UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

Abordagem Baseada em Objetivos para Construção de Casos de Uso e Cenários

por

ROBERTO VEDOATO

Dissertação submetida à avaliação,
como requisito parcial para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Marcelo Soares Pimenta
Orientador

Porto Alegre, junho de 2003.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Vedoato, Roberto

Abordagem Baseada em Objetivos para Construção de Casos de Uso e Cenários / por Roberto Vedoato. Porto Alegre: PPGC da UFRGS, 2003.

91 f.:il.

Dissertação (Mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR-RS, 2003. Orientador: Pimenta, Marcelo.

1. Engenharia de Software 2. Interação Humano-Computador 3. Construção de Casos de Uso e Cenários 4. Abordagem Orientada a Objetivos I. Pimenta, Marcelo Soares II. Título.

A minha mãe, com todo amor e carinho

Agradecimentos

Agradeço a Deus e a Nossa Senhora, em primeiro lugar, por guiarem meus passos ao longo da vida;

À minha família, pais Ademar Vedoato e Ofélia Chimentão Vedoato e irmãos Flávio Anselmo Vedoato e Valéria Vedoato, pelo constante amor, apoio e incentivo, dando-me suporte para chegar até aqui. Tenho certeza de meu sucesso ser também o deles;

A todos professores do Instituto de Informática, em especial a meu orientador, Marcelo Soares Pimenta, pessoa que me inspirou e, desde o princípio, não poupou esforços para me orientar. Sem dúvida, meus conhecimentos em ES e IHC cresceram profundamente, trabalhando com ele. Ao ilustre mestre, minha sincera gratidão;

A meus grandes amigos: Rafael Batistela Parisotto, Fábio Luis Secco, Cássio Duran Savioli, Bertoldo Prellwitz e Ronald Pablo Meneses pelos ideais e companheirismo;

Aos colegas de turma, particularmente a Carlos Gomiero Maluf, Dionísio Pinheiro de Oliveira, Fernando Manchini Serenato, Leonardo Mota Pinheiro e Marcel Watanabe, pelo esforço coletivo que fez com que obtivéssemos maior proveito do mestrado;

A todos aqueles que, de forma direta ou indireta, contribuíram à realização deste trabalho.

Sumário

Lista de Abreviaturas

ATM	Automatic Teller Machine
CREWS	Cooperative Requirements Engineering With Scenarios
CTT	ConcurTaskTree
ER	Engenharia de Requisitos
ES	Engenharia de Software
GBRAM	Goal-Based Requirements Analysis Method
GOMS	Goal Operators Methods Selection
GRASP	General Responsibility Assignment Software Patterns
HTA	Hierarquical Task Analysis
IHC	Interação Humano-Computador
IU	Interface do Usuário
LEL	Language Extended Lexicon
MAD	Méthode Analytique de Description
OMT	Objecto Modeling Technique
OO	Orientado a Objeto
OOSE	Object-Oriented Software Engineering
RC	Requirements Chunks
RNF	Requisito não Funcional
TAREFA	Task Analysis based Requirements Engineering FrAmework
UAN	User Action Notation
UML	Unified Modeling Language
XML	Extensible Markup Language

Lista de Figuras

Lista de Tabelas

Resumo

Para o desenvolvimento de sistemas interativos que respeitem critérios de usabilidade em adição aos critérios de qualidade convencionais, é necessário que, desde suas primeiras etapas, as áreas de Engenharia de Software (ES) e de Interação Humano-Computador (IHC) sejam consideradas, simultaneamente e de maneira integrada. Essas duas áreas investigam modelos, conceitos, técnicas e práticas que refletem diferentes perspectivas sobre a atividade de desenvolvimento, uma orientada mais ao sistema (ES) e outra, mais ao usuário (IHC). Para conciliar estas perspectivas, é necessário o estabelecimento de um entendimento mútuo e a utilização conjunta e integrada de conceitos, técnicas e práticas de desenvolvimento de ambas as áreas. Este trabalho visa mostrar as possibilidades desta integração, através da combinação dos conceitos de Casos de Uso (*Use Cases*) e Cenários (*Scenarios*), importantes técnicas de modelagem amplamente utilizadas respectivamente nas áreas de ES e IHC, em diferentes contextos, com diferentes visões; mas apresentando similaridades valiosas para propiciarem o uso complementar de ambas as técnicas. Para sistematizar esta integração, é proposta uma abordagem teleológica – baseada em objetivos – de construção sistemática de casos de uso com quatro diferentes níveis de abstração, desde os mais abstratos casos de uso essenciais até os cenários, aqui utilizados como instâncias concretas de casos de uso. Com esta abordagem, pretende-se construir um modelo de casos de uso que permita especificar requisitos funcionais, conjuntamente com requisitos de interação, de maneira compreensível e praticável e que sirva como ponto de partida à continuidade do desenvolvimento orientado a objetos de software. Com o intuito de exemplificar a proposta, é descrita e discutida a aplicação passo a passo desta abordagem a um exemplo.

Palavras-chave: Engenharia de Software, Interação Humano-Computador, Construção de Casos de Uso e Cenários, Abordagem Orientada a Objetivos.

TITLE: "GOAL-BASED APPROACH FOR CONSTRUCTION OF USE CASES AND SCENARIOS"

Abstract

For the development of interactive systems that respect usability criteria in addition to the conventional quality criteria, it is necessary to consider simultaneously and jointly from their first stages, the areas of Software Engineering (SE) and of Human Computer Interaction (HCI). Those two areas investigate models, concepts, techniques and practices that reflect different perspectives about the development activity, one more oriented to the system (SE) and other, more to the user (HCI). To conciliate these perspectives, it is necessary the establishment of a mutual understanding and the united and integrated use of concepts, techniques and practices of development of both areas. This work aims to show the possibilities of this integration, through the combination of the concepts of Use Cases and Scenarios, important modelling techniques widely used respectively in the areas of SE and HCI, in different contexts, with different visions; but presenting valuables similarities to propitiate the complemental use of both techniques. To systematize this integration, is proposed a teleological approach - goal-based - of systematic construction of use cases with four different abstraction levels, from the most abstract essentials use cases to the scenarios, here used as concrete instances of use cases. With this approach, it intends to build a use case model to allow to specify functional requirements, jointly with interaction requirements, in a comprehensible and practicable fashion and that serves as starting point to the continuity of the object oriented software development. With the intention of exemplifying the proposal, the application step by step of this approach to an example is described and discussed.

Keywords: Software Engineering, Human Computer Interaction, Use Cases and Scenarios Construction, Goal-Driven Approach.

1 Introdução

Freqüentemente, sistemas interativos são desenvolvidos com visões isoladas das áreas de Engenharia de Software (ES) e de Interação Humano-Computador (IHC). ES, tradicionalmente, prioriza aspectos essencialmente funcionais dos sistemas, como eficiência, manutenibilidade e portabilidade, conferindo ao desenvolvimento uma orientação funcional em detrimento da operacional. Já IHC foca principalmente a usabilidade dos sistemas, concentrando atenção aos aspectos de interação e não considerando adequadamente os aspectos enfatizados pela ES [CYB 98]. Essas duas abordagens refletem diferentes perspectivas, uma mais orientada ao sistema (tecnológica) e outra mais orientada ao usuário (humana), sobre a mesma atividade de desenvolvimento.

Recentes trabalhos de pesquisa convergem para idéia central de, para desenvolver sistemas interativos úteis e usáveis, ser necessário considerar as perspectivas de ES e IHC, desde as primeiras etapas do desenvolvimento do sistema, exigindo a utilização conjunta e integrada de conceitos, técnicas e metodologias de desenvolvimento de ambas as áreas. Porém, essa interseção não é ainda nem natural nem objetiva: cada área considera aspectos diferentes e, muitas vezes, disjuntos do sistema, sem nenhuma correspondência explícita e sistematicamente estabelecida, tendo como conseqüência o fracionamento de requisitos [PIM 2000]. Compor o desenvolvimento de sistemas com grupos multidisciplinares, com diferentes pontos de vista sobre o processo de desenvolvimento, tem como potencial problema promover um entendimento mútuo da mesma tarefa de desenvolvimento e dos objetivos comuns [ABO 94].

Esta dissertação visa mostrar as possibilidades desta integração, através da combinação dos conceitos de Casos de Uso (*Use Cases*) e Cenários (*Scenarios*), importantes técnicas de modelagem, amplamente utilizadas, respectivamente nas áreas de ES e IHC, em diferentes contextos, com diferentes visões; mas apresentando similaridades valiosas para propiciarem o uso complementar de ambas as técnicas. Ambos são descrições narrativas. Cenários são utilizados em IHC para diversos fins, todavia particularmente úteis para inspecionarem atividades humanas ao usar um, presente ou futuro, artefato [CAR 95] e mediarem a comunicação entre grupos multidisciplinares, enquanto que casos de uso são usualmente utilizados, em ES, para modelarem requisitos funcionais e manipularem modelos de objetos [JAC 94a]. Baseando-se em estudo realizado previamente [VED 2001], tem-se a convicção de serem conceitos valiosos para combinar as diferentes perspectivas das duas áreas citadas, permitindo uma busca integrada da qualidade interna e externa dos sistemas. Fazendo uso do conceito de níveis de abstração e partindo do enfoque de que cenários são instâncias concretas de casos de uso, podem-se combinar os propósitos e usos complementares de ambas as técnicas.

Contudo, abordagens apontam que casos de uso apresentam problemas conceituais [REG 95a, REG 95b] e a maneira pela qual são comumente usados na *Unified Modeling Language* (UML) [BOO 99], não considera adequadamente aspectos de usabilidade [CON 2000b].

Várias pesquisas têm focado, principalmente, as funções e os usos de cenários e casos de uso, no processo de desenvolvimento de sistemas, enquanto que o processo de

construção destes, muitas vezes, é negligenciado. Desenvolvedores, freqüentemente, não sabem como identificar casos de uso, o que incluir neles, qual o nível de detalhamento, como representá-los e como estruturá-los. A abordagem original de casos de uso [JAC 92, JAC 94a, JAC 94b, JAC 94c], posteriormente adotada na UML, não define precisamente nenhum formato específico para descrever seus conteúdos, nem um processo sistemático para construí-los. A falta de guias para a construção de casos de uso é certamente uma das desvantagens das abordagens baseadas em casos de uso para Engenharia de Requisitos (ER). Caso não sejam devidamente construídos, ou, ainda, sejam ambíguos, incompletos ou inconsistentes, as interações que os casos de uso representam também herdarão essas propriedades [ROL 98a].

Ao se criar um software, não se planeja e se desenvolve apenas um artefato, criam-se novas possibilidades para ações e interações humanas [CAR 95]. Todo software é uma ferramenta. Como boa ferramenta, deve suportar o trabalho de alguém, tornar o trabalho mais fácil, mais rápido, mais simples e mais flexível. Para desenvolverem-se sistemas que suportem adequadamente o uso, trabalhos de pesquisa enfatizam que se precisa entender melhor as tarefas realizadas pelas pessoas e aplicar de maneira mais eficiente a compreensão das tarefas no processo de desenvolvimento de software (ver ciclo tarefa-artefato [CAR 95] na seção 2.1.4).

As atividades de pessoas, no trabalho, podem ser descritas como "tarefas". Segundo [STO 95] o termo "tarefa" é definido como: *"uma tarefa é um objetivo junto com conjuntos ordenados de ações que o satisfariam em contextos apropriados"*. Baseando-se na teoria da ação, sabe-se que, antes de executar uma seqüência de ações, elaboram-se planos, e o ponto inicial de um plano é a formulação de um objetivo [CAR 95]. Seguindo esta perspectiva, está claro que, para desenvolver um sistema interativo, é necessário, em primeiro lugar, conhecer os objetivos dos usuários para, então, propor a especificação dos mecanismos que sustentarão as tarefas necessárias para alcançá-los.

Trabalhos de pesquisa já reconhecem a importância da relação entre objetivos e casos uso. Segundo Kaindl, somente se pode entender as interações descritas em um caso de uso, quando se conhece seu objetivo [KAI 95]. Quando se está familiarizado com um sistema, os objetivos das interações parecem óbvios; todavia, no processo de elicitação de requisitos e modelagem de tarefas de um novo sistema, ainda desconhecido, saber o objetivo é uma informação crucial para a compreensão de cada caso de uso.

Na realidade, outras abordagens também consideram serem os objetivos fundamentais para construção de cenários e/ou casos de uso [KLA 93, POT 95, KAV 96, REG 96, COC 97a, COC 97b, ROL 98b, CON 2000a]; porém poucos oferecem um modo de se utilizarem complementarmente as duas técnicas para a determinação dos requisitos. Na maioria das abordagens, ou não existe uma notação para objetivos, ou não existe um processo sistemático de construção e validação de casos de uso, a partir dos objetivos, ou, até mesmo não, existem ambos.

Para sistematizar esta integração, este trabalho tem como principal objetivo investigar os conceitos de cenários e casos de uso e propor uma abordagem teleológica – baseada em objetivos – de construção sistemática de casos de uso com quatro

diferentes níveis de abstração, desde os mais abstratos casos de uso essenciais até os cenários, aqui utilizados, como instâncias concretas de casos de uso.

Propõe-se uma abordagem que, a partir de um modelo explícito de objetivos dos usuários, refinados e representados, permite definir, refinar e representar os objetivos do sistema que servirão como base para a construção de casos de uso em quatro diferentes níveis de abstração, essencial, singular, operacional e concreto, cada um relativo a um tipo de conhecimento específico.

A abordagem é um processo interativo e iterativo de construção, refinamento, experimentação, modificação e integração de casos de uso. Propõe-se a construção de casos de uso essenciais, nível mais abstrato, pelo analista, a partir dos objetivos; ainda pelo analista, o refinamento de casos de uso essenciais, visando definir casos de uso singulares; a instanciação dos casos de uso concretos, cenários; e, quando desejável, a definição dos casos de uso operacionais, a partir dos casos de uso singulares; a experimentação e avaliação da funcionalidade e do comportamento dos cenários pelos usuários, visando investigar suas reais necessidades; a revisão dos casos de uso pelo analista, procurando adequá-los às expectativas dos usuários; e, finalmente, a integração, pelo analista, dos casos de uso obtidos neste processo recursivo.

Com esta abordagem, pretende-se construir um modelo de casos de uso que permita especificar requisitos funcionais conjuntamente com os de interação, de maneira compreensível e praticável e que sirva como ponto de partida para a continuidade do desenvolvimento orientado a objetos de software.

Os objetivos específicos do trabalho são os seguintes:

- Combinar os conceitos de cenários e casos de uso, contemplando mutuamente as perspectivas de ES e IHC;
- Promover uma abordagem de construção de casos de uso centrada no uso (*usage-centered*);
- Investigar e adotar notações casos de uso adequadas para cada nível de abstração;
- Considerar aspectos problemáticos de casos de uso, abordados por pesquisadores da área, e propor maneiras de resolvê-los ou contorná-los;
- Sistematizar a construção de casos de uso e cenários, conduzindo e orientando o autor do caso de uso, através de diretrizes, heurísticas e *templates*;
- Permitir compatibilidade com técnicas de modelagem conhecidas por grande parte dos desenvolvedores, como, por exemplo, a UML, de modo a possibilitar a continuidade do desenvolvimento;

Acredita-se que a combinação de casos de uso e cenários com uma boa compreensão das tarefas e do contexto organizacional em que elas são executadas, juntamente com a explicitação dos objetivos, previnam a possibilidade de que especificações se tornem desencontradas das reais necessidades dos usuários e devam ser a base para um processo de desenvolvimento de software interativo.

Com o intuito de exemplificar a proposta, é descrita e discutida a aplicação passo a passo desta abordagem a um exemplo.

O restante do trabalho é estruturado como segue: no próximo capítulo, conceitos e fundamentos de cenários e casos de uso, sob a ótica, respectivamente, de IHC e ES, são apresentados e discutidos; em seguida, no capítulo 3, apresenta-se uma visão panorâmica sobre abordagens teleológicas e comparam-se algumas propostas; no capítulo 4, apresenta-se a proposta; e, no capítulo 5, ilustra-se a aplicação passo a passo da abordagem ao exemplo do caixa eletrônico; por fim, no capítulo 6, são apresentadas considerações finais e perspectivas de trabalhos futuros.

2 Cenários e Casos de uso: Fundamentos e Conceitos

2.1 Cenários: Visão de IHC

Segundo definição do dicionário Aurélio [FER 99], o termo "cenário" significa: *"Lugar onde ocorre algum fato, ou onde decorre a ação, ou parte da ação, de uma peça, romance, filme, etc"*. Em IHC, o comum significado para o termo "cenário", é uma instância representativa de uma interação entre usuário e sistema [CAM 92]. Um cenário é uma descrição narrativa informal e concreta a respeito de um uso de um sistema por uma pessoa.

2.1.1 Aplicação

Na literatura de IHC, há um amplo consenso sobre os benefícios da utilização de cenários, no processo de desenvolvimento de software, e, também, que cenários podem ser usados para muitos propósitos diferentes. Em IHC, cenários são particularmente úteis para

- ilustrar como um usuário provavelmente efetue tarefas particulares com um, presente ou futuro, sistema;
- prover uma representação de projeto, formulada em ações e experiências das pessoas que, efetivamente, usarão a tecnologia;
- avaliar a usabilidade de sistemas;
- guiar o projeto de interfaces dos usuários (IUs);
- testar teorias de IHC;
- mediar a comunicação entre grupos multidisciplinares [CAM 92, CAR 2000].

2.1.2 Notação

Cenários geralmente são representados por narrativas textuais, em linguagem natural, que, freqüentemente, são aprimoradas por gráficos, diagramas e imagens [RYS 2000]. Podem também ser representados através de protótipos, *storyboards*, vídeos, maquetes, ou até mesmo por situações físicas planejadas para suportar as atividades de um usuário [CAR 2000]. Um exemplo de cenário pode ser observado na figura 2.1.

Maria deseja utilizar o caixa eletrônico do banco X para retirar RS 50,00 em dinheiro de sua conta corrente; Ela passa seu cartão no leitor do caixa eletrônico pronto para ser usado. O sistema valida o cartão e, em seguida, requisita a senha a Maria. Pelo teclado, Maria digita sua senha. O sistema valida a senha digitada e mostra na tela o menu de opções de transações. Maria escolhe a opção saque, tocando no botão com esta denominação que está na tela; em seguida escolhe R$ 50,00 no menu de possíveis quantidades de dinheiro. O caixa automático verifica a quantia escolhida, a existência de fundos, aprova a transação e libera a quantia correta e um recibo do saque. Maria pega o dinheiro e o recibo nos locais especificados, o sistema debita a quantia da conta corrente, exibe uma mensagem, pedindo para o cliente certificar-se de que está de posse do seu cartão magnético; e, por fim, mostra novamente o menu de opções de transações.

FIGURA 2.1 – Exemplo de cenário

2.1.3 Estrutura

Uma coleção de cenários não possui uma estrutura definida, é representada através de um conjunto não estruturado.

2.1.4 Uso

Cenários são utilizados tanto de maneira formal quanto informal, variando seu uso de partes constituintes do processo de desenvolvimento; por exemplo, no desenvolvimento baseado em cenários e no desenvolvimento participativo de software; até usos mais espontâneos; por exemplo, para propósitos de comunicação [KLA 93].

Em geral, desenvolvedores parecem não considerar cenários como parte integrante do método de desenvolvimento [ABO 94]. Estudos empíricos demonstram que programadores avaliam projetos, simulando cenários mentalmente [BEN 93]. O uso informal de cenários pode ser observado, quando desenvolvedores têm algum ponto de discordância, exploram opções de projeto, procuram esclarecer requisitos do sistema etc. [KLA 93].

Quando utilizados de maneira informal, cenários são construídos *ad hoc*, nenhum método específico é utilizado para gerá-los ou avaliá-los [ABO 94]; seus conteúdos provêem de muitas fontes diferentes, desde experiências dos desenvolvedores e conhecimento do domínio a reais observações dos usuários. Não existem compromissos com o que um cenário deve conter ou o quão representativo deve ser para uma futura situação de uso [KLA 93]. Os usos informais ou implícitos de cenários são predecessores dos usos mais formais ou explícitos.

Segundo Carroll [CAR 95], o desenvolvimento tecnológico tem sido dificultado pela incompleta compreensão da prática. Geralmente, computadores são vistos como artefatos da atividade humana por meio de especificações funcionais, o que gera uma visão idealizada de uso, não de como um usuário ativaria uma função e experimentaria seus efeitos. Computadores são mais do que funcionalidade. Eles inevitavelmente reestruturam atividades humanas, criando novas possibilidades, assim como novas dificuldades [CAR 2000]. Cenários são meios concretos para descreverem situações existentes e para projetarem futuras situações de uso, facilitando a comunicação efetiva entre usuários e desenvolvedores sobre os requisitos do sistema e opções de projeto.

Extrair dos usuários aquilo que desejam e de que necessitam não é uma tarefa fácil. A ciência cognitiva tem demonstrado que a inteligência humana está organizada em *chunks*, que reconhecem e respondem a situações específicas. A resolução de problemas, pelos humanos, tende a ser altamente situada; por exemplo, ao invés de realizarem planos detalhados antecipadamente, pessoas respondem aos detalhes de uma situação, conforme eles aparecem [BEN 93]. Representar o uso de um sistema através de um conjunto de cenários torna o uso explícito, isso pode ajudar a programadores e analistas a focarem atenção na compreensão das pessoas e suas tarefas, aspectos, muitas vezes, implícitos nos sistemas [CAR 2000]. O real uso de um sistema pode ser concretamente descrito por um conjunto de cenários [CAR 95].

Enquanto abordagens tradicionais lidam com a complexidade do processo de desenvolvimento, através de abstração, o desenvolvimento baseado em cenários utiliza concretização. Ao invés de desenvolver software, listando requisitos, funções e módulos

de código, o desenvolvedor foca primeiramente as tarefas dos usuários que precisam ser suportadas pelo sistema e, então, através de cenários, realiza descrições dessas para guiar todo o processo de desenvolvimento.

Na abordagem baseada em cenários, estes são construídos, baseados no ciclo tarefa-artefato: as tarefas que as pessoas estão engajadas e aquelas que elas desejam realizar definem os requisitos da futura tecnologia, incluindo novos artefatos de IHC. Os novos artefatos criam novas possibilidades para as tarefas humanas, novas maneiras de executar coisas familiares e atividades completamente novas para fazer; mas, também, criam novas complexidades de aprendizagem e performance, novas interações entre tarefas e, naturalmente, novas dificuldades e novos erros para as pessoas. Essas novidades, eventualmente, tornam-se requisitos dos próximos desenvolvimentos tecnológicos, provocando novas transações [CAR 95]. A figura 2.2 ilustra o ciclo tarefa-artefato.

FIGURA 2.2 – Ciclo tarefa-artefato

Para construir representações de cenários, não se pode contar somente com observações. Caso se desejar que cenários representem um papel pró-ativo, guiando o planejamento e desenvolvimento de um novo sistema, precisa-se ser capaz de criar representações de cenários, antes que qualquer versão do sistema tenha sido desenvolvida. Podem-se construir cenários que representam futuras situações de uso, através do uso coordenado de observações empíricas e de abstrações das teorias da atividade humana [CAR 95].

Portanto, o desenvolvimento baseado em cenários é uma técnica, orientada a tarefa, para fazer uma projeção do uso de um artefato, antes de construí-lo; nele, cenários podem ser usados ao longo de todo o ciclo de vida do software, desde a análise de requisitos e projeto do sistema até a documentação, o treinamento e a avaliação de protótipos.

2.2 Casos de Uso: Visão de ES

Casos de uso foram introduzidos por Jacobson como parte da metodologia de desenvolvimento de software OOSE (*Object-Oriented Software Engineering*) [JAC 92]. Sua definição original é *"Um caso de uso é uma maneira específica de usar um sistema usando alguma parte da funcionalidade. Constitui um curso completo de interação que acontece entre um ator e o sistema"*.

Por definição, o termo “caso de uso”, introduzido nesta metodologia, não é sinônimo do termo “cenário”. Um cenário deve ser entendido como uma instância específica de um caso de uso [JAC 94a], ao passo que o caso de uso é uma coleção de cenários, representando múltiplos caminhos.

A idéia geral de um caso de uso é representar seqüências de interações entre um sistema, mesmo que ainda não implementado, e o mundo externo ao sistema; ou seja, descreve maneiras de usar um sistema.

Muitas metodologias de desenvolvimento de software possuem alguma noção de casos de uso. Em [REG 96] pode-se encontrar uma visão geral, na forma de uma tabela comparativa, sobre como casos de uso são aplicados às metodologias Orientadas a Objeto (OO): OOSE, *Object Modeling Technique* (OMT) e Booch. A tabela 2.1 é uma transcrição parcial desta visão geral.

TABELA 2.1 – Visão geral de casos de uso em métodos OO [REG 96]

Metodologia	**OOSE**	**OMT**	**Booch**
Conceitos	Casos de uso, ator, exceção, extensão *(extends)*, inclusão *(uses).*	Caso de uso, ator, cenário, pré e pós-condição, exceção, *adds.*	Caso de uso, cenário, iniciador, pré e pós-condições.
Notação na fase de análise de requisitos	Linguagem natural para descrever casos de uso. Diagramas para descrever relações entre casos de uso.	Linguagem natural com algumas recomendações para estruturação.	Linguagem natural.
Função no processo de desenvolvimento	Guia todo processo, é usado para identificar objetos e para criar projetos robustos.	É usado para reforçar a análise de requisitos e para identificar objetos.	É usado para identificar objetos, para concepção e para planejamento de versão.
Metodologia para criação de casos de uso	Algumas heurísticas. Questões para responder a cada ator ajudam a identificar casos de uso.	Lista de ações passo a passo. Cenários são combinados ou generalizados em casos de uso.	Prescreve o planejamento de cenários como uma atividade e provê algumas poucas recomendações.

As semânticas dos conceitos, relacionados a casos de uso nos diferentes métodos, não são correspondentes e existe uma significante inconsistência entre os métodos com relação a como os conceitos são interpretados [REG 96].

Posteriormente, os autores das três metodologias, Jacobson (OOSE), Booch (Booch) e Rumbaugh (OMT), uniram e estenderam suas abordagens através de uma linguagem de modelagem unificada a *Unified Modeling Language* (UML), que emergiu como um padrão entre desenvolvedores de software OO. Conseqüentemente, neste trabalho, dar-se-á enfoque à UML.

A UML é uma linguagem padrão para elaboração da estrutura de projetos de software, empregada para a visualização, especificação, construção e documentação de artefatos que façam uso de sistemas complexos de software [BOO 99]. Ela integra técnicas de modelagem de diversas metodologias de desenvolvimento. Muitos dos conceitos sobre casos de uso, originalmente associados à metodologia OOSE, foram integrados à especificação da UML.

UML é linguagem de modelagem, não é metodologia, portanto é somente parte do método de desenvolvimento de software. A UML não prescreve explicitamente um processo de utilização; porém, para obter maior proveito da UML é preciso que o processo tenha a característica de ser baseado em casos de uso. Isto significa que casos de uso são utilizados como principal artefato para estabelecer o comportamento desejado do sistema, para verificação e validação da arquitetura do sistema, para a realização de testes e para comunicação entre os envolvidos no desenvolvimento do sistema [BOO 99].

A definição formal de caso de uso, segundo a UML [BOO 99], é *"Caso de uso é uma descrição de um conjunto de seqüências de ações, inclusive variantes, que um sistema realiza para produzir um resultado de valor observável a um ator"*.

Um ator representa um agente externo ao sistema que de alguma forma participa de um caso de uso. É chamado de ator um papel que uma pessoa, um dispositivo de hardware, ou, até mesmo, outro sistema desempenha com o software [BOO 99]. Atores ajudam a delimitar o sistema em desenvolvimento, por representarem agentes externos que interagem com o mesmo.

Existe distinção entre os conceitos de ator e usuário. Usuário é um conceito não formal. Ator representa um papel específico que um usuário pode atuar, enquanto que o usuário é alguém que utiliza o sistema [JAC 94a].

Um caso de uso é uma descrição narrativa de um processo do domínio da aplicação. Ele representa um requisito funcional do sistema [BOO 99] e captura o comportamento pretendido do sistema; sem, no entanto, especificar como este comportamento é implementado, descreve o que o sistema faz e não como é feito. Casos de uso proporcionam uma visão externa do sistema; através deles, o sistema é visto como uma caixa-preta (*black box*).

