
[MexnBF_L]:*callback*...leg(t)_(cd←cd)!.qtd	quantidade de ícones com *callback* que chama caixa de diálogo dos n objetos barra de ferramentas que não está relacionada a legendas item com *callback* que chama caixa de diálogo do objeto menu, por cada legenda titulo do objeto menu
***CALLBACK* L3 e [Me_L]LEGENDA L8** **(caixas de diálogo relacionadas a ações diretas) e (título)**	
[MexnBF_L]:*callback*...leg(t)_(cd→ação).qtd	quantidade de legendas item com *callback* que chama caixa de diálogo do objeto menu que está relacionada a ícones com *callback* que chama ação direta dos n objetos barra de ferramentas, numa relação de 1 pai para n filhos, por cada legenda título do objeto menu
LEGENDA DO OBJETO MENU e *TOOLTIP* DO OBJETO BARRA DE FERRAMENTAS **(que possuem a mesma *callback*)**	
[MexnBF_L]:*callback*...(legenda→*tooltip*).sn	valor sim ou não, legenda item do objeto menu é igual à *tooltip* associada ao ícone dos n objetos barra de ferramentas; depende de [nBF_L]:ícone...*tooltip*.sn.qtd
[MexnBF_L]:*callback*...(legenda→*tooltip*).sn.qtd	quantidade de * = s
LEGENDA DO OBJETO MENU e LEGENDA DOS n OBJETOS BARRA DE FERRAMENTAS **(que possuem a mesma *callback*)**	
[MexnBF_L]:*callback*...(legenda→legenda).sn	valor sim ou não, possuindo a mesma *callback*, legenda item do objeto menu é igual à legenda associada a um ícone dos n objetos barra de ferramentas; depende de [Me_L]:legenda...leg(i).qtd e [nBF_L]:ícone...legenda.sn.qtd
[MexnBF_L]:*callback*...(legenda→legenda).sn.qtd	quantidade de * = s
ÍCONE DO OBJETO MENU e ÍCONE DOS n OBJETOS BARRA DE FERRAMENTAS **(que possuem a mesma *callback*)**	
[MexnBF_L]:*callback*...(ícone→ícone).sn	valor sim ou não, possuindo a mesma *callback*, ícone associado à legenda item do objeto menu é igual ao ícone dos n objetos barra de ferramentas; depende de [Me_L]:legenda...leg(i)_ícone.sn.qtd
[MexnBF_L]:*callback*...(ícone→ícone).sn.qtd	quantidade de * = s

4.10.4. Contexto CD

Para as tabelas seguintes, referentes ao contexto CD (Caixa de Diálogo), prevê-se a relação entre o objeto janela e todos os demais objetos nele contidos. Como já foi explicado anteriormente, esse contexto fará uso de métricas já estabelecidas em todos os contextos anteriores. Também, não existe a categoria lógica nesse contexto, visto que é impossível prever a lógica entre todos os objetos presentes numa caixa de diálogo.

A categoria Gráfica (Tabela 22) aborda 2 focos: área (e.g., quantidade de cores e sobreposição) e legenda (e.g., tipo de fonte). Para exemplificar o foco área, pode-se levantar o número de cores utilizado na caixa de diálogo e ainda levantar o número de cores por cada quadrante. Pode-se também levantar se existe sobreposição de objetos na caixa de diálogo. A categoria Textual (Tabela 23) aborda 1 foco: legenda (e.g., sinônimos).

Tabela 22: 8G - contexto Caixa de Diálogo e categoria Gráfica

ÁREA G1: uso, cores, margens, sobreposição e desalinhamento	
[CD_G]:área...uso	razão da área usada pelos objetos presentes na caixa de diálogo e a área disponível
[CD_G]:área...uso_sup	*, na parte superior
[CD_G]:área...uso_inf	*, na parte inferior
[CD_G]:área...uso_esq	*, na parte esquerda
[CD_G]:área...uso_dir	*, na parte direita
[CD_G]:área...uso_q1	*, no quadrante 1 (superior direito)
[CD_G]:área...uso_q2	*, no quadrante 2 (superior esquerdo)
[CD_G]:área...uso_q3	*, no quadrante 3 (inferior esquerdo)
[CD_G]:área...uso_q4	*, no quadrante 4 (inferior direito)
[CD_G]:área...cores.qtd	quantidade de cores (de fundo) diferentes entre os objetos presentes na caixa de diálogo
[CD_G]:área...cores_sup.qtd	*, na parte superior
[CD_G]:área...cores_inf.qtd	*, na parte inferior
[CD_G]:área...cores_esq.qtd	*, na parte esquerda
[CD_G]:área...cores_dir.qtd	*, na parte direita
[CD_G]:área...cores_q1.qtd	*, no quadrante 1
[CD_G]:área...cores_q2.qtd	*, no quadrante 2
[CD_G]:área...cores_q3.qtd	*, no quadrante 3
[CD_G]:área...cores_q4.qtd	*, no quadrante 4
[CD_G]:área...margens.sn	valor sim ou não, objeto(s) ultrapassa(m) o valor das margens da caixa de diálogo
[CD_G]:área...margens.sn.qtd	quantidade de * = s
[CD_G]:área...margens_sup.sn.qtd	quantidade de * = s, na parte superior
[CD_G]:área...margens_inf.sn.qtd	quantidade de * = s, na parte inferior
[CD_G]:área...margens_esq.sn.qtd	quantidade de * = s, na parte esquerda
[CD_G]:área...margens_dir.sn.qtd	quantidade de * = s, na parte direita
[CD_G]:área...margens_q1.sn.qtd	quantidade de * = s, no quadrante 1
[CD_G]:área...margens_q2.sn.qtd	quantidade de * = s, no quadrante 2
[CD_G]:área...margens_q3.sn.qtd	quantidade de * = s, no quadrante 3
[CD_G]:área...margens_q4.sn.qtd	quantidade de * = s, no quadrante 4
[CD_G]:área...desalinh_h.sn	valor sim ou não, desalinhamento horizontal entre objetos presentes na caixa de diálogo existe
[CD_G]:área...desalinh_h.sn.qtd	quantidade de * = s
[CD_G]:área...desalinh_h_sup.sn.qtd	quantidade de * = s, na parte superior
[CD_G]:área...desalinh_h_inf.sn.qtd	quantidade de * = s, na parte inferior
[CD_G]:área...desalinh_v.sn	valor sim ou não, desalinhamento vertical entre objetos presentes na caixa de diálogo existe
[CD_G]:área...desalinh_v.sn.qtd	quantidade de * = s
[CD_G]:área...desalinh_v_esq.sn.qtd	quantidade de * = s, na parte esquerda
[CD_G]:área...desalinh_v_dir.sn.qtd	quantidade de * = s, na parte direita
[CD_G]:área...sobreposição.sn	valor sim ou não, sobreposição entre objetos presentes na caixa de diálogo existe
[CD_G]:área...sobreposição.qtd	quantidade de * = s
LEGENDA G1: fonte	
[CD_G]:legenda...fonte(tipo).qtd	quantidade de tipos de fonte aplicados às legendas dos objetos presentes na caixa de diálogo
[CD_G]:legenda...fonte(estilo).qtd	quantidade de estilos de fonte aplicados às legendas dos objetos presentes na caixa de diálogo
[CD_G]:legenda...fonte(tamanho).qtd	quantidade de tamanhos de fonte aplicados às legendas dos objetos presentes na caixa de diálogo

[CD_G]:legenda...fonte(cor).qtd	quantidade de cores de fonte aplicadas às legendas dos objetos presentes na caixa de diálogo
LEGENDA G2 (*TOOLTIP*): fonte	
[CD_G]:legenda...*tooltip*_fonte(tipo).qtd	quantidade de tipos de fonte aplicados às *tooltips* associadas às legendas e ícones dos objetos presentes na caixa de diálogo
[CD_G]:legenda...*tooltip*_fonte(estilo).qtd	quantidade de estilos de fonte aplicados às *tooltips* associadas às legendas e ícones dos objetos presentes na caixa de diálogo
[CD_G]:legenda...*tooltip*_fonte(tamanho).qtd	quantidade de tamanhos de fonte aplicados às *tooltips* associadas às legendas e ícones dos objetos presentes na caixa de diálogo
[CD_G]:legenda...*tooltip*_fonte(cor).qtd	quantidade de cores de fonte aplicados às *tooltips* associadas às legendas e ícones dos objetos presentes na caixa de diálogo

Tabela 23: 8T - contexto Caixa de Diálogo e categoria Textual

LEGENDA T1: gramática e sinônimos	
[CD_T]:legenda...gramática3.sn	valor sim ou não, existe erro de concordância gramatical entre as legendas dos objetos presentes na caixa de diálogo
[CD_T]:legenda...gramática3.sn.qtd	quantidade de * = s
[CD_T]:legenda...sinônimos.sn	valor sim ou não, existem sinônimos entre as legendas dos objetos presentes na caixa de diálogo
[CD_T]:legenda...sinônimos.sn.qtd	quantidade de * = s
LEGENDA T2 (*TOOLTIP*): gramática e sinônimos	
[CD_T]:legenda...*tooltip*_gramática3sn.	valor sim ou não, existe erro de concordância gramatical entre as *tooltips* associadas às legendas dos objetos presentes na caixa de diálogo
[CD_T]:legenda...*tooltip*_gramática3.sn.qtd	quantidade de * = s
[CD_T]:legenda...*tooltip*_sinônimos.sn	valor sim ou não, existem sinônimos entre as *tooltips* associadas às legendas dos objetos presentes na caixa de diálogo
[CD_T]:legenda...*tooltip*_sinônimos.sn.qtd	quantidade de * = s

4.10.5. Contexto mnOBJETO/APL

O contexto mnOBJETO/APL é uma generalização dos contextos referentes a mn ocorrências de diferentes tipos de objetos presentes numa aplicação. Tal contexto aborda mn ocorrências do objeto janela (mnJa/APL) e mn ocorrências do objeto botão de comando (mnBC/APL). A seguir encontra-se a descrição de cada um desses contextos específicos e suas respectivas métricas.

4.10.5.1 Contexto mnJa /APL

Para as tabelas seguintes, referentes ao contexto mnJA/APL (duas ou mais Janelas por Aplicação), prevê-se a relação entre duas ou mais janelas dentro de uma mesma Aplicação. Como já foi explicado anteriormente, esse contexto fará uso de métricas já estabelecidas no contexto Ja.

A categoria Gráfica (Tabela 24) aborda 3 focos: área de trabalho (e.g., razão de aspecto), barra de título (e.g., cor) e legenda (e.g., tipo de fonte). Para exemplificar o foco área de trabalho, pode-se fazer uma comparação de razão de aspecto entre as n janelas, como pode ser visto na Figura 33 e Figura 34.

