

UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
BACHARELADO EM CIÊNCIAS DA COMPUTAÇÃO

**Uma Proposta de Modelagem de Ambientes com Computação
Ubíqua**

Caio Stein D'Agostini

Trabalho de conclusão de curso apresentada como parte dos requisito para obtenção do
grau Bacharel em Ciências da Computação

Florianópolis – SC

2006/2

Caio Stein D'Agostini

Modelagem de Ambientes de Computação Ubíqua

Trabalho de conclusão de curso apresentada como parte dos requisito para obtenção do grau Bacharel em Ciências da Computação

Orientadora: Professora Doutora Patrícia Vilain

Banca Examinadora

Professor Doutor Ricardo Pereira e Silva

Professor Doutor Mario Dantas

Sumário

Lista de Tabelas

Lista de Figuras

Resumo

1	Introdução	p. 10
1.1	Justificativa	p. 12
1.2	Objetivos	p. 14
1.2.1	Objetivos Gerais	p. 14
1.2.2	Objetivos específicos	p. 14
1.3	Metodologia	p. 15
2	Revisão Bibliográfica I - Computação Ubíqua	p. 16
2.1	Requisitos da Computação Ubíqua	p. 16
2.2	Exemplos de Serviços Sensíveis ao Contexto	p. 18
2.3	Mensageiro Instantâneo	p. 18
2.4	Agenda	p. 19
3	Revisão Bibliográfica II - Modelagem: Trabalhos Relacionados	p. 21
3.1	ContextUML	p. 21
3.1.1	Modelagem	p. 21
3.1.1.1	Contexto	p. 22
3.1.1.2	Fonte	p. 22

3.1.1.3	Objeto Sensível ao Contexto	p. 23
3.1.1.4	Modelagem da Sensibilidade ao Contexto	p. 23
3.1.2	Exemplo	p. 23
3.2	<i>TangO</i>	p. 24
3.2.0.1	Avarias	p. 27
3.3	Identificação de objetivos	p. 27
3.3.1	Exemplo	p. 29
3.4	Escopos e Estratégias de Propagação de Exceções	p. 29
3.4.1	Exemplo	p. 30
3.5	Cenário de <i>e-Learning</i> Sensível ao Contexto	p. 31
3.5.1	Exemplo	p. 33
3.6	Modelando Aplicações de Computação Ubíquas	p. 33
3.6.1	Modelo baseado em serviços	p. 33
3.6.2	Modelo baseado em informação	p. 34
3.7	Comparação dos trabalhos estudados	p. 37
3.7.1	Comentários	p. 40
4	Proposta	p. 43
4.1	MARVIn	p. 43
4.2	Visão Estrutural/Espacial	p. 44
4.3	Visão Dinâmica	p. 47
4.4	Deployment	p. 49
4.5	Funcional	p. 49
4.6	Resumo	p. 52
4.7	Comparação com MARVIn	p. 54
5	Exemplo de uma aplicação em um ambiente	p. 57

5.1	Agenda	p. 57
5.2	Passo 1	p. 58
5.3	Passo 2	p. 59
5.4	Passo 3	p. 61
5.5	Passo 4	p. 62
5.6	Passo 5	p. 63
5.7	Passo 6	p. 65
5.8	Passo 7	p. 67
6	Conclusão e Trabalhos Futuros	p. 71
6.1	Trabalhos Futuros	p. 72
	Referências	p. 74

Lista de Tabelas

1	Tabela comparativa dos trabalhos referenciados	p. 38
2	Tabela comparativa final	p. 54

Lista de Figuras

1	Meta-modelo do ContextUML (18)	p. 22
2	Exemplo de serviço Mensageiro Instantâneo	p. 24
3	Representação de um objeto identificado no mundo físico, e sua representação como Objeto Tangível	p. 25
4	Modelo dos aspectos físicos do habitat no sistema (10)	p. 26
5	Mapa do ambiente com Serviço Mensageiro Instantâneo	p. 26
6	Exemplo de análise de metas (1)	p. 29
7	Contexto exige que exceção seja tratada pela brigada de incêndio . . .	p. 31
8	Estratégia propaga a exceção para equipe de manutenção	p. 31
9	Estratégia de propagação para um caso do Serviço Mensageiro	p. 32
10	Exemplo de atividade alterando contexto e de contexto alterando a atividade (7)	p. 33
11	Ações do Serviço Mensageiro Instantâneo e como interagem com diferentes contextos	p. 34
12	Níveis de aplicação e utilidade (9)	p. 35
13	Serviço Mensageiro	p. 35
14	Resposta do sistema a uma ação do usuário (9)	p. 36
15	Diagrama com links estereotipados e anotações (9)	p. 37
16	Metamodelo da proposta	p. 43
17	Utilização de mapa para indicar localização física	p. 45
18	Alteração no meta-modelo de ContextUML	p. 45
19	Operação fornecida por múltiplos serviços	p. 45

20	Contexto alterado por mudança no ambiente	p. 47
21	Hierarquia dos Objetos	p. 47
22	Exemplo de atividade alterando contexto e de contexto alterando a atividade (7)	p. 48
23	Marcação de tempo no diagrama. (9)	p. 48
24	Serviços e como acessá-los (9)	p. 49
25	Decomposição dos objetivos: Previsão do funcionamento do sistema frente a novos serviços.	p. 50
26	Identificação de Falhas e Estratégia de Propagação da Exceção.	p. 51
27	Identificação de Falhas e Estratégia de Propagação da Exceção no diagrama de classes.	p. 52
28	Agendamento de Compromisso	p. 58
29	Grafo dos objetivos da Agenda	p. 59
30	Identificação dos Objetos Tangíveis.	p. 61
31	Delimitação de área física.	p. 61
32	Contextos no diagrama de classes	p. 64
33	Serviços possíveis no ambiente agrupados em escopo	p. 65
34	Identificação de falhas no processo de agendamento e estratégia de propagação.	p. 65
35	Relacionamento entre Serviços, Operações e Contextos da Agenda.	p. 66
36	Falha causada por dados insuficientes ou demora na resposta.	p. 68
37	Outra situação em que é aplicada a mesma estratégia.	p. 68
38	Estratégia de Propagação	p. 69
39	Tempo limite para resposta.	p. 70

Resumo

Quando diferentes dispositivos e aplicativos começam a funcionar em conjunto, aspectos físicos e comportamentais no ambiente devem ser levados em conta devido à integração entre o mundo físico e o virtual. Não há como listar todos dispositivos e aplicativos e novas características e comportamentos podem surgir devido à autonomia das partes do sistema, diferentes interpretações dos dados para derivar informações de contexto e capacidade de auto-organização.

As ferramentas de modelagem em uso atualmente são deficientes para estes cenários pois não tratam de aspectos como localização física ou a análise de possíveis comportamentos emergentes resultantes de alterações no ambiente. Além dos aspectos da computação móvel deve-se considerar aspectos do ambiente (computacional e também físico) em que a computação está inserida, que em um primeiro momento podem não parecerem ter relevância para o funcionamento da aplicação, mas que podem determinar o resultado final da operação de um serviço, o que justifica uma necessidade de integrar os aspectos da computação tradicional, da computação móvel e as características ambientais.

A adaptação de recursos de modelagem já existentes e adição de outros pode diminuir essas deficiências, quando não resolvê-las totalmente. Este trabalho apresenta propostas de mecanismos para trabalhar com sensibilidade a contextos, localização física, análise de metas e comportamentos emergentes e tratamento de falhas no funcionamento esperado das aplicações que executam em ambientes, mecanismos estes, que se utilizados em conjunto podem contribuir para uma melhor modelagem de um ambiente de computação ubíqua.

O envolvimento de diversas áreas de pesquisa envolvidas no desenvolvimento e aplicação da computação ubíqua significa que não é possível, em um único trabalho, abordar todos tópicos relevantes. Assim, este trabalho fornece somente uma visão superficial de um todo muito maior que pode ser explorado.

1 *Introdução*

O conceito de computação ubíqua surgiu com Mark Weiser em 1988 (20), baseado em seus trabalhos de pesquisa sobre interação homem-máquina, realizados na Palo Alto Research Center (PARC), subsidiária da Xerox. De acordo com Weiser, uma boa ferramenta é uma ferramenta invisível, que não interfira na consciência de quem a está utilizando, mantendo o foco na tarefa e não na ferramenta. Como exemplo de boa ferramenta, ele cita os óculos - enxerga-se o mundo e não os óculos (20). Assim sendo, computação ubíqua é a computação que se torna invisível ao integrar-se com o ambiente.

Uma tecnologia que se tornou invisível no dia-a-dia, seguindo o conceito de boa ferramenta de Weiser, é a escrita. A representação de linguagem falada com símbolos gráficos se faz presente desde livros e revistas até placas e sinais, tendo presença constante no mundo, sem necessidade de focar a atenção na mesma para que se a utilize.

O conceito de computação ubíqua pode se confundir com o conceito de computação pervasiva. Em algumas situações os termos podem ter até o mesmo significado, porém, para este trabalho, o conceito de computação ubíqua implica na existência de computação pervasiva, mas não o contrário.

Enquanto a computação ubíqua tem como objetivo prover informação em qualquer lugar, a qualquer momento, com dispositivos conversando entre si, com o intuito de simplificar as tarefas dos usuários, a computação pervasiva foca na utilização da tecnologia disponível através do uso de dispositivos diversos, mas independentes entre si (13). Hoje vivemos em um mundo onde a computação é pervasiva, temos computadores nos carros, aviões, eletrodomésticos, celulares e muitos outros lugares, porém eles funcionam independentemente, ou seja, não cooperam entre si. Apesar de poder parecer uma realidade distante, a integração da tecnologia no ambiente em que as pessoas convivem diariamente, é uma realidade, como mostrado por Monique (16).

Para que a computação ubíqua se torne possível, o computador deve ser transportado de seu mundo muitas vezes isolado e deve ser integrado no mundo real, desde o início

do desenvolvimento de suas aplicações. As aplicações devem ser capazes de utilizar o contexto do ambiente em que se localiza para se adaptarem às condições do mesmo.

O contexto é definido de acordo com as informações que caracterizam uma determinada situação e podem alterar o resultado de uma ação, dependendo de seu valor.

Uma das definições mais comuns de contexto é de Dey e Aboud: "...qualquer informação que pode ser usada para caracterizar a situação de uma entidade. Uma entidade é uma pessoa, local ou objeto, que é relevante para a interação entre um usuário e uma aplicação, incluindo o próprio usuário e aplicação"(8).