2.2.1 Aplicação

Caso de uso é uma técnica de modelagem que pode ser aplicada a qualquer metodologia de desenvolvimento de software, estruturada, ou OO. A semântica de um modelo de casos de uso relaciona-se fortemente com a de um modelo de objetos [JAC 94c]; porém, um modelo de casos de uso não substitui um de objetos. O modelo de casos de uso é uma visão externa de um sistema; o modelo de objetos é uma visão interna do mesmo sistema [JAC 94a].

Nos últimos anos, casos de uso têm recebido muita atenção na área de ES, como meio para elicitar, documentar e validar requisitos, no entanto, isto não significa que estão limitados a ER [REG 96, RYS 2000]; eles podem ser usados ao longo de todo o ciclo de vida do desenvolvimento do software.

Casos de uso são utilizados para diversas finalidades, entre as quais:

- Estruturar complexos modelos de objetos, em visões manejáveis;
- Capturar, documentar e validar requisitos funcionais de sistemas;
- Gerenciar a complexidade de desenvolvimento, focando um aspecto distinto por vez;
- Compreender e estabelecer o comportamento desejado do sistema;

- Ganhar percepção sobre modelos alternativos de software;
- Fornecer uma descrição consistente e clara sobre as responsabilidades a serem cumpridas pelo sistema, funcionando como uma espécie de contrato;
- Facilitar a comunicação entre os envolvidos no desenvolvimento do sistema;
- Envolver usuários no desenvolvimento de software;
- Verificar e validar a arquitetura de sistemas;
- Realizar testes de sistemas;
- Servir como fonte para construção de manuais;
- Derivar modelos conceituais;
- Reengenharia de software.

2.2.2 Notação

Existe uma grande variedade de estilos para descrever o conteúdo narrativo de um caso de uso. A abordagem original de casos de uso, proposta por Jacobson e, posteriormente, adotada na UML, não define precisamente nenhum formato ou *template* específico para descrever o conteúdo de um caso de uso. Originalmente, casos de uso não possuem notação formal; eles são representados na forma de narrativa textual contínua.

Os problemas deste estilo de narrativa são numerosos. Não há nenhuma separação clara entre o lado do usuário e o do sistema. A narrativa mistura exigências internas e externas e salta irregularmente entre perspectivas internas e externas. Os elementos essenciais à natureza do problema são misturados com decisões de projeto, e a falta de estrutura força o leitor a seguir o texto inteiro apenas para obter uma idéia geral do caso de uso [CON 2000b].

Um outro estilo comum para representar casos de uso é a seqüência numerada; nele, a narrativa é escrita como uma série de etapas numeradas. A figura 2.3 ilustra um exemplo extraído de [CON 2000b].

1. O caso de uso inicia, quando o cliente introduz um cartão no caixa automático. O sistema lê e valida a informação do cartão.
2. O sistema aguarda pela senha. O cliente entra com a senha. O sistema valida a senha.
3. O sistema pergunta que operação o cliente deseja executar. O cliente seleciona "saque de dinheiro".
4. O sistema pede a quantia. O cliente entra com a quantia.
5. O sistema pede o tipo. O cliente seleciona o tipo de conta (poupança, conta corrente).
6. O sistema comunica-se com a rede do caixa automático para validar o número da conta, o cartão, a senha e a disponibilidade da quantia pedida.
7. O sistema pergunta ao cliente se ele deseja um recibo. Esta etapa é executada somente se houver papel para a impressão do recibo.
8. O sistema pede que o cliente retire o cartão. O cliente retira o cartão. (Esta é uma medida de segurança para assegurar que os clientes não deixem seus cartões na máquina.).
9. O sistema libera a quantia pedida.
10. O sistema imprime o recibo.
11. O caso de uso termina.

FIGURA 2.3 – Exemplo de caso de uso na notação de seqüência numerada [CON 2000b]

Uma vantagem deste estilo é evidente: a separação em etapas distintas facilita a leitura do caso de uso para a obtenção de uma visão geral. Todavia, sofre muito dos mesmos problemas do estilo narrativo contínuo. Apesar da segmentação em etapas discretas, as etapas individuais mesclam ações do sistema e do usuário.

Ambos os estilos de narrativa, textual contínua e seqüência numerada, apresentam abundância de palavras. Devido ao fato de não possuírem quase nenhuma estrutura, esses estilos de narrativa requerem, para clareza, que a perspectiva seja repetidamente declarada - o sistema faz isto, o usuário escolhe isso, o sistema termina algo mais -, o que contribui para suas loquacidades. Mesmo assim, o limite entre o que está dentro do sistema e o que está fora não está explicito e, somente, poderá ser discernido através de uma leitura cuidadosa da narrativa inteira [CON 2000b].

A solução simples para isto, é separar completamente da interação o lado do usuário e o do sistema. Para casos de uso, esta separação foi originalmente sugerida por Wirfs-Brock, sendo criado o estilo de narrativa com partição [CON 2000b]. Neste estilo, a narrativa de casos de uso é dividida em duas colunas: ação do usuário e resposta do sistema. Assim, o limite das perspectivas internas e externas torna-se óbvio e explícito, sem repetição desnecessária. Por exemplo, a figura 2.4 é uma narrativa com partição do caso de uso, sacar dinheiro, representado na figura 2.3, como narrativa com seqüência numerada:

ação do usuário	**resposta do sistema**
insere o cartão no caixa automático	
	lê o cartão
	solicita a senha
entra com a senha	
	valida a senha
	mostra o menu de opções
seleciona a opção	
	mostra o menu de conta
seleciona a conta	
	solicita a quantia
entra com a quantia	
	mostra a quantia
confirma a quantia	
	retorna o cartão
pega o cartão	
	libera o dinheiro se disponível

FIGURA 2.4 – Exemplo de caso de uso na notação de narrativa com partição [CON 2000b]

Este estilo de narrativa é bem mais legível e apresenta uma menor verbosidade do que os outros estilos apresentados, anteriormente. Naturalmente, nada impede que se numerem as ações e as respostas, tornando-o ainda mais útil.

Casos de uso podem ainda ser representados por pseudocódigos. Embora tais expressões pareçam oferecer precisão e possam ser confortáveis e familiares aos engenheiros de software, elas, raramente, são compreensíveis aos usuários comuns [CON 2000b].

2.2.3 Estrutura

Em adição aos de casos de uso, Jacobson também introduziu um modo de representá-los graficamente: o diagrama de casos de uso, o qual também é parte integrante da UML.

Um diagrama de caso de uso exibe uma coleção de casos de uso, atores e seus relacionamentos. Essa notação permite visualizar um caso de uso em separado de sua realização e no contexto com outros casos de uso [BOO 99].

Graficamente, um caso de uso é representado por uma elipse com linhas contínuas, incluindo um nome que o diferencia dos demais; um ator é representado por uma figura esquematizada, um "boneco de palitos"; e as associações entre atores e casos de uso são representadas por linhas. O escopo do sistema é explicitado por uma caixa que envolve os casos de uso. A figura 2.5 mostra um exemplo de diagrama de caso de uso.

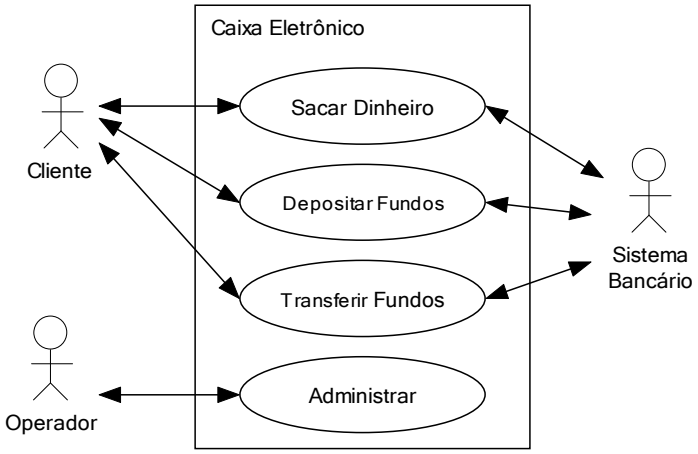

FIGURA 2.5 – Exemplo de diagrama de casos de uso

Diagramas de caso de uso fornecem um modo de descrever a visão externa do sistema e suas interações com o mundo exterior, representando uma visão de alto nível de funcionalidade intencional, mediante o recebimento de um tipo de requisição de usuário [FUR 98]. Esses diagramas definem a total funcionalidade do sistema e são importantes principalmente, para organização e modelagem de comportamento do sistema [JAC 94a, BOO 99].

2.2.4 Relacionamentos

Na UML, a conexão entre um ator e um caso de uso é denominada de associação. Ela indica que o ator e o caso de uso comunicam entre si, cada um com a possibilidade de enviar e receber mensagens.

Podem existir relacionamentos entre casos de uso, aplicados com a finalidade de fatorar comportamentos comuns e variantes, evitando redundâncias e aumentado o reuso. Os relacionamentos existentes na UML são generalização, inclusão e extensão [BOO 99].

Um relacionamento de generalização entre casos de uso indica que um caso de uso pode compartilhar o comportamento definido em um ou mais casos de uso. É um mecanismo usado para identificar comportamentos reutilizáveis.

O relacionamento de inclusão entre casos de uso, identificado pelo estereotipo “inclui” (do inglês “include”), significa que um caso de uso incorpora explicitamente o comportamento de outro caso de uso. O caso de uso, incluído, nunca permanece isolado, apenas é instanciado como parte de alguma base maior que o inclui. Este tipo de relacionamento é utilizado para evitar descrever o mesmo fluxo de eventos várias vezes, fatorando o comportamento comum em um caso de uso próprio. O relacionamento de inclusão é essencialmente um exemplo de delegação [BOO 99]. A figura 2.6 ilustra um exemplo de relacionamento de inclusão.

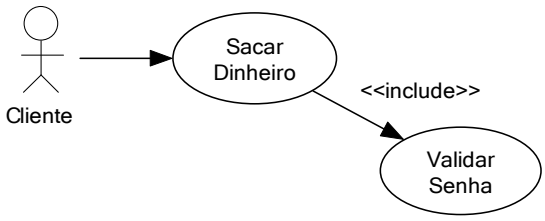

FIGURA 2.6 – Exemplo de relacionamento de inclusão

Já o relacionamento de extensão entre casos de uso, identificado pelo estereotipo “estende” (do inglês “extend”), significa que um caso de uso incorpora implicitamente o comportamento de um outro caso de uso, em um local especificado indiretamente pelo caso de uso estendido. Este relacionamento é usado para modelar um comportamento que ocorre em situações específicas, de modo a separar os comportamentos opcionais do comportamento obrigatório de um caso de uso. A figura 2.7 é um exemplo deste tipo de relacionamento.

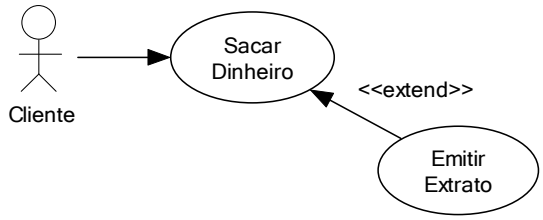

FIGURA 2.7 – Exemplo de relacionamento de extensão

Organizar os casos de uso, extraindo o comportamento comum, por meio de relacionamentos de inclusão, e diferenciando as variantes, através de relacionamentos estendidos, é uma parte importante para a criação de um conjunto de casos de uso simples, equilibrado e compreensível, para o sistema [BOO 99].

2.3 Discussão sobre Cenários e Casos de Uso

Existem ainda várias discussões ao redor de cenários e casos de uso. Os termos “cenário” e “caso de uso” significam diferentes coisas para diferentes pessoas: suas

definições, muitas vezes, não são claras, ora sendo utilizados como sinônimos, ora como conceitos distintos.

Em [COC 97a, COC 97b], encontra-se um estudo onde é apontado que as diversas definições de casos de uso diferem sobre quatro dimensões:

- Propósito: casos de uso podem ter o propósito de descrever histórias dos usuários ou especificar requisitos;
- Conteúdo: o conteúdo de um caso de uso pode ser contraditório ou consistente; quando consistente, pode estar na forma de narrativa textual ou notação formal;
- Pluralidade: casos de uso podem ser apenas um cenário ou conter múltiplos cenários;
- Estrutura: uma coleção de casos de uso pode ser representada através de uma estrutura formal, uma estrutura informal, ou, simplesmente, através de um conjunto não estruturado.

A definição original de casos de uso proposta por Jacobson [JAC 92] e, posteriormente, adotada na UML, tem o propósito de especificar requisitos, com o conteúdo na forma de uma narrativa textual consistente, contendo múltiplos cenários e possuindo uma estrutura semiformal.

Entretanto, o conceito de casos de uso ainda é vago e confuso, e existem vários aspectos problemáticos que precisam ser tratados. Abordagens apontam que casos de uso apresentam problemas conceituais, alguns deles são destacados em [REG 95a, REG 95b, CON 2000b] e mostrados a seguir:

- Notação e conteúdo: O que exatamente casos de uso contém? Como são organizados os conteúdos? O que significam os conteúdos?
- Expressões idiomáticas: Qual linguagem é usada para expressar os conteúdos que definem um caso de uso?
- Granularidade: O tamanho e o nível de detalhamento de cada caso de uso é escolhido arbitrariamente?
- Cobertura: Casos de uso podem garantir apenas uma cobertura parcial de todos possíveis usos do sistema?
- Contexto: Casos de uso ocorrem sobre quaisquer situações ou sobre condições específicas?
- Interseção: Casos de uso são independentes ou podem sobrepor-se total ou parcialmente? Como modelar os relacionamentos?
- Concorrência: Como modelar as concorrências internas e entre casos de uso?

Outro problema pertinente é que, na UML, casos de uso geralmente são referentes à interação sob a ótica do sistema, não considerando adequadamente aspectos de usabilidade. Conforme já destacado em [CON 2000a], observa-se que, na UML, a corrente definição de casos de uso, desviou-se da ênfase original de Jacobson, no uso: *"Um caso de uso é uma maneira específica de usar um sistema..."*; para um ponto de vista mais centrado no sistema: "*Caso de uso é uma descrição de um conjunto de seqüências de ações... que um sistema realiza..."*. O foco está no que o sistema executa, não no que o usuário faz ou deseja.

Casos de uso têm sido usados na ES para ganhar percepção em possíveis modelos de software, não focando as tarefas dos usuários, enquanto que cenários, em IHC,

capturam primeiramente o comportamento dos usuários. Deve-se atentar para não se modelar apenas a perspectiva do sistema (figura 2.8) ou a perspectiva do usuário (figura 2.9). É necessário estar centrado nas tarefas dos usuários e nas responsabilidades do sistema para suportá-las, ou seja, no uso do sistema.

Consultar Extrato
1. O caixa eletrônico captura a identificação.
2. O caixa eletrônico valida a identificação.
3. O caixa eletrônico coleta a transação extrato.
4. O caixa eletrônico coleta o período.
5. O caixa eletrônico valida o período
6. O caixa eletrônico emite o recibo do extrato.
7. O caixa eletrônico fica pronto para nova transação.

FIGURA 2.8 – Caso de uso com a perspectiva do sistema

Consultar Extrato
1. O cliente fornece sua identificação.
2. O cliente pede o extrato.
3. O cliente escolhe o período
4. O cliente pega o recibo do extrato.
5. O cliente sai.

FIGURA 2.9 – Caso de uso com a perspectiva do usuário

Como visto anteriormente, casos de uso e cenários são importantes técnicas de modelagem, amplamente utilizadas respectivamente em IHC e ES, em diferentes contextos, com diferentes visões; mas apresentando muitas similaridades. Embora ambos sejam descrições narrativas sobre interações, existem diferenças entre estes conceitos. A tabela a seguir destaca resumidamente algumas delas:

TABELA 2.2 – Quadro sintético, destacando diferenças entre cenários e casos de uso [VED 2001]

Conceito	**Cenário**	**Caso de Uso**
Aplicação	Comumente utilizado em IHC para inspecionar atividades humanas ao usar um, presente ou futuro, artefato. Concentra-se em aspectos de usabilidade.	Comumente utilizado em ES para modelar requisitos funcionais e manipular modelos de objetos. Concentra-se em aspectos funcionais.
Pluralidade	Uma seqüência específica de ações, um único caminho.	Conjunto de seqüências de ações, vários caminhos.
Nível de abstração	Concreto.	Diferentes níveis de abstração, mas tipicamente mais abstratos que cenários.
Notação	Descrito em linguagem natural.	Descrito em linguagem natural, semiformal ou formal.
Estrutura	Não há estrutura para representar um conjunto de cenários.	Uma coleção de casos de uso possui uma estrutura. Na UML, são os diagramas de casos de uso.
Abordagem	Provê uma abordagem *bottom-up*. Cenários são generalizados e sintetizados em casos de uso.	Provê uma abordagem *top-down*. Casos de uso são refinados respeitando suas propriedades estruturais.

Contudo, tem-se a convicção de serem conceitos valiosos para se combinarem as diferentes perspectivas das áreas de IHC e ES.

Fazendo uso do conceito de níveis de abstração e partindo do enfoque de cenários serem instâncias concretas de casos de uso, podem-se combinar os propósitos e usos complementares de ambas as técnicas. De fato, cenários têm sido utilizados com uma base comum entre diferentes tipos de modelagem. Um método de construção que integrasse as duas técnicas seria útil; pois permitiria uma busca integrada da qualidade interna e externa dos sistemas e, conseqüentemente, o desenvolvimento de sistemas interativos úteis e usáveis.

A integração destes conceitos às metodologias de desenvolvimento de software é também um importante aspecto a ser considerado. Em alguns casos, os conceitos são usados informalmente; em outros, fazem parte do processo de desenvolvimento; e, num ponto de destaque, quando suas potencialidades são exploradas mais efetivamente, são eles utilizados ao longo de todo o ciclo de vida do software, guiando o processo de desenvolvimento.

Todavia, não se podem representar através de casos de uso todos os requisitos de um sistema. Casos de uso são mais adequados para representar os requisitos comportamentais e os requisitos funcionais. Demais requisitos não funcionais (RNFs), como formato de dados, requisitos de performance, regras de negócio e fórmulas complexas, devem ser representados em outros formatos. Porém, casos de uso funcionam como uma estrutura central capaz de conectar vários tipos de requisitos, podendo ser ligadas a casos de uso informações associativas. Utilizando a metáfora da roda (figura 2.10), considera-se casos de uso como o eixo de uma roda que conecta outras informações, associadas aos raios, levando a diferentes direções [COC 2000]. Esta é uma das razões pelas quais muitos consideram casos de uso como o elemento central na ER e no processo de desenvolvimento, como um todo.

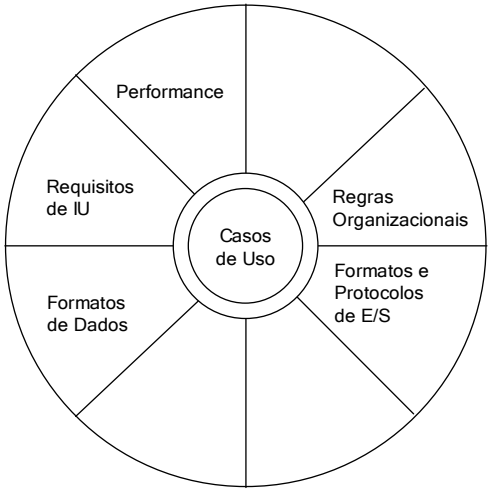

FIGURA 2.10 – Modelo de requisitos “eixo e raios” [COC 2000]

Segundo [CYS 2001] os RNFs, devem ser tratados desde as primeiras fases do desenvolvimento de software, trazendo para os casos de uso as informações e ações necessárias para satisfazerem o conjunto de RNFs elicitados. Os ciclos de requisitos

funcionais (visão funcional) e de RNFs (visão não funcional) devem ser integrados através do uso de pontos convergentes [CYS 2001]. De fato, os RNFs são modelados através de outra notação, como, por exemplo, RNFs Framework [CYS 2001]), não através de casos de uso; embora estes permitam ligar as duas visões. Casos de uso ajudam na clarificação do inter-relacionamento entre requisitos funcionais e RNFs.

Várias pesquisas têm focado principalmente as funções e os usos de cenários e casos de uso, no processo de desenvolvimento de sistemas, enquanto que o processo de construção destes, muitas vezes, é negligenciado. Desenvolvedores, freqüentemente, não sabem como identificar casos de uso, o que incluir neles, qual o nível de detalhamento, como representá-los e como estruturá-los. A abordagem original de casos de uso [JAC 92, JAC 94a, JAC 94b, JAC 94c], posteriormente adotada na UML [BOO 99], não define precisamente nenhum formato específico para descrever seus conteúdos, nem um processo sistemático para construí-los. A falta de guias para a construção de casos de uso é certamente uma das desvantagens das abordagens baseadas em casos de uso para ER. Caso não sejam devidamente construídos, ou, ainda, sejam ambíguos, incompletos ou inconsistentes, as interações que os casos de uso representam também herdarão essas propriedades [ROL 98a].

3 Abordagens Baseadas em Objetivos: Visão Panorâmica

3.1 Introdução

Ao se criar um software, não se planeja e não se desenvolve apenas um artefato, criam-se novas possibilidades para ações e interações humanas [CAR 95]. Todo software é uma ferramenta. Como boa ferramenta, deve suportar o trabalho de alguém, tornar o trabalho mais fácil, mais rápido, mais simples e mais flexível. Para desenvolverem-se sistemas que suportam adequadamente o uso, trabalhos de pesquisa enfatizam que se precisa entender melhor as tarefas realizadas pelas pessoas e aplicar de maneira mais eficiente a compreensão das tarefas no processo de desenvolvimento de software (ver ciclo tarefa-artefato [CAR 95] na seção 2.1.4).

As atividades de pessoas, no trabalho, podem ser descritas como "tarefas". Segundo [STO 95] o termo "tarefa" é definido como: *"uma tarefa é um objetivo junto com conjuntos ordenados de ações que o satisfariam em contextos apropriados"*. Baseando-se na teoria da ação, sabe-se que, antes de executar uma seqüência de ações, elaboram-se planos, e o ponto inicial de um plano é a formulação de um objetivo [CAR 95]. Seguindo esta perspectiva, está claro que, para desenvolver um sistema interativo, é necessário, em primeiro lugar, conhecer os objetivos dos usuários para, então, propor a especificação dos mecanismos que sustentarão as tarefas necessárias para alcançá-los.

Enquanto análises de sistema tradicionais focam "quais" características um sistema irá suportar, abordagens baseadas em objetivos focam "por que" sistemas são construídos, provendo motivação e argumento para justificar os requisitos de software [ANT 96]. Possibilitam assim uma análise mais ampla do contexto no qual o sistema irá operar. Focar objetivos, ao invés de requisitos específicos, permite analistas comunicarem com *stakeholders* - pessoas envolvidas no sistema em desenvolvimento -, usando uma linguagem baseada em conceitos que ambos têm familiaridade [ANT 96]. Todavia, especificar os requisitos de um sistema, a partir dos objetivos dos usuários, não é uma tarefa trivial. **Objetivos** são estados finais desejáveis de serem alcançados, não ações a serem realizadas [ANT 96]. O refinamento de objetivos em soluções de projeto é um processo de múltiplos estágios; vagos objetivos de alto nível devem ser refinados em concretos objetivos formais [KAV 96]. Isto é necessário, pois somente objetivos primitivos podem ser operacionalizados - traduzidos em ações atômicas do usuário, do hardware ou do sistema – para, então, se tornarem requisitos operacionais na especificação final de requisitos [POT 95].

Um dos fatores positivos de se enfatizarem objetivos, como fontes para o desenvolvimento, é que eles são consideravelmente estáveis [ANT 96]. Vários sistemas podem ser propostos para serem realizados os objetivos dos usuários: eles diferirão basicamente em como os objetivos são operacionalizados, ou seja, quais ações serão executadas para que os objetivos sejam alcançados e quem ou o que as realizará. Isto implica que é possível, a partir dos objetivos, especificar as futuras tarefas com o sistema, ou seja, as tarefas interativas [PIM 97].

Deve-se considerar que a operacionalização dos objetivos não deve ser predominantemente *top-down*, por ser altamente interativa [KAV 96]. Derivação de casos de uso e refinamento de objetivos são atividades do desenvolvimento de software que se apóiam mutuamente [POT 95, ROL 98b]. O conhecimento teleológico preocupa-se com os propósitos específicos para os quais o sistema foi projetado, enquanto a análise de casos de uso é dedicada a preencher a lacuna entre tais propósitos abstratos e a real estrutura e comportamento do sistema [KAV 96].

Casos de uso especificam um modo de operacionalizar um objetivo. Assim, é possível obter um conjunto de casos de uso, analisando objetivos, alocação de objetivos etc. Reciprocamente, é possível retornar e elaborar objetivos, quando se caminha através de um caso de uso, já que ele é uma forma natural para se descreverem as circunstâncias nas quais um objetivo pode falhar ou ser bloqueado, facilitando a descoberta de novos objetivos e a consideração de alternativas para a operacionalização dos mesmos. A análise do caso de uso pode fornecer percepções concretas sobre o comportamento do macrosistema - ambiente geral no qual o software é desenvolvido e deve operar - e as razões para tal, possibilitando a identificação de soluções mais plausíveis [POT 95, ANT 98a].

Trabalhos de pesquisa já reconhecem a importância da relação entre objetivos e casos uso. Segundo Kaindl, somente se podem entender as interações descritas em um caso de uso, quando se conhece seu objetivo [KAI 95]. Quando se está familiarizado com um sistema, os objetivos das interações parecem óbvios; todavia, no processo de elicitação de requisitos e modelagem de tarefas de um novo sistema, ainda desconhecido, saber o objetivo é uma informação crucial para a compreensão de cada caso de uso. De acordo com a abordagem [KAI 95] os propósitos da utilização de casos de uso são claramente definidos, já os propósitos das interações descritas nestes, geralmente são ignorados ou deixados implícitos. Faltam representações adequadas de objetivos em casos de uso.

Na realidade, outras abordagens também consideram serem os objetivos fundamentais para construção de cenários e/ou casos de uso [KLA 93, POT 95, KAV 96, REG 96, COC 97a, COC 97b, ROL 98b, CON 2000a]; porém poucos oferecem um modo de se utilizarem complementarmente as duas técnicas para a determinação dos requisitos. Na maioria das abordagens, ou não existe uma notação para objetivos, ou não existe um processo sistemático de construção e validação de casos de uso, a partir dos objetivos, ou, até mesmo não, existem ambos.

3.2 Algumas Propostas

Nesta seção, são apresentadas sucintamente três abordagens baseadas em objetivos para construção de casos de uso e/ou cenários. Notifica-se que se utilizará, ao longo deste trabalho, os termos “cenário” e “casos de uso”, de acordo com o entendimento apresentado no capítulo 2, embora possam ser utilizadas terminologias diferentes em algumas das abordagens referenciadas.

3.2.1 Abordagem de Potts [POT 94, POT 95]

Em [POT 94, POT 95], é apresentada a proposta de Colin Potts para a construção de casos de uso. Nesta abordagem, conceitos teleológicos são utilizados para gerarem casos de uso esquemáticos, utilizados para compreender as necessidades dos usuários.

É proposto um esquema de caso de uso com estrutura teleológica (figura 3.1) que, utilizado em conjunto com uma estratégia de refinamento de objetivos, guia a produção de casos de uso. O princípio é que a seção narrativa de um caso de uso consiste em uma seqüência de episódios, cada um correspondente a algum objetivo ou obstáculo. É **obstáculo** definido como comportamento ou, outro objetivo que bloqueia a realização de um dado objetivo e **episódio** é definido como um conjunto particular de ações (plano), executadas sob condições, que tornam possível a descrição operacional de um objetivo do domínio do problema [POT 95].

Scenario → Setting Narrative Evaluation
Setting → Background Roles
Roles → Role*
Narrative → Episode+
Episode → Goal Action Outcome

Legend: Asterisks represent zero or more repetitions, braces represent optional categories, and the vertical bar represents alternatives.

FIGURA 3.1 – Esquema de casos de uso [POT 95]

Em [POT 95], é apontado que, diferentemente das intuições dos usuários, a hierarquia dos objetivos ajuda a delimitar a cobertura dos casos de uso. Porém, um potencial número ilimitado de casos de uso, pode ser derivado, a partir dessa hierarquia. Heurísticas são utilizadas para guiarem a obtenção do conjunto de casos de uso "salientes". Um caso de uso saliente tem um propósito, não é redundante e auxilia os usuários e desenvolvedores a levantarem e entenderem algum aspecto do sistema proposto que deve ser resolvido antes de poder ser dito que o sistema atende as necessidades dos usuários.