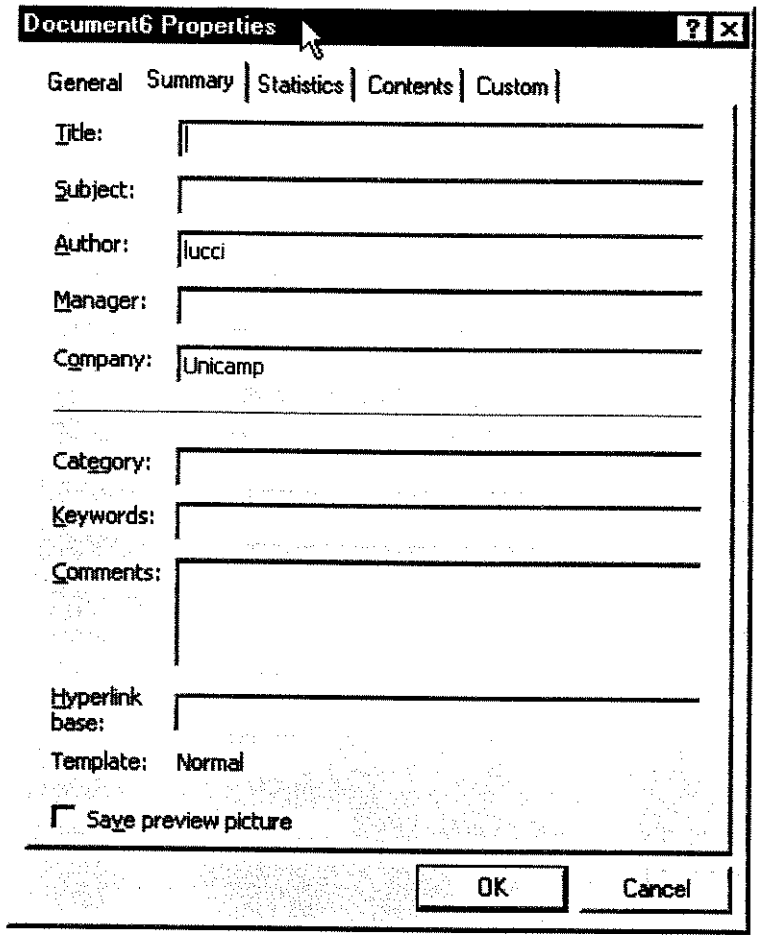

Figura 33: Exemplo da razão de aspecto da Janela "*Properties*"

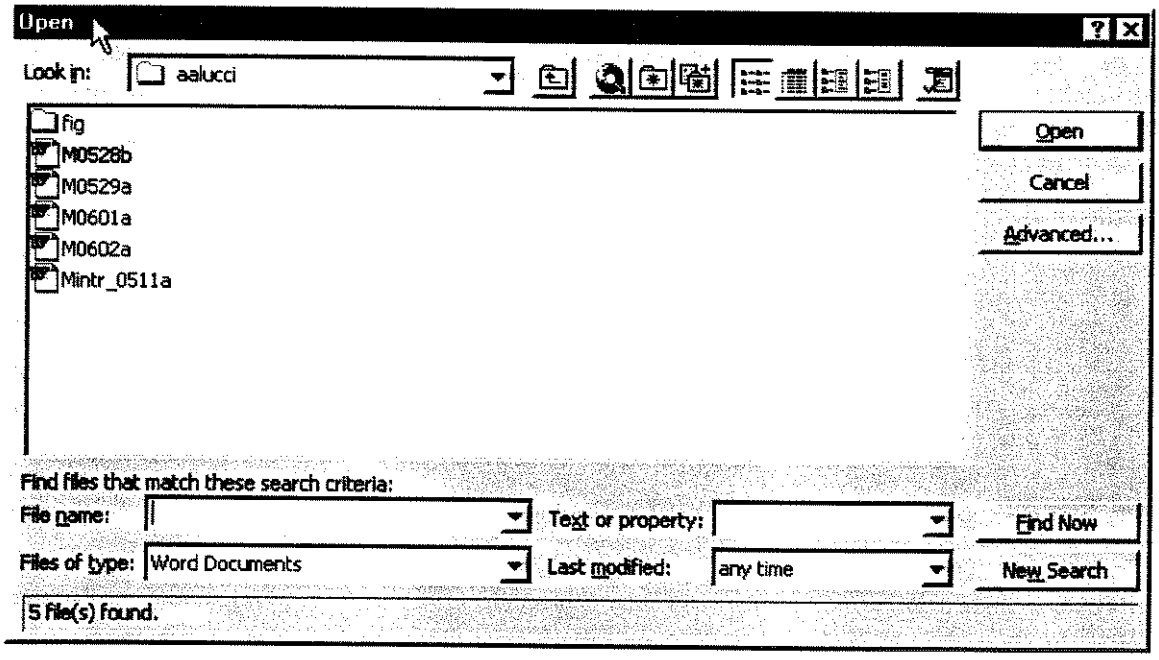

Figura 34: Exemplo da razão de aspecto da Janela "*Open*"

A categoria Textual (Tabela 25) aborda 1 foco: barra de título (e.g., ortografia). A categoria Lógica (Tabela 26) aborda 2 focos: geral (e.g., mobilidade e redimensão) e legenda (e.g., botão de controle). Para exemplificar o foco geral, pode-se levantar a informação de condição de mobilidade, ou seja, se ela pode ser posicionada em qualquer espaço da tela, e se ela pode ser redimensionada, como por exemplo, aumentada nas laterais. Para exemplificar o foco legenda, pode-se levantar a quantidade de janelas que possuem os botões de controle (minimizar, maximizar/restaurar e fechar).

Tabela 24: 9G - contexto duas ou mais Janelas e categoria Gráfica

ÁREA DE TRABALHO	
[nJa_G]:área...razão_aspecto.min	valor mínimo entre as n ocorrências da métrica [Ja_G]:área...razão_aspecto
[nJa_G]:área...razão_aspecto.max	valor máximo entre as n ocorrências da métrica [Ja_G]:área...razão_aspecto
[nJa_G]:área...razão_aspecto.med	valor médio entre as n ocorrências da métrica [Ja_G]:área...razão_aspecto
[nJa_G]:área...quadrado.min	valor mínimo entre as n ocorrências da métrica [Ja_G]:área...quadrado
[nJa_G]:área...quadrado.max	valor máximo entre as n ocorrências da métrica [Ja_G]:área...quadrado
[nJa_G]:área...quadrado.med	valor médio entre as n ocorrências da métrica [Ja_G]:área...quadrado
[nJa_G]:área...cor	valor sim ou não, existem diferentes cores entre as n ocorrências da métrica [Ja_G]:área...cor
[nJa_G]:área...cor.qtd	quantidade de * = s
[nJa_G]:área...localização.sn	valor sim ou não, existem diferentes valores entre as n ocorrências da métrica [Ja_G]:área...localização
[nJa_G]:área...localização.sn.qtd	quantidade de * = s
BARRA DE TÍTULO	
[nJa_G]:barra...cor.sn	valor sim ou não, existem diferentes cores entre as n métricas [Ja_G]:barra.cor
[nJa_G]:barra...cor.sn.qtd	quantidade de * = s
LEGENDA	
[nJa_G]:legenda...caracteres.qtd.min	valor mínimo entre as n ocorrências da métrica [Ja_G]:legenda...caracteres.qtd
[nJa_G]:legenda...caracteres.qtd.max	valor máximo entre as n ocorrências da métrica [Ja_G]:legenda...caracteres.qtd
[nJa_G]:legenda...caracteres.qtd.med	valor médio entre as n ocorrências da métrica [Ja_G]:legenda...caracteres.qtd
[nJa_G]:legenda...fonte(tipo).sn	valor sm ou não, existem diferentes tipos de fonte entre as n ocorrências da métrica [Ja_G]:legenda...fonte(tipo)
[nJa_G]:legenda...fonte(tipo).sn.qtd	quantidade de * = s
[nJa_G]:legenda...fonte(estilo).sn	valor sim ou não, existem diferentes estilos de fonte entre as n ocorrências da métrica [Ja_G]:legenda...fonte(estilo)
[nJa_G]:legenda...fonte(estilo).qtd	quantidade de * = s
[nJa_G]:legenda...fonte(tamanho).qtd	valor sim ou não, existem diferentes tamanhos de fonte entre as n ocorrências da métrica [Ja_G]:legenda...fonte(tamanho)
[nJa_G]:legenda...fonte(tamanho).qtd	quantidade de * = s
[nJa_G]:legenda...fonte(cor).qtd	valor sim ou não, existem diferentes cores de fonte entre as n ocorrências da métrica [Ja_G]:legenda...fonte(cor)
[nJa_G]:legenda...fonte(cor).qtd	quantidade de * = s

Tabela 25: 9T - contexto duas ou mais Janelas e categoria Textual

BARRA DE TÍTULO: LEGENDA	
[nJa_T]:legenda...ortografia.sn.qtd	quantidade de erros de ortografia entre as n métricas Ja.T.legenda...ortografia.sn
[nJa_T]:legenda...gramática1.sn.qtd	quantidade de erros de gramática entre as n métricas [Ja_T]:legenda...gramática1.sn
[nJa_T]:legenda...capitalização.sn.qtd	quantidade de erros de capitalização entre as n métricas [Ja_T]:legenda...capitalização.sn
[nJa_T]:legenda...abreviação.sn.qtd	quantidade de erros de abreviação entre as n métricas [Ja_T]:legenda...abreviação.sn
[nJa_T]:legenda...gramática2.sn	valor sim ou não, existem erros de concordância gramatical entre as legendas dos n objetos janela
[nJa_T]:legenda...gramática2.sn.qtd	quantidade de * = s
[nJa_T]:legenda...sinônimos.sn	valor sim ou não, existem sinônimos entre as legendas dos n objetos janela
[nJa_T]:legenda...sinônimos.sn.qtd	quantidade de * = s

Tabela 26: 9L - contexto duas ou mais Janelas e categoria Lógica

GERAL	
[nJa_L]:...mobilidade.sn.qtd	quantidade de condições de mobilidade = sim entre as n ocorrências da métrica [Ja_L]:...mobilidade.sn

[nJa_L]:...redimensão.sn.qtd	quantidade de condições de redimensionamento = sim entre as n ocorrências da métrica [Ja_L]:...redimensão.sn
LEGENDA (ÍCONE)	
[nJa_L]:legenda(ícone)... ícone.sn.qtd	quantidade de ícones associados às legendas entre as n métricas [Ja_L]:legenda(ícone)... ícone.sn
LEGENDA (BOTÃO DE CONTROLE)	
[nJa_L]:legenda(botcont)... botcont.sn.qtd	quantidade de objetos janela com botões de controle entre as n ocorrências da métrica [Ja_L]:legenda(botcont)...botcont.sn
[nJa_L]:legenda(botcont)... botcont.sn.opc.qtd	quantidade de cada combinação de botões de controle entre as n ocorrências da métrica [Ja_L]:legenda(botcont)...botcont.sn.opc

4.10.5.2 Contexto mnJa /APL

Para as tabelas seguintes, referentes ao contexto mnBC/APL (dois ou mais Botões de Comando por Aplicação), prevê-se a relação entre dois ou mais Botões de Comando dentro de uma mesma Aplicação. Como já foi explicado anteriormente, esse contexto fará uso de métricas já estabelecidas nos contextos BC e nBC.

A categoria Gráfica (Tabela 27) aborda 2 focos: geral (e.g., razão de aspecto) e legenda (e.g., tipo de fonte). Para exemplificar o foco geral, razão de aspecto, pode-se partir da Figura 35, que contém botões de comando em diferentes caixas de diálogo da aplicação. Pode-se perceber diferenças claras entre o botão "*Exceptions...*" e os dois botões "*OK*".

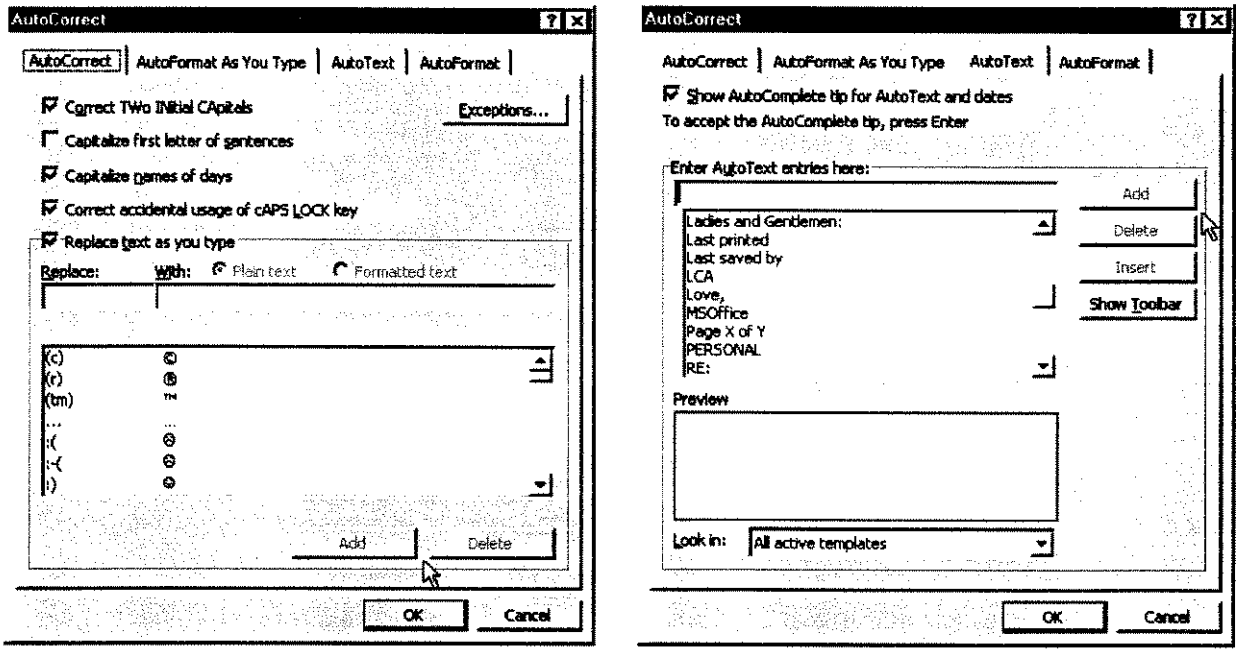

Figura 35: Exemplo de comparação entre botões de comando por aplicação

A categoria Textual (Tabela 28) aborda 1 foco: legenda (e.g., ortografia). A categoria Lógica (Tabela 29) apresenta 1 foco: geral (e.g., localização padrão). Para exemplificar o foco geral, localização padrão, pode-se perceber tal problema através da Figura 35, com os botões "*Add*" e "*Delete*".