Outra definição é a de Satyanarayanan, que diz que o contexto de um usuário de uma aplicação sensível ao contexto consiste de atributos como localização física, estado fisiológico, estado emocional, histórico pessoal, padrões diários de comportamento, entre outros, que se fornecidos para um assistente humano, podem ser usados para tomada de decisões sem necessidade de interromper o usuário a todo momento (17).

Não é difícil perceber, que não se pode enumerar os aspectos importantes em todas as situações, invalidando definições específicas de contexto (8).

As informações para caracterizar um contexto podem vir do mundo físico como do mundo virtual, que algumas vezes podem se misturar. A interação com o mundo físico, principalmente com pessoas, aumenta ainda mais a complexidade e o dinamismo do ambiente. Ambientes de computação ubíqua são ambientes complexos e heterogêneos, que oferecem grande conectividade entre serviços e interação com o mundo físico e em que a conectividade entre serviços sensíveis ao contexto cria um ambiente dinâmico em que as aplicações não têm um comportamento determinístico mas sim dependente do contexto que se apresenta.

Pessoas não percebem ambientes físicos (escritório, piso de uma loja, estádio) e virtuais (desktop de um computador, funcionalidades de um telefone celular) como entidades completamente separadas, já que objetos e processos comumente têm representações nos dois mundos. Assim, se faz necessário identificar construções capazes de representar elementos tanto do mundo real, tanto objetos físicos como conceitos, como do mundo virtual, da maneira mais genérica possível e que possibilite projetar ambientes que dêem um melhor suporte às atividades física e virtuais relacionadas (15).

1.1 Justificativa

Existem diversas aplicações sensíveis ao contexto, porém, de maneira genérica, estas são sensíveis a contextos que:

- Não se alteram no futuro (contexto relativo a informações do passado ou presente)
- Acabaram de mudar (contexto relativo a informações do presente).

Estas aplicações não são sensíveis ao contexto que será relevante no futuro, pois ignoram a possibilidade de mudanças no mesmo (2).

Segundo Ingstrup (10), existe um sentimento intuitivo de os métodos de análise e projeto e ferramentas de modelagem utilizadas tradicionalmente serem insuficientes para a construção de sistemas pervasivos, por exemplo, elas não apresentam meios para tratar da localização física, entre outras insuficiências.

Para que as aplicações no ambiente sejam capazes de se adaptarem e preverem as necessidades de seus ocupantes - contexto futuro - sem necessidade de constante interrupção do usuário, é preciso mais do que somente a presença de todas informações necessárias para a definição de um contexto.

Cada aplicação, executando distribuída no ambiente ou embarcada em dispositivos, vai depender de um conjunto particular de informações para o seu funcionamento, cada uma com um grau de importância e precisão também particular.

Assim as mesmas construções capazes de representar de maneira genérica os elementos do mundo real e virtual, devem ser capazes de quantificar e qualificar essas informações, em atributos e medidas que vão depender de cada caso - não se pode enumerar os aspectos importantes em todas as situações (15).

Da maneira como acontece o desenvolvimento atualmente, desenvolvedores de aplicações sensíveis ao contexto devem projetar, devido a falta de modelos adequados, tudo o que é relacionado ao gerenciamento de contexto, incluindo sua coleta, disseminação e uso em uma maneira *ad-hoc*, o que resulta em um desenvolvimento trabalhoso e demorado (18).

Um modelo capaz de modelar de maneira clara o conjunto mais genérico de informações de contexto e que ofereça notações para facilitar a representação de características de contextos específicos, pode contribuir com a redução no tempo de desenvolvimento e

diminuir as chances de que certas informações sejam negligenciadas, o que resulta em uma melhor correspondência entre projeto e resultado final da implementação.

1.2 Objetivos

1.2.1 Objetivos Gerais

O objetivo deste trabalho é fazer um levantamento de informações (tanto do mundo físico como virtual) relevantes às implementações de ambientes de computação ubíquos, que ao serem ignoradas ou dadas pouca importância na modelagem do ambiente, podem resultar em um funcionamento insatisfatório dos serviços no ambiente, ou com qualidade menor do que seria possível.

A partir destas informações, deve ser feita uma composição de construções de outros modelos que representem, satisfatoriamente, soluções que atendam as necessidades dos sistemas pervasivos.

1.2.2 Objetivos específicos

Para alcançar os objetivos gerais definidos para o trabalho, os seguintes objetivos específicos foram listados:

- Levantamento de requisitos necessários para o desenvolvimento de ambientes de computação ubíqua e para a implementação de aplicações para o mesmo.
- Identificação de informações que não são adequadamente tratadas no desenvolvimento tradicional e que são essenciais para representação adequada de contextos e conseqüentemente, para o funcionamento dos ambientes de computação ubíqua.
- Estudo de propostas para modelagem que se propõem a atender as necessidades do desenvolvimento de ambientes de computação ubíqua, através de construções capazes de modelar de maneira eficiente as informações necessárias, como informações de localização, preferências, etc.
- Aplicação dos modelos aos problemas levantados: Aplicação de partes ou todo, dos diferentes modelos identificados, para verificar suas eficiências e eficácias, para modelar o conjunto de informações identificadas como inadequadamente tratadas nos modelos habituais.
- Composição de um modelo, ou proposta de um novo, utilizando como referência os trabalhos estudados, que seja capaz de representar o conjunto mais genérico possível de informações de contexto propostas.

- Verificação da eficiência do modelo proposto, através de sua aplicação na modelagem dos mesmos problemas utilizados para identificar as falhas e pontos fracos dos modelos tradicionais.

1.3 Metodologia

O trabalho foi realizado e está organizado da seguinte maneira:

No capítulo 2.1 - Requisitos da Computação Ubíqua, requisitos que os modelos devem atender são listados e explicados.

O capítulo 2.2 apresenta exemplos de aplicações que são utilizados no decorrer do trabalho.

Foram buscados trabalhos relativos à modelagem de aplicações de computação ubíqua, sensibilidade a contexto, programação orientada à agentes e integração físico-digital. Os trabalhos estudados são apresentados no capítulo 3 Trabalhos Relacionados.

A contribuição de cada trabalho para atender os requisitos do capítulo 2.1 é comparada em uma tabela ao final do capítulo 3.

Após a análise dos trabalhos referenciados, um modelo é proposto reunindo contribuições de todas as fontes. A proposta é apresentada no capítulo 4 e é utilizada pra modelar uma aplicação no capítulo 5 - Exemplo de uma aplicação em um ambiente.

O trabalho termina com as conclusões e sugestões de trabalhos futuros relacionados a este, que podem vir a ser desenvolvidos.

2 Revisão Bibliográfica I - Computação Ubíqua

Neste capítulo são apresentados requisitos identificados em trabalhos relacionados, que devem ser atendidos para o desenvolvimento de aplicações de computação ubíqua e também são apresentados exemplos de aplicações que são utilizados no decorrer deste trabalho.

2.1 Requisitos da Computação Ubíqua

De acordo com Johanson (12), alguns requisitos devem ser cumpridos para que um sistema seja ubíquo, como extensibilidade, desacoplamento referencial e temporal (serviços devem poder funcionar de maneira assíncrona), percepção de instantaneidade (tempo de resposta ao usuário deve parecer instantâneo), tolerância a falhas e capacidade de recuperação e escalabilidade.

Outros tipos de requisitos, com características bem diferentes, referentes à modelagem e não à aplicação, devem ser cumpridos para que os requisitos (e seu cumprimentos) da aplicação citados anteriormente possam ser verificados durante o desenvolvimento das aplicações.

- **Identificação dos objetos físico-digitais:** Sistemas ubíquos devem interagir com aplicativos, objetos no mundo físico e pessoas. Elementos de um sistema devem ser identificados e representados corretamente no universo onde estão inseridos, seja somente no mundo físico, digital, conceitual, ou mesmo em todos estes ao mesmo tempo, de forma a manter todas as informações de ambos universos (físico e digital) centralizadas em um mesmo modelo (19)(14)(10)(15).
- **Localização:** Objetos com capacidade computacional estão dispersos pelo mundo, logo a localização dos mesmos dentro dos limites de um ambiente é um dado impor-

tante (10)(19). A proximidade de uma pessoa com um objeto pode, por exemplo, indicar a intenção da pessoa de utilizá-lo.

- Tratamento de informações: Diferentes serviços podem utilizar dados coletados das mesmas fontes de maneira diferente, assim como diferentes fontes podem prover dados relativos a um determinado contexto de maneira diferente. A entrada e saída das aplicações (tipo de dado, se é alimentada de maneira contínua ou discreta, tipo de saída, etc) devem ser facilmente identificadas, com o benefício de facilitar a integração de novas aplicações em um ambiente, sejam elas partes para prover uma funcionalidade ou provedores completos da mesma (9).
- Serviços: Um ambiente de computação ubíqua pode receber e liberar diversos serviços diferentes, dos quais o ambiente não necessita para manter-se funcionando, mas os quais pode utilizar se presentes. É útil modelar o ambiente com diferentes serviços, independentes entre si e que seja fácil de verificar como se relacionam com outros serviços. Também deve-se planejar as conseqüências da indisponibilidade de algum serviço essencial para o funcionamento esperado do ambiente.
- Alteração do funcionamento em função do contexto: A alteração no contexto em que um componente de um ambiente (uma pessoa ou uma aplicação) executa uma tarefa altera os resultados das ações realizadas, ou desencadeia outras ações. É importante identificar qual contexto (qual a informação responsável, como uma localização, valor de temperatura ou iluminação em um ambiente) pode causar uma determinada alteração nas tarefas realizadas (7).
- Alteração do contexto em função do funcionamento: Ações realizadas no ambiente, sejam por pessoas, objetos ou aplicações, podem alterar um determinado contexto. A identificação das alterações do contexto em função do funcionamento é necessária para a visualização de como uma ação pode alterar o comportamento de outras aplicações ou pessoas, através do item anterior (7).
- Utilidade (funcionalidade de um serviço) e Serviço (Meio de acesso à funcionalidade): Devem ser identificadas efetivamente as funcionalidades do ambiente e os objetos que fornecem a funcionalidade. Um mesmo objeto pode servir de acesso para diversas funcionalidades e uma mesma funcionalidade pode ser acessada por meio de diversos serviços(9).
- Identificação de comportamento emergente: Conforme novos serviços são inseridos em um ambiente, as possibilidades de combinação entre as interações das funciona-

lidades disponíveis são muitas, o que impossibilita uma análise exaustiva do sistema para identificar como será a alteração no ambiente quando novos serviços são inseridos. Deve ser fácil identificar como que os componentes do sistema interferem no cumprimento das metas esperadas das aplicações (1)(4), para identificar qual a consequência dos mesmos no funcionamento do sistema.