De acordo com a abordagem, várias atividades são necessárias antes que os casos de uso esquemáticos possam ser produzidos. Os objetivos para o domínio devem ser identificados e decompostos; obstáculos devem ser identificados e analisados para decidir quais merecem futura atenção; e, um ou mais sistemas que cumprem os objetivos identificados devem ser propostos. Com respeito à decomposição de objetivos, apenas é observada a similaridade com a análise hierárquica de tarefas [POT 95] e, para a identificação de obstáculos, é sugerida a realização de um questionamento sistemático [POT 94]. Os relacionamentos entre os objetivos e atores, assim como os relacionamentos entre objetivos, obstáculos e objetivos defensivos ou de suavização são representados por tabelas.

Segundo Potts [POT 95], casos de uso esquemáticos têm a vantagem de poder esclarecer antes da implementação como o sistema proposto, ou propostas alternativas irão afetar os reais objetivos dos usuários em situações atípicas de uso; mas significantes.

3.2.2 Abordagem de Cockburn [COC 97a, COC 97b, COC 98, COC 2000]

Em [COC 97a, COC 97b, COC 98, COC 2000] é proposta, por Alistair Cockburn, uma teoria que utiliza objetivos como o elemento chave dos casos de uso. A abordagem considera que os objetivos podem resumir as funções do sistema, em termos compreensíveis e verificáveis.

O foco principal da abordagem é a escrita de casos de uso efetivos, sendo apresentas técnicas de como pensar, como redigir sentenças e em que seqüência trabalhar. Os modelos e as técnicas apresentados na abordagem têm sido aplicados e avaliados em projetos reais. Os trabalhos [COC 97a, COC 97b, COC 2000] são descritos como resultados destas experiências.

Na abordagem de Cockburn, é proposto um modelo para descrever os casos de uso que utiliza conceitos teleológicos; mas que, diferentemente, da abordagem apresentada anteriormente, não possui uma estrutura episódica. O modelo pode ser usado tanto na forma textual quanto na tabular; em ambos os casos, a descrição dos casos de uso é divida em seções [COC 98]:

- Nome do caso de uso: uma pequena frase verbal descrevendo o objetivo;
- Objetivo no contexto: uma descrição mais detalhada a respeito do objetivo, caso necessário;
- Escopo: qual sistema deve ser considerado uma "caixa-preta" sobre o ponto de vista do projeto;
- Nível: o caso de uso pode ser de algum dos três tipos: resumo, tarefa primária ou subfunção;
- Pré-condição: as condições esperadas do contexto, antes de iniciar o caso de uso;
- Condição de termino bem sucedido: as condições esperadas do contexto, após a realização completa do caso de uso;
- Condição de termino com falha: as condições esperadas do contexto, após o abandono do caso de uso;
- Ator principal: o nome ou a descrição do ator principal;
- Gatilho (*Trigger*): a ação sobre o sistema que inicia o caso de uso;
- Curso principal: seqüência de passos do curso normal, descrevendo ações, a partir da ativação do caso de uso;
- Extensões: lista de condições, cada uma se referindo ao passo do curso principal;
- Subvariações: as subvariações que, eventualmente, possam causar bifurcação no caso de uso;
- Informações adicionais: outros dados importantes para o caso de uso, como prioridade, performance, freqüência, atores secundários e casos de uso relacionados.

A abordagem de Cockburn [COC 97a, COC 97b, COC 2000] faz uso de metáforas para representar os objetivos e os casos de uso. Os objetivos em seus diversos níveis são estruturados hierarquicamente, formando um gráfico que se assemelha a um "veleiro" (do inglês "*sailing ship*"), ilustrado na figura 3.2. Já casos de uso são associados a "calças listradas" (do inglês "*striped trousers*") com pernas de sucesso e falha (figura 3.3). O cinto da calça é o objetivo que aglutina todos os cenários, cada listra um cenário, e cada linha, em um cenário, uma ação primitiva ou um objetivo de

um caso de uso subordinado. Essa analogia provê uma visão sintetizada da coleção dos caminhos de um caso de uso.

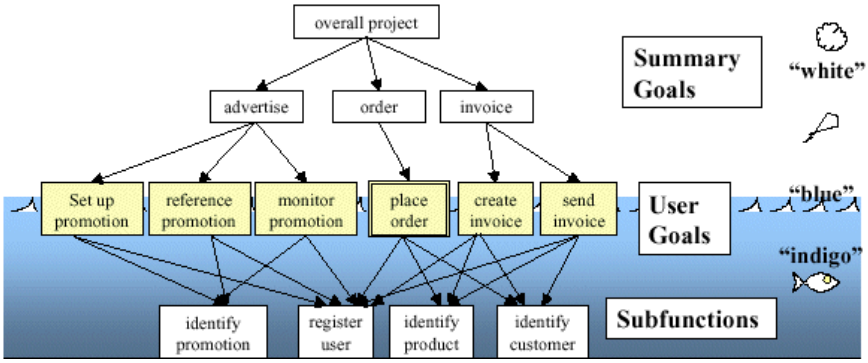

FIGURA 3.2 – Metáfora do "veleiro" (*sailing ship*) [COC 2000]

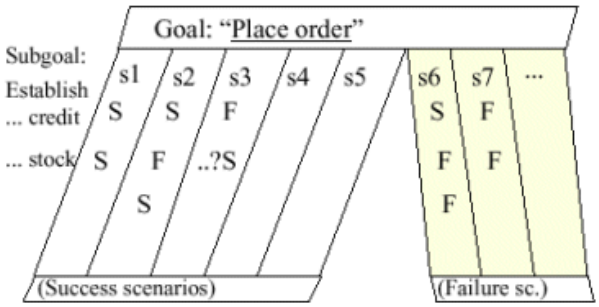

FIGURA 3.3 – Metáfora da "calça-listrada" (*striped trousers*) [COC 2000]

Em [COC 97a, COC 97b, COC 2000], a explosão do número de casos de uso é evitada, utilizando três técnicas: casos de uso subordinados, extensões e variações. Para ligar casos de uso a requisitos de interface, de performance e aos atores, é sugerido o uso de tabelas auxiliares.

3.2.3 Abordagem de Rolland [ROL 98a, ROL 98b, ROL 99]

A abordagem CREWS-*L'Ecritoire* de Colette Rolland é parte integrante do projeto CREWS (*Cooperative Requirements Engineering With Scenarios*), um longo projeto de pesquisa, iniciado em 1996, pela Comunidade Européia. A abordagem visa elicitar requisitos, explorando o relacionamento bidirecional entre a autoria de cenários textuais e a descoberta de objetivos. Para cada objetivo descoberto, é escrito um cenário que, posteriormente, é analisado para descobrir novos objetivos, o que leva à descrição de novos cenários e assim sucessivamente. Os requisitos elicitados são expressos na forma de uma hierarquia de objetivos e cenários.

A idéia central da abordagem CREWS-*L'Ecritoire* é o conceito de *Requirements Chunks* (RC) definido como um par de objetivo e cenário. Um objetivo é de natureza intencional; e um cenário, de natureza operacional; um RC é uma possível maneira de alcançar um objetivo. Os RCs são inter-relacionados por composição, alternativa e

refinamento, constituindo um sistema de conceitos, denominado *Requirement Chunck Model* (RC *Model*).

Os RCs são organizados hierarquicamente em 3 níveis de abstração: contextual, funcional e físico. O nível contextual tem o propósito de identificar os serviços que um sistema deve prover para uma organização. O foco do nível funcional é nas interações, entre o sistema e seus usuários, necessárias para alcançarem os serviços atribuídos ao sistema ao nível contextual. O nível físico foca quais ações internas o sistema precisa para executar as interações selecionadas ao nível funcional.

Uma ênfase particular é dada na exploração de cenários textuais, descritos em linguagem natural pelos *stakeholders*. São propostas guias para autoria de cenários e elicitação de objetivos.

A elicitação de objetivos é baseada em análise lingüística de declarações de objetivos. O processo guiado de autoria de cenários é dividido em dois estágios principais: a escrita dos cenários e a correção dos mesmos. Para guiar a escrita dos cenários são propostas guias de estilo e conteúdo referentes a um modelo conceitual e um modelo lingüístico de cenários. Para guiar a correção dos cenários é proposto um conjunto de regras. Cenários escritos, em prosa informal, são progressivamente transformados em textos estruturados e não ambíguos, integrados ao RC *Model*. Através de estratégias de composição e alternativa, diferentes cenários são integrados em um caso de uso.

3.2.4 Análise Crítica das Abordagens

A análise destes trabalhos permitiu a avaliação da importância de relacionar objetivos com casos de uso. A abordagem de Potts [POT 94, POT 95], em especial, clarifica a importância da consideração de obstáculos. A abordagem de Cockburn [COC 97a, COC 97b, COC 98, COC 2000] é um relato prático de como casos de uso têm sido utilizados em projetos e possui valiosas guias para a construção e descrição de casos de uso, assim como sugere o uso de uma simples, mas eficiente, lista para relacionarmos atores a objetivos. Diferentemente das duas abordagens anteriores, a CREWS-*L'Ecritoire* de Rolland [ROL 98a, ROL 98b, ROL 99] utiliza primeiramente cenários e propõe análises sintáticas e semânticas destes para a elicitação de objetivos e obstáculos e, também, para a integração dos cenários em casos de uso. As análises lingüísticas, sintáticas e semânticas, são relativamente complexas. Deste modo, precisa-se de ferramentas especializadas para utilizar a abordagem CREWS-*L'Ecritoire*, mesmo para pequenos projetos. De fato, em [ROL 98b] já é apresentada uma ferramenta para a aplicação da abordagem.

As idéias e conceitos presentes nessas três abordagens foram cruciais para a elaboração da proposta. Cada uma delas apresenta pontos de destaque; porém nenhuma delas atende a todo o conjunto de critérios que são julgados pertinentes para uma construção sistemática, que considere mutuamente as perspectivas de ES e IHC. A seguir é apresentada uma tabela comparativa que sintetiza a análise das abordagens.

TABELA 3.1 – Comparação entre abordagens baseadas em objetivos

Abordagem / **Critério**	**Potts**	**Cockburn**	**Rolland**
Investigação das singularidades	Um questionamento sistemático e a criação de ações defensivas e corretivas	Um questionamento sistemático e algumas guias.	Algumas guias
Delimitação do conjunto de casos de uso	Algumas heurísticas para determinar a saliência dos casos de uso. Uso do conceito de episódios	Uso de técnicas de subordinação, extensão e variação	Sugere que os objetivos delimitam o número de casos de uso. Uso do conceito de episódios
Uso complementar dos conceitos de cenários e casos de uso	Apenas é sugerido, sem especificar, que cenários podem ser instanciados para serem avaliados pelos usuários	Cenários não são usados para experimentação com os usuários	Cenários são usados apenas num primeiro momento para elicitar objetivos e posteriormente são integrados em casos de uso. Não são, porém, usados para experimentação com os usuários
Avaliação da usabilidade com os usuários	Não	Não	Não
Diferentes níveis de abstração	Não	Sumário, objetivos do usuário e subfunções	Contextual, funcional e físico
Nível de abstração que estabelece ligação com abordagens OO	Não	Não	Não
Notações definidas	Sim	Sim	Sim
Guias para a descrição textual dos casos de uso	Não	Sim	Sim
Relações temporais nos casos de uso	Não	Não	Não
Fluxo dos eventos	Seqüencial	Seqüencial	Seqüencial
Consideração de eventos assíncronos	Não	Sugere tratá-los em uma seção separada, usando asterisco (*) para identificá-los	Não
Representação de Objetivos	Apenas é sugerida uma hierarquia de objetivos similar a modelos de tarefas	Metáfora do veleiro	*Requirements Chunks* organizados hierarquicamente
Representação da ontologia	Não	Não	Não

4 Abordagem Proposta

4.1 Introdução

Este trabalho investiga e propõe uma abordagem que faz uso de conceitos teleológicos para gerar, estruturar e validar casos de uso e cenários. Tem como principal objetivo investigar os conceitos de cenários e casos de uso, e integrar e sistematizar a construção destes, conciliando as perspectivas de ES e IHC.

O trabalho tem como base algumas idéias da abordagem TAREFA (*Task Analysis based Requirements Enginnering Framework*) [PIM 97], concebida para sistematizar a engenharia de requisitos de sistemas interativos que já adota casos de uso e cenários como fios condutores para a integração das áreas de ES e IHC. Esta abordagem é uma evolução da abordagem TAREFA [PIM 97], de forma que os aspectos construtivos de casos de uso são aprimorados, novos aspectos são investigados e a construção de casos de uso e cenários é, afinal, sistematizada. Uma descrição detalhada da abordagem será apresentada ao longo dos próximos dois capítulos.

Propõe-se uma abordagem que, a partir de um modelo explícito de objetivos dos usuários, refinados e representados, permite definir, refinar e representar os objetivos do sistema que servirão como base para a construção de casos de uso em quatro diferentes níveis de abstração, essencial, singular, operacional e concreto, cada um relativo a um tipo de conhecimento específico.

A abordagem é um processo interativo e iterativo de construção, refinamento, experimentação, modificação e integração de casos de uso. Em resumo, propõe-se a construção de casos de uso essenciais, nível mais abstrato, pelo analista, a partir dos objetivos; ainda pelo analista, o refinamento de casos de uso essenciais, visando definir casos de uso singulares; a instanciação dos casos de uso concretos, cenários; e, quando desejável, a definição dos casos de uso operacionais, a partir dos casos de uso singulares; a experimentação e avaliação da funcionalidade e do comportamento dos cenários pelos usuários, visando investigar suas reais necessidades; a revisão dos casos de uso pelo analista, procurando adequá-los às expectativas dos usuários; e, finalmente, a integração, pelo analista, dos casos de uso obtidos neste processo recursivo.

O diagrama ilustrado na figura 4.1 representa uma visão geral do processo de construção sistemática de casos de uso e cenários. Ao final do processo, deseja-se obter um modelo de casos de uso que permita especificar requisitos funcionais, conjuntamente com requisitos de interação, de maneira compreensível e praticável e que sirva como ponto de partida para a continuidade do desenvolvimento de software.

Acredita-se que a combinação de casos de uso e cenários com uma boa compreensão das tarefas e do contexto organizacional em que elas são executadas, juntamente com a explicitação dos objetivos, previnam a possibilidade de que especificações se tornem desencontradas das reais necessidades dos usuários e devam ser a base para um processo de desenvolvimento de software interativo.

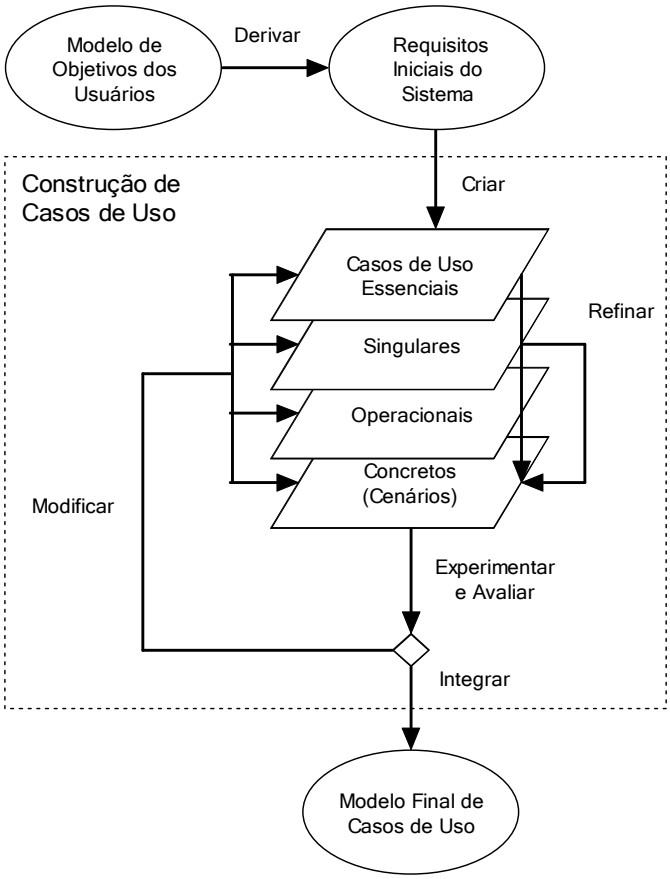

FIGURA 4.1 – Visão geral da abordagem teleológica e dos quatro níveis de abstração de casos de uso

4.2 Atividades Preliminares

A proposta, em epígrafe, está inserida no âmbito da ER; mas especificamente na ER de sistemas interativos. Casos de uso têm um papel importante na ER, contudo, é de se lembrar que algumas atividades são necessárias, antes que os casos de uso possam ser produzidos. Levando em consideração o que foi apresentado no capítulo 3 (seção 3.1), é fácil compreender que, para considerarem-se, adequadamente, aspectos de usabilidade, os objetivos dos usuários são fundamentais. Deve-se, então, ter como ponto de partida no processo de construção de casos de uso um modelo explícito de objetivos dos usuários, refinados e representados.

Não cabe aqui, neste trabalho, discutir quais são as melhores técnicas de coleta e determinação de objetivos dos usuários e de informação contextual. Sinta-se à vontade para usar o processo de sua preferência, como, por exemplo, GBRAM (*Goal-Based Requirements Analysis Method*) [ANT 96, ANT 98b], mas lembre-se que é preciso, sobretudo, identificar todos os atores que interagem com o sistema; identificar seus objetivos; decompô-los; organizá-los; e representá-los.

Sugere-se, assim, o uso do subprocesso de análise, proposto em TAREFA [PIM 97], que tem o objetivo de compreender o contexto do domínio do problema e de identificar os requisitos dos usuários. A análise é composta por análise contextual, elicitação dos requisitos dos usuários (explicitamente solicitados) e determinação dos

requisitos dos usuários, chegando a um conjunto de requisitos de usuários (explícitos, implícitos e organizacionais) e um conjunto de modelos para representar os conhecimentos obtidos durante a análise contextual [PIM 2000].

Uma contribuição original de TAREFA [PIM 97] é que a hierarquia de objetivos dos usuários é obtida e representada através de um tipo particular de modelo de tarefa. A análise de tarefa emergiu da ergonomia, como um importante apoio ao desenvolvimento de sistemas interativos, devido ao fato de ser um método empírico de compreender como as pessoas executam suas tarefas [PIM 96]. Uma análise de tarefa produz um modelo explícito de tarefas em um domínio, denominado modelo de tarefas, representando dois tipos de informação: objetivos e ações.

Em [PIM 97], é proposto um modelo, derivado diretamente de um modelo de tarefas, onde o elemento principal é a estrutura dos objetivos e subobjetivos, denominado **modelo de tarefa minimal**. Além da organização hierárquica dos objetivos, há a descrição da organização e da interdependência deles. A organização é a organização temporal na qual o usuário alcança suas tarefas (sucessão, intercalação, paralelismo, etc). A interdependência é uma sucessão imposta pelas relações entre as entradas e saídas das tarefas. O modelo de tarefa minimal é um modelo abstrato, sem considerações tecnológicas, representando as intenções do usuário, ao invés de ações, e serve como base para a construção dos casos de uso. É utilizada a notação MAD (*Méthode Analytique de Description*), neste modelo; mas qualquer modelo de tarefas com essas características pode ser usado, como, por exemplo, GOMS (*Goal Operators Methods Selection*), HTA (*Hierarquical Task Analysis*) ou CTT (*ConcurTaskTree*).

Entretanto, a determinação dos requisitos dos usuários não é suficiente para a definição dos requisitos do sistema, pois os requisitos dos usuários tendem a ser muito gerais e não provêem indicações ou detalhes sobre o comportamento (interno ou externo) do sistema. O modelo de tarefas descreve os objetivos dos usuários, sem a preocupação com as funções subjacentes do sistema. Também em TAREFA [PIM 97], são propostas duas atividades para a definição dos requisitos iniciais do sistema, chamados “iniciais”, por representarem o ponto de vista do usuário sobre os requisitos do sistema. A primeira é uma **Re-engenharia de Tarefas,** baseada na informação contextual coletada, que tem o intuito de reorganizar as tarefas que se deseja corrigir para eliminar as redundâncias e as ineficiências, antes de propor uma solução computacional; a segunda é uma **Alocação de Funções**, para determinar que tarefas serão manuais, automáticas ou interativas (maiores detalhes em [PIM 97]). Após a execução desses dois passos, os **requisitos iniciais do sistema** são determinados e pode-se pelas atividades de construção, de experimentação, de modificação e integração dos casos de uso, especificar as características que um sistema tem de possuir para satisfazer os requisitos dos usuários.

Para apoiar a construção dos casos de uso, também é interessante relacionar explicitamente os objetivos representados nos modelos de tarefas minimais com seus atores iniciadores. Para tal propósito, utiliza-se a lista ator-objetivo, proposta em [COC 2000]. A lista ator-objetivo também ajuda a priorizar e particionar o trabalho de desenvolvimento e é atualizada, conforme novos objetivos são descobertos na análise dos casos de uso. Esta lista ator-objetivo é composta pelos seguintes campos:

- Prioridade Organizacional: Prioridade do objetivo para a organização;

- Ator Iniciador: Nome do ator que inicia o caso de uso, ou seja, que tem o objetivo;
- Objetivo: Objetivo extraído do modelo de tarefa minimal;
- Prioridade: Prioridade na qual o sistema tem de suportar o objetivo;
- ID do Caso de Uso: Identificador do caso de uso que suporta o objetivo.

Opcionalmente o campo prioridade pode ser dividido em três:
- Prioridade Organizacional: Prioridade do objetivo para a organização;
- Complexidade Técnica: Complexidade ou dificuldade estimada pelo grupo de desenvolvimento para prover esta função;
- Prioridade de Desenvolvimento: Prioridade na qual o sistema tem de suportar o objetivo.

Esta divisão torna possível separar as necessidades da organização dos custos de desenvolvimento para derivar a prioridade de desenvolvimento [COC 2000].

Segundo Leite [LEI 97], o desenvolvimento de software baseado em casos de uso, apóia-se no conceito de que a utilização da linguagem do problema é benéfica na interação entre usuários e desenvolvedores. Produzir casos de uso, com a linguagem do usuário, ao invés da linguagem do desenvolvedor, contribui para o sistema resultante refletir o domínio do usuário [MCD 94]. Defende-se então, a construção de um **modelo ontológico** do vocabulário do problema usado pelo usuário, para representar e estabelecer uma terminologia unificada (ontologia). Essa unificação terminológica é especialmente importante, porque diferentes casos de uso podem ser descritos e analisados por pessoas ou grupos diferentes, e ela auxilia a melhorar a comunicação e estabelecer um entendimento comum entre os *stakeholders*. Em adição, a ontologia pode ajudar a encontrar e prevenir redundâncias; problema que, geralmente aparece, conforme aumenta o número de casos de uso. Pode-se representar o modelo ontológico através da notação LEL (*Language Extended Lexicon*) [LEI 97], centrada na idéia de que uma descrição circular dos termos da linguagem melhora a compreensão do macrosistema. O modelo ontológico já deve ser previamente gerado, ao determinarem-se os objetivos dos usuários; mas é, gradualmente, estendido e revisado, conforme os casos de uso são identificados e inspecionados.

4.3 Modelo de Casos de Uso

Considerando o já exposto até o momento, é fácil compreender que o conceito de casos de uso, apresentado nesta abordagem, está ancorado no paradigma teleológico. O entendimento de cenários e casos de uso leva em conta uma combinação de várias abordagens da literatura [BOO 99, CAR 95, COC 2000, CON 2000a, PIM 97, POT 95].

Caso de uso é um modelo narrativo de um conjunto de seqüências de interações entre atores e um sistema, agrupadas por um **objetivo,** comum do ator iniciador, incluindo suas variantes e representando potenciais ou reais situações de uso do sistema, ao ser o objetivo alcançado ou não. Um **cenário,** por sua vez, é uma instância específica e representativa de uma seqüência de interações de um caso de uso, tendo um resultado particular com respeito ao objetivo do ator iniciador. Corresponde à descrição de um comportamento do ambiente e do sistema que aparece em uma situação concreta e restrita de uso.

Um **ator** representa um papel desempenhado por agentes externos ao sistema: usuários, dispositivos de hardware, outros sistemas. É uma categoria de usuários com comportamentos e objetivos comuns, identificando um papel genérico e não uma ocorrência específica. O ator que tem o objetivo e inicia a interação é chamado de **ator iniciador**, os demais que assistem o sistema, para satisfazer o objetivo, são chamados de **atores participantes**.

As **seqüências de interações**, muitas vezes, também são referidas como cursos de interações. Alguns cursos terminam atingindo o objetivo, outros não. Um caso de uso agrupa todos os cursos de sucesso e falha. Casos de uso não ocorrem em qualquer situação, é necessário considerar o **contexto**, que demarca seu **escopo**, caracteriza seus elementos e define suas **condições**. As condições são predicados do estado do sistema e do ambiente que devem ser considerados antes - pré-condicões - ou depois - pós-condições - das seqüências de interações.

Tem-se então, que na abordagem, casos de uso devem ser vistos como planos de execução de objetivos, ao invés de meras seqüências cronológicas de eventos. Conforme já observado em [COC 97a, COC 97b], uma das dificuldades desta perspectiva é o fato de objetivos existirem em vários níveis (objetivos, subobjetivos, ações individuais) e a operacionalização deles requerer uma freqüente mudança entre os níveis.

Outra característica relevante observada por [KAI 95, PIM 97] é que um caso de uso é, ao mesmo tempo, uma especificação da estrutura, da funcionalidade e do comportamento de uma situação de uso. Como toda a especificação, tem dois aspectos: o técnico, pois age como ponto de origem para a concepção e a implementação do sistema; e o social, por servir para comunicação entre os *stakeholders*, em particular, o analista e o usuário.

Para lidar com esses aspectos, utiliza-se o conceito de níveis de abstração. Abstração é definida como um mecanismo para esconder detalhes, a fim de focar aspectos essenciais; e o refinamento é utilizado para descrever gradualmente os casos de uso em diferentes níveis de abstração.

Conforme já mencionado anteriormente, adotam-se nesta abordagem quatro níveis de abstração para descrever os casos de uso; cada um adaptado a um determinado objetivo e relativo a um tipo específico de conhecimento e de representação. São eles:

- **Essencial**: Casos de uso ao nível mais abstrato. São construídos, a partir dos objetivos de tarefas, presentes nos modelos de tarefas minimais. Devem ser realizados para a satisfação dos objetivos das atividades do usuário, sempre consideradas num contexto organizacional;
- **Singular**: Casos de uso essenciais, refinados através da determinação de singularidades - desvios do curso "normal" que incluem problemas, anormalidades, exceções, interrupções e obstáculos -. Para cada singularidade, pode haver a identificação de possíveis ações defensivas e corretivas que o sistema poderia realizar para, respectivamente, evitar ou corrigir um problema. Portanto, um caso de uso essencial pode dar origem a diversos casos de uso singulares; um para cada ocorrência de uma singularidade;
- **Operacional**: Um caso de uso operacional descreve o comportamento de um caso de uso singular, em termos de objetos e operações, a um nível similar ao

do Projeto OO. Estes casos de uso estão a um nível de abstração abaixo dos casos de uso singulares e são voltados, especificamente, à geração preliminar de arquiteturas de sistemas OO;

- **Concreto**: Casos de uso, no menor nível de abstração. Um caso de uso concreto é uma ocorrência de uma execução particular de um caso de uso singular do ponto de visto do usuário. Ele corresponde a uma descrição das características de uma situação concreta de uso - protótipos das IUs, comportamento detalhado do usuário e do sistema, condições de uso, singularidades ocorridas -, sem levar em consideração os detalhes técnicos e arquitetônicos do sistema ao nível operacional. Por definição, corresponde ao conceito de cenário utilizado em IHC.

Não é necessário fazer a descrição completa de um caso de uso de acordo com os quatro níveis. Por exemplo, não há necessidade de se fazer um refinamento, passo a passo, de um caso de uso essencial até o concreto: o caso de uso operacional é opcional e seu uso só é interessante para a concepção orientada a objeto.