Tabela 27: 10G - contexto dois ou mais Botões de Comando por aplicação e categoria Gráfica

GERAL	
[mnBC_G]:...razão_aspecto.min	valor mínimo entre as m ocorrências da métrica [nBC_G]:razão_aspecto.min
[mnBC_G]:...razão_aspecto.max	valor máximo entre as m ocorrências da métrica [nBC_G]:razão_aspecto.max
[mnBC_G]:...razão_aspecto.med	valor médio entre as m ocorrências da métrica [nBC_G]:razão_aspecto.med
[mnBC_G]:...quadrado.min	valor mínimo entre as m ocorrências da métrica [nBC_G]:quadrado.min
[mnBC_G]:...quadrado.max	valor máximo entre as m ocorrências da métrica [nBC_G]:quadrado.max
[mnBC_G]:...quadrado.med	valor médio entre as m ocorrências da métrica [nBC_G]:quadrado.med
[mnBC_G]:...cor_dif.qtd	quantidade de diferentes cores entre as m ocorrências da métrica [nBC_G]:cor_dif.qtd
LEGENDA G1: fonte	
[mnBC_G]:legenda...fonte(tipo_dif).qtd	quantidade de diferentes tipos de fonte entre as m métricas [nBC_G]:legenda...fonte(tipo_dif).sn.qtd
[mnBC_G]:legenda...fonte(estilo_dif).qtd	quantidade de diferentes estilos de fonte entre as m métricas [nBC_G]:legenda...fonte(estilo_dif).sn.qtd
[mnBC_G]:legenda...fonte(tamanho_dif).qtd	quantidade de diferentes tamanhos de fonte entre as m métricas [nBC_G]:legenda...fonte(tamanho_dif).sn.qtd
[mnBC_G]:legenda...fonte(cor_dif).qtd	quantidade de diferentes cores de fonte entre as m métricas [nBC_G]:legenda...fonte(cor_dif).sn.qtd
LEGENDA G2: caracteres	
[mnBC_G]:legenda...caracteres.qtd.min	valor mínimo entre as m ocorências da métrica [nBC_G]:legenda...caracteres.qtd.min
[mnBC_G]:legenda...caracteres.qtd.max	valor máximo entre as m ocorências da métrica [nBC_G]:legenda...caracteres.qtd.max
[mnBC_G]:legenda...caracteres.qtd.med	valor médio entre as m ocorências da métrica [nBC_G]:legenda...caracteres.qtd.med

Tabela 28: 10T - contexto dois ou mais Botões de Comando por aplicação e categoria Textual

LEGENDA	
[mnBC_T]:legenda...ortografia.sn.qtd	quantidade de erros de ortografia entre as m métricas [nBC_T]:legenda...ortografia.sn.qtd
[mnBC_T]:legenda...gramática1.sn.qtd	quantidade de erros de gramática entre as m métricas [nBC_T]:legenda...gramática1.sn.qtd
[mnBC_T]:legenda...capitalização.sn.qtd	quantidade de erros de capitalização entre as m métricas [nBC_T]:legenda...capitalização.sn.qtd
[mnBC_T]:legenda...abreviação.sn.qtd	quantidade de erros de abreviação entre as m métricas [nBC_T]:legenda...abreviação.sn.qtd
[mnBC_T]:legenda...gramática3.sn	valor sim ou não, existem erros de concordância gramatical entre as legendas dos m objetos botão de comando
[mnBC_T]:legenda...gramática3.sn.qtd	quantidade de * = s
[mnBC_T]:legenda...sinônimos.sn	valor sim ou não, existem sinônimos entre as legendas dos m objetos botão de comando
[mnBC_T]:legenda...sinônimos.sn.qtd	quantidade de * = s

Tabela 29: 10L - contexto dois ou mais Botões de Comando por aplicação e categoria Lógica

GERAL	
[mnBC_L]:...grupo_padrão.sn.qtd	quantidade de grupos padrão de botões de comando entre as m ocorrências da métrica [nBC_L]:...grupo_padrão.sn.qtd
[mnBC_L]:...grupo_padrão_ícones.qtd.min	valor mínimo entre as m ocorrências da métrica [nBC_L]:...grupo_padrão_ícones.qtd
[mnBC_L]:...grupo_padrão_ícones.qtd.max	valor máximo entre as m ocorrências da métrica [nBC_L]:...grupo_padrão_ícones.qtd
[mnBC_L]:...grupo_padrão_ícones.qtd.min	valor médio entre as m ocorrências da métrica [nBC_L]:...grupo_padrão_ícones.qtd
[mnBC_L]:...grupo_padrão_localização_padrão.sn	valor sim ou não, existe problema de localização padrão (para cada grupo padrão) entre as n ocorrências da métrica [nBC_L]:...grupo_padrão_localização
[mnBC_L]:...grupo_padrão_localização_padrão.sn.qtd	quantidade de * = s

4.10.6. Contexto iOBJETO/APL

O contexto iOBJETO/APL é uma generalização dos contextos referentes a relação entre diferentes objetos presentes numa aplicação. Entre os objetos trabalhados, a única relação tratada refere-se ao objeto menu (Me) e as mn ocorrências do objeto janela (mnJa/APL). A seguir encontra-se a descrição desse contexto e suas respectivas métricas.

4.10.6.1 Contexto Me x mnJa/APL

Para a tabela seguinte, referentes ao contexto Me x mnJa/APL (Menu versus duas ou mais Janelas por Aplicação), prevê-se a relação entre o objeto Menu e duas ou mais Janelas dentro de uma mesma aplicação. Como já foi explicado anteriormente, esse contexto fará uso de métricas já estabelecidas nos contextos Menu (Me) e mnJa/APL. Também, esse contexto não possui categoria gráfica nem textual, por não possuírem nenhuma relação desse tipo.

A categoria Lógica (Tabela 30) aborda 1 foco: legenda. As únicas métricas previstas nesse contexto, referem-se à relação de igualdade entre a legenda item do objeto menu com a legenda da barra de título do objeto janela, como pode ser visto na Figura 36.

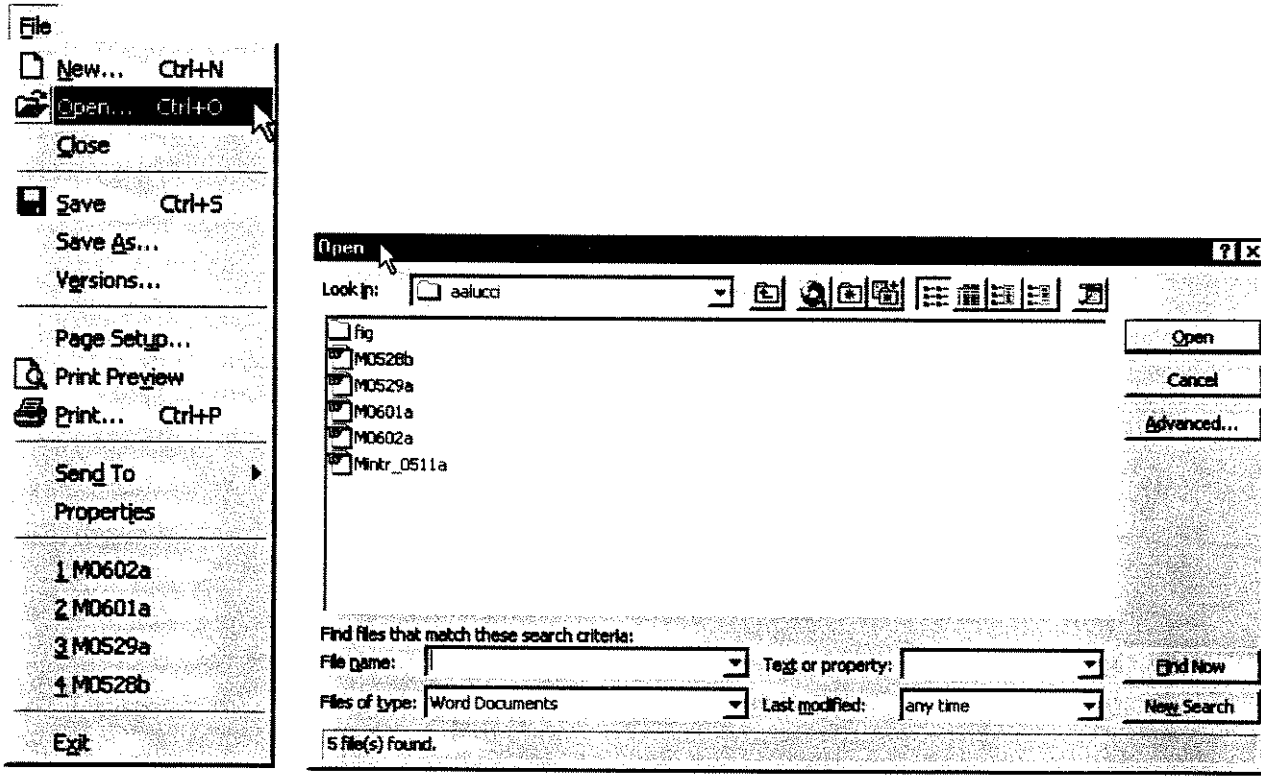

Figura 36: Exemplo de relação entre legendas do objeto menu e janela

Tabela 30: 11L - contexto Menu versus duas ou mais Janelas por aplicação e categoria Lógica

LEGENDA	
[MexnJa_L]:legenda...igual.sn	valor sim ou não, possuindo a mesma *callback* que chama caixa de diálogo, legenda item do objeto menu é igual à legenda da barra de título do objeto janela
[MexnJa_L]:legenda...igual.sn.qtd	quantidade de * = s

5. CONCLUSÃO

5.1. RESUMINDO...

O objetivo dessa dissertação de mestrado é explorar a relação entre as grandes áreas de Interação Humano-Computador e Engenharia de Software através da geração de uma base de métricas objetivas.

Pretende-se mostrar que a utilização de uma ferramenta construtora de interface não é condição suficiente para o sucesso de uma interface e consequentemente do produto de software do qual ela faça parte. É necessário que diversos conceitos relacionados à HCI, como as mais de 750 (setecentos e cinquenta) métricas levantadas nessa dissertação, estejam "empacotados" e disponíveis aos *designers* de interface através de, por exemplo, ferramentas *CASE* voltadas para avaliação de interface. A existência de uma ferramenta como essa minimiza a quantidade de recursos (custo, pessoal e tempo) alocados à tarefa de avaliação de interface, prevista em qualquer ciclo de desenvolvimento de produto de software, além de favorecer (não necessariamente garantir) a usabilidade do produto final que chega às mãos do usuário.

Por essa dissertação, a interface precisa ser encarada metaforicamente como um mapa, uma metáfora, por onde o usuário não pode ficar "perdido", mas achar o caminho através da "informação" (conceito bastante diferente de "dados"). Cabe ao *designer* aprender uma linguagem visual efetiva o suficiente para comunicar as informações necessárias, sempre com o enfoque no usuário final.

Foram explorados conceitos sobre as áreas de Engenharia de Software, Interação Humano-Computador e *Design* de Interação Visual. Sobre a relação entre ES e HCI, foi visto que ainda há muito a ser trabalhado. Com relação às Ferramentas CASE, foram apresentados dois tipos principais para essa dissertação: ferramentas *CASE* construtoras de interface e ferramentas *CASE* voltadas para avaliação de interface, capazes de avaliar a interface criada, fornecendo ao *designer*, sugestões para sua melhoria. Para "alimentar" esse tipo de ferramenta CASE, foi desenvolvido um conjunto de métricas objetivas para consistência de interface.