2.2 Exemplos de Serviços Sensíveis ao Contexto

Aqui são apresentados alguns serviços e contextos envolvidos, que podem fazer parte de algum ambiente ubíquo, utilizados como exemplo no decorrer do trabalho.

2.3 Mensageiro Instantâneo

O objetivo é fornecer um serviço de mensagens instantâneas internas entre os usuários de um ambiente, como um escritório ou laboratório.

Os usuários do sistema (funcionalidade oferecida em um ambiente) têm liberdade para se movimentar pelo ambiente sem se preocupar com o recebimento das mensagens. As mensagens são direcionadas automaticamente para o equipamento mais próximo do usuário que seja capaz de mostrá-la.

O serviço é resultado da operação conjunta de um conjunto de outros serviços mais simples, que são os seguintes:

- **Cliente Mensageiro:** Serve de interface entre o usuário e o serviço mensageiro. Realiza saída de dados (mostra mensagens e listas de usuários conectados e não conectados no ambiente) e entrada de dados (usuário digita mensagem para outro usuário).
- **Servidor Mensageiro:** Recebe todas mensagens enviadas e as encaminha para o destinatário. Também armazena mensagens com destinatários que não estão no ambiente até que o mesmo se conecte e elas possam ser entregues.
- **Localizador:** Fornece a localização de um determinado usuário no ambiente.
- **Autenticador:** Verifica usuário e senha e autoriza a utilização dos serviços do ambiente.

- **Entrada:**É a primeira interface do usuário com o ambiente. É o serviço pelo qual o usuário pede que seja liberado seu acesso. Quando o acesso é liberado ou negado, a entrada dá a resposta correspondente ao usuário, abrindo a porta ou recusando o pedido.

2.4 Agenda

Este serviço deve se encarregar do gerenciamento da carga horária, reuniões, compromissos e outras atividades em um grupo de trabalho, tirando a parte da carga burocrática das atividades (como agendamento de reuniões e conferência de disponibilidade de horários).

O sistema deve trabalhar com as seguintes condições:

- Existe a necessidade de acompanhamento contínuo do desenvolvimento dos projetos por diferentes equipes internas.

Encontros futuros dos grupos precisam ser agendados de acordo com a disponibilidade de todos os envolvidos e reuniões realizadas sem planejamento prévio devem ser registradas.

- A carga horária de atividades realizada pelos bolsistas deve ser de X horas semanais porém com flexibilidade de horário, o que dificulta o acompanhamento.
- Membros do grupo podem tentar agendar compromissos pessoais ou compromissos coletivos. Compromissos coletivos, quando agendados, devem esperar a confirmação de todos envolvidos. Compromissos pessoais não podem conflitar com compromissos do grupo.
- O grupo pode consultar os compromissos do grupo, mas não pode consultar os compromissos pessoais agendados por outra pessoa.

O sistema deve ser capaz de, através das funcionalidades dos serviços que o compõem, realizar todas suas atividades da maneira menos invasiva possível. Se houver um serviço executando que é capaz de atribuir prioridades aos diversos tipos possíveis de compromissos e se existirem, por exemplo, duas atividades agendadas para o mesmo horário, o sistema de agendamento deve ser capaz de decidir qual evento não agendar, ou se deve remarcar algum horário, sem a necessidade de intervenção humana.

Além de fornecer uma funcionalidade, esta funcionalidade deve aparentar ter comportamento inteligente.

3 Revisão Bibliográfica II - Modelagem: Trabalhos Relacionados

Existem diversos trabalhos que podem trazer contribuições para a modelagem de aplicações de computação ubíqua e em alguns casos foram desenvolvidos especificamente para este propósito.

Alguns trabalhos são apresentados e comparados neste capítulo.

3.1 ContextUML

Sheng e Benatallah (18) afirmam que há uma falta de abordagens específicas para a formalização do desenvolvimento de aplicações sensíveis ao contexto e conseqüentemente este desenvolvimento é trabalhoso e consome muito tempo. Para resolver este problema, eles apresentam ContextUML, uma linguagem de modelagem baseada em UML para a formalização do desenvolvimento de aplicações sensíveis ao contexto.

Os desenvolvedores devem identificar o tipo de informação necessária para as funcionalidades a serem implementadas e como derivá-la. Podem existir diversos provedores de contexto fornecendo o mesmo tipo de informação, assim como pode acontecer de não existir nenhum provedor de contexto para o tipo de informação que se procura.

3.1.1 Modelagem

A modelagem através do modelo ContextUML identifica diferentes elementos de uma aplicação sensível ao contexto e trata separadamente cada um deles (18). O meta-modelo é apresentado na figura 1.

É importante destacar que no trabalho de Sheng e Benatallah, diferente do que é

apresentado na figura 1, cada mensagem pode pertencer somente a uma operação, ou seja, mais de uma operação não pode utilizar a mesma mensagem, o que no contexto de seu trabalho faz sentido, onde as operações funcionam individualmente. Em uma situação em que as operações podem uma atender às necessidades umas das outras, as mensagens podem pertencer a diversas operações.

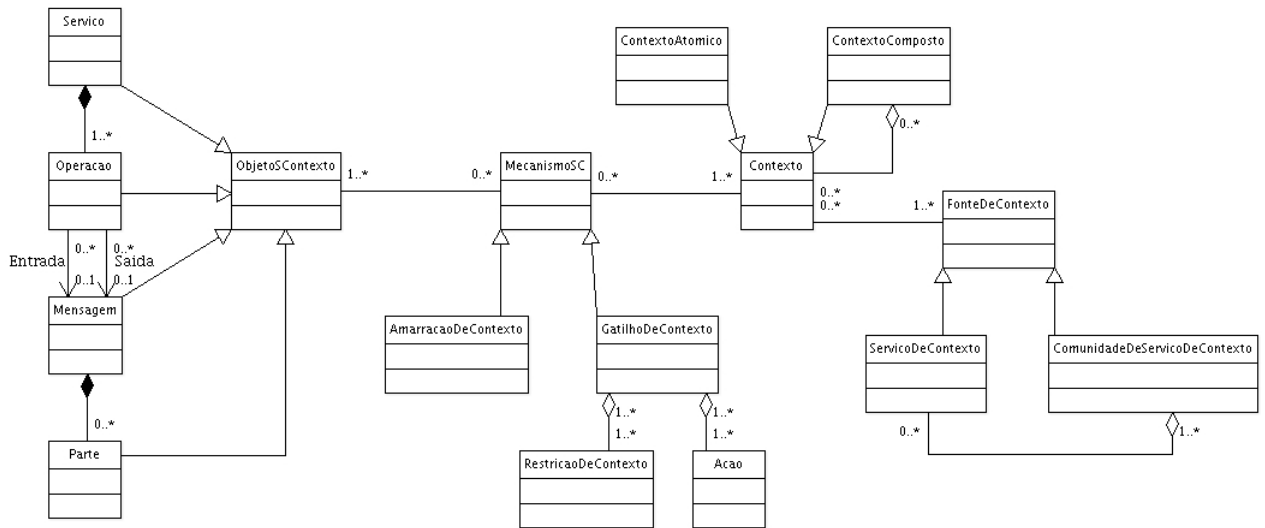


Figura 1: Meta-modelo do ContextUML (18)

3.1.1.1 Contexto

Contexto é uma classe que modela as informações de contexto e se especializa em ContextoAtomico e ContextoComposto (18). O primeiro provem diretamente de alguma fonte (ver 3.1.1.2). O último agrega múltiplos contextos, tanto atômicos como compostos e permite que uma grande variedade de contextos seja modelada.

3.1.1.2 Fonte

FonteDeContexto modela os recursos dos quais são derivadas as informações de contexto. As fontes também se especializam (18). Podem ser ServicosDeContexto nos quais o contexto é fornecido por um provedor de contexto, ou ComunidadeDeServicosDeContexto, que agrega múltiplos serviços que provêm contextos, com uma única interface, abstraíndo-os em uma única fonte.

3.1.1.3 Objeto Sensível ao Contexto

A classe ObjetoSContexto se especializa em Serviço, Operação, Mensagem e Parte (18). A entrada e saída das operações é feita por mensagens representadas por objetos Mensagem e Parte, que podem ser usados para identificar um parâmetro da mensagem. Um serviço pode possuir diversas Operações, mas uma operação é parte de apenas um Serviço.

3.1.1.4 Modelagem da Sensibilidade ao Contexto

Os mecanismos para tratar a sensibilidade ao contexto são modelados pela classe MecanismoSC, com especializações AmarracaoDeContexto e GatilhoDeContexto (18).

A primeira especialização modela a amarração do contexto com objetos sensíveis a um determinado contexto e a última modela a situação em que um serviço automaticamente se adapta baseado nas informações de contexto. Esses mecanismos são atribuídos a objetos do tipo ObjetoSC através de uma relação chamada AtribuicaoDeMecanismo (18).

3.1.2 Exemplo

Na figura 2 é apresentada a modelagem do serviço Mensageiro Instantâneo, com os diferentes serviços que o compõem e contextos envolvidos.

Uma solução amplamente aceita (por (11)(12)(5), entre outros) como meio para desenvolver um ambiente ubíquo, é a de sistemas baseados em eventos, em que ocorre a utilização de troca de mensagens para a comunicação entre serviços e para transmissão de resultados e parâmetros de operações realizadas pelos serviços. As diferentes classes definidas pelo meta-modelo mapeiam diretamente os diferentes elementos envolvidos (serviço, operação e mensagem).

As Mensagens identificam os diferentes tipos de eventos e as Operações o tratamento que é dado para os eventos.

Mesmo com um grande número de classes presentes no diagrama de classe, não há acoplamento entre os diversos Serviços. Quase todos os relacionamentos são definidos entre Operação e Mensagens, mensagens que podem ser utilizadas por Operações de diversos serviços. A mensagem é utilizada para especificar exatamente qual informação é utilizada pela operação, dentre as informações contidas em um determinado contexto.