Existe uma grande variedade de estilos de representação, utilizados para descrever o conteúdo narrativo de um caso de uso (maiores detalhes em [ANT 98a, CON 2000a]). Mas a notação, na qual os casos de uso são escritos, tem um profundo efeito em suas utilidades e na qualidade dos artefatos que resultam [CON 2000a]. Para cada um dos níveis de abstração, adota-se uma notação específica de acordo com o propósito do nível.

4.3.1 Casos de Uso Essenciais

O conceito de caso de uso essencial é similar ao de caso de uso essencial de Constantine [CON 2000a], diferindo basicamente no fato de que os casos de uso essenciais de Constantine [CON 2000a] são expandidos, pois descrevem além do curso "normal", as alternativas importantes ou exceções que podem surgir em relação ao curso "normal"; enquanto que os casos de uso essenciais, aqui apresentados, são básicos e de alto nível, descrevendo apenas o curso "normal" das atividades, onde o objetivo é alcançado, sem nenhuma obstrução. Considera-se benéfico o uso de casos de uso essenciais como casos de alto nível; pois, desta maneira, tornam-se úteis para compreender rapidamente o grau de complexidade e a funcionalidade do sistema proposto. Concentra-se primeiro em como o sistema deve funcionar normalmente, antes de serem investigadas as possíveis alternativas e falhas. Desta forma, evita-se uma sobrecarga de complexidade e pode-se aumentar o paralelismo e a produtividade do desenvolvimento.

Os casos de uso essenciais são expressos numa forma ideal, relativamente livre de tecnologia e de detalhes de implementação; ou seja, as decisões de projeto são postergadas e abstraídas, principalmente aquelas relacionadas com a IU [LAR 2000]. Eles são abstratos e modelam as intenções e não as ações dos usuários; as responsabilidades e não as respostas do sistema [CON 2000a]. Casos de uso não devem ser entendidos como *logs* de atividades humanas, ao contrário, devem ser abstratos e orientados a objetivos, significativos e discutíveis pelos *stakeholders*.

Um caso de uso essencial não especifica uma seqüência de interações concretas, mas uma seqüência de passos abstratos, permitindo um variável número de

implementações [BID 2001] e encorajando a inovação criativa [CON 2000a]. É um mecanismo esquemático para ajudar a pensar na maneira cujo software será utilizado pelos usuários. Neste sentido, um caso de uso essencial é uma projeção do futuro uso do sistema pelos usuários. Além disso, como são adaptados ao nível de abstração do usuário, eles podem servir como um eficiente meio de comunicação entre o usuário e analista.

Existe uma sutil diferença entre objetivos e intenções. Objetivo é um estado final desejado de um sistema, sendo corretamente descrito em termos estáticos como estado e características de objetos. Já uma intenção, em contraste, é dinâmica e representa o sentido ou o progresso, ao invés de um estado final. *"Objetivos são destinos, enquanto que as intenções representam a jornada do usuário"* [CON 2000a]. Ainda sobre este aspecto, Kavakli [KAV 96] destaca que "necessidades" são difíceis de serem averiguadas e que é mais apropriado considerar requisitos como "tendências" – aquilo que pessoas tentam fazer quando tem oportunidade [KAV 96] –. No trabalho, "tendência" é cognominada como "intenção". Uma intenção deve ser entendida como uma versão operacional de um objetivo, portanto ela pode ser identificada e testada, observando-se o comportamento das pessoas [KAV 96]. Neste sentido, casos de uso essenciais podem definir estratégias sistemáticas para se encontrar os requisitos dinâmicos do sistema.

Nos casos de uso essenciais, a motivação para o usuário é a intenção de atingir um objetivo e a motivação para o sistema é a responsabilidade de cumprir as obrigações. Responsabilidade é, então, uma expressão de "o que" precisa ser feito, sem necessariamente detalhar "como" será feito. Para este mais alto-nível de descrição de um caso de uso, uma responsabilidade é uma abstração das ações, a serem executadas pelo sistema para transformar o estado atual no estado final desejado, onde o objetivo da tarefa interativa, representada pelo caso de uso, é alcançado. As responsabilidades (funções) do sistema sustentarão as tarefas interativas para atingir os objetivos e podem ser classificadas em dois tipos: responsabilidade receptiva do sistema, onde o evento do sistema é uma recepção de uma intenção ou uma preparação para ação do usuário; e a responsabilidade expressiva do sistema, onde o evento é uma resposta do sistema a uma intenção [PIM 97].

Este tipo de construção permite que os projetos de IUs e do software sejam feitos em paralelo, ambos a partir dos casos de uso essenciais. Casos de uso essenciais foram originalmente criados para apoiar o projeto de IUs, uma vez que se é levado a deduzir, de forma correta, a concepção da interface, a partir da lógica de utilização do sistema como suporte à realização de tarefas, e não, a partir da lógica de funcionamento das funções da aplicação, como usualmente realizado. Eles também tendem a permanecerem corretos por um longo período de tempo, uma vez que excluem decisões de projeto e descrevem, apenas, a essência dos casos de uso, isto permite a identificação de padrões de casos de uso, sendo úteis no desenvolvimento de novos projetos [BID 2001].

Notação

Encontram-se na literatura diferentes formas para representar casos de uso: textual, animada, protótipos etc. A UML propõe notações gráficas, como os diagramas de casos de uso e os diagramas de seqüência para descreverem os eventos de um caso de

uso, entretanto estas notações não permitem a expressão necessária aos casos de uso para efetiva aquisição e validação de requisitos [ANT 98a]. Além disto, os diagramas de seqüência levam o desenvolvedor a considerar mensagens passadas entre objetos do software, ao invés de manter uma perspectiva externa do usuário e um foco na natureza das tarefas.

Diversos autores recomendam a representação textual em linguagem natural. Existem claras vantagens da narrativa textual. A principal razão é o fato de que narrativa textual provê a maneira de expressar o problema em uma representação facilmente compreensível, permitindo uma comunicação mais efetiva entre os *stakeholders*, no desenvolvimento da aplicação. Clientes e usuários geralmente não possuem um vasto conhecimento de notações de ES: a narrativa textual, em linguagem natural, não exige treinamento formal ou o uso de ferramentas sofisticadas, s*takeholders* tem familiaridade com esta notação [COC 97a, COC 97b, ROL 98a]. Desta forma, é possível que pessoas de outras disciplinas entendam mais claramente os requisitos do sistema.

Os casos de uso essenciais foram estruturados em seis seções: informação descritiva, contexto, narrativa, relação temporal, anexos e histórico, sendo as duas últimas opcionais. A figura 4.2 ilustra a estrutura do caso de uso essencial.

Informação Descritiva		
ID:	Nome:	
Contexto		
Escopo: Ator Iniciador: Atores Participantes: Prioridade: Freqüência: Recursos: Pré-condições: Pós-condições:		
Narrativa		
Intenções do usuário	Responsabilidades do sistema	
(eventos)	(eventos)	
Relação Temporal		
(expressão)		
Anexos		
Âncoras:		
Histórico		
Autor:	Data:	Versão:

FIGURA 4.2 – Caso de uso essencial estruturado

A primeira seção, informação descritiva, contém dados que identificam e explicam o propósito do caso de uso; ou seja, relaciona o caso de uso com o objetivo. É composta pelos elementos:

- ID: meramente um identificador único do caso de uso; útil no processo de desenvolvimento para rastreamento (*tracking*) e para uso em ferramentas;
- Nome: declaração de seu objetivo. O nome do caso de uso é uma frase que expressa o propósito do caso de uso. A primeira palavra deve ser um verbo no tempo presente na voz ativa, como sugerido em [COC 2000, ROL 98b];

A seguir, representado na figura 4.3, tem-se um exemplo de um caso de uso essencial, para o objetivo *Consultar extrato*, descrito de acordo com esta estrutura que foi definida.

<table>
<tr><td colspan="2">Informação Descritiva</td></tr>
<tr><td>ID: UC3</td><td>Nome: Consultar extrato</td></tr>
<tr><td colspan="2">Contexto</td></tr>
<tr><td colspan="2">Escopo: caixa eletrônico
Ator Iniciador: cliente
Atores Participantes: -
Prioridade: 5
Freqüência: diversas vezes ao dia
Recursos: identificação, conta, extrato, recibo
Pré-condições: caixa eletrônico pronto para ser usado
Pós-condições: caixa eletrônico pronto para ser usado; cliente com o recibo do extrato</td></tr>
<tr><td colspan="2">Narrativa</td></tr>
<tr><td>Intenções do usuário</td><td>Responsabilidades do sistema</td></tr>
<tr><td>1. Fornece identificação ao sistema</td><td></td></tr>
<tr><td></td><td>2. Aceita a identificação do cliente
3. Valida a identificação do cliente
4. Oferece transações ao cliente</td></tr>
<tr><td>5. Pede a transação extrato</td><td></td></tr>
<tr><td></td><td>6. Aceita a transação extrato
7. Pede o período do extrato ao cliente</td></tr>
<tr><td>8. Escolhe o período do extrato</td><td></td></tr>
<tr><td></td><td>9. Aceita o período do extrato
10. Valida o período do extrato
11. Emite o recibo do extrato</td></tr>
<tr><td>12. Pega o extrato</td><td></td></tr>
<tr><td></td><td>13. Registra transação
14. Oferece transações ao cliente</td></tr>
<tr><td>15. Escolhe Sair</td><td></td></tr>
<tr><td colspan="2">Relação Temporal</td></tr>
<tr><td>1 ||| (5 8) 12 15</td><td></td></tr>
</table>

FIGURA 4.3 – Exemplo de caso de uso essencial

A seção de contexto contém dados que contextualizam o caso de uso. Seus elementos são melhores explicados a seguir:

- Escopo: termo utilizado para delimitar qual é o sistema em discussão. Tipicamente, o escopo é considerado tão óbvio pelos autores dos casos de uso que eles sequer o mencionam. Todavia, em um grupo de vários autores e leitores o escopo do caso de uso pode não ser tão óbvio [COC 2000];
- Ator Iniciador: nome do papel que tem o objetivo e inicia a interação;
- Atores Participantes: demais atores que participam do caso de uso;
- Prioridade: prioridade de desenvolvimento do caso de uso;
- Freqüência: freqüência de utilização do caso de uso por unidade de tempo;
- Recursos: recursos necessários ao caso de uso. Podem ser concretos, por exemplo, o cartão bancário; ou abstratos, a conta corrente;
- Pré-condições: predicados do estado do sistema e do ambiente que devem ser considerados, antes das seqüências de interações. Não podem depender de informações obtidas, dentro do próprio caso de uso. São documentadas separadamente, pois não precisam ser verificadas, novamente, ao longo do caso de uso;

- Pós-condições: predicados do estado do sistema e do ambiente que devem ser encontrados após a execução do caso de uso.

A terceira seção, a narrativa, é o corpo narrativo do caso de uso e descreve os aspectos dinâmicos, a interação. Esta seção é estruturada através do estilo de narração numerada particionada; ou seja, a narrativa é segmentada em etapas individuais numeradas, dispostas em duas colunas: intenção do usuário e responsabilidade do sistema. Esta notação torna explícito o limite das perspectivas internas e externas e têm maior legibilidade e uma menor loquacidade do que outros estilos de narração (maiores detalhes em [CON 2000a]). É então, razoavelmente adequada para descrição dos principais aspectos dos casos de uso, por permitir uma comunicação mais fácil entre os envolvidos no desenvolvimento do software.

Surge aqui uma pergunta pertinente: Como representar as etapas dos demais atores participantes? Haverá colunas adicionais? Neste caso, onde outros atores, além do ator iniciador, participam da interação, sugere-se ser utilizado o estilo de seqüência numerada na seção narrativa. Mas certifique-se que esteja explicitando o ator, em cada etapa, e descrevendo intenções e responsabilidades.

Deve-se também, atentar para o fato de que, nem sempre, uma interação é estritamente seqüencial. Cada intenção pode ter relações temporais diferentes com as outras intenções do mesmo caso de uso. Por exemplo, no caso de uso essencial *Consultar extrato*, a intenção *Fornece a identificação ao sistema* pode ser executada de maneira integrada com as intenções *Pede a transação extrato* e *Escolhe o período do extrato*. Este aspecto de ordenação opcional ou flexível é uma situação muito comum em modelagem de tarefas; mas, raramente, é modelado explicitamente no corpo narrativo de casos de uso [CON 2000a]. Em TAREFA [PIM 97], é definido um conjunto de sinais (figura 4.4) para tratar deste aspecto, inspirado na notação UAN (*User Action Notation*). A maioria destas relações é adotada por modelos de tarefas e modelos de diálogo.

Relações temporais	**Sinais da Notação**
1.Grupo	(A)
2.Opção (Alternativa)	A | B
3.Paralelo	A | | B
4.Repetição	A^{+}, A^{n}, A^{*}
5.Interrupção	$B \rightarrow A$
6.Entrelaçamento	A ||| B
7.Simultaneidade	A, B
8 Ordem independente	A & B
9.Esperar por tempo *n*	A (t > *n* seconds) B

FIGURA 4.4 – Sinais de relações temporais e de ordenação [PIM 97]

Deste modo, tem-se, após a seção narrativa do caso de uso, a seção de relação temporal, a qual contém uma expressão, que faz uso deste conjunto de sinais e dos identificadores dos eventos, para lidar com as relações temporais como partes intrínsecas dos casos de uso.

A quinta seção, anexos, contém âncoras a quaisquer informações que precisam ser especificadas ao caso de uso, como, por exemplo, o identificador de um requisito organizacional; e, finalmente, a sexta seção contém dados históricos e de autoria.

4.3.2 Casos de Uso Singulares

Casos de uso singulares são casos de uso expandidos que modelam, além do curso "normal" de um caso de uso essencial, os potenciais desvios que podem mudar a interação do curso "normal". Estes desvios - englobando as anormalidades, as exceções, os problemas, os enganos, as interrupções e os obstáculos [POT 95] - são chamados de singularidades, donde vem a expressão "caso de uso singular". Por definição, uma singularidade é a causa de um problema [PIM 97]. Para o nível dos requisitos, um problema é uma diferença semântica entre a situação atual e uma situação desejada [PIM 97]. Então, é necessário sempre ligar uma singularidade a um problema, e um problema a um objetivo.

Os casos de uso essenciais são refinados através da determinação de potenciais singularidades, a cada situação específica e singular obtém-se um caso de uso singular. Portanto, um caso de uso essencial pode dar origem a diversos casos de uso singulares. Esta concentração em pontos específicos ajuda a refinar a compreensão dos requisitos de um sistema proposto.

A consideração de singularidades é uma questão muito importante e pode ser crucial para usabilidade de um sistema. Deve-se pensar em alternativas, para serem alcançados os objetivos, quando ocorrer alguma singularidade. Todo objetivo pode ser bloqueado de certa maneira por condições ou eventos, no sistema ou no ambiente. Levar em conta as possíveis singularidades força o desenvolvedor e, também, o usuário, a pensarem em soluções flexíveis e robustas para estas situações reais, onde o uso acontece em modos não antecipados pelo desenvolvedor, e não para situações idealizadas [POT 95]. Devem-se capturar as singularidades antes de se preocupar em como lidar com elas. Um exemplo de singularidade para o caso de uso singular (*Consultar Extrato*, ilustrado na figura 4.7) é apresentado na figura a seguir.

ID da singula-laridade	**ID do episódio onde a singularidade acontece**	**ID da ação do episódio onde a singularidade acontece**	**Problema**	**Singularidade (Causa do problema)**	**Ações (D)efensivas ou (C)orretivas**
S06	EP1	7	Senha não aceita	Senha incorreta, provida pelo cliente	(C) Permitir o cliente fornecer a senha, até 4 vezes consecutivas (D) Após 4 tentativas, o cartão é bloqueado

FIGURA 4.5 – Exemplo de singularidade para o caso de uso singular *Consultar extrato*

Para cada singularidade identificada são propostas possíveis ações defensivas e corretivas que o sistema poderia realizar para, respectivamente, evitar ou corrigir seus efeitos. Os casos de uso singulares permitem avaliar a conseqüência de singularidades e possíveis mecanismos defensivos e corretivos; e, também, possibilitam estabelecer proposições de novas ações para alcançar um determinado objetivo e descobrir novos objetivos para uma nova situação futura.

Naturalmente, um problema é o potencial número ilimitado de casos de uso, necessários a representar todas as diferentes possibilidades de progresso de uma interação. A solução, como proposto em [POT 95], é definir um subconjunto de casos de uso em termos de singularidades e combinações de singularidades consideradas, relevantes, – "salientes" [POT 95] – por eles terem características que ajudam

exemplificar aspectos cruciais do comportamento do sistema proposto e não são redundantes. Em TAREFA [PIM 97], as singularidades relevantes são divididas em duas categorias não-ortogonais: singularidades críticas, obstruções ao curso normal do caso de uso essencial; e singularidades representativas, obstruções que aparecem mais freqüentemente. Além disso, uma combinação de singularidades também é considerada singularidade. A relevância das singularidades individuais, por conseguinte, é determinada independente do domínio da aplicação, e a relevância da combinação é dependente [POT 95].

Provavelmente, não sejam conhecidas todas as condições de um caso de uso, desde o principio; a análise de singularidades ajuda a identificar as pré e pós-condições, que levarão o usuário a seguir um determinado curso de interação. Devemos distinguir as condições que separam o curso normal dos outros possíveis cursos.

Notação

Para serem representados, adequadamente, os casos de uso neste nível de abstração, definiu-se como pré-requisito que a estrutura adotada permitisse a associação de conceitos teleológicos: atores, objetivos, singularidades etc.

Um conceito adicional é fundamental à descrição dos casos de uso essenciais, o conceito de **episódios** (ver seção 3.2.1). Como toda narrativa, casos de uso podem ser descritos de maneira episódica. Assim, a seção narrativa de um caso de uso singular consiste em uma seqüência de episódios, cada um correspondente a alguma intenção ou singularidade que bloqueia a realização dessa intenção.

O conceito de episódio é recursivo. Cada episódio pode ser considerado como um pequeno caso de uso [LEI 97]; mas, no papel de caso de uso, informações adicionais são necessárias. Este conceito auxilia a lidar com grandes conjuntos de casos de uso, por permitir o reuso de partes dos casos de uso. Um mesmo episódio pode aparecer em vários casos de uso, evitando assim a duplicação do trabalho de especificação de casos de uso e oferecendo um modelo de casos de uso mais compacto. Diversos autores já utilizam o conceito de episódios para estruturar casos de uso [REG 96, POT 95, LEI 97, ROL 98a].

O elemento básico do modelo de caso de uso é a **ação**, podendo ser atômica ou composta. As ações atômicas são divididas em duas categorias: ações de comunicação (*o cliente insere seu cartão no caixa eletrônico*); e ações internas (*o caixa eletrônico verifica que o cartão é valido*) [ROL 98a]. Uma ação composta é formada por uma organização, temporal ou de dependência lógica, de ações atômicas. As ações são descritas pelo par de termos o agente, seguido pela ação. O agente pode ser tanto o sistema quanto um ator. Recomenda-se numerar as ações, do mesmo modo que se numeram os eventos nos casos de uso essenciais.

Adota-se para a representação dos casos de uso singulares uma notação que segue um “esquema gramatical”. Segundo Potts [POT 95], casos de uso são um tipo de história, então é possível utilizar um esquema de caso de uso que é semelhante a um esquema de história. Um esquema gramatical não é uma gramática geradora, no senso conhecido da Teoria das Linguagens, porque permite deixar indeterminada a estrutura de certos elementos - tipicamente por uma descrição informal [PIM 97]. Esta

representação é perfeitamente adaptada para abordagens baseadas em objetivos, como evidenciado pela explícita associação de uma intenção a cada episódio no caso de uso [ANT 98a]. Representações de casos de uso que permitem os analistas e *stakeholders* considerarem singularidades são fontes especialmente ricas para identificarem novos requisitos. Casos de uso esquemáticos orientados a objetivos parecem apoiar a identificação e refinamento de requisitos mais que outras representações [ANT 98a].

O esquema gramatical, notação dos casos de uso singulares, é apresentado na figura 4.6. O modelo episódico de casos de uso provê uma estrutura para a especificação dos casos de uso. A especificação em si é escrita em linguagem natural estruturada.

<Caso de uso>:=	<Informação Descritiva>
	<Contexto>
	<Narrativa>
	<Anexos>
	<Histórico>
<Informação Descritiva>:=	<ID>
	<Nome>
<Nome >:=	<Objetivo> |
	(<Objetivo> <Nome do episódio>)
<Contexto>:=	<Escopo>
	<Atores>
	<Prioridade>
	<Freqüência>
	<Recursos>
	<Predicados>
<Atores>:=	<Ator Iniciador>
	<Ator Participante>*
<Recursos>:=	<Recurso>*
<Predicados>:=	<Pré-condição>+
	<Pós-condição>+
<Narrativa>:=	<Episódio>+
<Episódio>:=	<ID>
	<Nome do episódio>
	<Predicados>
	<Plano>
<Nome do episódio>:=	<Intenção> |
	(<Intenção > <Singularidade>)
<Plano>:=	<Ação>*
<Ação>:=	<ID>
	<Agente>
	<Nome da ação>
<Agente>:=	<Sistema> |
	<Ator>
<Anexos>:=	<Ancora>*
<Histórico>:=	<Data>
	<Autor>
	<Versão>

Legenda: A* = A aparece 0 ou mais vezes; A+ = A aparece 1 ou mais vezes; A | B = A ou B

FIGURA 4.6 – Esquema gramatical para casos de uso singulares

A figura 4.7 ilustra um exemplo de caso de uso singular descrito de acordo com o esquema gramatical. Este caso de uso singular é o refinamento do caso de uso essencial *Consultar Extrato* representado na figura 4.3.

Informação Descritiva
ID: UC3
Nome: Consultar extrato
Contexto
Escopo: caixa eletrônico
Ator Iniciador: Cliente
Ator Participante: -
Prioridade: 5
Freqüência: diversas vezes ao dia
Recursos: cartão, senha, conta, extrato, período do extrato, recibo
Pré-condição: caixa eletrônico pronto para usar; cliente com cartão;
Pós-condição: caixa eletrônico pronto para usar; cliente com o cartão; cliente com o recibo do extrato
Narrativa
Episódio
ID: EP1
Nome: Fornece identificação ao sistema
Pré-condição: caixa eletrônico pronto para usar; cliente com o cartão
Pós-condição: caixa eletrônico pronto para usar ou em uso; cliente com o cartão; cartão e senha do cliente válidos
Plano:

Nº	Agente	Ação
1	Cliente	Passa seu cartão no leitor do caixa eletrônico
2	Sistema	Aceita o cartão do cliente
3	Sistema	Valida o cartão do cliente
4	Sistema	Pergunta a senha do cliente
5	Cliente	Fornece sua senha ao sistema
6	Sistema	Aceita a senha do cliente
7	Sistema	Valida a senha do cliente

Episódio
ID: EP2
Nome: Pede extrato
Pré-condição: caixa eletrônico pronto para usar
Pós-condição: caixa eletrônico pronto para usar; pedido de transação extrato aceito
Plano:

Nº	Agente	Ação
1	Sistema	Oferece transações ao cliente
2	Cliente	Pede a transação extrato
3	Sistema	Aceita a transação extrato

Episódio
ID: EP3
Nome: Escolhe o período do extrato
Pré-condição: caixa eletrônico pronto para usar; pedido de transação extrato aceito
Pós-condição: caixa eletrônico pronto para usar; pedido de transação extrato aceito; período do extrato válido
Plano:

Nº	Agente	Ação
1	Sistema	Pede o período do extrato ao cliente
2	Cliente	Escolhe o período do extrato
3	Sistema	Aceita período do extrato
4	Sistema	Valida o período do extrato

Episódio
ID: EP4
Nome: Pega o recibo
Pré-condição: caixa eletrônico pronto para usar; cartão e senha do cliente válidos; período do extrato válido
Pós-condição: caixa eletrônico pronto para usar; cartão e senha do cliente válidos; período do extrato válido; cliente com o recibo
Plano:

Nº	Agente	Ação
1	Sistema	Emite o recibo da transação extrato para o cliente
2	Cliente	Pega o recibo
3	Sistema	Registra a transação extrato

Episódio
ID: EP5
Nome: Sair
Pré-condição: caixa eletrônico pronto para usar
Pós-condição: caixa eletrônico pronto para usar
Plano:

Nº	Agente	Ação
1	Sistema	Oferece transações ao cliente
2	Cliente	Pede a transação sair
3	Sistema	Aceita a transação sair
4	Sistema	Registra a transação sair

FIGURA 4.7 – Exemplo de caso de uso singular

4.3.3 Casos de Uso Operacionais

A um nível de abstração abaixo dos casos de uso singulares, os casos de uso operacionais estão descritos ao nível mais técnico e são voltados, especificamente, para a geração preliminar de Arquiteturas de sistemas OO. Logicamente, que outras classes de objetos podem surgir durante o projeto e a implementação do software. Este nível baseia-se principalmente na distribuição dos comportamentos descritos nos casos de uso singulares em objetos que participam dos casos de uso. O foco que, anteriormente, era usuário e sistema é alterado para objetos do sistema que realizam os objetivos. Os casos de uso operacionais podem ser omitidos, quando não for desejada uma concepção OO.

Neste nível, procura-se compreender como os componentes do software, em especial os objetos, interagem uns com os outros e com o ambiente externo. O sistema é considerado como um conjunto de objetos que participam dos casos de uso. Deseja-se identificar objetos e suas responsabilidades para cobrir o comportamento do sistema a um nível particular de abstração.

A identificação de classes e objetos é um ponto de muita importância em abordagens OO. É reconhecido o fato de que a modelagem conceitual, que identifica conceitos do domínio do problema, é pertinente tanto para abordagens de desenvolvimento OO quanto para o desenvolvimento de interfaces. Porém modelos conceituais só modelam componentes estáticos, representam conceitos do mundo real não componentes de software. A identificação dos objetos não deve se limitar a objetos tangíveis do mundo real; precisa-se identificar um conjunto preliminar de objetos que permitam a realização do comportamento dos casos de uso. Autores já enfatizam que se devem modelar os aspectos comportamentais [RUB 92, WIR 2001]. A vantagem da equivalência com o mundo real continua sendo aplicada; mas não se limita aos objetos tangíveis.

Não é o mérito deste trabalho discutir qual a melhor prática de determinação de objetos e de atribuição de responsabilidades. Pode-se, por exemplo, construir um modelo conceitual, usando estratégias de encontrar conceitos baseando-se em listas de categorias de conceitos e estratégias de encontrar conceitos com a identificação de substantivos; e, posteriormente, aplicar os padrões GRASP (*General Responsibility Assignment Software Patterns*) para a atribuição de responsabilidades [LAR 2000]. O que de fato se destaca é que os aspectos comportamentais têm uma grande importância.

Alguns autores alegam que casos de uso têm uma abordagem de decomposição funcional e que, por isto, não são uma boa fonte para serem identificados objetos e classes. Discorda-se desta visão. De fato, casos de uso não são realmente orientados a objetos; todavia já se encontra destacado em [JAC 94c] que a semântica do modelo de caso de uso (visão externa) interfere fortemente na semântica do modelo de objetos (visão interna) e que estes modelos estão intimamente ligados: *"Há um claro mapeamento de um modelo de casos de uso em um modelo de objetos. Neste mapeamento, cada caso de uso é executado por um número de objetos participantes onde cada objeto tem um papel específico no caso de uso. Isto significa que todo o comportamento definido para um caso de uso é distribuído para estes diferentes objetos, os quais cooperam para fazer o caso de uso acontecer."* [JAC 94c]. Através de casos de uso pode-se obter uma perspectiva mais dinâmica dos objetos, identificam-se os objetos no contexto de uso.

Porém, para ser modelada esta colaboração entre os objetos, precisa-se saber que objetos estão envolvidos; que operações estão envolvidas e em quais objetos; e qual a seqüência de operações. Na prática, existe uma lacuna entre as descrições narrativas dos casos de uso e os modelos de objetos.

Focando responsabilidades, tanto nos modelos de casos de uso quanto nos modelos de objetos, pode-se estabelecer um elo de ligação entre ambos. Em pontos significantes do processo de desenvolvimento, a habilidade de verificar as responsabilidades dos objetos com as expressas nos casos de uso representa uma forma valiosa de verificar se o projeto continua correspondendo aos requisitos [BID 2001]. O uso do conceito de responsabilidade na ER e no projeto representa um importante fator na possibilidade de rastreamento (*traceability*).