Todos esses conceitos foram passados para que fosse possível entender que a grande quantidade de métricas geradas tem a função de favorecer uma efetiva interação humano-computador, através da avaliação objetiva do *design* de interação visual. Somente dessa maneira a interface poderá comunicar, passando informação ao usuário, não somente dados. Um bom *design* de interação visual dependerá dos sólidos conhecimentos que o *designer* tenha de HCI e da ferramenta *CASE* construtora de interface que ele tiver à sua disposição. Ainda assim, essas não serão condições suficientes para garantir a "usabilidade" da interface gerada.

Para compreendê-las, foram propostos conceitos, tais como avaliação objetiva e subjetiva, métricas objetivas e subjetivas, contextos e categorias, além da estrutura das métricas. Após essa conceituação, foram apresentadas as métricas objetivas, resultado do cruzamento entre contextos e categorias.

Nessa dissertação, trabalhou-se com a plataforma *Windows95*, desenvolvida pela *Microsoft*, por ser uma plataforma de amplo alcance. Todas as métricas levantadas foram baseadas em "problemas" que o autor detectou tanto no processador de texto *Word97*, uma aplicação desenvolvida com objetos da *Microsoft* (desenvolvida por centenas de pessoas, entre elas analistas de sistemas, programadores e *designers*), e na ferramenta construtora de interface VB, uma ferramenta que usa objetos da *Microsoft* para gerar aplicações.

Descobrir que a ferramenta VB, ao oferecer comercialmente os objetos da plataforma *Windows95*, é incapaz de consistir aspectos básicos da interface, o que as mais de 750 (setecentos e cinquenta) métricas dessa dissertação propõem-se a fazer, foi uma descoberta muito importante e a prova mais concreta da importância desse trabalho! Essa descoberta também evidencia a necessidade de trabalhos futuros.

5.2. TRABALHOS FUTUROS

O trabalho de geração de métricas objetivas para consistência de interface visual, voltado para um conjunto específico e limitado de objetos, contextos e categorias, possibilitou a concepção de mais de 750 (setecentos e cinquenta) métricas. É sabido que existem pontos que não foram explorados, e que merecem igual atenção. Portanto, como trabalhos futuros, sugere-se trabalhar nas seguintes frentes (não necessariamente nessa sequência):

- explorar mais métricas objetivas para consistência de interface visual, baseadas nesses objetos, contextos e categorias levantados na dissertação
- implementar as métricas, disponibilizando-as para uma ferramenta *CASE* de avaliação de interface
- explorar características "dinâmicas" dos objetos (e.g., menu "acinzentado", resultado da função não disponível, devido ao estado do sistema)
- explorar mais objetos

- explorar mais contextos
- explorar mais categorias
- explorar novas métricas objetivas para consistência de interface visual, baseadas nesses novos objetos, contextos e categorias
- explorar métricas objetivas para consistência de interface multimídia
- explorar métricas subjetivas para consistência de interface visual
- explorar métricas subjetivas para consistência de interface multimídia

Esses trabalhos futuros poderão vir a ter um forte impacto na indústria de desenvolvimento de produtos de software, uma vez que o *design* e a posterior avaliação de interface têm recebido grande atenção nos últimos anos. Evidentemente, essas métricas aplicam-se ao estado atual da arte em que se encontram as áreas de pesquisa de Engenharia de Software e Interação Humano-Computador, e seu cruzamento. Uma vez que os paradigmas de interação evoluam, será necessário redirecionar a linha de pesquisa traçada nessa seção.

Para finalizar, espera-se que essa dissertação tenha contribuído com um passo inicial viável para uma solução mais ambiciosa no que diz respeito a problemas referentes a avaliação da interface, ampliando significativamente o estado da arte dessa área de pesquisa.

6. REFERÊNCIAS BIBLIOGRÁFICAS

[ACM92] ACM SIGCHI. **Curricula for Human-Computer Interaction**. 1992.

[ANDER88] ANDERSEN, N. B. Are "Human factors" Human? **The Computer Journal**. vol.31. n.5. 1988.

[BASS94] BASS, L., ABOWD, G., KAZMAN, R. Issues in the Evaluation of User Interface Tools. In: **Lecture Notes in Computer Science** (LNCS) 896. Workshop on SE-HCI. ICSE'94 (Itália). 1994. p.17-27.

[BERRY92] BERRY, D. M. **Academic Legitimacy of Software Engineering Discipline**. technical report. CMU/SEI-92-TR-34. Nov. 1992. 70p.

[BILLI94] BILLINGSLEY, P. The Pace Quickens... **SIGCHI Bulletin**. vol.26. n.3. p.08-10.

[CALLA95] CALAHAN, J. et al. An empirical comparison of pie vs. linear menus. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.79-88.

[CARD83] CARD, S., MORAN, T., NEWELL, A. **The Psychology of Human-Computer Interaction**. Lawrence Erlbaum Associates. 1983.

[CHIME94a] CHIMERA, R. Working Group on Evaluating User Interfaces and User Interface Tools. In: **Lecture Notes in Computer Science** (LNCS) 896. Workshop on SE-HCI. ICSE'94 (Itália). 1994. p.09-13.

[CHIME94b] CHIMERA, R. Platform Independent GUI Builders: Advance Software Enginueering to Handle HCI Issues. In: **Lecture Notes in Computer Science** (LNCS) 896. Workshop on SE-HCI. ICSE'94 (Itália). 1994. p.28-34.

[COUTA94] COUTAZ, J. e TAYLOR, N. Introduction to the Workshop on Software Engineering and Human-Computer Interaction: Joint Research Issues. In: **Lecture Notes in Computer Science** (LNCS) 896. Workshop on SE-HCI. ICSE'94 (Itália). 1994. p.1-3.

[CRAWF90] CRAWFORD, C. Lessons from Computer Game Design. In: LAUREL, B. (ed) **The Art of Human-Computer Interface Design**. Addison-Wesley. 1990. p. 103-111.

[CROSS92] CROSS II, J. H., CHIKOFSKY, E. J., MAY Jr, C. H. Reverse Engineering. **Advances in Computers**. vol.35. p.199-254. 1992.

[DAY93] DAY, M.C. e BOYCE, J. Human Factors in Human-Computer System Design. **Advances in Computer**. vol.36. p.333-431. 1993.

[GASEN93] GASEN, J. B., PERLMAN, G. HCI Education Survey now available. **SIGCHI Bulletin**. vol.25. n.3. p.07-08. Jul. 1993.

[GASEN94a] GASEN, J. B., PERLMAN, G., KELO, M. A. Update on the HCI Education Survey. **SIGCHI Bulletin**. vol.26. n.2. p.08-11. Apr. 1994.

[GASEN94b] GASEN, J. B. Getting to the "Core" of the Matter. **SIGCHI Bulletin**. vol.26. n.4. p.10-11. Oct. 1994.

[GASEN95] GASEN, J. B. Who's to say? Essential Elements of HCI Education. **SIGCHI Bulletin**. vol.27. n.3. p.16. Jul. 1995.

[GONZA95] GONZALEZ, C. Visual Design of Interaction, Dialog, or Interface? **SIGCHI Bulletin**, vol.27. n.1. p. 12-13. Jan. 1995.

[HECKE94] HECKEL, P. **Software Amigável: Técnicas de Projeto de Software para uma melhor interface com o usuário**. Campus. 1993.

[HEFLE94] HEFLEY, W. E. et al. **Integrating Human Factors with Software Engineering Practices**. technical report. CMU-CS-94-175. Jul. 1994. 9p.

[HIX93a] HIX, D., HARTSON, H.R. **Developing User Interfaces: Ensuring Usability through Product and Process**. Wiley. 1993. Cap. 4: Iterative, Evaluation-Centered User Interaction Development. p.95-116.

[HIX93b] HIX, D., HARTSON, H.R. **Developing User Interfaces: Ensuring Usability through Product and Process**. Wiley. 1993. Cap. 5: An Overview of System Analysis and Design. p.117-145.

[HIX93c] HIX, D., HARTSON, H.R. **Developing User Interfaces: Ensuring Usability through Product and Process**. Wiley. 1993. Cap. 1: Ensuring Usability in Human-Computer Interaction. p.01-12.

[HIX93d] HIX, D., HARTSON, H.R. **Developing User Interfaces: Ensuring Usability through Product and Process**. Wiley. 1993. Cap. 2: User Interaction Design Guidance: Standards, Guidelines, and Style Guides. p.15-55.

[HIX93e] HIX, D., HARTSON, H.R. **Developing User Interfaces: Ensuring Usability through Product and Process**. Wiley. 1993. Cap. 11: User Interface Development Tools. p.341-360.

[HOWAR95] HOWARD, S. User Interface Design and HCI: Identifying the Training Needs of Practitioners. **SIGCHI Buletin**. vol.27. n.3. p.17-22. Jul. 1995.

[HUTCH86] HUTCHINS, E. L., HOLLAN, J. D., NORMAN, D. A. Direct Manipulation Interfaces. In: NORMAN, D., DRAPER, S. (eds) **User Centered System Design: New Perspectives in Human-Computer Interaction**. Lawrence Erlbaum Associates. 1986. p.87-124.

[ISO9126] International Organization for Standardization. **ISO/IEC 9126**. 1992.

[JOHNS92] JOHNSON, P. **Human-Computer Interaction: Psychology, Task Analysis and Software Engineering.** McGraw-Hill. 1992.

[KELLER94] KELLER, R. K.User Interface Tools: A Survey and Perspective. In: **Lecture Notes in Computer Science** (LNCS) 896. Workshop on SE-HCI. ICSE'94 (Itália). 1994. p.225-231.

[KOVED95] KOVED, L. SHNEIDERMAN, B. Embedded menus: selecting items in context. pTime stress effects on the menu selection systems. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.67-77.

[LUCCA92] LUCCA, Waldo L. de. **Uma avaliação do processo de seleção de ferramentas CASE no mercado brasileiro**. São Carlos: CCET, UFSCAR, 1992. Dissertação (Mestrado) – Centro de Ciências Exatas e Tecnologia, Universidade Federal de São Carlos, 1992. 177p.

[LUCEN94] LUCENA, F.N. e LIESENBERG, H.K.E. **Interfaces Homem-Computador: uma primeira introdução**. relatório técnico (Unicamp). DCC-94-07. Set. 1994. 29p.

[MAHAJ97] MAHAJAN, R, e SHNEIDERMAN, B. Visual and Textual Consistency Checking Tools for Graphical User Interfaces. **IEEE Transactions on Software Engineering**, vol.23. n.11. Nov. 1997. p.722-735.

[MANTE89] MANTEI, M. M. An HCI Continuing Education Curriculum for Industry. **SIGCHI Bulletin**. vol.20. n.3. p.16-18. Jan. 1989.

[MARSH90] MARSH, S. Human-Computer Interaction: An Operational Definition. **SIGCHI Bulletin**. vol.22. n.1. p.16-22. Jul. 1990.

[MYERS95] MYERS, B. User Interface Software Tools. **ACM Transactions on Computer-Human Interaction**. vol.2, n.1. Mar. 1995. p.64-103.

[MYERS97] MYERS, B. **UIMS, Toolkits, Interface Builders**. Versão revisada de [MYERS95]. (NIELSEN, J. (ed) Handbook of User Interface Design. 1997) Disponível em http://www.cs.cmu.edu/afs/cs/user/bam/www/toolnames.html

[NEGRO95] NEGROPONTE, N. **A Vida Digital**. Companhia das Letras. 1995.

[NEWMA95] NEWMAN, W. M., LAMMING, M. G. **Interactive System Design**. Addison-Wesley. 1995.

[NORMA72] NORMAN, D. A., LINDSAY, P. H. **Human Information Processing: An Introduction to Psychology**. Academic Press. 1972.

[NORMA86] NORMAN, D. A. Cognitive Engineering. In: NORMAN, D. A., DRAPER, S. (eds) **User Centered System Design: New Perspectives in Human-Computer Interaction**. Lawrence Erlbaum Associates. 1986. p.31-62.

[OAKES92] OAKES, K. S., SMITH, D., MORRIS, E. **Guide to CASE Adoption**. technical report. CMU/SEI-92-TR-15. Nov. 1992. 60p.

[PLAIS95] PLAISANT, C., SEARS, A. Touchscreens interface for alphanumeric data entry. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.195-204.

[POTTE95] POTTER, R. L., WELDON, L. J., SHNEIDERMAN, B. Improving the accuracy of touchscreens: an experimental evaluation of three strategies. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.161-169.

[POWEL90] POWELL, J. E. **Designing User Interfaces**. Microtrend. 1990. Cap. 3: Visual Design. p.35-73.