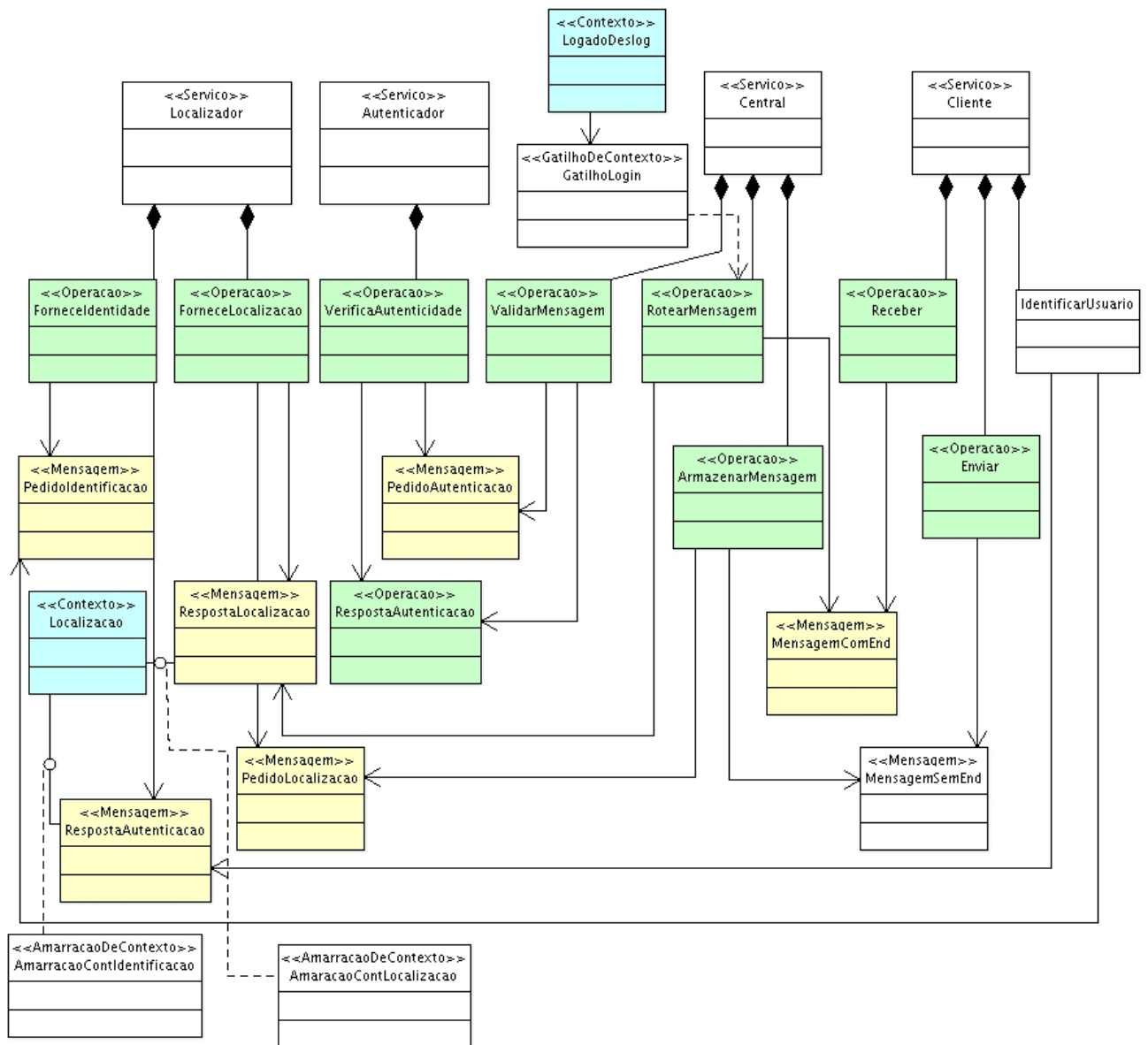


Figura 2: Exemplo de serviço Mensageiro Instantâneo

A relevância dos contextos no funcionamento do sistema é facilmente identificada através do relacionamento do contexto e do evento (Mensagem) que este altera, no caso de uma amarração com o contexto que causa alterações constantes no funcionamento, ou no caso do relacionamento do contexto com uma determinada Operação que tem seu funcionamento alterado em função de alguma condição atrelada ao contexto.

3.2 *TangO*

Nesta seção é apresentado o modelo TangO. Nele, Daniel May (14) propôs a utilização da idéia de Objetos Tangíveis.

Elementos de ambientes pervasivos que possuem elementos de mais de um espaço (físico, informativo e conceitual) são modelados através de Objetos tangíveis. Suas propriedades são compatíveis com sistemas multi-agentes, em que os agentes têm liberdade de ir e vir, também compatível com o comportamento humano (14). Um objeto pode estar presente em um ambiente mas não ser mais necessário para o funcionamento do sistema, porém continua no ambiente e outros objetos podem interagir com ele.

Essa condição é conflitante com a definição de um sistema como algo do qual uma entidade ou faz parte do sistema ou faz parte do ambiente (externo ao sistema), sem a possibilidade de um meio termo.

Na figura 3 é mostrado o exemplo de um projetor em uma sala de projeção, com funcionalidades para mostrar slides e se conectar, representando tanto seus aspectos físicos como virtuais. A classe do lado esquerdo é o objeto no mundo físico e a do lado direito, sua representação como objeto tangível.

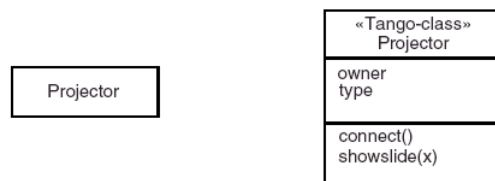


Figura 3: Representação de um objeto identificado no mundo físico, e sua representação como Objeto Tangível

Os Objetos Tangíveis existem em Habitats. Habitats são os contextos lógicos em que os objetos existem e interagem. O relacionamento entre os Objetos Tangíveis em um Habitat define uma Associação (14).

Habitats podem ser físicos e lógicos. Ambos tem sua existência limitada no tempo (quando foi criado até o momento que deixa de existir) e o físico tem limitações espaciais (como sua dimensão). Diversos Habitats podem existir e se relacionar em um mesmo nível, ou podem ser locais a outros, como uma loja dentro de um centro comercial. Conseqüentemente, Objetos Tangíveis podem habitar diversos Habitats ao mesmo tempo (14). Na figura 4 é mostrada a modelagem de um cenário em que um projetor pode exibir conteúdo de PDAs (*personal digital assistant*), tanto o projetor quanto os PDAs são apresentados como objetos tangíveis, identificados por pequenos círculos sobre suas representações no mapa.

Na figura 5 é mostrado o ambiente onde será executado o serviço Mensageiro Instan-

tâneo. São identificadas as aplicações Cliente (nos *Desktops* e no PDA) que têm como informação de contexto importante, sua localização, assim como o usuário.

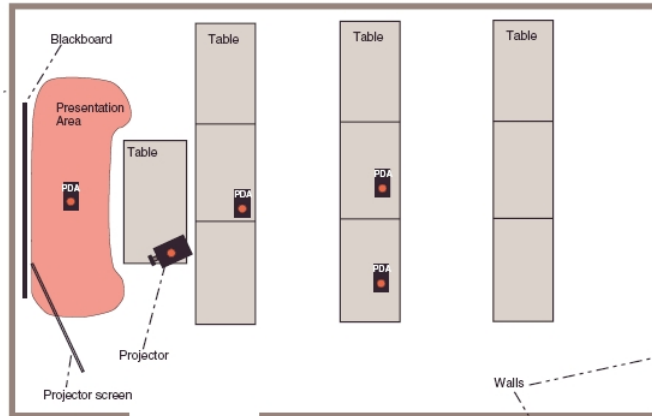


Figura 4: Modelo dos aspectos físicos do habitat no sistema (10)

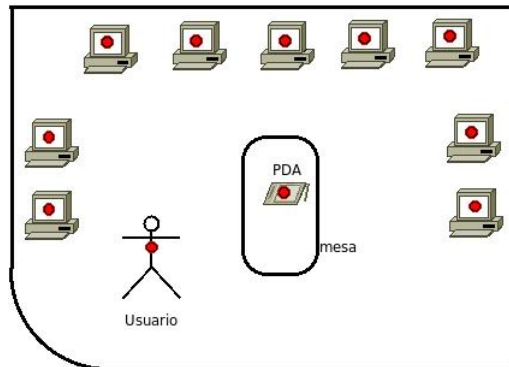


Figura 5: Mapa do ambiente com Serviço Mensageiro Instantâneo

A Associação define a dependência de um Objeto Tangível com um determinado contexto. Elas podem ser temporárias (um cliente de passagem por uma loja, se relaciona com o ambiente onde está, porém logo deixará o local) ou permanentes (14).

Um conceito parecido com o de Objetos Tangíveis é o de entidades pivotais, apresentado por Pederson. Objetos que possuem função física-digital e são o foco da atenção de uma atividade, são considerados entidades pivotais, enquanto todos os outros objetos que seriam identificados em TangO como objetos tangíveis, são considerados parte do ambiente (19) .

Outros trabalhos foram feitos dando continuidade à proposta de TangO, dentre eles o trabalho de Ingstrup(10) que contribuiu com o conceito de *Avaria*, apresentado na próxima seção.

3.2.0.1 Avarias

A idéia de avaria corresponde à idéia de Weiser de que uma boa ferramenta é uma ferramenta invisível, que não interfira na consciência de quem a está utilizando, mantendo o foco na tarefa e não na ferramenta (20).

No lugar de um mundo povoado por objetos (mundo físico), os objetos somente passam a existir para as pessoas quando uma avaria acontece (10).

Uma avaria ocorre quando um objeto que normalmente é aceito sem que seja necessário seu reconhecimento ou identificação explícita em um determinado contexto, passa a ser percebido, devido a sua ausência ou seu funcionamento não correspondente com o esperado (10). Por exemplo, quando uma pessoa que trabalhe em um escritório utiliza uma caneta como ferramenta para escrever freqüentemente esta está quase sempre presente. O objetivo da pessoa não é utilizar a caneta, mas fazer anotações. Assim ela só irá pensar na caneta no momento em que for fazer a anotação e não conseguir encontrá-la, ou acabar a tinta por exemplo.

Um ambiente de computação ubíqua deve evitar que a atenção da tarefa seja transferida para a ferramenta utilizada para a realização da mesma. Um avaria tira a atenção da pessoa da atividade que ela realiza e a transfere para a ferramenta, mais especificamente, para o problema com a ferramenta, portanto, uma atividade de desenvolvimento de um sistema deve identificar junto aos casos de uso, as principais distrações que podem acontecer (10).

Como as distrações vão atuar sobre o usuário final, a participação do mesmo na modelagem é vital. Se o sistema for modelado com uma correspondência direta entre o comportamento nas ações humanas, é mais fácil garantir a adaptação e antecipação (10).

3.3 Identificação de objetivos

Esta seção apresenta uma proposta de desenvolvimento de sistemas orientados a agentes que utiliza modelagem orientada a objetivos, apresentado em (1).