A noção de responsabilidades, nos casos de uso, é compatível com a noção de responsabilidade de objetos; ambos descrevem comportamento, sem descrever implementação [BID 2001]. Ao atribuir responsabilidades iniciais aos objetos, devem-se considerar as responsabilidades requeridas, nos casos de uso. Estas responsabilidades são atribuídas baseadas no conhecimento do domínio e em heurísticas.

Para a identificação de objetos e atribuição de responsabilidades, segue-se a abordagem OOSE [JAC 92], onde uma das fases da análise é construir o modelo de análise, onde a arquitetura é baseada em três tipos de objetos: objetos entidade, objetos de interface e objetos de controle. Esse modelo especifica a composição de objetos e a associação entre eles. **Objetos entidade** representam a informação persistente, mantida pelo sistema; **objetos de interface** representam as interações entre atores e o sistema; **objetos de controle** representam as tarefas executas pelos usuários e suportadas pelo sistema.

Modelar um sistema com estes três tipos de objetos tem várias vantagens. Primeiro, proporciona ao desenvolvedor heurísticas simples para distinguir conceitos semelhantes; mas diferentes. Segundo, a abordagem dos três tipos de objeto resulta em objetos menores e mais especializados. Terceiro, a abordagem dos três tipos de objeto conduz a modelos mais aptos a mudanças [BRU 2000]. A diferenciação dos objetos de interface permite isolar funcionalidade, representada por objetos entidade e de controle, da interação humano-computador, representada pelos objetos de interface, garantindo a possibilidade de serem realizadas modificações mais facilmente, sem efeitos em cascata. Para distinguir entre diferentes tipos de objetos, a UML provê o mecanismo de estereótipos.

Notação

Uma vez definida a arquitetura, devem ser modeladas as interações entre os objetos. Para representar os casos uso, neste nível de abstração, adotam-se os diagramas de seqüência, propostos na UML [BOO 99].

Um importante papel, que casos de uso executam em desenvolvimentos orientados a objeto, é a elucidação das responsabilidades de interação. Para este propósito, os diagramas de seqüência são uma clara representação. Um diagrama de seqüência é um modo formal de assegurar que são identificados todos objetos e operações, inseridos num determinado caso de uso e mostra como objetos participantes

interagem para cumprir o caso de uso [JAC 94c]. Na análise de requisitos, os diagramas de seqüência são usados para ajudarem identificar novos objetos participantes e comportamento ausente.

Construindo diagramas de seqüência, não só se modela a ordem da interação entre os objetos; mas, também, distribui-se o comportamento do caso de uso. Em outras palavras, designa-se a cada objeto responsabilidades na forma de um conjunto de operações. Estas podem ser compartilhadas por qualquer caso de uso no qual um determinado objeto participa. Se um objeto participar em mais de um caso de uso sua definição deverá ser idêntica; ou seja, o comportamento de uma operação deverá ser o mesmo nos diferentes diagramas de seqüência em que ela aparecer [BRU 2000].

O diagrama de seqüência é de fato um elemento pivô à transição de análise em projeto OO. Ele foca comportamento de alto-nível; assim, aspectos de implementação não deverão ser tratados neste momento.

Nos diagramas de seqüência, cada coluna representa um objeto que participa da interação. A ordem das colunas não é significante; entretanto devem ser posicionadas de modo a oferecer maior clareza. Pode-se, também, ter colunas que representam os atores. O eixo vertical representa o tempo, de cima para baixo. Objetos interagem entre si, enviando mensagens, representadas por setas. A recepção de uma mensagem por um objeto aciona a execução de uma operação que pode enviar mensagens a outros objetos. As mensagens podem ter argumentos (parâmetros). Rótulos nas setas representam nomes e argumentos das mensagens. A figura 4.8 ilustra um exemplo de um diagrama de seqüência.

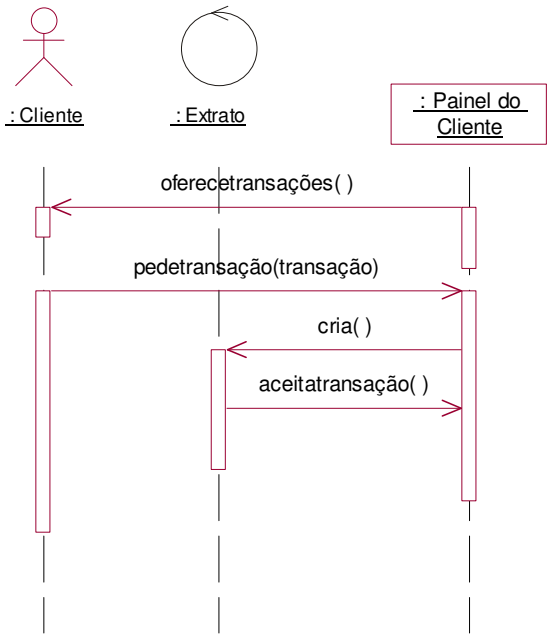

FIGURA 4.8 – Exemplo de diagrama de seqüência

4.3.4 Casos de Uso Concretos (Cenários)

Casos de uso concretos são casos de uso ao menor nível de abstração. Um caso de uso concreto é uma instância específica e representativa de um caso de uso singular, tendo um resultado particular com respeito ao objetivo do ator iniciador. Ele descreve um comportamento do ambiente e do sistema que aparece em uma situação concreta e restrita de uso, presente ou futura, sob o ponto de vista de um usuário. Por definição, corresponde ao conceito de cenário utilizado em IHC (ver seção 2.1).

Os cenários não expõem somente a funcionalidade do sistema; mas declarações específicas sobre como o usuário irá acessar a funcionalidade e o que ele experimentará, fazendo isto [CAR 2000]. Eles expressam uma visão parcial do desenvolvimento e, dessa forma, expõe o desenvolvimento à crítica.

Durante estágios de pré-implementação, analistas e *stakeholders* podem não compreender muitas de suas decisões propostas; as decisões e comportamentos podem ser clarificados através de exemplos concretos. Através de cenários, cada decisão pode ser avaliada e documentada em termos de suas conseqüências específicas.

Cenários facilitam o melhor entendimento do futuro sistema, forçando os envolvidos no desenvolvimento a prestarem atenção em particularidades do uso real [ANT 98a]. Podem elucidar como o sistema proposto irá suportar práticas concretas de trabalho. Ajudando assim a decidir se as atuais soluções operacionais dos objetivos satisfazem as reais necessidades dos usuários.

Sendo concretos, também forçam que os princípios mais abstratos, suposições, teorias e métodos de cada disciplina ou abordagem sejam concretizados [ABO 94]. Funcionam como uma base comum para comunicação entre grupos multidisciplinares.

Cenários podem endereçar conflitos entre reflexão e ação, entre situações típicas e críticas e entre situações desejáveis e indesejáveis [BØD 2000], e relacionam a maneira prática e teórica de pensar [ABO 94]. É um meio de incorporar o conhecimento e a experiência dos usuários no processo de desenvolvimento.

Notação

Cenários ajudam desenvolvedores e usuários a administrarem a complexidade conceitual de desenvolvimento de software, clarificando as conseqüências comportamentais concretas dos requisitos ou propostas de desenvolvimento. Outra estratégia de desenvolvimento relacionada, que faz isto, é a prototipagem. Porém prototipagem usualmente é direcionada a testar sistemas resultantes do desenvolvimento e não a pensar no desenvolvimento, conforme é conduzido [CAR 2000]. Cenários podem ser usados independentemente da prototipagem, e protótipos podem ser usados na ausência de cenários, contudo o uso conjunto tem se mostrado benéfico [ANT 98a, CAR 2000].

Então, para o nível concreto é adotada a combinação de uma descrição textual em linguagem natural e um protótipo de "baixo nível", adequado para exemplificar os aspectos concretos dos casos de uso para o usuário. Este nível não se preocupa com detalhes de plataforma tecnológica, nem com detalhes de interface; mas, unicamente,

com que ela faz. A forma de narrativa textual, contendo informação sobre situações e usuários ilustrativos, parece ser mais memorável e inteligível à maioria dos leitores que a forma estruturada [POT 95]. A seguir, a figura 4.9 ilustra um exemplo de um caso de uso concreto.

Maria deseja utilizar o caixa eletrônico do banco X para retirar RS 50,00, em dinheiro, de sua conta corrente; Ela passa seu cartão no leitor do caixa eletrônico, pronto para ser usado. O sistema valida o cartão e, em seguida, requisita a senha a Maria. Pelo teclado, Maria digita sua senha. O sistema valida a senha digitada e mostra na tela o menu de opções de transações. Maria escolhe a opção saque, tocando no botão com esta denominação que está na tela; em seguida escolhe R$ 50,00 no menu de possíveis quantidades de dinheiro. O caixa automático verifica a quantia escolhida, a existência de fundos, aprova a transação e libera a quantia correta e um recibo do saque. Maria pega o dinheiro e o recibo nos locais especificados, o sistema debita a quantia da conta corrente, exibe uma mensagem, pedindo para o cliente certificar-se de que está de posse de seu cartão magnético; e, por fim, mostra novamente o menu de opções de transações.

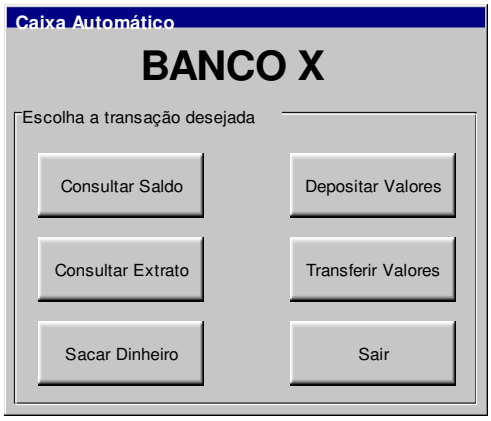

FIGURA 4.9 – Exemplo de caso de uso concreto

4.4 Construção de Casos de Uso

Acredita-se que as abordagens *top-down* e *bottom-up* de criação de casos de uso e cenários devam ser complementares e que possam ser incorporadas como atividades iterativas, num mesmo método. Nesta abordagem, o processo de construção de casos de uso é interativo e iterativo; entretanto, a princípio, segue a abordagem *top-down*, onde casos de uso são construídos e refinados, sucessivamente, pelo analista, respeitando suas propriedades estruturais. O uso de notações definidas para cada um dos níveis de abstração, juntamente com a utilização de heurísticas e diretrizes, conduz a construção dos casos de uso. Este processo incremental resume-se em

- construir os casos de usos essenciais, a partir da hierarquia de objetivos representada pelos modelos de tarefas minimais e da definição das responsabilidades essenciais do sistema. Cada caso de uso é construído, associando atores, objetivos, intenções e responsabilidades;
- construir os casos de uso singulares, refinando os casos de uso essenciais e definindo as singularidades, associadas às intenções e responsabilidades;
- construir os casos de uso operacionais definindo os objetos e as operações, descritos nos casos de uso singulares, caso a concepção seja orientada a objeto;
- construir (instanciar), a partir dos casos de uso singulares, os casos de uso concretos que serão experimentados, recursivamente, com os usuários.

Deve-se estar atento ao fato de a construção de casos de uso implicar em escrever, refinar, experimentar, modificar e integrar casos de uso.

A seguir, apresentam-se algumas guias para conduzirem o processo de construção de casos de uso:

- Procure definir os atores e seus objetivos, antes de escrever os casos de uso;
- Explicite o contexto do caso de uso, especialmente as pré e pós-condições;
- Descreva os casos de uso como planos de execução de objetivos, não como meras seqüências cronológicas de eventos;
- Descreva as intenções dos usuários e as responsabilidades do sistema, não os detalhes de IU;
- Para cada caso de uso, escreva e revise o curso normal onde o objetivo é alcançado, antes de escrever os cursos alternativos ou de exceção;
- Liste as singularidades e avalie suas relevâncias, antes de escrever os cursos de como o sistema deve tratá-las;
- Utilize as questões "por quê?" para encontrar o nível de abstração acima e "como?" para encontrar o nível de abstração abaixo;
- Revise a ontologia, constantemente, ao gerar ou alterar os casos de uso.

Apoiado nas abordagens de [COC 2000, ROL 98, ACH 98] definem-se algumas diretrizes que têm a função de prover recomendações sobre o estilo esperado das narrativas dos casos de uso aos níveis essencial e singular. As diretrizes estão representadas na figura a seguir.

1. Use o mesmo nível de abstração, ao longo do caso de uso;
2. A narrativa deve representar um único curso de ações. Cursos alternativos, variações e exceções devem ser descritos, separadamente;
3. Descreva o curso das ações esperadas, não as ações inesperadas, impossíveis ou irrelevantes. Evite o uso de negações, advérbios, e verbos modais na descrição de uma ação;
4. Mantenha a IU fora do caso de uso; tenha certeza de estar descrevendo as motivações (intenções e responsabilidades) e não os movimentos de manipulação de interface;
5. Escreva o caso de uso como uma lista de ações discretas. Cada ação deve iniciar em uma nova linha;
6. Escreva cada ação como uma frase onde um objetivo é alcançado, usando a seguinte estrutura: sujeito + verbo + objeto direto + frase preposicional. Exemplo: O sistema deduz a quantia da conta corrente;
7. Use verbos na voz ativa e no tempo presente, ao descrever as ações, a fim de evitar a omissão ou esquecimento de algum ator;
8. Tenha certeza de que o ator que executa a ação e o objeto da ação está visível em cada passo;
9. Caso utilize a notação de duas colunas ao nível essencial, pode-se omitir o nome do ator iniciador e sistema, uma vez que a própria notação já os torna explícitos;
10. Caso participem dois atores do mesmo tipo, num caso de uso, discrimine-os. Por exemplo, use os nomes "cliente C1" e "cliente C2";
11. Evite utilizar referências anafóricas: ele, ela, dele, sua. Use os nomes do modelo ontológico;
12. Evite o uso de sinônimos e homônimos. Use termos consistentes com a terminologia unificada;
13. Não utilize "verifica se" substitua por "valida" ou "verifica que". Não utilize checagem de uma condição, elas não dão uma seqüência na narrativa, substitua por validação. Por exemplo, "O sistema verifica se a senha está correta", substitua por "O sistema verifica que a senha está correta";
14. Não utilize "se... condição... então". Sempre que encontrar esta construção olhe para a sentença anterior, provavelmente encontrará uma checagem. Corrija ambas as sentenças. Por exemplo, "O sistema verifica se a senha está correta", "Se a senha está correta, o sistema apresenta opções ao usuário", substitua por "O sistema verifica que a senha está correta", "O sistema apresenta opções ao usuário".

FIGURA 4.10 – Diretrizes para a descrição textual de casos de uso

Estas diretrizes devem estar disponíveis ao autor do caso de uso (analista), no momento da descrição. Acredita-se que a qualidade dos casos de uso, produzidos, possa aumentar, quando as diretrizes são aplicadas corretamente.

4.5 Visão Macro do Processo de Construção de Casos de Uso

Uma representação tabular concisa das atividades e dos resultados esperados de cada uma delas, no processo de construção de cenários e casos de uso proposto, é apresentada na tabela 4.1.

TABELA 4.1 – Visão macro do processo de construção de casos de uso

Atividades	**Resultados**
1. Encontre todos os atores que interagem com o sistema.	• Lista de atores
2. Encontre e refine os objetivos dos atores para com o sistema (ver análise hierárquica de tarefas, seção 4.2).	• Modelos de tarefas minimais • Modelo ontológico
3. Encontre os requisitos iniciais do sistema: 3.1 Re-engenharia de tarefas; 3.2 Alocação de Funções.	• Modelos de tarefas minimais refinados • Lista de requisitos iniciais do sistema • Lista ator-objetivo • Diagramas de casos de uso (opcionalmente)
4. Construa os casos de uso essenciais: 4.1 Selecione um objetivo de alto nível; 4.2 Contextualize o caso de uso; 4.2 Utilize as intenções do modelo de tarefa minimal num caso de uso; 4.3 Defina as responsabilidades iniciais do sistema; 4.4 Verifique e expresse as relações temporais entre as intenções; 4.5 Associe os requisitos não funcionais que tem influência sobre este caso de uso.	• Casos de uso essenciais
5. Construa os casos de uso singulares: 5.1 Selecione um caso de uso essencial para refinar; 5.2 Capture as pré e pós-condições e escreva o caso de uso singular para o curso normal; 5.3 Investigue e avalie as singularidades e as ações defensivas e corretivas; 5.4 Construa os casos de uso singulares às singularidades consideradas relevantes.	• Lista de singularidades / ações defensivas e corretivas • Tabela de sucesso e falha de episódios • Casos de uso singulares
6. Construa os cenários: 6.1 Selecione um caso de uso singular para instanciar; 6.2 Crie um protótipo de interface para apoiar o teste.	• Cenários • Protótipos
7. Teste os cenários com os atores.	• Lista de modificações
8. Modifique os casos de uso: 8.1 Propague a modificação até o menor nível de abstração; 8.2 Retorne ao passo 7, até que se chegue a um consenso.	• Casos de uso modificados
9. Integre os casos de uso.	• Casos de uso validados e integrados
10. Construa os casos de uso operacionais: 10.1 Defina os objetos e atribua as responsabilidades.	• Casos de uso operacionais

Observa-se que essas atividades não são necessariamente seqüenciais; na prática, algumas delas podem ser realizadas em paralelo ou entrelaçadas. São apresentados de

modo linear, por questão de clareza. Além disto, o processo de construção de casos de uso não segue um modelo em cascata e, sim, um modelo iterativo.

5 Aplicação Passo a Passo ao Caixa Eletrônico

Para ilustrar a aplicação de nossa proposta, nas próximas seções, são utilizados trechos de um exemplo: o caixa eletrônico, o qual o acrônimo, comumente utilizado, é ATM, do inglês *Automatic Teller Machine*.

5.1 Descrição do Exemplo

Atualmente, os caixas eletrônicos oferecem uma vasta gama de serviços. Entretanto, no exemplo apresentado não são tratados todos os possíveis serviços de um caixa eletrônico. Salienta-se que o exemplo é de um sistema hipotético, por o objetivo aqui ser apresentar a proposta e não conceber completamente uma máquina de caixa eletrônico. Portanto, selecionam-se algumas das funções mais comuns dos caixas eletrônicos, considerando suficientes para ilustrarem a aplicação da abordagem.

Tem-se, assim, como definição do problema: Definir o software interativo de um para permitir que o auto-atendimento nos serviços bancários mais freqüentes como saque, consulta de saldo, consulta de extrato, transferência e depósito, seja realizado pelos clientes do banco, em questão.

Têm-se várias razões para a escolha do caixa eletrônico, como exemplo:

- Este tipo de exemplo é onipresente na literatura, adotado como *benchmark* de vários métodos [KAI 95, JAC 94a, ROL 98a, COC 2000]. Assim, seu uso facilita a comparação de nossa abordagem com outros trabalhos;
- Não requer um conhecimento especializado do domínio para compreendê-lo. De fato, assume-se que os leitores tenham alguma experiência no uso diário dos caixas eletrônicos. Infelizmente, por isto é difícil imaginar que se esteja trabalhando no domínio de caixas eletrônicos, como se tais máquinas não existissem. Pela mesma razão, porém, este exemplo é útil para o propósito deste trabalho, por esta familiaridade ajudar a explicar as noções e conceitos discutidos;
- A especificação dos requisitos do caixa eletrônico ilustra problemas típicos de requisitos de sistemas. Ele tem algumas características muito dependentes da organização, às vezes escondidas por suposições implícitas e consideradas evidentes; há uma ampla oportunidade para diferentes interpretações e diferentes pontos de vista sobre os requisitos e vários aspectos, freqüentemente, não considerados adequadamente pelos caixas eletrônicos existentes;

5.2 Construção de Casos de Uso

Será seguido para este exemplo o conjunto de passos descrito na tabela 4.1 (ver visão macro do processo de construção de casos de uso, seção 4.5). As seções subseqüentes detalham a realização das atividades presentes nesta tabela.

5.2.1 Atividades Preliminares

Sabe-se que, para levar a cabo a construção dos casos de uso e cenários, algumas atividades devem ser previamente realizadas. O passo inicial é encontrar os atores que interagem com o sistema. O levantamento dos atores iniciadores foca a atenção do desenvolvedor nas pessoas que irão utilizar os sistema e auxilia na elicitação de um maior número de objetivos no começo do desenvolvimento.

Após terem sido identificados os atores que interagem com o sistema de caixa eletrônico; determinados seus objetivos; e coletadas as informações contextuais, aplica-se uma re-engenharia de tarefas para as tarefas que se deseja corrigir. Às vezes, o solucionar um problema é resultado de uma modificação organizacional ou de atitude frente a uma tarefa, ao invés de uma inovação tecnológica. Por exemplo, nas informações contextuais coletadas não se tinha nenhuma restrição que impunha a seqüência mostrada na figura 5.1, que é provavelmente a conseqüência do uso freqüente desta seqüência.

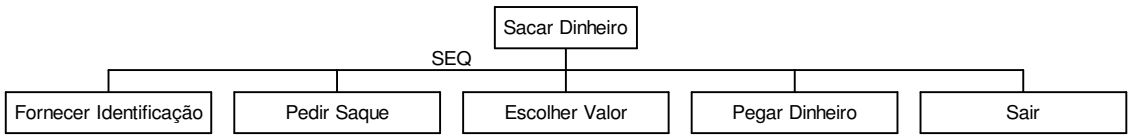

FIGURA 5.1 – Modelo de tarefa minimal *Sacar Dinheiro*, antes da re-engenharia de tarefas

Sabe-se que o cliente deve *Fornecer Identificação*, antes de *Pegar Dinheiro*; mas não, necessariamente, antes de realizar a seqüência que agrupa *Pedir Saque* e *Escolher Valor*. Raciocínio similar é aplicado aos modelos de tarefas minimais *Consultar Saldo* e *Consultar Extrato*. Os modelos de tarefas minimais para os objetivos *Sacar Dinheiro*, *Consultar Saldo* e *Consultar Extrato*, após a re-engenharia de tarefas, estão ilustrados respectivamente nas figuras 5.2, 5.3 e 5.4. *PAR* indica que as subtarefas podem ser executadas em paralelo, e *SEQ* indica que o ordenamento deve ser seqüencial. Utilize o modelo de tarefas de sua preferência, como por exemplo, ConcurTaskTree [PAT 2001], contanto que a representação utilizada ilustre a decomposição de tarefas em subtarefas, hierarquicamente, como estruturas de árvores e, também, represente o ordenamento temporal da tarefa.

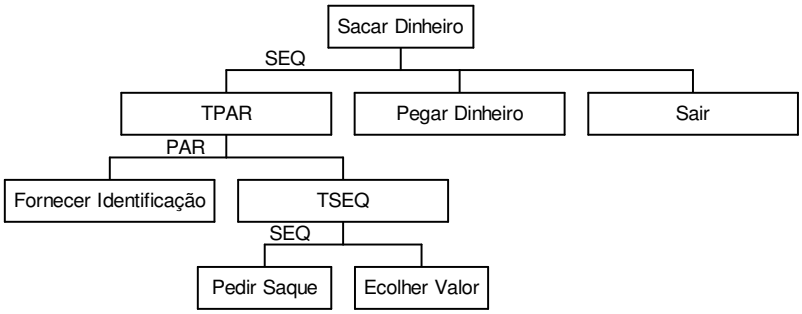

FIGURA 5.2 – Modelo de tarefa minimal *Sacar Dinheiro*

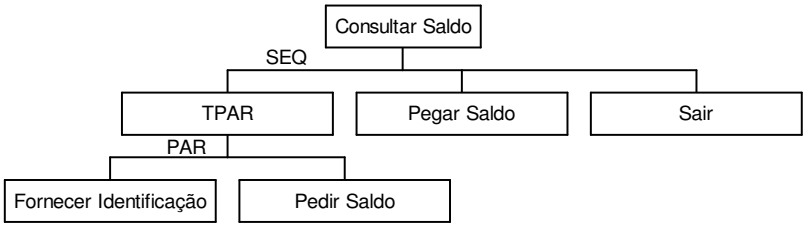

FIGURA 5.3 – Modelo de tarefa minimal *Consultar Saldo*

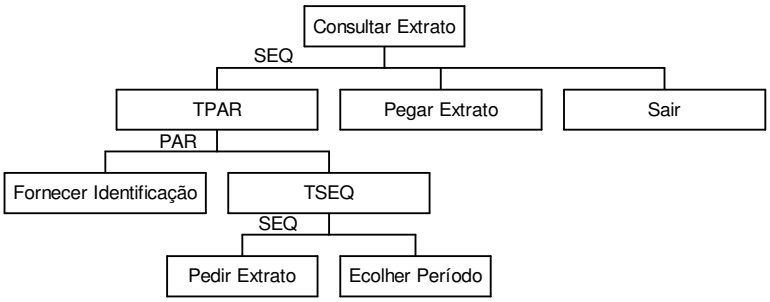

FIGURA 5.4 – Modelo de tarefa minimal *Consultar Extrato*

Após a re-engenharia das tarefas, realiza-se a alocação de funções. Ela determina quais atividades são afetadas pela introdução do sistema e qual será o papel do sistema, como suporte a cada atividade. Também é importante para ajudar a avaliar custos e benefícios das alternativas do sistema para os usuários e a organização. Nela, o desenvolvedor enumera as diferentes alternativas de apoio do computador para cada atividade e os *stakeholders* podem avaliar qual é a mais adequada para sua tarefa e a mais praticável com os recursos colocados à disposição. Na tabela 5.1 pode-se observar um exemplo de alocação de função para a emissão de recibos das transações com o caixa eletrônico, de acordo com a perspectiva do banco.

TABELA 5.1 – Exemplo de alocação de função para emissão de recibos das transações com o caixa eletrônico

Alternativa	**Função de usuário**	**Função do Sistema**	**Custo**	**Benefício**
1. Emissão do recibo para todas transações, sem interferência do usuário	Nenhuma intervenção do usuário	Imprimir os recibos para todas transações	Preço de papel	O tempo de uso é minimizado, favorecendo o número de transações na mesma unidade de tempo
2. Emissão do recibo de acordo com escolha do usuário	Decidir se quer ou não o recibo e responder a pergunta dada	Perguntar para o usuário se quer ou não o recibo de suas transações; Imprimir os recibos desejados, de acordo com a resposta	O uso é prolongado por causa do diálogo mais sofisticado	Uso reduzido de papel

Tendo optado pela primeira alternativa, tem-se então que, ao final de qualquer transação bancária, no caixa eletrônico, o cliente deverá receber automaticamente um recibo com informações sobre a transação e o saldo de sua conta.

Posteriormente, a estas duas atividades, definem-se os requisitos iniciais do sistema; obtém-se uma lista informal de requisitos iniciais do sistema (tabela 5.2), derivada diretamente dos requisitos dos usuários.

TABELA 5.2 – Lista informal dos requisitos iniciais do sistema para o caixa eletrônico

Identificador do Requisito	**Requisitos Iniciais do Sistema**
RS1	O sistema deve exigir o uso do cartão bancário, em todas transações
RS2	Cada cartão bancário está vinculado a uma conta de um cliente
RS3	Todo cartão bancário tem uma senha
RS4	O número de toques e o caminho de interação devem ser minimizados
RS5	O sistema deve registrar as transações realizadas
RS6	O sistema deve solicitar a confirmação das transações
RS7	O sistema deve permitir a definição de período flexível na consulta do extrato
RS8	O sistema deve emitir um recibo para cada transação
...	...

Em seguida, cria-se a lista ator-objetivo para o caixa eletrônico, para apoiar a construção dos casos de uso essenciais, representada na tabela 5.3. É estabelecida uma escala de prioridade que varia de 1 a 5. Caso for conveniente, classifique-os como primários, secundários ou opcionais [LAR 2000].

TABELA 5.3 – Lista ator-objetivo para o caixa eletrônico

Ator iniciador	**Objetivo**	**Prioridade organizacional**	**Complexidade técnica**	**Prioridade**	**ID do caso de uso**
Cliente	Sacar dinheiro	Alta	Complexo	5	UC1
Cliente	Consultar saldo	Alta	Simples	5	UC2
Cliente	Consultar extrato	Alta	Médio	4	UC3
Operador	Reabastecer o dinheiro	Média	Simples	5	UC4
...	...			...	...