[PREEC90] PREECE, J., KELLER, L. Key Issues in HCI Curriculum Design. **SIGCHI Bulletin**. vol.22. n.1. p.67-69. Jul. 1990.

[PRESS94a] PRESSMAN, R. S. **Software Engineering: a practitioner's approach**. 3a. edição. Adaptado por Darrel Ince. editora McGraw-Hill. 1994. cap. 1: Software and Software Engineering. p.3-39.

[PRESS94b] PRESSMAN, R. S. **Software Engineering: a practitioner's approach**. 3a. edição. Adaptado por Darrel Ince. editora McGraw-Hill. 1994. cap. 24: The Road ahead. p.771-783.

[PRESS94c] PRESSMAN, R. S. **Software Engineering: a practitioner's approach**. 3a. edição. Adaptado por Darrel Ince. editora McGraw-Hill. 1994. cap. 22: Computer-Aided Software Engineering. p.723-747.

[PRESS94d] PRESSMAN, R. S. **Software Engineering: a practitioner's approach**. 3a. edição. Adaptado por Darrel Ince. editora McGraw-Hill. 1994. cap. 23: Integrated CASE Environments. p.749-770.

[REED94] REED, P. ANSI/HFES Software User Interface Standardization: Critical Issues. **SIGCHI Bulletin**. vol.26. n.2. p.12-15. Apr. 1994.

[SEARS95] SEARS, A., SHNEIDERMAN, B. High precision touchscreens: design strategies and comparisons with a mouse. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.171-185.

[SHNEI92] SHNEIDERMAN, B. **Designing the User Interface: Strategies for Effective HCI**. Addison-Wesley. 2nd ed. 1992.

[SHNEI95a] SHNEIDERMAN, B. Direct Manipulation. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.13-15.

[SHNEI95b] SHNEIDERMAN, B. Direct Manipulation: a step beyond programming lnguages. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.17-37.

[SHNEI95c] SHNEIDERMAN, B. Touchingscreens now offer compelling issues. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.187-193.

[SILVA95] SILVA, E. J. **Representação em Projeto de Interfaces Homem-Computador: Estudo, Aplicação e Propostas de Extensão do Formalismo UAN**. Campinas: FEE, UNICAMP, 1995. Dissertação (Mestrado) – Faculdade de Engenharia Elétrica, Universidade Estadual de Campinas, 1995. 139p.

[TUFTE83] TUFTE, E. **The Visual Display of Quantitative Information**. Graphics Press. 1983.

[VBonline] Manual On-Line do Visual Basic 5.0.

[VEJA95] A máquina e o homem. **Revista Veja** (Especial). Computador: o micro chega às casas. ano 28. n.48. p.06-13. Dez. 1995.

[VERTE90] VERTELNEY, L., ARENT, M., LIEBERMAN, H. Two Disciplines in Search of an Interface: Reflections on a Design Problem. In: LAUREL, B. (ed) **The Art of Human-Computer Interface Design**. Addison-Wesley. 1990. p.45-55.

[WADLO93] WADLOW, M. G. Visual Interaction Design Special Interest Area: An Introduction. **SIGCHI Bulletin**. v.25, n. 1, p. 52-53. Jan. 1993.

[WADLO94a] WADLOW, M. G. Design as a Way of Life. **SIGCHI Bulletin**. v.26, n. 1, p. 07-08. Jan. 1994.

[WADLO94b] WADLOW, M. G. The Zen of Interface Design. **SIGCHI Bulletin**. v.26, n. 2, p. 16-17. Apr. 1994.

[WALKE90] WALKER, J. Through the Looking Glass. In: LAUREL, B. (ed) **The Art of Human-Computer Interface Design**. Addison-Wesley. 1990. p. 439-447.

[WALLA95] WALLACE, D., ANDERSON, N. S., SHNEIDERMAN, B. Time stress effects on the menu selection systems. In: SHNEIDERMAN, B. (ed). **Sparks of Innovation**. Ablex. 1995. p.89-97.

[WINOG96] WINOGRAD, T. **Bringing Design to Software**. Addison-Wesley. 1996.

[WURMA91] WURMAN, R. S. **Ansiedade de Informação**. 1991.

ANEXO A
REVISÃO BIBLIOGRÁFICA DE HCI

A.1. INTRODUÇÃO

O objetivo desse anexo é apresentar ao leitor a revisão bibliográfica realizada na área de HCI,. Em função da área ser bem recente, estão sendo levantadas informações a sobre os últimos anos da história de HCI, a partir de 1982, tomando por base a primeira edição do CHI. Essa revisão encontra-se na *home-page* do autor (http://www.dca.fee.unicamp.br/~lucci). Constam da revisão, um levantamento de eventos, periódicos e listas de assinatura da internet, além de uma pesquisa de diversos padrões ISO relacionados a HCI. Para o levantamento de eventos, a tabela apresentada na home-page oferece *links* para páginas próprias dos eventos. Essa revisão de eventos e periódicos foi baseada em [SHNEI87] [ACM92], e contou com pesquisa nas bibliotecas da Unicamp e USP (em São Paulo) e pesquisa na internet.

A.2. EVENTOS

Não houve a preocupação em conceituar diferenciar cada um dos diferentes "tipos" de eventos (e.g., encontros, conferências, congressos), visto que eles já estavam auto-denominados. Foram levantados diversos eventos internacionais referentes à HCI, com uma grande riqueza de temas. Embora existam diversos, não foram apresentados *workshops*, que geralmente são eventos menores que ocorrem em larga escala. Limitou-se a apresentar grandes eventos, com intervalos mínimos de um ano entre suas edições. Informações sobre eventos nacionais podem ser encontrados no Anexo C.

Pela tabela a seguir, pode-se ver os eventos, realizados entre 1982 e 1996, com o número da sua edição. Essa numeração tem o objetivo de ajudar o leitor a visualizar o quão recente essa área de pesquisa é. Por exemplo, o CHI, realizado em 1996, normalmente é conhecido como CHI'96. Mas essa nomenclatura não passa ao leitor a informação de que essa foi a 14a. (décima quarta) edição desse evento, uma informação importante para essa dissertação. Também pode-se ver por essa tabela que alguns eventos já estão bastante sedimentados, como é o caso das edições do CHI e do HFES.

A diferença do CHI para o HFES é que o segundo não é limitado à computação. Os demais encontros também são bastante conhecidos, mas são mais restritos, e não têm o alcance dos dois primeiros. Também existem diversos outros eventos que foram encontrados que oferecem espaço para discussão de assuntos relacionados à HCI, mas cujo foco não é esse, não estando, portanto, apresentados na tabela.

A Tabela 31 também indica uma grande concentração de eventos a partir de 1990. Isto prova que mudanças significativas realmente começaram a acontecer nessa área nos últimos anos. A maior prova disso é que 1997 foi um ano repleto de eventos.

A.3. PERIÓDICOS

Diversos periódicos (jornais e revistas) foram consultados entre o período de 1987 a 1996 (ano em que o levantamento foi feito), de maneira que se cobrisse os últimos dez anos de publicações. Na tabela a seguir, pode-se ter uma idéia da amplitude de títulos existentes (não necessariamente à disposição). São duas as tabelas apresentadas: a Tabela 32 apresenta os periódicos disponíveis que foram pesquisados pelo autor e a Tabela 33, os periódicos que não estavam disponíveis e que não foram pesquisados pelo autor. Dos periódicos selecionados, diversos (que não estão em negrito) apresentam uma amplitude de variados temas, dos quais às vezes, consegue-se extrair informações sobre HCI. Já os periódicos em negrito e alinhados à direita, representam a maior fonte de informações, cujo escopo está limitado a HCI (exceto *Human Factors*, que tem um escopo mais abrangente). Os periódicos *SIGCHI Bulletin* e *Human Factors* são os mais antigos, o que confirma o trabalho realizado pelas suas entidades responsáveis, através dos eventos *CHI* e *HFES*, também os mais antigos do gênero (vide Tabela 31).

Tabela 31: Relação de Eventos Internacionais

EVENTO	ANO (mil novecentos e...)															
Acrônimo do Evento (Nome completo do Evento) Entidade Responsável + parceiros	**82**	**83**	**84**	**85**	**86**	**87**	**88**	**89**	**90**	**91**	**92**	**93**	**94**	**95**	**96**	**97**
CHI (Conference on Human Factors in Computing Systems) SIGCHI.	1	2	-	3	4	5	6	7	8	9	10	11 *	12	13	14	15
HFES (HFES Annual Meeting) Human Factors and Ergonomics Society.	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
INTERACT IFIP TC13 + parceiros.	-	-	-	-	-	-	-	-	-	-	1?	2? *	3?	4?	5	6 #
APCHI (Asia Pacific Conference on HCI) ITI + parceiros.	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	2 #
OZCHI (Australian HCI Conference) ?	-	-	-	-	-	-	-	-	-	1?	2?	3?	4?	5?	6	7 #
EWHCI (East-West International Conference on HCI) ?	-	-	-	-	-	-	-	-	-	1	2	3	4?	5?	6?	7?
HCI International (International Conf. on Human-Computer Interaction) ?	-	-	1	-	-	2	-	3	-	4?	-	5?	-	6?	-	7
HCI (Human-Computer Interaction) British HCI Group	-	-	-	-	1?	2?	3?	4?	5?	6?	7?	8?	9?	10	11	12
UIST (User Interface Software Technology) SIGCHI, SIGGRAPH, SIGSOFT + parceiros.	-	-	-	-	-	-	1	2	3	4	5	6	7	8	9	10
IUI (Intelligent User Interface) SIGART, SIGCHI, AAAI e British HCI Group + parceiros.	-	-	-	-	-	-	1	-	-	-	-	2	-	-	-	3
UI (User Interface) User Interface Engineering.	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	2

LEGENDA:

*	INTERCHI'93 união do INTERACT'93 e CHI'93
#	INTERACT'97 em conjunto com APCHI'97 e OZCHI'97
?	dúvida, probabilidade
-	não existe
AAAI	American Association for Artificial Intelligence
ITI	Information Technology Institute
IFIP TC13	International Federation of Information Processing Technical Committee on HCI
SIGART	Special Interest Group on Artificial Intelligence (ACM)
SIGCHI	Special Interest Group on Human-Computer Interaction (ACM)
SIGSOFT	Special Interest Group on Software Engineering (ACM)
ACM	Association for Computer Machinery

Tabela 32: Relação de Periódicos pesquisados

PERIÓDICOS pesquisados	ANO (mil novecentos e...) volume.números									
	87	88	89	90	91	92	93	94	95	96
IEEE Communications	26.12	27.12	28.12	29.12	30.12	31.12	32.12	33.12	34.12	35.12
IEEE Computer	37.12	38.12	39.12	40.12	41.12	42.12	43.12	44.12	45.12	46.12
IEEE Software	4.6	5.6	6.6	7.6	8.6	9.6	10.6	11.6	12.6	13.6
IEEE Comput. Science and Eng. Magazine	-	-	-	-	-	-	-	1.4	2.4	3.4
IEEE Computer Graphics and Applications	7.6	8.6	9.6	10.6	11.6	12.6	13.6	14.6	15.6	16.6
ACM Communications	30.12	31.12	32.12	33.12	34.12	35.12	36.12	37.12	38.12	39.12
ACM Interactions	-	-	-	-	-	-	-	1.4	2.4	3.4
ACM SIGCHI Bulletin	18.2/ 19.2	19.2/ 20.2	20.2/ 21.2	21.2/ 22.2	23.4	24.4	25.4	26.4	27.4	28.4
ACM Transactions on CHI (TOCHI)	-	-	-	-	-	-	-	1.4	2.4	3.4
HFES Human Factors	29.6	30.6	31.6	32.6	33.6	34.6	35.4	36.4	37.4	38.4
Computer and Graphics	11.4	12.4	13.4	14.4	15.4	16.4	17.6	18.6	19.6	20.6
Advances in Computer	26	27	28/29	30/31	32/33	34/35	36/37	38/39	40/41	42/43
BCS The Computer Journal	30.6	31.6	32.6	33.6	34.6	35.6	36.8	37.?	38.10	39.?
Lectures Notes in Computer Science	+	+	+	+	+	+	+	+	+	+

LEGENDA:

? dúvida, probabilidade
- não existe
+ não possui volume, só número; não está indicado na tabela porque são publicações isoladas, referentes a eventos de diversas áreas, dentre elas ES e HCI
IEEE Institute of Electrical and Electronics Engineers
ACM Association for Computer Machinery
HFES Human Factors and Ergonomics Society
BCS British Computer Society

Tabela 33: Relação de Periódicos não pesquisados

PERIÓDICOS não pesquisados	ANO (mil novecentos e...) volume.números									
	87	88	89	90	91	92	93	94	95	96
BCS People and Computers	2	3	4	5	6	7/8?	9	10	11	12
BCS Interacting with Computers			1.4	2.4	3.4	4.4	5.4	6.4	7.4	8.4
BCS Interfaces	?	?	?	?	?	?	?	?	?	?
Human-Computer Interaction	?	?	?	?	?	?	?	?	?	?
Int. Journal of Man-Machine Studies	?	?	?	?	?	?	?	-	-	-
Int. Journal of Human-Computer Studies *	-	-	-	-	-	-	-	?.12	?.12	?.12
Int. Journal of HCI	?	?	?	?	?	?	?	?	?	?