De acordo com o trabalho, a utilização de agentes apresenta um paradigma social, onde os agentes, independentes, podem ou devem trabalhar em conjunto para alcançar resultados (1).

"Conhecimento especializado não é freqüentemente disponível em um único agente

(onisciência). O conhecimento que é distribuído em várias fontes (agentes) pode ser integrado para uma maior completude quando for necessário, motivado pelo aumento da complexidade: complexos, distribuídos, dinâmicos, imprevisíveis" (1).

Através da utilização de agentes, é possível se obter comportamento emergente utilizando-se uma arquitetura estática. A dificuldade de desenvolver um sistema está em manter um equilíbrio entre o comportamento pró-ativo (os agentes atuam por conta própria) e comportamento reativo (os agentes atuam sob um estímulo). Nem sempre o comportamento resultante da composição de agentes pode ser compreendido em função somente do comportamento dos agentes individualmente (1).

Os sistemas falham em atender aos interesses das organizações de que fazem partes devido à ausência de compreensão da organização pelos projetistas e pela incapacidade de acomodar todas as mudanças organizacionais que ocorrem, o que promove a engenharia de requisitos como uma fase crítica no desenvolvimento de sistemas devido a necessidade de balancear os fatores técnicos e os sociais/organizacionais (1).

A utilização do *framework* NFR (4) é apresentada como solução que, ao relacionar as diferentes propriedades dos agentes, facilita a identificação de comportamentos emergentes e fornece uma visão da colaboração proveniente das características dos agentes sobre o resultado final do sistema e facilita o balanceamento dos fatores técnicos e organizacionais (1).

Os objetivos do framework NFR é representar requisitos não funcionais específicos, relacionar as decisões de projeto aos requisitos não funcionais, justificar decisões, auxiliar a detecção de defeitos e considerar as vantagens e desvantagens decorrentes de diferentes aspectos do projeto. O framework trata os requisitos não funcionais como metas em conflito ou em sinergia para alcançar um objetivo (4).

O framework utiliza uma estrutura de grafo e utiliza a noção de satisfação das metas, o que significa que as decisões de projeto geralmente apenas contribuem parcialmente para alcançar a meta, assim o software resultante irá satisfazer os requisitos não funcionais dentro de uma margem, mas não irá atendê-los em absoluto (4).

São identificadas metas (*hard-goals*) e metas flexíveis (*soft-goals*). Os agentes são os donos das metas. As metas podem ser atribuídas a agentes já identificados ou agentes podem ser atribuídos a metas (1). O objetivo da identificação das metas e agentes não é a identificação de como fazer, mas sim de porque fazer.

Na elaboração das metas são utilizadas questões:

- Por que para a exploração de metas altas, de contexto,
- Como para a exploração de operações,
- De que outra maneira para a exploração de alternativas.

As metas podem ajudar, prejudicar, realizar ou quebrar outras metas, condições respectivamente representadas através por +, -, ++ e - - e têm precedências entre si.

3.3.1 Exemplo

As metas são avaliadas para determinar obstruções (quando não podem ser alcançadas) e identificar as características que contribuem para a satisfação dos *soft-goals*. Na figura 6 são mostrados objetivos para o Sistema Mensageiro, que tem meta de ser preciso (a mensagem deve ser enviada para próximo do destinatário), barato, entre outras.

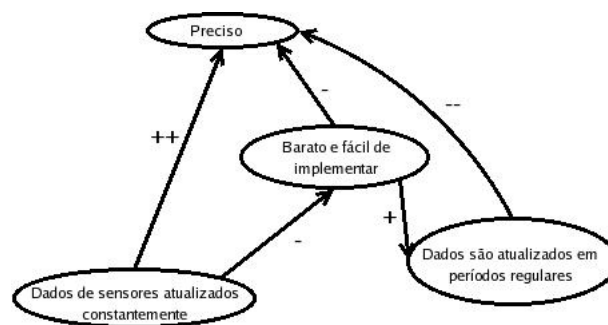


Figura 6: Exemplo de análise de metas (1)

A atualização dos dados dos sensores em intervalos periódicos de tempo (não contínuo), contribui para fazer o sistema barato, porém impossibilita a precisão, como mostrado na figura 6

3.4 Escopos e Estratégias de Propagação de Exceções

Em geral as aplicações utilizam mecanismos de tratamento de exceção fornecidos pelas linguagens em que são implementadas. Essa solução não é adequada para aplicações sensíveis ao contexto (6).

A caracterização de uma exceção depende do contexto, já que um mesmo estado do sistema pode significar uma exceção em uma determinada situação e não em outra.

Desse modo, a propagação de exceções nas aplicações sensíveis ao contexto deve levar em consideração as mudanças de contexto (6).

Um contexto que define a ocorrência de uma exceção é um contexto excepcional. "O objetivo de um contexto excepcional é facilitar a definição de situações excepcionais, pois a ocorrência de um contexto excepcional está diretamente relacionada à ocorrência de uma exceção" (6).

Diferentes tratadores podem ser necessários para tratar uma exceção, dependendo do contexto excepcional. Escopos são utilizados para a identificação de quais tratadores devem ser acionados em cada caso. A aplicação desenvolvida em (6) depende da localização física dos usuários, define os seguintes escopos: grupos de dispositivos, dispositivo, servidor e região. Quando ocorre uma exceção, uma busca sensível ao contexto por um tratador é realizada, seguindo uma ordem crescente dos escopos. Primeiro a busca é feita por dispositivo, caso nenhum tratador seja encontrado, a busca parte para outros grupos de dispositivos na mesma região, outras regiões no mesmo servidor e finalmente no escopo de servidor (6).

O trabalho de Pederson, também propõem a utilização de escopos e acrescenta a idéia de classificar as ações, utilizando a idéia de hierarquias de contenção, em intra ou extra-manipulação. Intra-manipulação indica que a ação terá conseqüências em como os objetos se relacionam nos níveis abaixo do nível do objeto de referência (como uma loja dentro de um centro comercial), e extra, tem conseqüências sobre os objetos no mesmo nível da referência (15).

Além da busca por escopo, podem ser definidas estratégias de propagação de erro para alertar os tratadores (6). No exemplo apresentado no trabalho, um parque de diversões, um incêndio próximo a uma atração do parque deve alertar a brigada de incêndio - situação como a mostrada na figura 7 - porém, é provável que devido a proximidade, um incêndio cause danos nas estruturas das atrações, assim a equipe de manutenção também deve ser alertada - como é mostrado na figura 8.

3.4.1 Exemplo

No Serviço Mensageiro, quando uma mensagem é enviada, o sensor do serviço que fornece a localização dos destinatários pode apresentar um problema. Uma estratégia pode ser definida, que pode por exemplo, notificar outros serviços que tem a mesma capacidade de fornecer a localização de usuários para atender ao pedido de localização

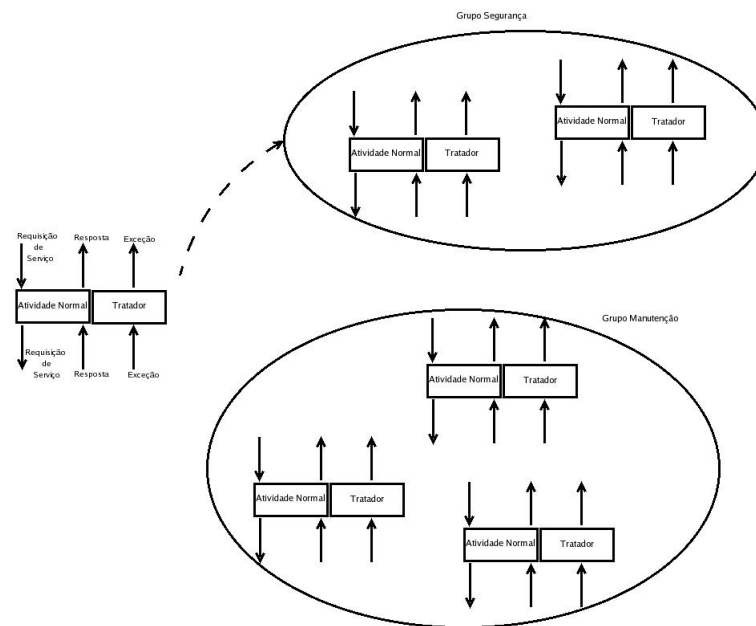


Figura 7: Contexto exige que exceção seja tratada pela brigada de incêndio

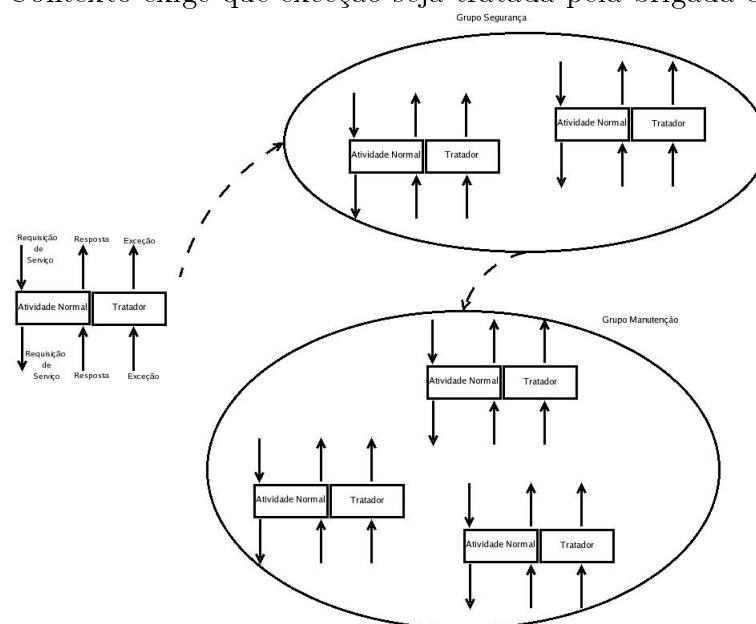


Figura 8: Estratégia propaga a exceção para equipe de manutenção

realizado e notifica também, uma equipe de manutenção próxima ao sensor para solicitar seu conserto.