Algumas pessoas podem preferir utilizar a lista de ator-objetivo para ter uma visão global dos casos de uso que estão sendo desenvolvidos; enquanto outras podem optar pelos diagramas de casos de uso. Um diagrama de casos de uso, representando os atores iniciadores e os casos de uso, pode servir para o mesmo propósito; porém necessita-se de notas adicionais para explicitar as prioridades.

Lembra-se também, que o modelo ontológico é inicialmente criado, conforme se realiza a análise de tarefas. Alguns exemplos na notação LEL são representados na figura 5.5. A LEL é baseada na simples idéia: *"entender a linguagem do problema sem se preocupar em entender o problema"* [LEI 97]. Nela, cada termo é um símbolo, e cada símbolo é uma entrada expressa em termos de noção e resposta comportamental. A noção deve tentar esclarecer o significado do símbolo e seus relacionamentos fundamentais com os outros símbolos. A resposta comportamental deve especificar a conotação do símbolo no macrosistema. Cada símbolo pertence a uma categoria e deve ser representado por um ou mais sinônimos [CYS 2001].

Símbolo: Cliente/Correntista
Noção:
- Alguém que possua uma *conta* em uma *agência* do *banco*.
- Tem um *cartão* e conhece sua *senha*.

Resposta Comportamental
- Cliente faz um *saque*.
- Cliente faz um *depósito*.
- Cliente faz uma *transferência.*
- Cliente *consulta o saldo*.
- Cliente *consulta o extrato*.

Símbolo: Consulta o Saldo/ Consultar Saldo
Noção:
- Ação requisitada pelo *cliente*.
- Consiste em verificar a quantia disponível na *conta*.

Resposta Comportamental
- O sistema imprime o *recibo* do *saldo* da *conta*.

FIGURA 5.5 – Exemplos de entradas na notação LEL para o caixa eletrônico

Então, é chegado o momento de construir os casos de uso, iniciando pelo nível mais abstrato: os casos de uso essenciais.

5.2.2 Construção de Casos de Uso Essenciais

Um usuário tem o objetivo de causar um certo efeito (F) no ambiente, sob certas condições, tem-se assim F como parte dos requisitos. Se um artefato ou dispositivo pode criar o efeito desejado, então é possível atribuir o efeito como uma função (responsabilidade) do artefato. Cabe ao desenvolvedor criar um sistema que possua esta responsabilidade, a fim de auxiliar um usuário a alcançar seu objetivo [CHA 96]. Em sistemas interativos, os objetivos são alcançados por certas interações entre o usuário e o sistema, isto é, por relacionamentos entre intenções do usuário e responsabilidades do sistema, modelados através de casos de uso.

Construir os casos de uso essenciais implica em definir os dois componentes da interação usuário-sistema: as intenções do usuário, derivadas diretamente da estrutura hierárquica de objetivos do modelo de tarefa minimal e as responsabilidades correspondentes do sistema.

A determinação das responsabilidades do sistema é uma atividade criativa de domínio específico. Podem ser concebidos vários sistemas que cumprem os objetivos dos usuários, eles diferirão no modo com que os objetivos são operacionalizados. Devem ser realizadas propostas alternativas do sistema [POT 95]. Casos de uso permitem definir com maior precisão como o usuário deseja usar as tarefas interativas [PIM 97]; casos de uso ajudam a avaliar, refinar e escolher a proposta [POT 95]. Na realidade, as responsabilidades do sistema devem ser refinadas, durante todo o processo de desenvolvimento até a definição detalhada do comportamento do sistema [PIM 97].

A organização dos eventos, no caso de uso essencial, segue a organização lógica e temporal do modelo de tarefa minimal. O uso de seqüência de eventos, como narração, é encorajado baseado na abordagem de Pimenta [PIM 97]: *"Mesmo entrelaçando tarefas, para uma dada instância, o usuário só manipula uma tarefa interativa por vez, independente da possibilidade de paralelismo do computador"*. Um caso de uso é

resultado da escolha de uma seqüência. Em [PIM 97], são propostas as heurísticas para escolha das seqüências: seqüência habitual do usuário; seqüência prévia (formal) da organização; e novas seqüências propostas para teste do usuário.

A figura 5.6 ilustra o caso de uso essencial que modela a seqüência habitual do usuário para realizar o objetivo *Sacar dinheiro*. O nome do caso de uso é objetivo principal do modelo de tarefa minimal (*Sacar dinheiro*) que, junto com a identificação (*UC1*) e nome o ator iniciador (*Cliente*), provêem da explicita associação entre atores e objetivos, representados aqui, através da lista ator-objetivo.

Informação Descritiva	
ID: UC1	**Nome:** Sacar dinheiro
Contexto	
Escopo: caixa eletrônico **Ator Iniciador:** cliente **Atores Participantes:** - **Prioridade:** 5 **Freqüência:** diversas vezes ao dia **Recursos:** identificação, conta, valor do saque, dinheiro, recibo **Pré-condições:** caixa eletrônico, pronto para ser usado **Pós-condições:** caixa eletrônico pronto para ser usado; cliente com o valor do saque em dinheiro; cliente com o recibo do saque	
Narrativa	
Intenções do usuário	**Responsabilidades do sistema**
1. Fornece a identificação ao sistema	
	2. Aceita a identificação do cliente
	3. Valida a identificação do cliente
	4. Oferece transações ao cliente
5. Pede a transação saque	
	6. Aceita a transação saque
	7. Pede o valor do saque ao cliente
8. Escolhe o valor do saque	
	9. Aceita o valor do saque
	10. Valida o valor do saque
	11. Altera o saldo da conta do cliente
	12. Emite o recibo da transação saque para o cliente
	13. Libera o valor do saque em dinheiro para o cliente
14. Pega o dinheiro e o recibo	
	15. Registra a transação
	16. Oferece transações ao cliente
17. Escolhe Sair	

FIGURA 5.6 – Caso de uso essencial *Sacar dinheiro*

Antes de escrever a narrativa do caso de uso, procura-se contextualizá-lo. Além do ator iniciador, deve-se definir os outros elementos contextuais que delimitam o caso de uso, propostos na estrutura dos casos de uso essenciais (figura 4.2). O escopo deste caso de uso é o *caixa eletrônico*, uma vez que se está discutindo apenas o sistema caixa eletrônico; a rede de computadores e os outros sistemas estão fora do escopo do desenvolvimento. Explicitando o escopo, permite-se que os leitores dos casos de uso vejam rapidamente o que está dentro dos limites do sistema, o que, muitas vezes, não é tão claro apenas pelo nome do caso de uso ou do ator iniciador.

No caso de uso, em questão, não participam outros atores; desta forma, o elemento atores participantes permanece vazio. A prioridade de desenvolvimento do

caso de uso é oriunda de uma anterior discussão com os *stakeholders* ao criar-se a lista ator-objetivo. O motivo pelo qual foi construído primeiramente este caso de uso é que sua prioridade é a mais alta, dentro da escala que se definiu, prioridade 5.

A freqüência de utilização do caso de uso é especialmente útil para definir a IU do sistema. Se um caso de uso possuir uma alta taxa de utilização, dever-se-á tornar o seu acesso e sua interação a mais direta possível.

Os recursos restringem o caso de uso e sinalizam importantes aspectos dos requisitos não funcionais. No exemplo em questão, está-se definindo o caso de uso *Sacar dinheiro* onde é necessário ter uma identificação, uma conta, um valor, um recibo e dinheiro.

Através dos recursos, também se pode elicitar objetivos complementares, necessários para obter uma descrição da total funcionalidade do sistema, aplicando o princípio consumo/produção [ROL 98b]. De acordo com este princípio, qualquer recurso, produzido em um caso de uso deve ser consumido no mesmo ou em outro caso de uso e vice-versa. Por exemplo, argumentando sobre a produção e o consumo do recurso *dinheiro*, identifica-se um evento onde ele é consumido: a intenção *pegar dinheiro.* Mas onde ele é produzido? Por quem? Não se identifica no caso de uso um evento responsável pela produção deste recurso, isto indica que se deve ter outro caso de uso que descreva o objetivo *abastecer o caixa eletrônico com dinheiro*.

Outra utilidade dos recursos é auxiliar a identificação de objetos e atributos ao nível operacional.

Deve-se, também, especificar às pré e pós-condições dos casos de uso. Descreve-se um caso de uso, de modo que ele transforme o estado atual no estado final desejado. O caso de uso não pode iniciar se os estados das pré-condições não são verdadeiros. As pós-condições descrevem os estados resultantes da execução do caso de uso. Para o caso de uso *Sacar dinheiro*, define-se a pré-condição *caixa eletrônico pronto para ser usado*, o caso de uso só iniciará se esta condição é válida e, sendo uma pré-condição, ela não é averiguada ao longo do caso de uso. A pós-condição mais direta que se determina é o estado final que o ator iniciador deseja, ao executar o caso de uso e atingir seu objetivo; ou seja, *cliente com o valor do saque em dinheiro*. Para serem determinadas as demais pós-condições, também se baseia na propriedade de que as pré-condições devam estar inclusas nas pós-condições para garantirem um funcionamento autocontido [ROL 98b]. Neste caso, a pré-condição *caixa eletrônico pronto para ser usado* deve fazer parte das pós-condições. Quando não se consegue incluir uma pré-condição nas pós-condições de um caso de uso, significa que devamos ter um caso de uso singular para tratar dessa exceção.

O uso de pré e pós-condições apóia significativamente um desenvolvimento centrado no uso, e essa prática expressa a ordem de determinadas tarefas relacionadas. De fato, casos de uso com pré e pós-condições fornecem meios lógicos e diretos de modelar o *workflow*, aspecto da estrutura da tarefa freqüentemente negligenciado na modelagem de casos do uso [CON 2000a].

Ao se escrever a seção narrativa do caso de uso, cada subobjetivo do modelo de tarefas minimal é considerado como uma intenção do usuário. A cada intenção, as

responsabilidades receptivas e expressivas são definidas. Por exemplo, para a intenção *Fornece identificação ao sistema (1)* definimos a responsabilidade receptiva *Aceita a identificação do cliente (2)* e a responsabilidade expressiva *Valida a identificação do cliente (3).*

Deve-se lembrar que um dos requisitos iniciais do sistema é a emissão de recibo para cada transação do caixa eletrônico. Como o objetivo do usuário é sacar dinheiro, acredita-se que a emissão do recibo deva preceder a liberação do dinheiro, pois o ator, atingindo seu objetivo final, tenderia a esquecer de pegar o recibo, com maior freqüência. Casos de uso facilitam argumentações desta natureza.

De forma análoga se obtêm os outros casos de uso essenciais, como, por exemplo, os casos de uso essenciais *Consultar Saldo* (figura 5.7) e *Consultar Extrato* (apresentado na seção 4.3.1, figura 4.3).

Informação Descritiva	
ID: UC2	**Nome:** Consultar saldo
Contexto	
Escopo: caixa eletrônico **Ator Iniciador:** cliente **Atores Participantes:** - **Prioridade:** 5 **Freqüência:** diversas vezes ao dia **Recursos:** identificação, conta, saldo, recibo **Pré-condições:** caixa eletrônico pronto para ser usado **Pós-condições:** caixa eletrônico pronto para ser usado; cliente com o recibo do saldo	
Narrativa	
Intenções do usuário	**Responsabilidades do sistema**
1. Fornece a identificação ao sistema	
	2. Aceita a identificação do cliente 3. Valida a identificação do cliente 4. Oferece transações ao cliente
5. Pede a transação saldo	
	6. Aceita a transação saldo 7. Emite o recibo do saldo para o cliente
8. Pega o saldo	
	9. Registra a transação 9. Oferece transações ao cliente
10. Escolhe Sair	

FIGURA 5.7 – Caso de uso essencial *Consultar Saldo*

Como se está especificando um novo caixa eletrônico, note que a intenção *Fornece a identificação ao sistema* não está presa a nenhuma tecnologia específica, não se especifica que o cliente insere seu cartão e fornece sua senha, embora um dos requisitos organizacionais seja que o uso do cartão do cliente é indispensável, para que ele realize as transações com o caixa eletrônico. Pode-se aqui discutir com os *stakeholders* a possibilidade da adoção de novas tecnologias para a identificação dos clientes, como, por exemplo, leitura de impressão digital. O caso de uso essencial é independente de tecnologia e permite um variável número de implementações.

Sabe-se que uma interação nem sempre é estritamente seqüencial. Então, faz-se uso dos identificadores dos eventos e do conjunto de sinais temporais definido no capítulo 4 (ver figura 4.4 na seção 4.3.1) para expressar as relações temporais entre as

intenções nos próprios casos de uso. Por exemplo, no caso de uso essencial *Sacar dinheiro* ter-se-ia a expressão 1 ||| (5 8) 14 17, isto é, conforme a organização lógica e temporal do modelo de tarefa minimal *Sacar Dinheiro* (figura 5.2) sabe-se que a intenção *Fornece a identificação ao sistema (1)* deve ser executada antes da intenção *Pega o dinheiro e o recibo (14)*; mas pode ser executada de maneira integrada com as intenções *Pede a transação saque (5)* e *Escolhe o valor do saque (8).*

Esta prática tem um efeito positivo sobre a usabilidade, por encorajar o desenhista de IU a modelar, como seqüências completamente ordenadas, somente as interações onde a ordem é realmente fixa ou determinada. Desta forma, o termo "seqüência", utilizado na definição do conceito de caso de uso, torna-se impreciso, pois não indica uma sucessão estritamente ordenada. As expressões temporais também auxiliam na identificação e avaliação de pré e pós-condições dos episódios ao nível singular.

Existe ainda outro tipo de eventos não tratado aqui até este momento, os eventos assíncronos. Como eles não se aplicam a um único ponto específico da narrativa de um caso de uso, são descritos, separadamente. Sugere-se utilizar um tipo de identificação diferente da dos outros eventos para evitar confusões. Como, por exemplo, use letras, ao invés de números. Exemplificando, caso existisse o requisito de que o cliente pudesse cancelar a realização da tarefa *Sacar dinheiro* a qualquer momento que lhe fosse requisitada alguma informação, ter-se-ia o caso de uso ilustrado na figura 5.8.

Informação Descritiva	
ID: UC1	**Nome:** Sacar dinheiro
Contexto	
Escopo: caixa eletrônico **Ator Iniciador:** cliente **Atores Participantes:** - **Prioridade:** 5 **Freqüência:** diversas vezes ao dia **Recursos:** identificação, conta, valor do saque, dinheiro, recibo **Pré-condições:** caixa eletrônico pronto para ser usado **Pós-condições:** caixa eletrônico pronto para ser usado; cliente com o valor do saque em dinheiro; cliente com o recibo do saque	
Narrativa	
Intenções do usuário	**Responsabilidades do sistema**
1. Fornece a identificação ao sistema	
	2. Aceita a identificação do cliente 3. Valida a identificação do cliente 4. Oferece transações ao cliente
5. Pede a transação saque	
	6. Aceita a transação saque 7. Pede o valor do saque ao cliente
8. Escolhe o valor do saque	
	9. Aceita o valor do saque 10. Valida o valor do saque 11. Altera o saldo da conta do cliente 12. Emite o recibo da transação saque para o cliente 13. Libera o valor do saque em dinheiro para o cliente
14. Pega o dinheiro e o recibo	
	15. Registra a transação 16. Oferece transações ao cliente
17. Escolhe Sair	
a. Cancela a transação saque	
	b. Aceita o cancelamento da transação saque c. Desfaz a transação saque
Relação temporal	
a → (1 ||| (5 8)) 14 17	

FIGURA 5.8 – Caso de uso essencial *Sacar Dinheiro*, com relação temporal e eventos assíncronos

A expressão de relação temporal também é atualizada para considerar este evento. Dessa forma, tem-se que, enquanto o cliente não *Pega o dinheiro e o recibo (14)*, ele pode *Cancelar a transação saque (a).*

5.2.3 Construção de Casos de Uso Singulares

A construção dos casos de uso singulares consiste em algumas atividades:

- Refinamento dos casos de uso essenciais, em casos de uso singulares;
- Identificação de singularidades;
- Elaboração de subobjetivos ou ações para evitar a ocorrência de singularidades ou corrigir seus efeitos;
- Finalmente, a construção de um caso de uso singular para cada singularidade considerada relevante.

A atividade de refinamento do caso de uso essencial, em caso de uso singular, é alcançada por Decomposição das intenções dos atores presentes, nos casos de uso essenciais, aqui consideradas como episódios, em ações mais elementares. O uso da questão "como?" auxilia nesta decomposição; Decomposição da informação manipulada nos passos dos casos de uso essenciais; e Refinamento de pré e pós-condições para o caso de uso e seus episódios.

Selecionou-se o caso de uso essencial *Sacar Dinheiro* para refinar. A figura 5.9 ilustra o caso de uso singular *Sacar Dinheiro* escrito através do esquema gramatical. Os elementos chave do esquema gramatical estão em negrito.

A intenção *Fornece a identificação ao sistema* deu origem ao episódio *EP1* de mesmo nome. Decompôs-se o recurso *identificação*, manipulado nesta intenção, em *cartão* e *senha*, conforme os requisitos iniciais do sistema. Em seguida, refinaram-se ações para lidar com esses novos recursos. Observa-se, também, que, após a ultima requisição de informação do cliente (episódio *EP3*; conjunto de ações *1,2,3,4* e *5*), adicionaram-se ações (ações *6,7* e *8*) para solicitar a confirmação da transação, conforme o requisito inicial do sistema *RS6*. Analogamente, escreveram-se os demais episódios.

Baseados na expressão temporal do caso de uso ao nível essencial, foram refinadas as pré e pós-condições. Observa-se que a pré-condição do episódio *Pede a transação saque (EP2)* está inserida no conjunto de pré-condições do episódio *Fornece a identificação ao sistema (EP1)*, isto implica que, sendo as pré-condições do episódio *EP1* verdadeiras, qualquer um dos dois episódios pode ser executado.

Uma vez refinado o caso de uso singular que modela o curso "normal", iniciou-se a investigação das singularidades para o caso de uso. Devem-se capturar as singularidades, antes de se preocupar em como lidar com elas.

Em geral, um modo intuitivo de serem identificadas as singularidades é analisar o caso de uso "normal" e realizar um questionamento sistemático [ANT 96]. Utilizaram-se questões do tipo:

- "What if…?" [COC 2000];
- "What can go wrong with this action?" [POT 94].

Informação Descritiva
ID: UC1
Nome: Sacar dinheiro
Contexto
Escopo: caixa eletrônico
Ator Iniciador: Cliente
Ator Participante: -
Prioridade: 5
Freqüência: diversas vezes ao dia
Recursos: cartão, senha, conta, valor do saque, dinheiro, recibo
Pré-condição: caixa eletrônico pronto para usar; cliente com cartão;
Pós-condição: caixa eletrônico pronto para usar; cliente com o cartão; cliente com o dinheiro; cliente com o recibo; valor do saque debitado da conta.
Narrativa
Episódio
ID: EP1
Nome: Fornece identificação ao sistema
Pré-condição: caixa eletrônico pronto para usar; cliente com o cartão
Pós-condição: caixa eletrônico pronto para usar ou em uso; cliente com o cartão; cartão e senha do cliente válidos
Plano:

Nº	Agente	Ação
1	Cliente	Passa seu cartão no leitor do caixa eletrônico
2	Sistema	Aceita o cartão do cliente
3	Sistema	Valida o cartão do cliente
4	Sistema	Pergunta a senha do cliente
5	Cliente	Fornece sua senha ao sistema
6	Sistema	Aceita a senha do cliente
7	Sistema	Valida a senha do cliente

Episódio
ID: EP2
Nome: Pede Saque
Pré-condição: caixa eletrônico pronto para usar
Pós-condição: caixa eletrônico pronto para usar; pedido de transação saque aceito
Plano:

Nº	Agente	Ação
1	Sistema	Oferece transações ao cliente
2	Cliente	Pede a transação saque
3	Sistema	Aceita a transação saque

Episódio
ID: EP3
Nome: Escolhe o valor do saque
Pré-condição: caixa eletrônico pronto para usar; pedido de transação saque aceito
Pós-condição: caixa eletrônico pronto para usar; pedido de transação saque aceito; valor de saque válido
Plano:

Nº	Agente	Ação
1	Sistema	Pede o valor do saque ao cliente
2	Cliente	Escolhe o valor do saque
3	Sistema	Aceita valor de saque
4	Sistema	Verifica que tem o valor do saque disponível no caixa eletrônico
5	Sistema	Valida o valor do saque com o saldo da conta do cliente
6	Sistema	Pede a confirmação da transação saque para o cliente
7	Cliente	Entra com a confirmação da transação saque
8	Sistema	Aceita a confirmação da transação saque

Episódio
ID: EP4
Nome: Pega o dinheiro e o recibo
Pré-condição: caixa eletrônico pronto para usar; cartão e senha do cliente válidos; valor de saque válido
Pós-condição: caixa eletrônico pronto para usar; cartão e senha do cliente válidos; valor de saque válido; cliente com o dinheiro e o recibo
Plano:

Nº	Agente	Ação
1	Sistema	Emite o recibo da transação saque para o cliente
2	Sistema	Libera o valor do saque em dinheiro para o cliente
3	Cliente	Pega o recibo
4	Cliente	Pega o dinheiro
5	Sistema	Altera o saldo da conta do cliente
6	Sistema	Registra a transação saque

Episódio
ID: EP5
Nome: Sair
Pré-condição: caixa eletrônico pronto para usar
Pós-condição: caixa eletrônico pronto para usar
Plano:

Nº	Agente	Ação
1	Sistema	Oferece transações ao cliente
2	Cliente	Pede a transação sair
3	Sistema	Aceita a transação sair
4	Sistema	Registra a transação sair

Anexos
Ancora: RS1; RS2; RS3; RS5; RS6; RS8

FIGURA 5.9 – Caso de uso singular *Sacar Dinheiro*

A inspeção das pré e pós-condições também auxilia na identificação de singularidades, assim como o uso de algumas guias derivadas da abordagem [COC 2000]:

- Considere cursos alternativos para cursos de sucesso;
- Comportamentos incorretos do ator iniciador; por exemplo, senha inválida;
- Inatividade do ator iniciador; por exemplo, tempo limite de entrada da senha esgotado;
- Resposta inadequada ou falta de resposta de um ator participante ou do sistema; por exemplo, tempo limite de espera por resposta esgotado;
- Cada ocorrência de um evento do tipo "o sistema valida" implica que existe a necessidade de tratar a falha da validação; por exemplo, valor de saque incorreto.
- Falha interna de componentes que fazem parte do sistema; por exemplo, impressora sem papel;

Desta forma, para cada caso de uso singular "normal", obtém-se uma lista de singularidades e, posteriormente, propõe-se ações defensivas e corretivas. Representa-se os relacionamentos entre objetivos/episódios/problemas/singularidades/ações corretivas e defensivas, através de tabelas. Como já mencionado anteriormente, cada singularidade deve ser associada a um problema; e um problema deve ser associado a um objetivo.

A tabela 5.4 apresenta um resumo das singularidades encontradas para o caso de uso singular *Sacar Dinheiro (UC1)*, ilustrado na figura 5.9.

Capturadas as singularidades, inicia-se a delimitação das singularidades relevantes. Propõe-se o uso de algumas heurísticas, adaptas de [POT 95], para guiar a obtenção do conjunto de casos de uso singulares relevantes:

- Cada objetivo da hierarquia de objetivos deve ser realizado com sucesso por um episódio em, pelo menos, um caso de uso. Esta heurística garante que todos objetivos da hierarquia serão considerados;
- O conjunto total dos casos de usos tem de conter, pelo menos, uma instância de cada categoria de singularidade - crítica ou representativa -; mas cada singularidade é associada somente a um caso de uso, sem repetições;
- Respeitar as dependências lógicas e temporais da hierarquia dos objetivos. Não há necessidade de considerar os objetivos fora da hierarquia.

Sendo assim, através destas heurísticas e baseados na tabela de singularidades e no caso de uso singular que representa o curso "normal", definem-se os demais casos de uso singulares relevantes. Um caso de uso singular é definido para cada singularidade, e para cada combinação de singularidades, consideradas relevantes, descrevendo se o episódio é afetado ou não pela conseqüência da singularidade. Há três tipos de conseqüências:

- O episódio é alcançado com sucesso (S - sucesso);
- O episódio não é realizado, porque a singularidade ocorreu no episódio anterior (F – falha);
- O episódio é alterado para considerar a singularidade (ID da singularidade).

TABELA 5.4 – Tabela de singularidades para o caso de uso *Sacar dinheiro*

ID da singula-laridade	**ID do episódio onde a sin-gularidade acontece**	**ID da ação do episódio onde a sin-gularidade acontece**	**Problema**	**Singularidade (Causa do problema)**	**Ações (D)efensivas ou (C)orretivas**
S01	EP1	2	Caixa eletrônico não está pronto para ser usado	Caixa eletrônico desligado	-
S02	EP1	2	Cartão não aceito	Leitor quebrado ou cartão inelegível	-
S03	EP1	3	Cartão inválido	Cartão inválido, provido pelo cliente	(C) Notificar o cliente
S04	EP1	3	Cartão inválido	Cartão bloqueado	(C) Notificar o cliente
S05	EP1	6	Senha não aceita	Cliente não entra com a senha a tempo (*timeout*)	(C) Notificar o cliente e retornar ao estado inicial, à pré-condição do episódio
S06	EP1	7	Senha não valida	Senha provida pelo cliente, incorreta	(C) Permitir o cliente fornecer a senha, até 4 vezes, consecutivas (D) Após 4 tentativas, o cartão fica bloqueado
S07	EP2	3	Pedido de transação incorreto	Pedir a transação incorretamente ou transação inválida	(D) Fazer o cliente escolher a transação de uma lista de transações (C) Permitir escolher a transação novamente
S08	EP2	3	Pedido de transação, não realizado	Cliente não pede a transação a tempo (*timeout*)	(C) Notificar o cliente e retornar ao estado inicial, à pré-condição do episódio
S09	EP3	3	Valor de saque incorreto	Cliente entrou com valores incorretos	(C) Notificar o cliente e permitir escolher o valor, novamente
S10	EP3	3	Valor de saque não escolhido	Cliente não entra com o valor de saque a tempo (*timeout*)	(C) Notificar o cliente e retornar ao estado inicial, à pré-condição do episódio
S11	EP3	4	Dinheiro insuficiente	O valor do saque escolhido excede a quantia disponível no caixa eletrônico	(C) Informar o cliente da quantia disponível e permitir que escolha esta quantia
S12	EP3	5	Saldo Insuficiente	O valor do saque escolhido excede o saldo da conta do cliente	(D) Exibir o saldo da conta do cliente (C) Notificar o cliente e permitir escolher o valor ,novamente
S13	EP3	8	Transação não confirmada	Cliente não confirma a transação a tempo (*timeout*)	(C) Notificar o cliente e retornar ao estado inicial, à pré-condição do episódio
S14	EP4	1	Recibo não emitido	Papel de recibo esgotado ou atolado	(C) Apenas notificar o cliente, pois o recibo não é indispensável
S15	EP4	1	Recibo não pegado	Cliente não pega o recibo	(D) Emitir o recibo antes de dispensar o dinheiro, a fim de minimizar a ocorrência desta singularidade
S16	EP4	2	Dispensador com defeito	Dinheiro enrosca durante a liberação	(C) Notificar o cliente, quando isto ocorrer (D) Tornar indisponível a transação de saque
S17	EP4	2	Dinheiro não pegado	Cliente não pega o dinheiro do dispensador, a tempo (*Timeout*)	(C) Recolher o dinheiro e cancelar a transação

Faz-se uso de uma tabela de sucesso e falha de episódios para a definição dos casos de uso singulares. Esta tabela tem como cabeçalho o objetivo que aglutina os possíveis casos de uso singulares que tratam de sua realização. A cada caso de uso singular identificado, são atribuídos um identificador e um nome. O nome deve ser sugestivo do comportamento do caso de uso. Para os casos de uso singulares que modelam casos diferentes do “normal”, adota-se o nome do caso de uso, como uma

combinação do objetivo do caso de uso, o nome do episódio onde a singularidade aparece e nome da singularidade. Um exemplo de definição de casos de uso singulares, para as singularidades, anteriormente encontradas, é mostrado na tabela 5.5.