LEGENDA:

? dúvida, probabilidade
- não existe
* antigo Int. Journal of Man-Machine Studies
BCS British Computer Society

A.4. LISTAS DE ASSINATURA

A seguir encontram-se citadas as listas de interesse disponíveis na internet das quais o autor participou nos últimos três anos. São listas voltadas para ES, HCI e *design* de interação visual.

NOME	OBSERVAÇÕES
sbc-l	Sociedade Brasileira de Computação – SBC (nacional) Anúncio e Discussão: eventos e questões de interesse do grupo mail para: majordomo@sbc.org.br conteúdo: subscribe sbc-l <endereço>
ihc-l	Grupo de Pesquisadores Brasileiros de HCI (nacional) Anúncios e Discussão: eventos e questões de interesse do grupo mail para: listproc@listserver.furb.rct-sc.br conteúdo: subscribe ihc-l <endereço>
seworld	Software Engineering World Anúncio: conferências, *workshops*, simpósios, edições especiais de jornais, etc mail para: majordomo@cs.colorado.edu conteúdo: subscribe seworld <endereço>
chi-announcements	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio: não comercial de conferências e cursos mail para: listserv@acm.org conteúdo: subscribe chi-announcements <endereço>
chi-intercultural	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio e Discussão: questões inter-culturais na comunidade de HCI mail para: listserv@acm.org conteúdo: subscribe chi-intercultural <endereço>
chi-social-action	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio e Discussão: questões sociais mail para: listserv@acm.org conteúdo: subscribe chi-social-action <endereço>
chi-students	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio e Discussão: questões de estudantes mail para: listserv@acm.org conteúdo: subscribe chi-students <endereço>
chi-tech-program	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio e Discussão: planejamento para o programa técnico da conferência CHI mail para: listserv@acm.org conteúdo: subscribe chi-tech-program <endereço>
chi-vision	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio e Discussão: direções futuras do SIGCHI mail para: listserv@acm.org conteúdo: subscribe chi-vision <endereço>
chi-educators	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio: novos livros, qualquer discussão relevante mail para: listserv@acm.org conteúdo: subscribe chi-educators <endereço>
chi-jobs	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio: serviços mail para: listserv@acm.org conteúdo: subscribe chi-jobs <endereço>
chi-publications	Special Interest Group in Computer Human Interaction (SIGCHI) da ACM Anúncio e Discussão: direções futuras sobre publicações do SIGCHI mail para: listserv@acm.org conteúdo: subscribe chi-publications <endereço>
visual-l	Visual Interaction Design community Anúncio e Discussão: mail para: listserv@vtvm1.cc.vt.edu conteúdo: subscribe visual-l <endereço>
bcs-hci	British HCI News Service Anúncio: mail para: mailbase@mailbase.ac.uk conteúdo: join bcs-hci <seunome>

A.5. PADRÕES ISO RELACIONADOS À HCI

Existem diversas entidades/associações/grupos voltadas à criação de Normas e padrões distribuídos ao redor do mundo (http://www.iso.ch/infoe/stbodies.html). Segundo o site da ISO, essas organizações estão divididas entre:

- **organizações de padrões internacionais**: entidades tendo atividades reconhecidas de padronização, que são reconhecidas a nível internacional, cuja função principal é, em função dos seus estatutos, a preparação, aprovação e adoção de padrões que são disponibilizados ao público; para ser membro, é necessário fazer parte de uma entidade nacional relevante de qualquer país;
- **entidades de padronização internacionais**: entidades que têm atividades reconhecidas de padronização a nível internacional;
- **entidades de padrões nacionais**: entidades tendo atividades reconhecidas de padronização, que são reconhecidas a nível nacional, cuja função principal é, em função dos seus estatutos, a preparação, aprovação e adoção de padrões que são disponibilizados ao público; e são elegíveis para serem os membros nacionais das organizações de padrões regionais e internacionais correspondentes.

O objetivo nesse anexo é relacionar algumas Normas que estejam voltadas às questões de HCI, oriundas de uma organização internacionalmente reconhecida, a ISO. Ao trabalhar com uma Norma ISO deve-se saber distinguir o seu estado, o que indicará a "maturidade" de discussão em cima da mesma, apresentados na Tabela 34 (http://www.iso.ch/infoe/proc.html). O estado das Normas será apresentado na tabela que apresenta algumas das Normas encontradas no site da ISO, relacionados à HCI (Tabela 35).

Um membro da ISO que estude suas Normas pertence a um subcomitê (SC, *subcommittee*), que pertence a um comitê técnico (TC, *technical committee*). Eventualmente o comitê pode ser um comitê técnico conjunto (JTC, *joint techical committee*), composto por membros de outra organização mundial de padronização, como por exemplo a IEC (*International Eletrotechnical Committee*). Essa informação é importante porque a listagem das Normas na Tabela 35 apresente tais termos. Para maiores informações sobre o funcionamento da ISO, basta ter acesso à sua *home-page* (http://www.iso.ch). Como informação adicional, alguns desses padrões são comentados na coluna *standards* do *SIGCHI Bulletin*.

Tabela 34: Tabela de Estágios de desenvolvimento de padrões ISO

STEPS	STEPS DESCRIPTION	STAGES
NP	New Work Item Proposal	Proposal
WD	Working Draft	Preparatory
CD	Committee Draft	Committee
DIS	Draft International Standard	Enquiry
FDIS	Final Draft International Standard	Approval
IS	International Standard	Publication

Tabela 35: Relação de Normas ISO relacionadas à HCI

TC 145 Graphical symbols (http://www.iso.ch/liste/TC145.html)		
ISO 7001:1990	IS	Public information symbols
ISO/TR 7239:1984	IS	Development and principles for application of public information
ISO 9186:1989	IS	Procedures for the development and testing of public information symbols
ISO 3461-1:1988	IS	General principles for the creation of graphical symbols – Part 1: Graphical symbols for use on equipment
ISO 3864:1984	IS	Safety colours and safety signs
JTC 1 information technology (http://www.iso.ch/liste/JTC1.html)		
ISO/IEC 10741-1:1995	IS	Information technology – User system interfaces – Dialogue interaction – Part 1: Cursor control for text editing
ISO/IEC 11581-1	DIS	Information technology – User system interfaces – Icon symbols and functions – Part 1: Icons – General
ISO/IEC 11581-2	DIS	Information technology – User system interfaces – Icon symbols and functions – Part 2: Object icons
ISO/IEC 11581-3	DIS	Information technology – User system interfaces – Icon symbols and functions – Part 3: Pointers
TC 159 Ergonomics (http://www.iso.ch/liste/TC159.html)		
ISO 10075:1991	IS	Ergonomic principles related to mental work-load – General terms and definitions
ISO 10075-2:1996	IS	Ergonomic principles related to mental work-load – Part 2: Design principles
ISO 7250:1997	IS	Basic human body measurements for technological design
ISO 9241-1:1997	IS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 1: General introduction
ISO 9241-2:1992	IS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 2: Guidance on task requirements
ISO 9241-3:1992	IS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 3: Visual display requirements
ISO 9241-4	DIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 4: Keyboard requirements
ISO 9241-5	DIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 5: Workstation layout and postural requirements
ISO 9241-6	DIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 6: Environmental requirements
ISO 9241-7	FDIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 7: Requirements for displays with reflections
ISO 9241-8	FDIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 8: Requirements for displayed colours
ISO 9241-9	CD	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 9: Requirements for non-keyboard c devices
ISO 9241-10:1996	IS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 10: Dialogue principles
ISO 9241-11	FDIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 11: Guidance on usablity
ISO 9241-12	DIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 12: Presentation of information
ISO 9241-13	IS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 13: User guidance
ISO 9241-14:1997	IS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 14: Menu dialogues
ISO 9241-15	IS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 15: Command dialogues
ISO 9241-16	DIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 16: Direct-manipulation dialogues
ISO 9241-17	FDIS	Ergonomic requirements for office work with visual display terminal (VDTs) – Part 17: Form-filling dialogues
ISO 9355-1	DIS	Ergonomic requirements for the design of displays and control actuators – Part 1: Human interactions with displays and control actuators
ISO 9355-2	DIS	Ergonomic requirements for the design of signals and control actuators – Part 2: Displays
ISO 13407	DIS	Human-centred design process for interactive systems
ISO 7731:1986	IS	Danger signals for work places – Auditory danger signals
ISO 8995:1989	IS	Principles of visual ergonomics – The lighting of indoor work systems
ISO 11428:1996	IS	Ergonomics – Visual danger signals – General requirements, design and testing
ISO 11429:1996	IS	Ergonomics – System of auditory and visual danger of information signals

9. cores de fundo diferentes (*distinct background colors*)	➡ o número de diferentes cores de fundo	• múltiplas cores podem indicar inconsistência • para cada caixa de diálogo
10. cores de frente diferentes (*distinct foreground colors*)	➡ o número de diferentes cores de frente	• múltiplas cores podem indicar inconsistência • para cada caixa de diálogo
11. analista de margens (*margin analyzer*)	➡ os valores mais frequentes das margens	• extensão da métrica margens (*margins*) • os valores mais frequentes são os valores de margem ótimos • lista as margens em todas as caixas de diálogo que estão inconsistentes com esses valores • calcula quais *widgets* da caixa de diálogo precisam ser movidos, por quantos *pixels*
12. concordância (*concordance*)	➡ todas as palavras que aparecem em cada caixa de diálogo	• ajuda os *designers* com o uso de palavras apropriadas, tais como *spelling*, *abbreviation*, *tense consistence*, consistência de caso, voz passiva/ativa • esta ferramenta foi quebrada
13. concordância de interface (*interface concordance*)	➡ as variantes de capitalização para todos os termos que aparecem em botões, labels, menus, etc para cada caixa de diálogo	• differences grafias podem ser aceitáveis
14. concordância de botões (*button concordance*)	➡ as variantes de capitalização, fontes, cores de frente e tamanhos diferentes dos botões	
15. tabela de layout de botões (*button layout table*)	➡ identifica inconsistências na colocação do botão, terminologia do botão e tamanhos diferentes do botão, local e globalmente	• de uma série de botões que ocorrem frequentemente juntos (e.g., *ok*, *cancel*, *close*, *help*), o primeiro botão da série é detectado na caixa de diálogo e o programa gera a altura, largura e posição relativa ao primeiro botão de cada botão detectado na lista (x + *offset*, y + *offset*)
16. corretor ortográfico (*interface speller*)	➡ todos os termos usados nos *widgets* da interface e apresentar (*output*) termos que não são achados no dicionário	• essa saída é filtrada por um arquivo que contém termos de computação e termos padrão de VB
17. cesto de terminologia (*terminology basket*)	➡ a coleção de termos de computação, incluindo suas diferentes formas de contração (*tense formats*)	• os formatos de contração e sinônimos podem ser inadvertidamente usados como sinônimos por *designers* (ex: *close*, *cancel*, *exit*, *end*, *quit* e *remove*, *removes*, *removed*, *removing*)