3.5 Cenário de *e-Learning* Sensível ao Contexto

Esse trabalho é voltado para o desenvolvimento de um ambiente sensível ao contexto específico para cenários de aprendizagem, porém o mesmo pode ser generalizado para

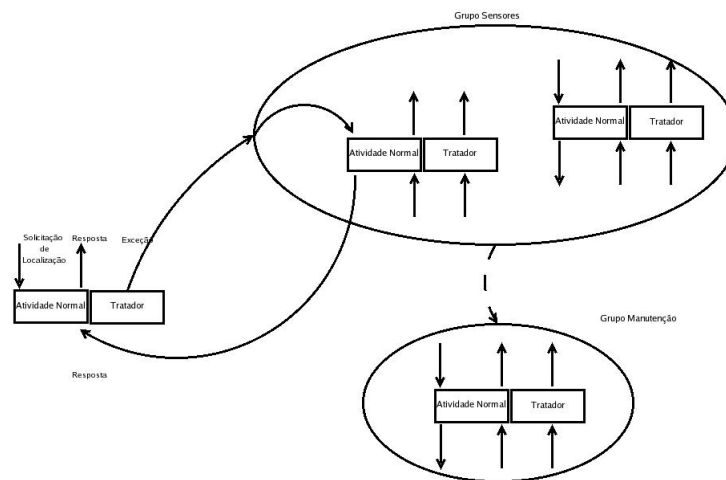


Figura 9: Estratégia de propagação para um caso do Serviço Mensageiro

desenvolvimento de ambientes sensíveis ao contexto em geral. O aspecto principal do projeto é capturar as seqüências de atividades usando diagramas de atividade UML.

Durante as atividades os diversos serviços se relacionam com diferentes contextos. Contextos não partilham uma mesma estrutura comum entre diferentes aplicações, assim é necessária uma divisão (7):

- Atributos Universalmente Aplicáveis: Data, hora, localização, etc.
- Atributos Específicos da Aplicação: Devem ser definidos como requisitos do domínio da aplicação.

O contexto em ambientes reais é implícito e concorrente, já que podem ser derivados diferentes contextos para diversas situações possíveis, o que gera problemas na modelagem de um contexto relevante. Somente alterações relevantes de contexto e as influências do contexto no fluxo da atividades são modelados (7).

São diferenciados três casos do relacionamento entre serviços e contexto, mostrados na figura 10 e explicados abaixo:

- Atividade altera o Contexto: Uma relação de dependência é estabelecida da atividade para o objeto de contexto.
- Contexto altera Atividade: A atividade é conduzida com variações conseqüentes dos atributos de algum objeto de contexto. Uma relação de dependência é estabelecida do contexto para a atividade.

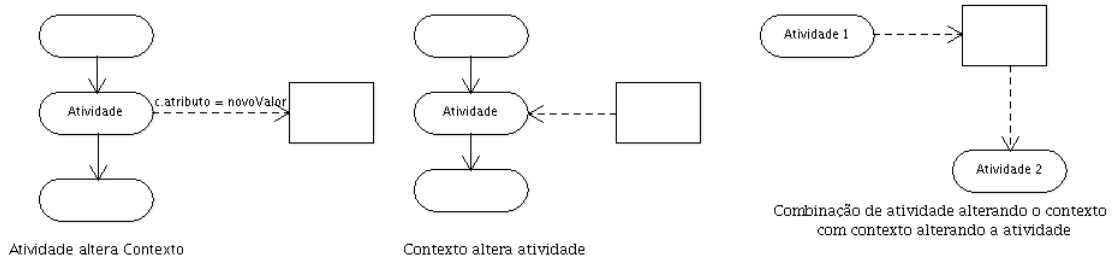


Figura 10: Exemplo de atividade alterando contexto e de contexto alterando a atividade (7)

- Contexto como Guarda: A transição entre nodos do diagrama de atividades é guardada por algum atributo de um objeto de contexto.

3.5.1 Exemplo

Na figura 11 é mostrada a modelagem do Serviço Mensageiro. Os contextos envolvidos são informações sobre usuários online, tipo de mensagens e interface gráfica necessária para exibir a mensagem e localização.

3.6 Modelando Aplicações de Computação Ubíquas

Em (9), o autor apresenta diferentes abordagens possíveis para modelagem de aplicações de computação ubíqua, que podem ser usadas em conjunto. Destas duas são apresentadas a seguir, baseada em serviços e na informação.

3.6.1 Modelo baseado em serviços

É uma visão da interação entre atores ou entidades em termos de serviços. A distinção entre serviços e a funcionalidade fornecida por eles, é feita pela ubiqüidade e pela invisibilidade. A funcionalidade deve ser acessado através de um utensílio (9), como pode ser visto na figura 12.

Na figura 13 as utilidades do Serviço Mensageiro (o serviço de receber e enviar mensagens provido pela aplicação Cliente e o serviço roteamento das mensagens provido pela aplicação servidor Central) são separadas das formas de acesso às mesmas. O cliente pode ser acessado por dispositivos diversos (computadores, celulares, *PDA*s, etc), enquanto as funcionalidades da Central não podem ser acessadas diretamente pelos usuários da apli-

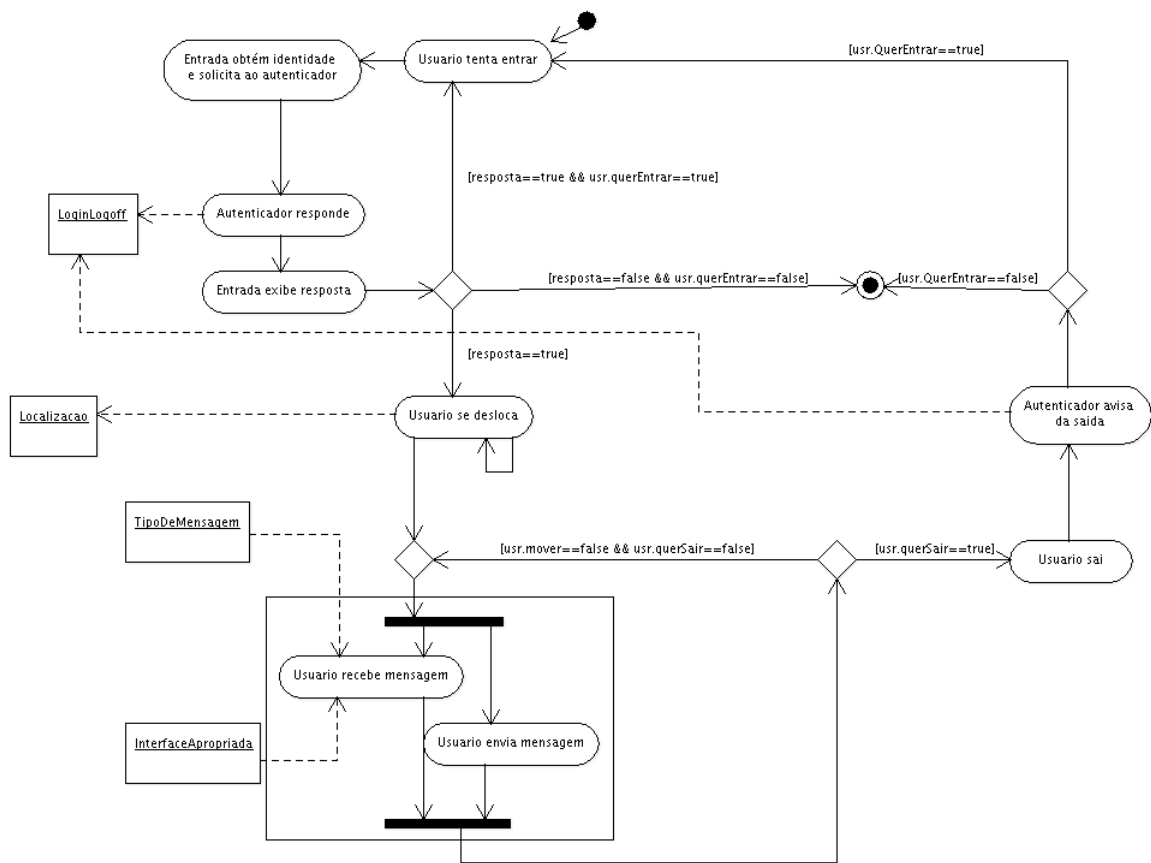


Figura 11: Ações do Serviço Mensageiro Instantâneo e como interagem com diferentes contextos

cação.

Um modelo baseado em serviços permite que se considere os serviços fornecidos independentemente da sua implementação e que novas aplicações propiciem a utilização de serviços já existentes, sem duplicar funcionalidades em cada aplicação (9).

Este tipo de modelo é útil quando já se tem requisitos detalhados das tarefas e atividades dos usuários do sistema.

3.6.2 Modelo baseado em informação

Este tipo de modelo provê uma base útil para análise de fatores humanos. Como a computação ubíqua tende a usar diferentes representações de informações que diferem da em uma interface habitual, é interessante modelar a informação armazenada, transformada e comunicada, assim como a sua utilização é feita em uma determinada tarefa.

A habilidade de uma aplicação de se adaptar e corresponder a diferentes contextos

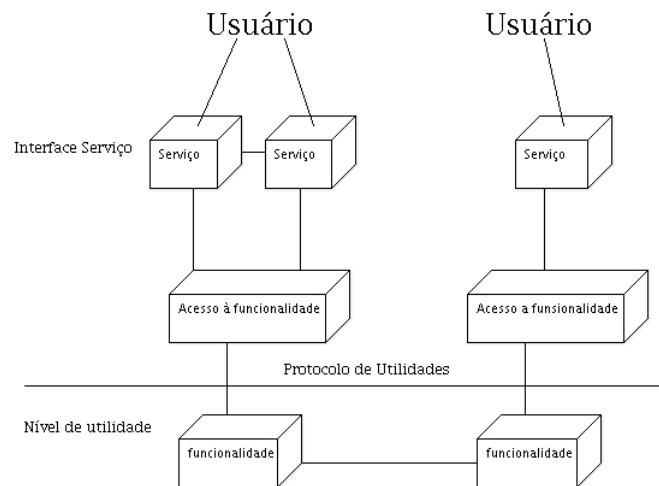


Figura 12: Níveis de aplicação e utilidade (9)

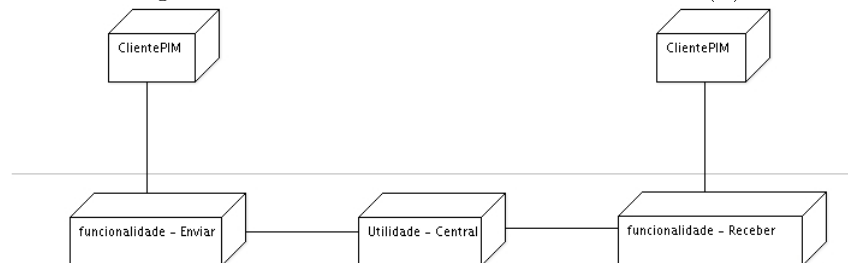


Figura 13: Serviço Mensageiro

depende inteiramente da informação disponível.