TABELA 5.5 – Tabela de sucesso e falha de episódios para definição dos casos de uso singulares do objetivo Sacar Dinheiro

Objetivo: Sacar dinheiro						
Id do caso de uso	**Nome do caso de uso**	**EP1**	**EP2**	**EP3**	**EP4**	**EP5**
UC1	Sacar dinheiro	S	S	S	S	S
UC4	Sacar dinheiro; Fornece identificação ao sistema; Caixa eletrônico desligado	S01	F	F	F	F
UC5	Sacar dinheiro; Fornece identificação ao sistema; Leitor quebrado ou cartão inelegível	S02	S	S	F	F
UC6	Sacar dinheiro; Fornece identificação ao sistema; Cartão inválido	S03	S	S	F	F
UC7	Sacar dinheiro; Fornece identificação ao sistema; Cartão bloqueado	S04	S	S	F	F
UC8	Sacar dinheiro; Fornece identificação ao sistema; *Timeout* da senha	S05	S	S	F	F
UC9	Sacar dinheiro; Fornece identificação ao sistema; Senha incorreta	S06	S	S	F	F
UC10	Sacar dinheiro; Pede saque; Transação inválida	S	S07	F	F	F
UC11	Sacar dinheiro; Pede saque; *Timeout* do pedido da transação	S	S08	F	F	F
UC12	Sacar dinheiro; Escolhe o valor do saque; Valor incorreto do saque fornecido	S	S	S09	F	F
UC13	Sacar dinheiro; Escolhe o valor do saque; *Timeout* da escolha do valor do saque	S	S	S10	F	F
UC14	Sacar dinheiro; Escolhe o valor do saque; Valor do saque excede o disponível, no caixa eletrônico	S	S	S11	F	F
UC15	Sacar dinheiro; Escolhe o valor do saque; Valor do saque excede o saldo da conta do cliente	S	S	S12	F	F
UC16	Sacar dinheiro; Escolhe o valor do saque; *Timeout* da confirmação da transação	S	S	S13	F	F
UC17	Sacar dinheiro; Pega o dinheiro e o recibo; Erro na emissão do recibo	S	S	S	S14	F
UC18	Sacar dinheiro; Pega o dinheiro e o recibo; *Timeout* para pegar o recibo	S	S	S	S15	F
UC19	Sacar dinheiro; Pega o dinheiro e o recibo; Dinheiro enrosca, durante a liberação	S	S	S	S16	F
UC20	Sacar dinheiro; Pega o dinheiro e o recibo; *Timeout* para pegar o dinheiro	S	S	S	S17	F

Para episódios com ordenamento seqüencial, ao ocorrer uma falha num episódio, pode-se, com pouco esforço, determinar que os episódios posteriores, necessariamente também falharão. Isto se deve ao fato de que as pós-condições de um episódio precedente fazem parte das pré-condições do episódio posterior.

Após a definição dos casos de uso singulares, cada um deles é descrito de acordo com a notação gramatical. Um exemplo de caso de uso singular *UC19* que trata da singularidade *Dinheiro enrosca durante a liberação (S16)* é apresentado na figura 5.10. Por motivos de clareza, apresenta-se um extrato deste caso de uso, detalhando apenas o episódio (*Pega o dinheiro e o recibo*), onde a singularidade ocorre, e as ações corretivas

e defensivas são propostas. Limita-se a citar os demais episódios, por eles serem integralmente reaproveitados do caso de uso "normal" (*UC1*). Observa-se que o plano (conjunto de ações) do episódio que trata desta singularidade não corresponde ao plano do episódio onde não ocorre nenhuma singularidade. Sendo assim, para identificar as variantes dos episódios "normais", adota-se o padrão de concatenar o identificador do episódio "normal" com o da singularidade. No exemplo, em questão, obteve-se o identificador *EP4.S16*.

Informação Descritiva
ID: UC19
Nome: Sacar dinheiro; Pega o dinheiro e o recibo; Dinheiro não liberado
Contexto
Ator Iniciador: Cliente
Ator Participante: -
Recursos: cartão, senha, conta, valor do saque, dinheiro, recibo
Pré-condição: caixa eletrônico pronto para usar; cliente com cartão;
Pós-condição: caixa eletrônico pronto para usar; cliente com o cartão; cliente sem o dinheiro; cliente sem o recibo; valor do saque não debitado na conta.
Narrativa
Episódio
ID: EP1
Nome: Fornece identificação ao sistema
Pré-condição: caixa eletrônico pronto para usar; cliente com o cartão
Pós-condição: caixa eletrônico pronto para usar ou em uso; cliente com o cartão; cartão e senha do cliente válidos
Episódio
ID: EP2
Nome: Pede Saque
Pré-condição: caixa eletrônico pronto para usar
Pós-condição: caixa eletrônico pronto para usar; pedido de transação saque aceito
Episódio
ID: EP3
Nome: Escolhe o valor do saque
Pré-condição: caixa eletrônico pronto para usar; pedido de transação saque aceito
Pós-condição: caixa eletrônico pronto para usar; pedido de transação saque aceito; valor de saque válido
Episódio
ID: EP4.S16
Nome: Pega o dinheiro e o recibo; dinheiro enrosca, durante a liberação
Pré-condição: caixa eletrônico pronto para usar; cartão e senha do cliente válidos; valor de saque válido
Pós-condição: caixa eletrônico pronto para usar; cartão e senha do cliente válidos; valor de saque válido; cliente sem o dinheiro e sem o recibo
Plano:

Nº	**Agente**	**Ação**
1	Sistema	Libera o valor do saque em dinheiro para o cliente
2	Sistema	Detecta falha no dispensador de dinheiro
3	Sistema	Notifica o cliente de que a transação saque não pode ser efetuada
4	Sistema	Bloqueia a transação saque

Episódio
ID: EP5
Nome: Sair
Pré-condição: caixa eletrônico pronto para usar
Pós-condição: caixa eletrônico pronto para usar
Anexos
Ancora: RS1; RS2; RS3; RS5; RS6; RS8

FIGURA 5.10 – Caso de uso singular *Sacar dinheiro; Pega o dinheiro e o recibo; Dinheiro não liberado*

Ao ser descrito este caso de uso singular, constatou-se que só se poderá emitir o recibo de realização da transação saque, depois de ter certeza de que foi realizada, com sucesso. Dessa forma, a alternativa de emitir o recibo, antes de liberar o dinheiro a fim de se minimizar o esquecimento do recibo, por parte do cliente, mostrou-se ineficiente, pois podem ocorrer problemas com o dispensador de dinheiro e a transação não se efetivar. Deve-se, assim, pensar em outras alternativas para evitar o esquecimento do recibo, como, por exemplo, a exibição de uma mensagem de realização do saque com sucesso e de solicitação, para o cliente pegar o recibo. Deste modo, o episódio *Pega o dinheiro e o recibo (EP4)* tornar-se-ia o episódio ilustrado na figura 5.11.

Episódio

ID: EP4

Nome: Pega o dinheiro e o recibo

Pré-condição: caixa eletrônico pronto para usar; cartão e senha do cliente válidos; valor de saque válido

Pós-condição: caixa eletrônico pronto para usar; cartão e senha do cliente válidos; valor de saque válido; cliente com o dinheiro e o recibo

Plano:

Nº	Agente	Ação
1	Sistema	Libera o valor do saque, em dinheiro, para o cliente
2	Cliente	Pega o dinheiro
3	Sistema	Altera o saldo da conta do cliente
4	Sistema	Informa o cliente do sucesso da transação saque e da emissão do recibo
5	Sistema	Emite o recibo da transação saque para o cliente
6	Cliente	Pega o recibo
7	Sistema	Registra a transação saque

FIGURA 5.11 – Episódio *Pega o dinheiro e o recibo*

Este exemplo permite que se compreenda o valor de considerar as singularidades e a iteratividade do processo de construção de casos de uso. Conforme se caminha aos níveis de menor abstração, podem-se avaliar com maior precisão as conseqüências de opções de desenvolvimento. Neste exemplo, trata-se de uma suposta falha de um componente interno do caixa eletrônico: o dispensador de dinheiro. As ações defensivas e corretivas adotadas podem não ser a melhor solução possível, o que se gostaria de destacar é que a consideração de singularidades e a análise dos casos de uso contribuem para a flexibilidade e a usabilidade do sistema.

5.2.4 Construção de Casos de Uso Operacionais

A construção dos casos de uso operacionais é apoiada em heurísticas adaptadas das abordagens de Jacobson [JAC 92] e Bruegge [BRU 2000], tanto para a determinação dos tipos de objetos responsáveis quanto para descrição dos diagramas de seqüência.

Para identificarem-se objetos entidade, analisa-se os casos de uso singulares. Análise de linguagem natural é um conjunto intuitivo de heurísticas para identificar objetos, atributos e associações de uma especificação de sistema. As heurísticas de Abbott [ABB 83], apresentadas na tabela 5.6, mapeiam tipos de palavras em componentes de modelo. Em geral, essas heurísticas funcionam bem para gerarem uma lista inicial de objetos candidatos, a partir de pequenas descrições, como as narrativas dos casos de uso [BRU 2000].

TABELA 5.6 – Heurísticas para mapear tipos de palavras em componentes de modelo [ABB 83]

Tipo de palavra	Componente de modelo	Exemplos
Substantivo próprio	Objeto	João
Substantivo impróprio	Classe	Cliente
Verbo de ação	Operação	Cria, submete, seleciona
Verbo de estado	Herança	É um tipo de
Verbo de posse	Agregação	Tem, consiste de, inclui, possui
Verbo modal	Restrições	Deve ser
Adjetivo	Atributo	Cartão Ouro

Em conjunto com as heurísticas de Abbott, podem-se usar as seguintes heurísticas para identificarem objetos entidade [BRU 2000]:

- Termos que desenvolvedores ou usuários precisam clarificar para entenderem o caso de uso;
- Procurar substantivos, nos casos de uso;
- Entidades do mundo real que o sistema precisa manter;
- Processos e procedimentos do mundo real que o sistema precisa manter;
- Fontes de dados.

Desenvolvedores nomeiam e descrevem brevemente os objetos, seus atributos e suas responsabilidades, conforme são identificados. Descrever objetos, mesmo que brevemente, permite aos desenvolvedores clarificarem os conceitos que estão usando e evita enganos [BRU 2000].

Objetos de interface modelam a interface do sistema com os atores, é através destes que atores se comunicam com o sistema. Em cada caso de uso, cada ator interage através de, pelo menos, um objeto de interface. O objeto de interface coleta a informação do ator e a traduz em um evento que pode ser usado pelos objetos entidade e, também, pelos objetos de controle, assim como traduz os eventos gerados pelo sistema em informação apresentada aos atores. Para identificar objetos de interface são utilizadas as heurísticas:

- Cada interação entre um ator e o sistema é alcançada pelo intermédio de, pelo menos, um objeto de interface;
- Em geral, periféricos, como impressora, teclado, tela ou outro periférico especial, são ligados à interação ator-sistema e cada um deles pode ser manipulado por um objeto de interface particular;
- Identificar notificações e requisições de informação aos atores;
- Não modelar os aspectos visuais da interface com objetos de interface, protótipos são mais adequados para tal.

Objetos de controle são responsáveis por coordenarem os objetos de interface e os objetos entidade. Objetos de controle, normalmente, não têm uma contraparte concreta no mundo real. Há freqüentemente uma forte relação entre um caso de uso e um objeto de controle. Um objeto de controle, normalmente, é criado no começo de um caso de uso e deixa de existir a seu fim. É responsável por coletar informação dos objetos de interface e despachar para objetos entidade. As heurísticas para identificarem objetos de controle são as seguintes:

- Depois de atribuir o comportamento de um caso de uso aos objetos de interface e objetos entidade, o comportamento restante é atribuído a um objeto de controle;
- Identificar um objeto de controle, por um caso de uso ou mais, se o caso de uso for complexo e puder ser dividido em fluxos mais curtos de eventos (episódios).

Os atributos dos objetos também podem ser identificados, usando as heurísticas de Abbott, em particular, frases de substantivo, seguidas por frases possessivas; ou frases adjetivas devem ser examinadas. No caso de objetos de entidade, qualquer propriedade que necessite ser armazenada pelo sistema é um atributo candidato.

Depois de identificados os objetos, modela-se as interações entre eles. Os casos de uso operacionais são representados por diagramas de seqüência, construídos com base nas seguintes heurísticas:

- A primeira coluna, comumente, deve corresponder ao ator que iniciou o caso de uso;
- A segunda coluna, geralmente, deve ser um objeto de interface, que o ator usa para iniciar o caso de uso;
- A terceira coluna, comumente, deve ser o objeto de controle que administra o resto do caso de uso;
- Objetos de controle são criados por objetos de interface que iniciam o caso de uso;
- Objetos de interface são criados por objetos de controle;
- Objetos entidade são acessados por objetos de controle e de interface;
- Objetos entidade nunca acessam objetos de interface ou de controle; isto torna mais fácil de compartilhar objetos entidade, entre casos de uso.

Um exemplo de objetos identificados e de diagrama de seqüência para o episódio *Fornece identificação ao sistema (EP4)* do caso de uso singular *Sacar dinheiro (UC1)* é mostrado, respectivamente, nas figuras 5.12 e 5.13. De fato, como episódios são utilizados como mecanismos de modularização, constrói-se um diagrama de seqüência para cada episódio dos casos de uso singulares. Ressalta-se que os episódios, que tratam de singularidades, também são representados em diagramas de seqüência distintos, para aumentarem a clareza da especificação.

Tipo de objeto	**Nome**	**Definição**
Entidade	Cartão	Objeto do domínio do problema
Entidade	Senha	Objeto do domínio do problema
Interface	Painel do Cliente	Objeto que realiza as interações via tela e teclado.
Interface	Leitor	Objeto que realiza as interações com o leitor de cartão
Controle	Identificação	Objeto que controla a transação que corresponde ao episódio *EP4*

FIGURA 5.12 – Objetos identificados no episódio *Fornece identificação ao sistema*

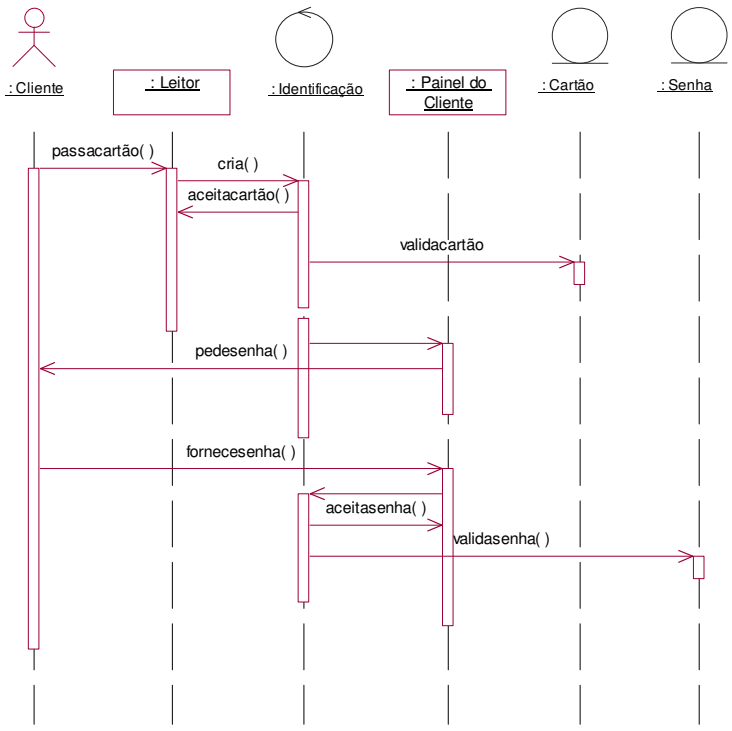

FIGURA 5.13 – Diagrama de seqüência para o episódio *Fornece identificação ao sistema*

5.2.5 Construção de Casos de Uso Concretos (Cenários)

A construção dos cenários, casos de uso concretos, é feita diretamente, a partir de casos de uso singulares. Cada ação dos episódios de um caso de uso escolhido é instanciada pela determinação de suas características concretas que servirão para a experimentação com um usuário específico.

Para instanciar os atores, faz-se uso da lista ator-pseudônimos. Esta lista é inicialmente criada no começo do desenvolvimento, tendo a função de manter a correspondência entre os nomes genéricos e específicos dos atores e, também, de manter informação sobre os atores, para auxiliar na definição do tipo de interação e, conseqüentemente, da IU. O objetivo e a descrição dos campos da lista ator-pseudônimo são apresentados a seguir:

- Ator: Nome usado nos casos de uso para descrever o papel;
- Definição: Definição informal do papel do ator no sistema;
- Nomes específicos: Freqüentemente, usados nas entrevistas e descrições textuais. Fazem referência a indivíduos ou elementos particulares;
- Tipo de uso: Freqüente, ocasional, especialista, genérico;
- Volume de uso: Número de transações / Unidade de tempo;
- Iniciador dos casos de uso: ID dos casos de uso onde o ator é o agente iniciador;
- Participante dos casos de uso: ID dos casos de uso onde o ator participa; mas não é o iniciador.

A tabela 5.7 representa um excerto da lista ator-pseudônimos para o caixa eletrônico.

TABELA 5.7 – Lista ator-pseudônimo para o caixa eletrônico

Ator	**Definição**	**Nomes específicos**	**Tipo de uso**	**Volume de uso**	**Iniciador dos casos de uso**	**Participante dos casos de uso**
Cliente	Pessoa que usa o caixa eletrônico. Não é esperado ser experiente no uso de computadores. Pode ter dificuldade de leitura, ser daltônico, etc.	Proprietário da conta, Maria, correntista	genérico	Diversas vezes, ao dia	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8, UC9, UC10, UC11, UC12, UC13, UC14, UC15, UC16, UC17, UC18, UC19, UC20	-
Operador	Pessoa que executa as operações de mantença do caixa eletrônico. Experiente no uso de computadores. Pode ter uma IU customizada.	João	especialista	Poucas vezes, ao dia	UC30	UC31
...	...	...	...	...	...	...

De posse dos casos de uso singulares, os projetistas de interface também possuem os elementos suficientes para, através de modelos de interface, especificarem as IUs e, assim, construírem os protótipos de "baixo nível", propriamente ditos. O ideal é que fossem desenvolvidos os protótipos, baseados em critérios de IHC, como os critérios ergonômicos [CYB 2000]. Todavia, independente de detalhes da IU, o ponto de crucial

importância é que os protótipos são construídos para suportarem as tarefas dos usuários. O quão difícil ou fácil for a interação com um artefato, estará altamente relacionada com qual informação está representada e como ela está representada, na IU. Uma IU deve ser projetada de maneira que as ações e operações que devem ser realizadas interagindo com o sistema, correspondam às maneiras naturais de pensar e agir do usuário. Um exemplo de caso de uso concreto foi ilustrado na figura 4.9 da seção 4.3.4.

5.2.6 Experimentação de Cenários

Erros de requisitos são freqüentes, devido à validação imprópria. Embora o uso de *templates*, heurísticas e diretrizes aumente a qualidade dos casos de uso produzidos, ele não garante que os casos de uso estejam livres de erros. A coleção de casos de uso abstratos também não é suficiente para garantir a usabilidade do sistema proposto. São necessários testes com os usuários e análise de interfaces.

Durante a experimentação de cenários são feitas avaliações para determinar como o sistema proposto atende as reais necessidades dos *stakeholders*. Revisões iterativas de casos de uso com *stakeholders* ajudam desenvolvedores a assegurar que os requisitos estão completos, consistentes e realísticos. São úteis para assegurar que os comportamentos dinâmicos, descritos pelo modelo, correspondem ao comportamento desejado.

A experimentação dos casos de uso concretos consiste em simular a interação, seguindo a seqüência descrita pelo cenário, usando conjuntamente os protótipos desenvolvidos. Esta simulação permite o usuário avaliar a funcionalidade e a usabilidade do sistema; isto pode implicar mudanças, em casos de uso, nos requisitos ou em ambos. O uso destes cenários não apenas possibilita verificar a atual solução proposta no real contexto de uso; mas, também, descobrir novos requisitos que emergem somente na prática. A rápida geração de idéias é uma das vantagens do uso conjunto de protótipos. Como resultado dessa avaliação, as conseqüências da introdução da nova tecnologia podem ser analisadas e as práticas de trabalho podem ser repensadas pelos usuários.

Freqüentemente, existem condições de falha que fazem referência a regras organizacionais que o desenvolvedor desconhece [COC 2000]. Cenários ajudam a elicitar estas condições uma vez que são testados com os próprios usuários.

Em TAREFA [PIM 97] encontram-se valiosas recomendações a serem utilizadas durante as sessões de experimentação de cenários com os usuários:

- Usar, sempre que possível, o cenário no contexto concreto de utilização;
- Usar um cenário, por vez;
- Explicar somente as alternativas possíveis;
- Deixar o usuário expor suas idéias, sem repressões.

Idealmente, pelo menos uma experimentação de cada caso de uso singular instanciado deve ser realizada com um usuário. Caso não seja possível realizar a experimentação com os reais usuários do futuro sistema, sugere-se realizar a experimentação dos cenários com membros da equipe de desenvolvimento, onde, então, deverá ser efetuada uma avaliação heurística; isto é, os membros da equipe avaliarão os cenários, baseados em seus conhecimentos empíricos e em heurísticas.

O caso de uso que modela o curso normal deve ser experimentado, primeiramente, por ilustrar como o sistema funcionará freqüentemente. Inclusive este teste pode ser realizado antes de se levantarem as singularidades e as ações defensivas e corretivas. Pois, sendo cenário um meio concreto, capaz de conduzir conjuntamente a ação e reflexão, pode servir para o usuário participar mais ativamente do processo de desenvolvimento, levantando possíveis casos de falha ou exceção, além de encontrar possíveis erros.

Mesmo alertando o usuário que a experimentação seria realizada com um protótipo de baixo nível e o objetivo não era avaliar detalhes de interface; porém a semântica da interface, o primeiro questionamento do usuário foi "*Mas vai ser desse jeito mesmo? Com essa aparência?*" O que veio a confirmar uma constatação encontrada em [COM 2000c] que, para o usuário, o sistema é a interface. Neste sentido, casos de uso essenciais mostram-se de grande valia, por permitirem que *designers* de interface tenham elementos suficientes para projetá-las: além do corpo narrativo que descreve a interação entre atores e sistema, o campo freqüência de uso e possíveis âncoras a RNFs são úteis na definição das IUs; as expressões de relação temporal auxiliam a definir o ordenamento dentro do caso de uso e os campos de pré e pós-condições ajudam a estabelecer o ordenamento entre os casos de uso.

5.2.7 Modificação de Casos de Uso

As críticas dos usuários são as principais origens das modificações dos casos de uso. Estas podem ser pequenas e gerar apenas variantes dos casos de uso concretos, mudando exclusivamente suas características concretas; ou grandes, provocando algumas alterações a nível operacional, singular ou até mesmo essencial do caso de uso, ou seja, propagando alterações para os níveis mais abstratos. O analista deve revisar os casos de uso, visando adequá-los às expectativas dos usuários. Os casos de uso modificados devem ser novamente instanciados e experimentados, junto ao usuário, até chegar-se, consensualmente, a um conjunto de requisitos considerado satisfatório pelos participantes do desenvolvimento.

Mudanças ao nível essencial lidam com a escolha de alternativas de desenvolvimento. Ao nível singular, descrevem diferentes possíveis realizações de uma tarefa. Mudanças ao nível operacional referem-se a diferentes maneiras de distribuir responsabilidades entre objetos.

Envolver mais efetivamente usuários no desenvolvimento de software, como co-autores, é muito importante. Porém devem-se balancear as sugestões dos usuários com a análise destas sugestões [MCD 94]. Deve-se lembrar que, enquanto um caso de uso abstrato representa o ponto de vista de um grupo de usuários (ator), um cenário representa o ponto de vista de um usuário específico. Usuários e desenvolvedores sugerem novas funções e interfaces, todavia é necessário realizar uma análise para determinar como melhor adotar cada sugestão.

5.2.8 Integração de Casos de Uso

Até o presente momento, cada caso de uso foi utilizado como um apoio à determinação dos requisitos do sistema, na perspectiva de uma situação específica de

uso, então dissociada dos outros casos de uso. Deste modo, obtém-se uma coleção solta de casos de uso separados, modelos parciais com aspectos restritos do sistema. Necessita-se integrar os casos de uso para alcançar uma visão global de utilização para um ator do sistema.

O número de possíveis relacionamentos entre os casos de uso cresce exponencialmente com o de casos de uso. Se esses relacionamentos forem formalizados, poderão ser mais facilmente identificados e suportados [ALS 99]. Episódios representam explicitamente o relacionamento entre casos de uso; os relacionamentos entre os casos de uso aparecem de forma direta na narrativa, não sendo necessária uma representação gráfica, como os diagramas de casos de uso da UML, para representá-los. Episódios comuns entre os casos de uso devem ser globalmente renumerados, visando à integração dos casos de uso.

No exemplo do caixa eletrônico, o episódio *Fornecer Identificação* e o episódio *Sair* são os mesmos para os casos de uso essenciais *Sacar Dinheiro*, *Consultar Saldo* e *Consultar Extrato*. Sendo assim, atribuí-se um identificador único para cada episódio e tem-se que o sistema do caixa eletrônico é composto pelos episódios apresentados na tabela 5.8.

TABELA 5.8 – Episódios do caixa eletrônico

ID	**Nome**
EP1	Fornece identificação ao sistema
EP2	Pede saque
EP3	Escolhe valor do saque
EP4	Pega o dinheiro e o recibo
EP5	Sair
EP6	Pede extrato
EP7	Escolhe período do extrato
EP8	Pega o recibo
EP9	Pede saque

6 Conclusão

Esta conclusão está articulada em quatro partes: a primeira resume as principais contribuições desta dissertação; a segunda compara a abordagem proposta com outras abordagens baseadas em objetivos; a terceira discute as limitações do trabalho e a quarta investiga perspectivas futuras de continuidade do trabalho.

6.1 Contribuições

Esta dissertação é um esforço para a integração das áreas de Engenharia de Software (ES) e Interação Humano-Computador (IHC), particularmente com a proposta de uma abordagem para construção sistemática e integrada de casos de uso e cenários. As contribuições podem ser resumidas pelos seguintes aspectos:

6.1.1 Compatibilidade com a UML e Resolução de Aspectos Problemáticos

A abordagem baseada em objetivos para construção de cenários e casos de uso proposta é totalmente compatível com a UML. De fato, a UML é apenas uma linguagem de modelagem e não uma metodologia de desenvolvimento. Sendo assim, ela não prescreve explicitamente nenhum processo de utilização de seus modelos.

Propõe-se uma definição do conceito de casos de uso diferente da UML; mas não incompatível. Da mesma forma que a definição de casos de uso da UML, na abordagem, casos de uso têm igualmente o propósito de especificar requisitos, com o conteúdo, na forma de uma narrativa textual consistente, contendo múltiplos cenários e possuindo uma estrutura semiformal. Porém, na abordagem, os objetivos dos usuários estão vinculados à própria definição do conceito de caso de uso (ver seção 4.1), deste modo, tem-se uma perspectiva centrada no uso e não no sistema como na UML (ver seção 2.3).

Ressalta-se que o modelo de casos de uso, adotado na UML, ainda apresenta problemas que a abordagem procurou resolver ou contornar. Com relação aos problemas conceituais discutidos na seção 2.3, tem-se o seguinte:

- Conteúdo: Casos de uso são planos de execução de objetivos, ao invés de meras seqüências cronológicas de eventos;
- Notação: Adoção de representações adequadas a cada nível de abstração, contendo elementos importantes para a usabilidade e funcionalidade;
- Expressões idiomáticas: Terminologia unificada com termos do domínio do problema;
- Granularidade: Flexível, porém não escolhida, arbitrariamente. Definem-se quatro níveis onde são considerados apenas os elementos relevantes a cada nível. Mas deve ser lembrado que o escopo de um caso de uso é definido como possibilitando no mínimo a cobertura (satisfação ou não) de um objetivo;
- Cobertura: A hierarquia de objetivos ajuda a delimitar a cobertura dos casos de uso;
- Contexto: O contexto onde o caso de uso ocorre é explicitado, sobretudo, através das pré e pós-condições;

- Interseção: O conceito de episódio é utilizado para se relacionarem casos de uso na própria descrição textual;
- Concorrência: O uso de expressões temporais para ordenamento interno e de pré e pós-condições para o ordenamento externo.