ANEXO B
MÉTRICAS PROPOSTAS POR MAHAJAN

Tabela 36: Métricas propostas por Mahajan

MÉTRICA	seu OBJETIVO é verificar...	OBSERVAÇÕES
1. **razão de aspecto (*aspect ratio*)**	➡ a razão entre altura e largura de uma caixa de diálogo	• desejável entre 0,5 a 0,8 • diálogos com funções similares devem ter mesma razão de aspecto
2. **total de widgets (*widget totals*)**	➡ o aninhamento entre todos os *widgets* e *widgets* de alto nível	• quanto maior a diferença maior o aninhamento
3. **área sem widgets (*non-widget area*)**	➡ a razão entre a área sem objetos com a área total do diálogo	• expresso em porcentagem, o número perto de 100 indica alta utilização e números baixos, menores que 30 indicam possibilidade de redesign
4. **densidade de widgets (*widget density*)**	➡ razão entre número de *widgets* de alto nível e área total da caixa de diálogo	• multiplicado por 100.000 para normalização • números maiores que 100 indicam grande número de *widgets* presentes em uma pequena área
5. **margens (*margins*)**	➡ o número de pixels entre a borda da caixa de diálogo e o *widget* mais próximo	• as margens esquerda, direita, superior e inferior deveriam ser aproximadamente iguais umas as outras, na caixa de diálogo e entre diferentes caixas de diálogo
6. **grades (*gridedness*)**	➡ o alinhamento dos *widgets*	• números de pilhas de *widgets* com as mesmas coordenadas x,y. altos valores (de x ou y) indicam possibilidade de *widgets* mal alinhados • sua extensão é grade de botões (*button gridedness*)
7. **balanço de áreas (*area balances*)**	➡ o balanço horizontal: razão entre área total de *widgets* na metade esquerda da caixa de diálogo em relação à direita ➡ o balanço vertical: razão entre a área total de *widgets* na metade superior da caixa de diálogo em relação à inferior	• altos valores variando entre 4 e 10 indicam telas mal balanceadas • o valor limite 10 representa uma caixa de diálogo branca, ou quase branca (ex: caixa de diálogo com só um *widget*, um botão)
8. **fontes distintas (*distinct typefaces*)**	➡ a variação de fontes (tamanho, negrito e itálico)	• quanto menor a variação melhor • medida em cada caixa de diálogo e entre as caixas de diálogo

As informações referentes à ISO 9241 também foram tiradas de um site de uma empresa (http://www.system-concepts.com/stds/status.html) que possui uma lista com o serviço de aviso de atualizações das partes da Norma. Note-se que o estado das partes 11, 13, 15 e 17 da Norma não são as mesmas entre os sites... (um problema grave na Internet, as informações fluem tão rapidamente que as informações podem estar desatualizadas).

ANEXO C
HCI NO BRASIL

Internacionalmente, principalmente em países que produzem software em massa, HCI é uma área bastante trabalhada e respeitada. O Brasil recém está desenvolvendo projetos que visam a exportar produtos de software nacionais. Ao desenvolver uma dissertação cujo foco é avaliação de interface de produto de software, espera-se estar contribuindo para a produção científica e tecnológica do país.

Não foram encontradas informações relevantes à atuação do Brasil na área de HCI. O que existe são pequenas contribuições, de pesquisadores que se encontram isolados. Os espaços para discussão ainda bastante "tímida" dessa área aconteceram eventualmente através de publicações de artigos, em edições do SBES (Simpósio Brasileiro de Engenharia de Software), e na terceira edição do WoMH, "III *Workshop* em Sistemas Multimídia e Hipermídia" (http://www.icmsc.sc.usp.br/womh97/), em 1997.

Para tentar mudar esse quadro, há pouco mais de um ano foi criada uma lista de discussão para pessoas da área, bem como um site na internet (http://www.inf.furb.rct-sc.br/ihc). Tais iniciativas produziram resultados bastante frutíferos. Da lista surgiu a idéia de formalizar um *workshop* voltado para HCI, vinculado ao SBES'98 (http://www.nce.ufrj.br/sbes97/home-p.htm), inicialmente. O IHC'98, "I *Workshop* sobre Fatores Humanos em Sistemas Computacionais: Compreendendo Usuários, Construindo Interfaces" (http://www.inf.puc-rio.br/~ihc98/index.html) será realizado em Maringá, durante dois dias, em outubro de 1998. Será a primeira vez que pesquisadores brasileiros da área de HCI terão a chance de se encontrar e dar início a um trabalho mais sério no reconhecimento e fortalecimento dessa área de pesquisa.

A partir desse evento, espera-se um maior reconhecimento da área de HCI frente à sociedade brasileira de computação, através do intercâmbio de informações entre os seus pesquisadores, bem como a formação de uma comunidade de HCI, forte e atuante.

ANEXO D
ESTUDO DE CASO

Para demonstrar alguns problemas encontrados com utilização de uma ferramenta construtora de interface, e a grande importância das métricas apresentadas nessa dissertação, foi usado o VB (*Visual Basic*) para desenvolver um executável que é uma "cópia", uma "simulação" muito básica de parte da interface do *Word97*, então chamado de "*simword97*". O objetivo é apresentar a interface de um sistema bastante conhecido simulando seu desenvolvimento por um *designer* pouco experiente. Não existe nenhuma pretensão de desenvolver o código da aplicação, tampouco oferecer uma interface completa ao usuário. Pretende-se tão somente apresentar os aspectos mais simples de uma interface, realçando três dos quatro objetos estudados nessa dissertação: janela, botão de comando e menu. Para fins de simplificação, não é apresentado o objeto barra de ferramentas. Os resultados conseguidos com o *simword97* podem ser vistos a seguir, a começar pela Figura 37, que mostra sua tela principal.

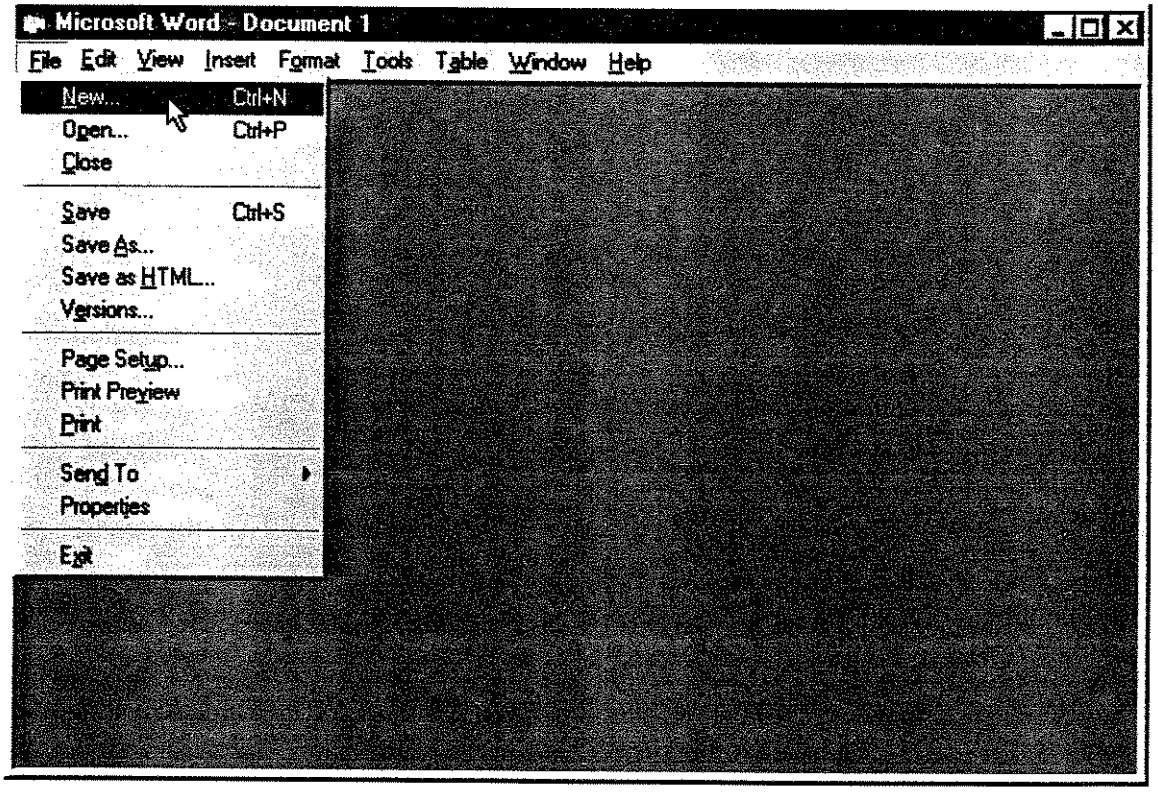

Figura 37: Tela principal do *simword97*

O *simword97* é um executável bastante simples. Embora o menu esteja completo e conforme o *Word97*, somente o menu "*File*" possui alguns ítens de menu que chamam caixas de diálogo (CD), sendo que nenhum deles chama uma ação direta. Outro detalhe importante: para simplificação, também optou-se por não utilizar ícones ao lado das legendas do objeto menu. Com tais simplificações, algumas métricas oferecidas na dissertação não podem ser exemplificadas nesse anexo.

Na Figura 37, ao observar o menu "*File*" pode-se notar diversos problemas. Os mais importantes serão mostrados a seguir. O primeiro problema é detectado pela métrica **[Me_L]:legenda...sublinhado_padrão.sn**. Por essa métrica, através de uma lista padrão, identifica-se que o sublinhado padrão de "*Open*" é a letra "o" e não "p".

Como consequência desse descuido, aparece o segundo problema: a métrica **[Me_L]:legenda...sublinhado_repetido.sn** vai levantar o fato de haver sublinhados repetidos nesse menu, assumindo valor lógico igual a "s". Ao procurar por todo o menu "*File*", deve-se procurar saber quantos sublinhados repetidos existem, fazendo uso da métrica **[Me_L]:legenda...sublinhado_repetido.sn.qtd**. Ao descobrir que só a letra "p" tem seu sublinhado repetido, essa métrica assume o valor igual a 1. Caso houvesse mais letras sublinhadas repetidas (e.g., a letra "N" do item de menu "*New*" se repetisse em algum outro item de menu do menu "*File*"), a métrica assumiria o valor igual a 2. Para cada letra com sublinhado repetido usa-se a métrica **[Me_L]:legenda...sublinhado_repetição.sn.qtd**, cujo objetivo é descobrir quantas repetições ocorrem para cada letra que tem sublinhado repetido. No exemplo anterior, tal métrica assume valor igual a 1, visto que existe uma única repetição da letra "p", ou seja, somente mais uma ocorrência da letra "p". Assumindo que o primeiro item de menu encontrado é o "*Open*", o item de menu "*Print*" seria considerado uma repetição.

Outro problema detectado somente na implementação, mas não coberto pelas métricas: embora todos os ítens de menu estejam sublinhados, e somente os ítens de menu "*Open*" e "*Print*" possuam a mesma letra "p" sublinhada, sem entrar em detalhes de codificação do VB, conseguiu-se fazer com que a letra "p" também ativasse o item de menu "*Print Preview*", ainda que esse último tenha a letra "v" sublinhada. Dessa forma, ao apertar intermitentemente a letra "p" , o menu ficou oscilando entre as três opções, como pode ser visto na Figura 38. Esse problema interno do VB, dos objetos desta ferramenta, acabou por gerar um problema a mais para o usuário.

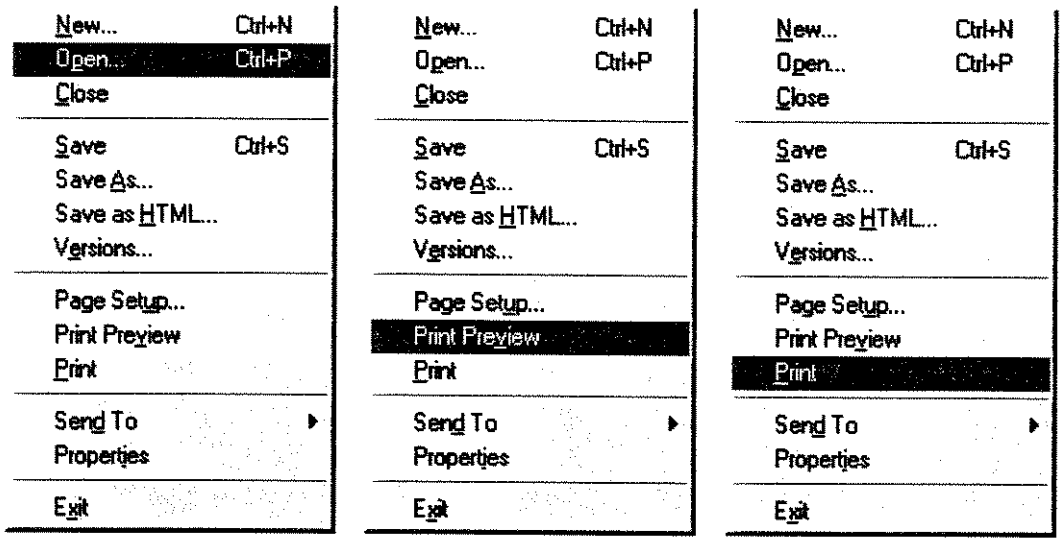

Figura 38: Ítens de menu ativados pela letra "p"

Ainda analisando o objeto menu da aplicação *simword97*, pode-se detectar pela métrica **[Me_L]:legenda...shortcut_padrão.sn** que o item de menu "*Open*" possui um *shortcut* que é padrão para outro comando (e.g., Ctrl+P normalmente é associado ao item de menu "*Print*"). Então existem, na verdade, dois problemas. Um deles é o comando "*Print*" não ter um *shortcut*; o segundo é o shortcut do item de menu "*Open*" possuir um *shortcut* não padrão. Esta métrica, por depender de uma lista padrão, só detecta uma associação de *shortcut* errada. Pela métrica **[Me_L]:legenda...shortcut_padrão.sn.qtd**, tem-se a quantificação dessas associações.