Da perspectiva de modelagem, a informação percebida pode ser considerada como informação dinâmica, enquanto informação como uma identidade associada a um objeto é considerada informação estática. A informação dinâmica pode ser atualizada de maneira contínua ou discreta (9).

Na UML, pode-se representar atualizações discretas das informações por meio de eventos e atualizações contínuas por meio de conexões (Doherty *apud* Doherty et al 2001(9)), conforme mostrado na figura 15. Nesta figura, os eventos e atualizações são representados pelos relacionamentos.

A informação pode ainda ser classificada segundo a proposta de Baber *apud* Doherty, em que a informação possui uma dimensão de tempo - armazenada, corrente ou prevista - e uma segunda dimensão que possui dimensões conceituais. Essas dimensões conceituais são conceitos de projeto como tarefa e aplicações, ou como eventos, etc (9).

Uma característica de ambientes de computação ubíqua é a percepção de instantaneidade (12). Doherty (9) sugere utilizar marcações de tempo nos diagramas dinâmicos,

como no de seqüência por exemplo, para registrar as restrições de tempo nos diagramas.

O exemplo apresentado pelo autor do trabalho é um cenário em que as condições de iluminação e aquecimento são ajustadas de acordo com as preferências de um usuário, representando através de um diagrama de colaboração, com o fluxo da informação, que modela a resposta do sistema a uma ação do usuário, como pode ser visto na figura 14.

A seguinte taxionomia para classificar a informação é usada (9):

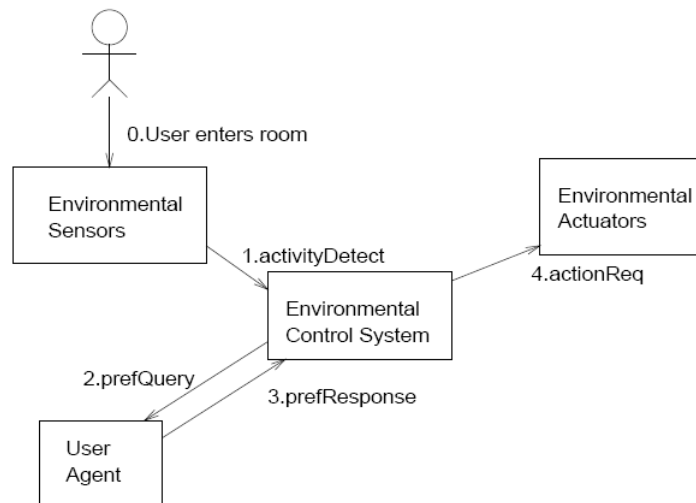


Figura 14: Resposta do sistema a uma ação do usuário (9)

- Evento: Atividade ou situação em um período de tempo determinado. Usuário entra na sala - informação corrente.
- Tarefa: Atividades específicas realizadas por alguém. Tarefa do ambiente é adaptar suas condições para estar de acordo com as preferências do usuário.
- Ambiente: Características físicas/espaciais. Há presença do usuário no espaço. A informação é corrente e implicitamente também armazenada já que não existe uma reatribuição das preferências quando o usuário deixa o espaço.
- Pessoa: Comunicação com o agente do usuário (algum dispositivo que o identifica) habilita preferências específicas da pessoa.
- Artefato: Objeto do mundo reconhecível para o computador. Reconhecimento que o usuário entrou na sala e a comunicação com o agente do usuário. O próprio autor manifesta que não é claro se deve ser feita uma distinção entre um agente físico (um objeto) do usuário que existe também no mundo virtual, como um crachá com um componente eletrônico para autenticar o portador, de um objeto somente digital.

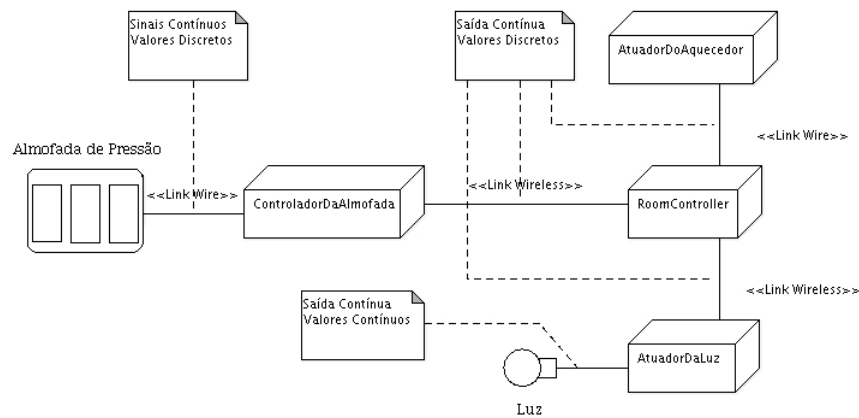


Figura 15: Diagrama com links estereotipados e anotações (9)

3.7 Comparação dos trabalhos estudados

A tabela 1 faz uma comparação entre os diferentes trabalhos referenciados.

Cada coluna é referente a um trabalho referenciado. Eles são abreviados da seguinte forma:

- *CUML* - Context UML
- *InfServ* - Modeling Ubiquitous Computing Applications (Informação e Serviços)
- *eLer* - e-Learning
- *Tango* - Tango
- *Obj* - Identificação de Objetivos
- *Escopo* - Escopos e Estratégias de Propagação de Exceções

As linhas da tabela têm objetivo de identificar nos trabalhos listados anteriormente, as seguintes características:

- A : Objetos em um sistema ubíquo são identificados tanto no mundo físico como no digital?
- B : Localização é um fator importante para a correta modelagem deste tipo de sistema?
- C : O sistema deve ou pode ser parcialmente ou totalmente modelado tomando como base a informação trabalhada no mesmo?
- D : O sistema deve ou pode ser parcialmente ou totalmente modelado tomando como base os serviços oferecidos aos usuários?

- E : É identificada a influência do contexto sobre o funcionamento dos seus componentes?
- F : São identificadas as alterações nos contextos, decorrentes do funcionamento dos componentes?
- G : Existe ou é necessária uma distinção entre utensílios, que provêm o acesso a um serviço mas não o fornecem e a utilidade, como o elemento que realmente a fornece?
- H : É possível identificar com facilidade comportamento emergente resultante de alterações feitas no ambiente, como inserção ou remoção de serviços?

As características foram escolhidas em função dos objetivos listados na seção 2.1 ou para destacar características encontradas em algum dos trabalhos estudados que o destaquem de alguma maneira.

Tabela 1: Tabela comparativa dos trabalhos referenciados

fator	Trabalhos Referenciados					
	CUML	InfServ	eLer	Tango	Obj	Escopo
A - Objetos físico-digitais	Não faz referência	SIM. Ao propor modelagem baseada na distribuição física da informação e capacidade de processamento, considera que objetos podem ser tanto elementos físicos como digitais.	Não faz referência	SIM. Utiliza conceito de Objetos Tangíveis.	Não faz referência	Não faz referência.
B - Localização	Não faz referência	Não faz referência.	Não faz referência	SIM. Localização é atribuída como atributos dos Objetos Tangíveis. Diagrama espacial (um mapa), com localização geográfica precisa.	Não faz referência	SIM. Escopos definem relações de contenção entre objetos no ambiente. Não permite muita precisão.
C - Informação	SIM. Utiliza Mensagens e Mecanismos de atribuição para identificar os diferentes tipos de informação e onde é utilizada (quais serviços).	SIM. Informação pode ser estática ou dinâmica, que pode ser discreta ou contínua.	SIM. Através da solução dos itens E e F .	Chama atenção que a Modelagem do Espaço Informativo e Conceitual é essencial. Proposta é bastante teórica e não propõe muitas soluções.	Não faz referência	Não faz referência

D - Serviço	SIM. Baseia a modelagem ao redor dos serviços presentes no ambiente	SIM. Baseia a modelagem ao redor dos serviços presentes no ambiente.	SIM. Baseia a modelagem ao redor dos serviços presentes no ambiente.	SIM. Baseia a modelagem ao redor dos serviços presentes no ambiente, providos pelos objetos tangíveis.	Não faz referência	Não faz referência.
E - Contexto altera funcionamento	SIM. Diferencia alterações constantes de alterações ocasionais com utilização de gatilhos de contexto e amarração de contexto. Identifica explicitamente um elemento Contexto.	NÃO, pois não identifica explicitamente um elemento Contexto.	SIM. O atual contexto em que uma aplicação funciona determina, ou ao menos influencia, o resultado das ações realizadas que dependem deste mesmo contexto.	NÃO. Não identifica explicitamente um elemento Contexto. Somente insere o Objeto Tangível em um Habitat, que fornece menos informações sobre o ambiente do que a utilização de diferentes contextos.	Não faz referência	Não faz referência.
F - Funcionamento altera contexto	Não representa esse tipo de acontecimento	Não identifica explicitamente um elemento contexto	Assim como o contexto pode influenciar uma ação, o resultado de ações realizadas pode ocasionar alterações em um contexto.	Não identifica explicitamente um elemento Contexto. O habitat é habitado por objetos tangíveis e é o contexto para as interações.	Não faz referência	Não faz referência
G - Utilidade / utilitário	SIM. Diferencia Objeto Sensível ao Contexto, Serviço e Operação (funcionalidade)	SIM. A funcionalidade, é o componente ubíquo, disponibilizado ao meio através de serviços que funcionam somente como interfaces com o ambiente.	Não faz referência	SIM. Distinção semelhante pode ser feita no caso de funcionalidades proporcionadas por objetos que existem somente no mundo digital mas que são acessados por Objetos Tangíveis (presentes no mundo físico).	Não faz referência	Não faz referência

H - Identificação e tratamento de comportamento emergente	Não faz referência	SIM. A modelagem da manipulação e armazenamento da informação provê uma base útil para análise de fatores humano e da qualidade de informação no ambiente	Não faz referência	Não faz referência	SIM. De-compõe os requisitos e resolve ambigüidades. Facilita identificação de sinergia ou conflito para realização das metas.	SIM. A definição de comportamentos genéricos para a propagação de exceções, facilitando o tratamento de exceções não previstas no desenvolvimento.
---	--------------------	---	--------------------	--------------------	--	--

3.7.1 Comentários

A seguir são apresentados comentários sobre cada item da tabela 1.

- Objetos em um sistema ubíquo são identificados tanto no mundo físico como no digital? : A identificação se um objeto no ambiente é físico e/ou digital é apresentada por meio de duas soluções semelhantes, Objetos Tangíveis(14)(10) e Entidades Pivotalis(19).