A UML não define precisamente nenhum formato ou *template* específico para descrever o conteúdo dos casos de usos. Isto faz com que desenvolvedores, freqüentemente, não saibam como identificar casos de uso, o que incluir neles, qual o nível de detalhamento, como representá-los e como estruturá-los. Por outro lado, esta negligência com o formalizar a especificação permitiu que fosse tida maior liberdade para adotar diferentes notações, adequadas a cada nível de abstração, sem que fossem incompatíveis com alguma notação, previamente definida.

Propõe-se a utilização de modelos adicionais para representar os objetivos dos usuários (modelo hierárquico de tarefas) e listas para auxiliar na construção dos casos de uso nos diferentes níveis de abstração (lista ator-objetivo, lista ator-pseudônimo, lista de singularidades etc.).

Com relação aos diagramas de casos de uso, destaca-se que a informação representada neles é similar a da lista ator-objetivo; mas a apresentação é diferente. Os diagramas de casos de uso são dispositivos mnemônicos de duas dimensões que servem para um propósito cognitivo: destacar relacionamentos. Deve-se usá-los para este propósito, não para substituir o texto. As elipses e as setas mostram apenas as ligações e decomposições e não ilustram o conteúdo dos casos de uso. Os diagramas, por si só, não definem os requisitos de interação e os requisitos funcionais do sistema. Não se deve substituir a seção narrativa dos casos de uso por diagramas. Os aspectos dinâmicos dos requisitos são representados no corpo narrativo dos casos de uso, e os diagramas devem servir como sumários, auxiliando o leitor a localizar o texto que ele precisa ler.

Sobre os relacionamentos propostos na UML, o uso do conceito de objetivos, aliado ao conceito de episódios e à adoção de níveis de abstração e de notações específicas, tornou claros os relacionamentos entre os casos de uso, na própria narrativa textual.

O relacionamento “inclui” (*includes*) da UML é facilmente representado na narrativa textual dos casos de uso. É um relacionamento normal entre um objetivo de mais alto nível com um de mais baixo nível. O caso de uso incluído descreve um objetivo de mais baixo nível do que o caso de uso base. A estrutura episódica facilita tratar este tipo de relacionamento. Cada passo em um caso de uso essencial é potencialmente um episódio (subcaso de uso). Se nunca refinar a intenção que o passo representa, então ela será simplesmente um passo; caso for refinada, dir-se-á que o caso de uso “inclui” o comportamento do episódio.

O relacionamento “estende” (*extends*) da UML é representado nesta abordagem, através de casos de uso singulares. O comportamento flui normalmente como no caso de uso base até o ponto onde a condição ocorre e, assim, a singularidade é tratada de maneira diferenciada. O conceito de episódios e singularidades auxilia tratar este tipo de relacionamento.

Quanto ao uso de notação gráfica, para expressar a narrativa dos casos de uso, também há algumas considerações. Embora os diagramas de seqüência possam igualmente representar interações entre atores, o uso desta notação gráfica para a seção narrativa dos casos de uso apresenta dois problemas de usabilidade. O primeiro é que usuários finais e outros *stakeholders* geralmente não têm familiaridade com a notação; o segundo é que diagramas não permitem demonstrar tudo o que se precisa escrever, geralmente é necessário colocar notas adicionais, nos diagramas. Estes são alguns dos motivos pelos quais não foi adotada a representação gráfica para a narrativa dos casos de uso (diagrama de seqüência) na fase de elicitação de requisitos; mas, apenas, ao nível operacional, restrito aos analistas e programadores.

6.1.2 Resumo de Resultados

1. Apresentou-se uma proposta onde se explorou o uso integrado de modelos de tarefas, cenários e casos de uso para o desenvolvimento de sistemas interativos úteis e usáveis. Propôs-se que casos de uso podem ser especialmente úteis para combinar a perspectiva técnica de ES com a perspectiva humana de IHC. Tendo os objetivos das atividades humanas, como ponto de partida, casos de uso apóiam o pensamento sobre as tarefas interativas; ou seja, sobre as interações entre usuários e sistema, a fim alcançarem objetivos, dentro de contextos de uso;
2. Promoveu-se uma construção sistemática de casos de uso, centrada no uso de um futuro sistema. Sistematizou-se a construção de cenários e casos de uso, através de guias, heurísticas e notações;
3. Encontrou-se uma base teórica, tanto na perspectiva de IHC (relação tarefa-objetivo [STO 95]), quanto na perspectiva de ES (relação função-objetivo [CHA 96]) para a proposição da abordagem baseada em objetivos de construção de casos de uso e cenários;
4. Demonstrou-se que conceitos teleológicos são importantes para organizar e estruturar casos de uso e permitem solucionar grande parte dos problemas conceituais de casos de uso;
5. Mostrou-se que o conceito de episódios tem um papel significante no gerenciamento dos casos de uso. Episódios podem ser usados como mecanismos de modularização. A estrutura episódica possibilita representar os relacionamentos entre os casos de uso, dentro da própria descrição narrativa;
6. A construção de casos de uso, delimitados pela estrutura hierárquica de objetivos, evita que se desenvolvam soluções técnicas para problemas irrelevantes;
7. O uso dos quatro níveis de abstração ajudou a lidar com a complexidade da construção e validação dos casos de uso e permitiu utilizar de maneira integrada casos de uso e cenários. À medida que a modelagem é realizada, o desenvolvedor consegue perceber mais claramente os efeitos de suas decisões e, assim, obter soluções mais adequadas;
8. Construíram-se primeiramente os casos de uso essenciais que representam a essência das tarefas. Ao nível singular, inspecionaram-se as singularidades, o que contribuiu para a usabilidade; ao nível concreto, cenários permitiram uma avaliação da usabilidade de um futuro sistema, antes da fase de projeto; já o nível operacional estabeleceu uma ligação com abordagens OO e permitiu a continuidade do desenvolvimento de sistemas nestas abordagens;

9. Através da abordagem, a concepção do conteúdo e a organização das IUs são deduzidas, a partir da lógica de uso do sistema, como suporte à realização de tarefas, e não, a partir da lógica de funcionamento das funções do sistema;
10. Casos de uso criados, a partir da abordagem, são gerados de maneira objetiva, ao invés de intuitiva. Casos de uso gerados, sistematicamente, permitem estudar os comportamentos não normativos dos usuários mais a fundo, o que contribuiu para uma investigação mais dirigida à usabilidade de sistemas.

6.2 Comparação com Outras Abordagens Baseadas em Objetivos

Tem-se aqui, representada na tabela 6.1, uma sucinta comparação da abordagem proposta com algumas outras abordagens baseadas em objetivos que foram apresentadas na seção 3.2 e comparadas na seção 3.2.4 (ver tabela 3.1).

TABELA 6.1 – Comparação com outras abordagens baseadas em objetivos

Abordagem / **Critério**	**Potts**	**Cockburn**	**Rolland**	**Abordagem Proposta**
Investigação das singularidades	Um questionamento sistemático e a criação de ações defensivas e corretivas	Um questionamento sistemático e algumas guias.	Algumas guias	Um questionamento sistemático, inspeção de pré e pós-condições, algumas guias, e a criação de ações defensivas e corretivas
Delimitação do conjunto de casos de uso	Algumas heurísticas para determinar a saliência dos casos de uso. Uso do conceito de episódios	Uso de técnicas de subordinação, extensão e variação	Sugere que os objetivos delimitam o número de casos de uso. Uso do conceito de episódios	Hierarquia de objetivos. Heurísticas para determinar os casos de uso relevantes. Uso do conceito de episódios
Uso complementar dos conceitos de cenários e casos de uso	Apenas é sugerido, sem especificar, que cenários podem ser instanciados para serem avaliados pelos usuários	Cenários não são usados para experimentação com usuários	Cenários são usados apenas num primeiro momento para elicitar objetivos e posteriormente são integrados em casos de uso. Não são, porém, usados para experimentação com usuários	Construção sistemática e integrada. Casos de uso e cenários são integrados em estruturas com diferentes níveis de abstração. Cenário como uma instância específica de um caso de uso, usado para experimentação com usuários
Avaliação da usabilidade com os usuários	Não	Não	Não	Através de cenários e protótipos de baixo nível
Diferentes níveis de abstração	Não	Sumário, objetivos do usuário e subfunções	Contextual, funcional e físico	Essencial, singular, operacional e concreto
Nível de abstração que estabelece ligação com abordagens OO	Não	Não	Não	Nível operacional
Notações definidas	Sim	Sim	Sim	Sim
Guias para a descrição textual dos casos de uso	Não	Sim	Sim	Sim
Relações temporais nos casos de uso	Não	Não	Não	Através da expressão de relação temporal
Fluxo dos eventos	Seqüencial	Seqüencial	Seqüencial	De acordo com a expressão de relação temporal (seqüencial ou ordenação flexível)
Consideração de eventos assíncronos	Não	Sugere tratá-los em uma seção separada, usando asterisco (*) para identificá-los	Não	Descritos separadamente e tratados na expressão de relação temporal
Representação de Objetivos	Apenas é sugerida uma hierarquia de objetivos similar a modelos de tarefas	Metáfora do veleiro	*Requirements Chunks* organizados hierarquicamente	Modelo de tarefa minimal
Representação da ontologia	Não	Não	Não	Sim

6.3 Limitações

A falta de aplicação da abordagem em um estudo de caso real é uma limitação clara do trabalho. Exemplos mais claros da integração da abordagem com uma metodologia de desenvolvimento, orientado a objetos, poderiam auxiliar os interessados em usar esta abordagem.

6.4 Perspectivas de Trabalhos Futuros

Perspectivas de continuidade do trabalho incluem:

- Aplicação da abordagem em estudos de casos reais (ver Limitações acima);
- Embora a ênfase seja no processo, acredita-se que o desenvolvimento uma ferramenta com a capacidade de navegação por *hyperlinks* e, também, capacidade de ordenamento, facilitaria a adoção da proposta, principalmente devido à iteratividade do processo. Uma possível ferramenta poderia ter esquemas, em XML (*Extensible Markup Language*), para a notação aos níveis essencial e singular, onde se poderia ter proveito de *hyperlinks* para navegar ao longo dos níveis de abstração e do modelo ontológico; geração assistida de cenários e protótipos; e mecanismos de ordenação e filtro, como nas planilhas eletrônicas, para trabalhar com as listas auxiliares. Em resumo, a ferramenta permitiria apoiar a construção dos casos de uso, obter rastreabilidade (*traceability*) e obter ainda, uma consistência da terminologia, mais facilmente;
- Uma possível extensão da abordagem seria avaliar os benefícios da categorização de objetivos sobre as duas dimensões ortogonais: tipo e assunto, a fim de auxiliar na questão de múltiplos pontos de vista, conforme sugerido em alguns trabalhos relacionados [ANT 98b, KAV 96].

Bibliografia

[ABB 83] ABBOTT, R. Program Design by Informal English Descriptions. **Communications of the ACM**, [S.l.], v. 26, n. 11, 1983.

[ABO 94] ABOULAFIA, A.; KLAUSEN, T.; JORGENSEN, A. H. Towards a Theoretical Underpinning fo Scenarios. In: INFORMATION SYSTEMS RESEARCH SEMINAR, IRIS, 17., 1994, Finland. **Proceedings...** [S.l.:s.n.], 1994. p.561-571.

[ACH 98] ACHOUR, C. Ben. Guiding Scenario Authoring. In: EUROPEAN-JAPANESE CONFERENCE ON INFORMATION MODELLING AND KNOWLEDGE BASES, 8., 1998, Finland. **Proceedings...** [S.l.]: H. Jaakola, H. Kangassalo, 1998. p.181-200.

[ALS 99] ALSPAUGH, T. A.; ANTÓN, A. I.; BARNES, T.; MOTT, B. An Integrated Scenario Management Strategy. In: IEEE INTERNATIONAL SYMPOSIUM ON REQUIREMENTS ENGINEERING, RE, 4., 1999, Limerick, Ireland. **Proceedings...** [S.l]: IEEE Computer Society, 1999. p.142-149.

[ANT 96] ANTÓN, A. I. Goal-Based Requirements Analysis. In: INTERNATIONAL CONFERENCE ON REQUIREMENTS ENGINEERING, ICRE, 1996, Colorado, USA. **Proceedings...** Colorado Springs: ICRE, 1996. p.136-144.

[ANT 98a] ANTÓN, A. I.; POTTS, C. A Representational Framework for Scenarios of Systems Use. **Requirements Engineering Journal,** [S.l.], v.3, n.3-4, p.219-241, 1998.

[ANT 98b] ANTÓN, A. I.; POTTS, C. The Use of Goal to Surface Requirements for Evolving Systems. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, ICSE, 20., 1998, Koyto, JP. **Forging New Links:** proceedings. [Los Alamitos: IEEE Computer Society], 1998. p.157-166.

[BEN 93] BENNER, M. K. et al. Utilizing Scenarios in the Software Development Process. In: INFORMATION SYSTEM DEVELOPMENT PROCESS, IFIP, 1993, Como, Italy. **IFIP WG8.1 Working Conference on Information System Development Process:** proceedings. [S.l.:s.n.], 1993

[BID 2001] BIDDLE, R.; NOBEL, J.; TEMPERO, E. Essential Use Cases and Responsibility in Object-Oriented Development. **Australian Computer Science Communications**, [S.l.], v. 24 , n. 1, Jan.-Feb. 2002.

[BØD 2000] BØDKER, S. Scenarios in User-Centered Design – Setting the Stage for Reflection and Action. **Interacting with Computers,** [S.l.], v. 13, n. 1, p.61-75, 2000.

[BOO 99] BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **The Unified Modeling Language User Guide.** Reading, MA: Addison-Wesley, 1999. 482 p.

[BRU 2000] BRUEGGE, Bernd; DUTOIT, Allen H. **Object-Oriented Software Engineering:** Conquering Complex and Changing Systems. [S.l.]: Prentice-Hall, 2000. 553p.

[CAM 92] CAMPBELL, R. L. Will the real scenario please stand up? **SIGCHI Bulletin,** [S.l.], v. 24, n. 2, p.6-8, 1992.

[CAR 95] CARROLL, J. M. Artifacts and Scenarios: An Engineering Approach. In: MONK, A.; GILBERT, N. (Ed.). **Perspectives on HCI:** Diverse Approaches. London: Academic Press, 1995. chap.6, p.121-144.

[CAR 2000] CARROLL, J. M. Five Reasons for Scenario-Based Design. **Interacting with Computers,** [S.l.], v. 13, n. 1, p.61-75, 2000.

[CHA 96] CHANDRASEAKARAN, B.; KAINDL, H. Representing Functional Requirements and User-System Interactions. In: NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, AAAI, 13., 1996, Portland, USA. **Proceedings…**Menlo Park: AAAI, 1996.

[COC 97a] COCKBURN, A. Structuring Use Cases with Goals (Part 1). **Journal of Object Oriented Programming,** [S.l.], v. 9, n. 5, p.35-40, Sept./Oct. 1997.

[COC 97b] COCKBURN, A. Structuring Use Cases with Goals (Part 2). **Journal of Object Oriented Programming,** [S.l.], v. 9, n. 6, p.56-62, Nov./Dec. 1997.

[COC 98] COCKBURN, A. **Basic Use Case Template.** Oct. 1998. Disponível em: <http://www.members.aol.com/acockburn/papers/uctempla.htm>. Acesso em: 08 jan. 2002.

[COC 2000] COCKBURN, A. **Writing Effective Use Cases**. Boston: Addison-Wesley, 2000.

[CON 99] CONSTANTINE, L.; LOCKWOOD, L. **Structured Use Case Form.** 1999. Disponível em: <http://www.foruse.com/Files/Forms/ taskcases6.pdf>. Acesso em: 20 fev. 2002.

[CON 2000a] CONSTANTINE, L. et al. **Structure and Style in Use Cases for User Interface Design.** 2000. Disponível em: <http://www.foruse.com/ Presentations/ structurestyle2.pdf>. Acesso em: 20 fev. 2002.

[CON 2000b] CONSTANTINE, L.; LOCKWOOD, L. **The Usability Challenge:** Can UML and the Unified Process Meet It? Australia, Nov. 2000. Disponível em: <http://www.foruse.com/Presentations/tools.pdf>. Acesso em: 03 mar. 2002.

[CON 2000c] CONSTANTINE, L.; LOCKWOOD, L. **What Do Users Want? Engineering Usability into Software.** Australia, Nov. 2000. Disponível em: <http://www.foruse.com/Presentations/tools.pdf>. Acesso em: 15 jan. 2002.

[CYB 98] CYBIS, W. A.; PIMENTA, M. S.; SILVEIRA, M. C.; GAMEZ, L. Uma Abordagem Ergonômica para o Desenvolvimento de Sistemas Interativos. In: WORKSHOP SOBRE FATORES HUMANOS EM SISTEMAS COMPUTACIONAIS, IHC, 1., 1998, Maringá-PR. **Compreendendo Usuários, Construindo Interfaces:** atas. [Rio de Janeiro: PUC-RJ], 1998.

[CYB 2000] CYBIS, W. A. **Abordagem ergonômica para IHC.** 2000. Apostila de curso. Laboratório de Utilizabilidade INE/UFSC, Florianópolis. Disponível em: <http://www.labiutil.inf.ufsc.br/apostila/apostila.htm>. Acesso em: 18 maio 2002.

[CYS 2001] CYSNEIROS, L. M.; LEITE, J. C. S. P. Driving Non-Functional Requirements to Use Cases and Scenarios. In: SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE, SBES, 15., 2001, Rio de Janeiro. **Anais...** Rio de Janeiro: UFRJ, 2001.

[FER 99] FERREIRA, Aurélio B. H. **Novo Dicionário Aurélio – Século XXI.** [S.l.]: Nova Fronteira, 1999.

[FOW 97] FOWLER, M.; SCOTT, K. **UML Distilled:** Applying the Standard Object Modeling Language. Reading: Addison-Wesley, 1997.

[FOW 2000] FOWLER, M.; SCOTT, K. **UML Distilled:** A Brief Guide to the Standard Object Modeling Language. 2nd ed. Reading: Addison-Wesley, 2000.

[FUR 98] FURLAN, J. D. **Modelagem de Objetos através da UML.** São Paulo: Makron Books, 1998.

[JAC 92] JACOBSON, I. et al. **Object Oriented Software Engineering:** A Use Case Driven Approach. Workingham: Addison-Wesley, 1992.

[JAC 94a] JACOBSON, I. Basic Use-Case Modeling. **Report on Object Analysis & Design,** [S.l.], v. 1, n. 2, Aug. 1994.

[JAC 94b] JACOBSON, I. Basic Use-Case Modeling (Continued). **Report on Object Analysis & Design,** [S.l.], v. 1, n. 3, Oct. 1994.

[JAC 94c] JACOBSON, I. Use Cases an Objects. **Report on Object Analysis & Design,** [S.l.], v. 1, n. 4, Dec. 1994.

[JAR 99] JARKE, M. Scenarios for Modeling. **Communications of the ACM,** New York, v. 42, n. 1, Jan. 1999.

[KAI 95] KAINDL, H. An Integration of Scenarios with Their Purposes in Task Modelling. In: SYMPOSIUM ON DESIGNING INTERATIVE SYSTEMS, DIS, 1995, Ann Arbor, USA. **Processes, Practices, Methods & Techniques:** proceedings. [Ann Arbor: ACM], 1995. p.227-235.

[KAI 99] KAINDL, H.; CARROLL, J. M. Symbolic Modeling in Practice. **Communications of the ACM,** New York, v. 42, n. 1, Jan. 1999.

[KLA 93] KLAUSEN, T.; BERSEN, N.O. COSITUE: Towards a Methodology for Constructing Scenarios. In: COGNITIVE SCIENCE APPROACHES TO PROCESS CONTROL, CSAPC, 1993, Copenhagen. **Designing for Simplicity:** proceedings. [S.l.:s.n.], 1993, p.1-16.

[KAV 96] KAVAKLI, E.; LOCOPOULOS, P.; FILIPPIDOU, D. Using Scenarios to Systematically Support Goal-Direct Elaboration for Information System Requirements. In: IEEE SYMPOSIUM AND WORKSHOP ON ENGINEERING OF COMPUTER-BASED SYSTEM, ECBS, 1996, Friedrichshafen, Germany. **Proceedings…** [Los Alamitos: IEEE Computer Society], 1996. p.308-314.

[KYN 92] KYNG, M. Scenario? Guilty! **SIGCHI Bulletin,** [S.l.], v. 24, n. 4, p. 8-9, Oct. 1992.

[LAR 2000] LARMAN, C. **Utilizando UML e Padrões:** uma Introdução à Análise e ao Projeto Orientados a Objetos. Tradução L. A. M. Salgado. Porto Alegre: Bookman, 2000.

[LEI 97] LEITE, J. C. S. P. et al. Enhancing a Requirements Baseline with Scenarios. **Requirements Engineering Journal**, [S.l.], v.2, n.4, p.184-198, 1997.

[MAR 2000] MARKOPOULOS, P. and MARIJNISSEN, P. UML as a representation for Interaction Design. In: OZCHI, 2000, Paris. **Proceedings...** [S.l.:s.n.], 2000. p. 240-249.

[MCD 94] McDANIEL, S. E.; OLSON, G. M.; OLSON, J. S. Methods in Search of Methodology-Combining HCI and Object Orientation. In: CHI, 1994, USA. **Proceedings...** New York: ACM Press, 1994. p. 145-151.

[MON 2000] MONTEIRO, C. de C. O.; BARBOSA, S.D.J.; de SOUZA, C.S. The Role of Designer-Generated Scenarios in Developing Web Apllications: A Case Study. In: WORKSHOP SOBRE FATORES HUMANOS EM SISTEMAS COMPUTACIONAIS, IHC, 3., Gramado-RS. **Muitas Faces em Interfaces:** anais. Porto Alegre: Instituto de Informática da UFRGS, 2000.

[PAT 99] PATERNÓ, F.; MANCINI, C. Developing Task Models from Informal Scenarios. In: ACM CHI, 1999. **Late Breaking Results:** proceedings. [Pittsburgh: ACM Press], 1999. p.228-229.

[PAT 2001] PATERNÓ, F. Task Models in Interactive Software Systems. In: CHANG, S.K. (Ed.). **Handbook of Software Engineering & Knowledge Engineering**, [S.l.]: World Scientific Publishing Co., 2001.

[PIM 96] PIMENTA, M. S.; BARTHET, M. Context Modelling for an Usability Oriented Approach to Interactive Systems Requirements Engineering. In: In: IEEE SYMPOSIUM AND WORKSHOP ON ENGINEERING OF COMPUTER-BASED SYSTEM, ECBS, 1996, Friedrichshafen, Germany. **Proceedings…** [Los Alamitos: IEEE Computer Society], 1996. p.315-321.

[PIM 97] PIMENTA, Marcelo S. **TAREFA:** Une Approche pour I'Ingénierie des Besoins des Systèmes Interactifs. 1997. 285p. Tese (Doutorado) - Université Toulouse, Toulouse.

[PIM 2000] PIMENTA, M. S. TAREFA: Uma Abordagem para Engenharia de Requisitos de Sistemas Interativos. In: JORNADAS IBEROAMERICANAS DE INGENIERÍA DE REQUISITOS Y AMBIENTES SOFTWARE**,** IDEAS, 2000, Cancún. **Memorias…** Cuernavaca: Centro Nacional de Investigación y Desarollo Tecnológico, 2000.

[POT 94] POTTS, C.; TAKAHASHI, K.; ANTÓN, A. **Inquiry-Based Scenario Analysis of Systm Requirements.** [S.l.]: Georgia Tech College of Computing, Jan. 1994. Technical Report.

[POT 95] POTTS, C. Using Schematic Scenarios to Understand User Needs. In: SYMPOSIUM ON DESIGNING INTERATIVE SYSTEMS, DIS, 1995, Ann Arbor, USA. **Processes, Practices, Methods & Techniques:** proceedings. [Ann Arbor: ACM], 1995. p.247-256.

[REG 95a] REGNELL, B.; KIMBLER, K.; WESSLÉN. A. Improving the Use Case Driven Approach to Requirements Engineering. In: IEEE INTERNATIONAL SYMPOSIUM ON REQUIREMENTS ENGINEERING, RE, 2., 1995, York, England. **Proceedings...** Los Alamitos: IEEE Computer Society, 1995.

[REG 95b] REGNELL, B. Usage Oriented Requirements Engineering. In: IEEE INTERNATIONAL SYMPOSIUM ON REQUIREMENTS ENGINEERING, RE, 2., 1995, York, England. **Consortium...** Los Alamitos: IEEE Computer Society, 1995.

[REG 96] REGNELL, B.; ANDERSSON, M.; BERGSTRAND, J. A Hierarchical Use Case Model with Graphical Representation. In: IEEE INTERNATIONAL SYMPOSIUM AND WORKSHOP ON ENGINEERING OF COMPUTER BASED SYSTEMS, Friedrichshafen. **Proceedings...** Los Alamitos: IEEE Computer Society, 1996.

[ROL 98a] ROLLAND, C.; ACHOUR, C. Ben. Guiding the Construction of Textual Use Case Specifications. **Data & Knowledge Engineering Journal**, Amsterdam, v. 25, n. 1-2, p.125-160, Mar. 1998.

[ROL 98b] ROLLAND, C.; SOUVEYET, C.; ACHOUR, C. Ben. Guiding Goal Modeling Using Scenarios. **IEEE Transactions on Software Engineering**, New York, v. 24, n. 12, p.1055-1071, Dec. 1998 .

[ROL 99] ROLLAND, C. et al. Experience With Goal-Scenario Coupling In Requirements Engineering. In: IEEE INTERNATIONAL SYMPOSIUM ON REQUIREMENTS ENGINEERING, RE, 4., 1999, Limerick, Ireland. **Proceedings...** [S.l.]: IEEE Computer Society, 1999.

[ROS 95] ROSSON, M. B.; CARROLL, J. M. Integrating Task and Software Development for Object-Oriented Applications. In: CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, CHI, 1995, Denver. **Human Factors in Computing Systems.** New York: ACM, 1995. p.377-384.

[RUB 92] RUBIN, K. S.; COLDBERG, A. Object Behavior Analysis. **Communications of the ACM,** New York, v. 35, n. 9, September 1992.

[RYS 2000] RYSER, J.; GLINZ, M. **SCENT:** A Method Employing Scenarios to Systematically Derive Test Cases for System Test. Zrich: Universitt Zrich, Institut fr Informatik, Berichte des Instituts fr Informatik, 2000.

[STO 95] STORRS, Graham. The Notion of Task in Human-Computer Interaction. In: HUMAN COMPUTER INTERACTION, HCI, 1995, Huddersfield. **Preceedings…** Huddersfield: Cambridge University Press, 1995. p.357-365.

[VED 2001] VEDOATO, Roberto. **Uma Revisão Bibliográfica sobre Cenários e Casos de Uso:** Rumo à Integração de Engenharia de Software e Interação Humano-Computador no Desenvolvimento de Sistemas Interativos. 2001. Trabalho Individual (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre. (TI-999).

[VED 2002a] VEDOATO, R.; PIMENTA, M. S. Rumo a uma Abordagem Teleológica para Construção de Cenários e Casos de Uso. In: SYMPOSIUM ON HUMAN FACTORS IN COMPUTER SYSTEMS, 5., 2002, Fortaleza. **Proceedings...** Fortaleza: SBC, 2002. p. 371-375.

[VED 2002b] VEDOATO, R.; PIMENTA, M. S. Uma abordagem teleológica para construção de casos de uso e cenários. In: CONFERENCIA LATINOAMERICANA DE INFORMÁTICA, CLEI, 28., 2002, Montevideo, Uruguai. **Program and Abstracts...** Montevideo, Uruguai: CLEI, 2002. 1 CD-ROM.

[VEM 95] VEMULAPALLI, C. A Use Case FAQ. In: **SIGPLAN Notices**, New York, v.30, n.10, Oct. 1995. Trabalho apresentado na 10. Conference on

Object-Oriented Programming Systems, Languages na Applications, OPSLA, 1995.

[WIR 2001] WIRFS-BROCK, R.; McKEAN, A. **A Brief Tour of Responsibility-Driven Design**. OOPSLA 2001. Disponível em: <http://www.wirfs-brock.com/pages/resources/zip/brief_tour_of_responsibility_driven_design_tutorial.zip>. Acesso em: 27 set. 2002.

[WOO 96] WOOD, Larry E.; ZENO, Ron. Transforming User-Centered Analysis into Concrete Design. **SIGCHI Bulletin**, New York, v. 28, n.4, Oct. 1996.