Pela métrica **[Me_L]:legenda...3pontos.sn** consegue-se verificar que o item de menu "*Print*" está sem os três pontos, o que é errado, visto que os três pontos indicam a chamada de uma caixa de diálogo. Pela métrica **[Me_L]:legenda...3pontos.sn.qtd**, tem-se a quantificação dessas associações.

Outro problema que não pode ser visualizado nesse anexo, pois exigiria "rodar" o programa, é a ativação do item de menu "*Open*". Esse item, ao invés de chamar a sua respectiva CD, chama a CD "*Print*", nitidamente um problema de repetição de *callbacks*. As três métricas que podem detectar tal problema são: **[Me_L]:*callback*...cd_funrepetida.sn**, que assumiria o valor "s" ao checar o item de menu "*Print*", **[Me_L]:*callback*...cd_funrepetida.sn.qtd**, que assumiria valor igual a 1 e **[Me_L]:*callback*...cd_funrepetição.sn** que assumiria valor igual a 1, por haver apenas uma repetição dessa *callback*.

Em relação ao objeto menu, deve-se acrescentar que, em função da simplificação do exemplo (e.g., ausência de ícones), muitas métricas não podem ser constatadas.

Para fazer uma análise dos objetos botão de comando e janela, é mais fácil mostrar os problemas variando entre os diferentes contextos, indo do mais simples ao mais complexo (e.g., no caso do objeto botão de comando existem os contextos [BC], [nBC] e [mnBC]), tendo como referência as caixas de diálogo às quais tais objetos pertencem.

Na CD da Figura 39, fazendo uma análise do contexto [BC] (objeto botão de comando), diversos outros problemas podem ser detectados. Por exemplo, as métricas **[BC_L]:*callback*...padrão.sn** e **[BC_L]:*callback*...sublinhado.sn** identificam que ambos os botões são padrão e o primeiro tem um caracter sublinhado, ao passo que o segundo, não. Através da métrica **[BC_L]:*callback*...sublinhado_padrão.sn** pode se detectar que o botão "*Cancel*" possui um sublinhado, o que não é padrão em nenhuma das aplicações desenvolvidas pela *Microsoft*. Particularmente os botões "*Ok*" e "*Cancel*" nunca têm caracteres sublinhados.

O contexto [nBC] (n ocorrências do objeto botão de comando presentes numa CD), para esta CD, não apresenta qualquer problema.

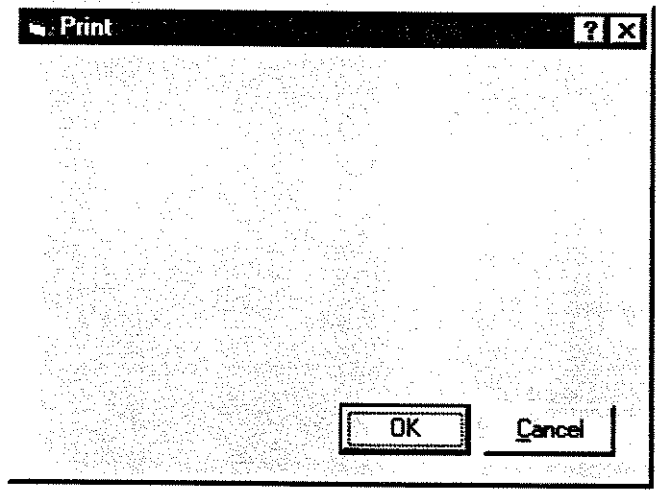

Figura 39: Caixa de Diálogo "*Print*"

Na Figura 40, uma análise do contexto [BC] não apresenta nenhum problema. Já uma análise do contexto [nBC], apresenta alguns. A métrica **[nBC_G]:...quadrado.sn** mostra que existe uma diferença entre os botões de comando: o objeto botão "*Cancel*" está maior que o botão "*OK*". Destaca-se aqui que a métrica **[nBC_G]...razão_aspecto.sn** seria incapaz de detectar tal diferença, visto que a razão de aspecto (relação altura/largura) dos objetos botão de comando é igual. A métrica **[nBC_G]:...quadrado.sn.qtd** quantifica as ocorrências. A questão do posicionamento do objetos botão de comando será visto no contexto [mnBC], mais a frente.

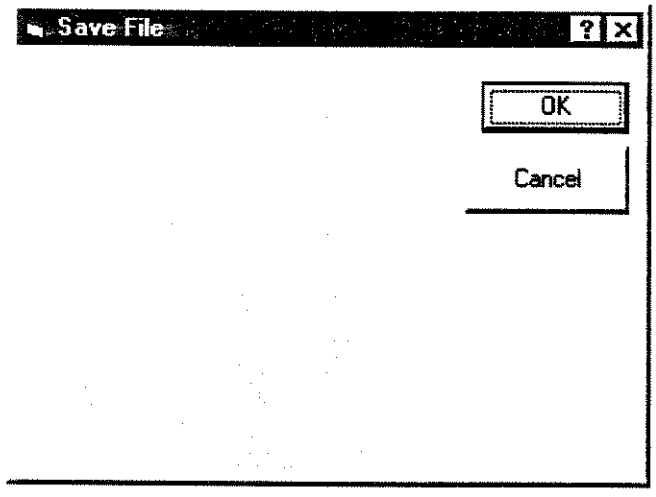

Figura 40: Caixa de Diálogo "*Save File*"

Na CD da Figura 41, fazendo uma análise do contexto [BC], a métrica **[BC_T]:legenda...ortografia.sn** detecta que o objeto botão com legenda "*Cancelk*" possui uma legenda com erro ortográfico.

O contexto [nBC] apresenta outros problemas: a métrica **[BC_G]:...sobreposição.sn** mostra que o objeto botão cuja legenda é "*OK*" está se sobrepondo ao objeto botão cuja legenda é "*Options*". A métrica **[BC_G]:...sobreposição.sn.qtd** quantifica as ocorrências. A métrica **[nBC_G]:...desalinh_h.sn** mostra que os botões estão desalinhados, sendo que a métrica **[nBC_G]:...desalinh_h.sn.qtd** quantifica tais ocorrências, tomando por base o primeiro botão.

Ainda no contexto [nBC], a métrica **[nBC_L]:legenda...seqpadrão.sn** identifica que os botões "*OK*" e "*Cancel*" (após correção) são uma sequência padrão e devem ser invertidos. A métrica **[nBC_L]:legenda...seqpadrão.sn.qtd** quantifica as ocorrências.

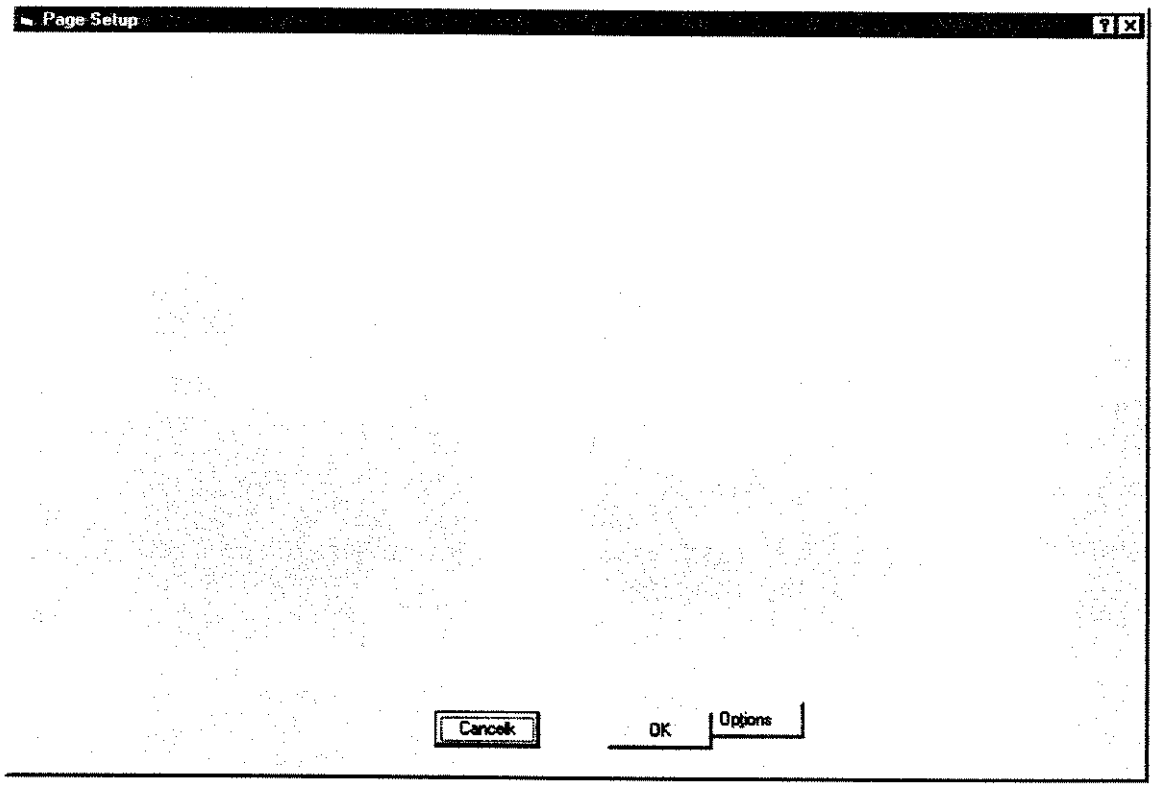

Figura 41: Caixa de Diálogo "*Page Setup*"

Uma vez vistos os contextos relacionados aos objetos e às caixas de diálogo, pode-se abordar o contexto da aplicação. Pelo exemplo dado, destacam-se os problemas a seguir.

Na Figura 40, no contexto [MexnJa] (menu relacionado às n janelas da aplicação), a métrica **[MexnJa]:legenda...igual.sn** aponta para o fato de que o item de menu do objeto menu e a legenda da barra de título do objeto janela não são iguais. O primeiro objeto apresenta a descrição "*Save*", ao passo que o segundo, "*Save File*". Para quantificar as ocorrências usa-se a métrica **[MexnJa]:legenda...igual.sn.qtd**.

No contexto [mnBC] (n ocorrências do objeto botão de comando na aplicação), comparando a Figura 39 com a Figura 40 e com a Figura 41, detectam-se mais problemas. A métrica **[mnBC_L]:...grupo_padrão_localização_padrão.sn** identificará que existem posições diferentes para o grupo padrão "*Ok*" e "*Cancel*" nessas CDs. Se o *designer* optar pela posição usada na CD "*Print*" (Figura 39), visto anteriormente, os demais estarão fora do padrão. A métrica **[mnBC_L]:...grupo_padrão_localização_padrão.sn.qtd** irá quantificar as ocorrências.

Os últimos problemas levantados neste exemplo referem-se ao contexto [nJa] (n ocorrências do objeto janela na aplicação). Embora a imagem capturada na Figura 41 não seja muito clara, essa CD possui uma semelhante razão de aspecto com as demais CDs. Entretanto, ela é bem maior em proporção, o que, por padrão seria errado. As métricas a seguir identificam o problema: **[nJa_G]:área...quadrado.min**, **[nJa_G]:área...quadrado.max** e **[nJa_G]:área...quadrado.med**.

É possível que alguns erros aqui apresentados propositadamente, de fato sejam difíceis de ocorrer. Mas a proposta dessas métricas é facilitar a verificação do desenvolvimento da interface, visto que um *designer* pode estar desenvolvendo concorrentemente inúmeras aplicações, cada qual com inúmeras caixas de diálogo, compostas por inúmeros objetos.