Dentre as duas soluções, somente para a utilização de Objetos Tangíveis é apresentado como representá-los através dos recursos oferecidos pela UML, pela utilização de estereótipo.

- Localização é um fator importante para a correta modelagem deste tipo de sistema? : A classificação de Pederson (15) conforme o escopo, utilizando a idéia de hierarquias de contenção, é uma maneira rápida e fácil de identificar que partes do ambiente modelado são sujeitas a alterações no seu estado ou funcionamento em função de uma mudança de ação ou estado em um objeto, entretanto, o trabalho não apresenta nenhuma forma fácil de integrar este conceito com os diagramas UML existentes.

Em seu trabalho, Ingstrup (10) propõe a utilização de um diagrama novo ao conjunto de diagramas UML, que representa a disposição física dos objetos presentes no ambiente através de um mapa. Além de indicar a localização e disposição, o mapa também pode representar áreas do ambiente que possuam características particulares (como um serviço só atuar na presença de um usuário dentro de uma determinada região).

- O sistema deve ou pode ser parcialmente ou totalmente modelado tomando como base a informação trabalhada no mesmo? : A maioria dos trabalhos não foca a modelagem na informação utilizada pelos serviços no ambiente, mas sim nos próprios serviços. A solução proposta em (7) (eLer) trata a informação que circula no ambiente, porém isto é abordado nos itens E e F. ContextUML(18) (CUML) identifica o tipo de informação utilizado por cada serviço através de Mensagens e Mecanismos de Atribuição. No trabalho (9) (InfServ) é expresso que uma abordagem para a modelagem de ambientes de computação ubíqua é realizar a modelagem baseada na informação que trafega no ambiente, porém não apresenta soluções concretas para realizar esse tipo de modelagem.
- O sistema deve ou pode ser parcialmente ou totalmente modelado tomando como base os serviços oferecidos aos usuários? : Todos os trabalhos apresentam soluções que se baseiam na modelagem ao redor das funcionalidades oferecidas pelos serviços no ambiente. O trabalho Escopo (1), apesar de se focar nos requisitos não funcionais, considera o efeito que a inserção ou remoção de um serviço e funcionalidades vai ter no funcionamento do ambiente.
- É identificada a influência do contexto sobre o funcionamento dos seus componentes? : A CUML identifica a influência do contexto sobre os serviços através dos objetos contexto e mecanismos sensíveis ao contexto. Em eLer a alteração na ação realizada por um serviço é representada através de um relacionamento com o contexto que influencia o mesmo, no sentido do contexto para a ação sendo realizada.
- São identificadas as alterações nos contextos, decorrentes do funcionamento dos componentes? : Nenhum dos trabalhos estudados, com exceção de eLer (7) se preocupa com a identificação da alteração do contexto (nos casos em que o contexto é identificado explicitamente).

A solução de (7) é a mesma apresentada para a alteração de uma ação em função de um contexto, porém o relacionamento é no sentido inverso.

CUML(18) identifica explicitamente quais serviços dependem de quais contextos. Relacionamentos inversos poderiam ser utilizados para identificar a formação e alteração em decorrência de ações, dos contextos existentes no ambiente.

- Existe ou é necessária uma distinção entre utensílios, que provêm o acesso a um serviço mas não o fornecem e a utilidade, como o elemento que realmente a fornece? : O trabalho InfServ (9) propõe, quando modelando em função dos serviços, que

seja feita uma distinção entre utilidade (o serviço) e utensílio (o meio de acesso ao serviço). De acordo com a proposta, uma mesma funcionalidade pode ser oferecida por diversos utensílios.

No modelo de ContextUML (18), um serviço pode possuir diversas operações (diversas funcionalidades), porém cada operação pertence a somente um serviço.

- É possível identificar com facilidade comportamento emergente resultante de alterações feitas no ambiente, como inserção ou remoção de serviços? : A utilização de escopo de tratadores e estratégias de propagação de exceções sensível ao contexto permite que soluções genéricas sejam definidas e que o desenvolvedor utilize o tempo que gastaria para identificar inúmeros comportamentos possíveis do sistema, solucionando outros problemas.

A confecção de um grafo relacionando as metas esperadas para o sistema permite que as conseqüências da inserção ou remoção de um novo serviço sejam previstas e analisadas.

4 Proposta

O objetivo deste capítulo é identificar recursos, como a utilização de diagramas, para a modelagem de aplicações de computação ubíqua, que utilizem, se possível, os recursos oferecidos pela UML ou que necessitem de poucas modificações no conjunto de notações oferecido por ela.

A proposta deve ser utilizada em conjunto com o método de desenvolvimento tradicional que o desenvolvedor costuma utilizar, acrescentando a modelagem de características da aplicação que poderiam passar despercebidas. Como base, é utilizado o ContextUML.

4.1 MARVIn

MARVIn significa *Modelo para Ambientes Reais/Virtuais Integrados*, por integrar aspectos do mundo físico (real) e do virtual. O nome também é homenagem à obra de Douglas Adams (3).

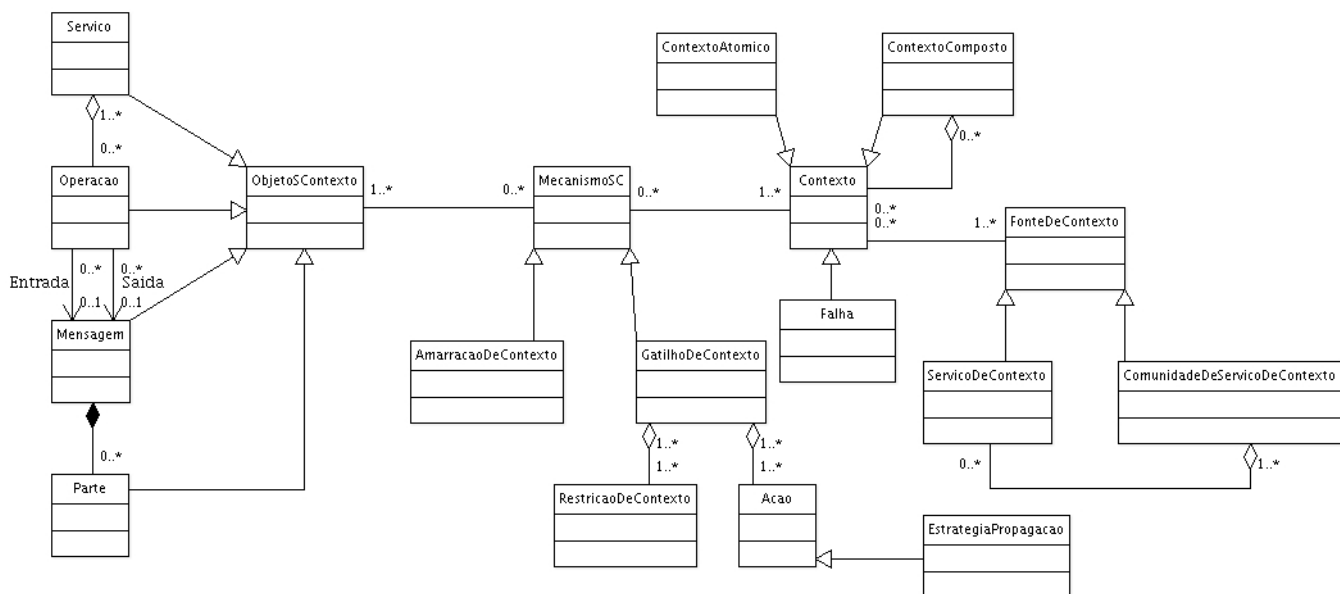


Figura 16: Metamodelo da proposta

O capítulo é dividido em quatro outras seções correspondentes a diferentes perspectiva de modelagem.

4.2 Visão Estrutural/Espacial

A identificação diferenciada de um objeto que possui representação tanto física como digital através de uma única entidade, pode ser feita utilizando os recursos oferecidos pela UML. A identificação do mesmo pode ser feita pela utilização de uma classe estereotipada como ObjetoTangível. O nome Objeto Tangível é um nome adequado para denominar os objetos que existem tanto no mundo físico como digital.

Os diversos componentes que existem em um ambiente de computação ubíqua não necessariamente se comunicam diretamente. É importante identificar o tratamento dado para a informação que trafega no ambiente através dos diversos componentes. A identificação do tipo de mensagem utilizada em uma comunicação pode ser realizada através de objetos Mensagem e Parte (que compõem a mensagem), o que facilita a identificação de como os diversos serviços oferecidos em um ambiente se relacionam.

Não somente objetos com capacidade computacional têm papel no funcionamento de um ambiente. Objetos puramente físicos, ou o próprio espaço físico, além dos objetos tangíveis, são relevantes para o desenvolvimento de uma aplicação em diversas situações.

Como um exemplo, uma aplicação para um teatro deve apresentar um comportamento esperado quando o ator estiver localizado sobre a área do cenário que será usado na próxima cena. A utilização de um mapa possibilita indicar não somente a localização de objetos tangíveis relevantes para a aplicação, mas também delimitações de área (indicar no mapa a área do cenário, assim passa a ser uma informação mais precisa para quem irá desenvolver a aplicação), limites físicos, disposição de objetos do ambiente, etc.

Na figura 17 é mostrado o ambiente de um teatro, em que no palco existem diferentes cenários de tamanhos diferentes. Os refletores usados para a iluminação devem iluminar o palco quando o ator estiver na área correspondente a um cenário.

A confecção de um mapa aproximado de um ambiente é uma necessidade não coberta pelos recursos oferecidos pela especificação UML, mas é um problema fácil de ser solucionado através da utilização de um simples editor gráfico ou lápis e papel. Um esboço aproximado da realidade, que permita identificar áreas e disposição dos objetos já contribui consideravelmente para uma melhor compreensão do ambiente sendo modelado.

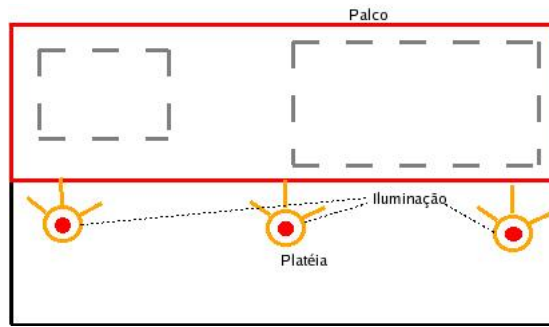


Figura 17: Utilização de mapa para indicar localização física

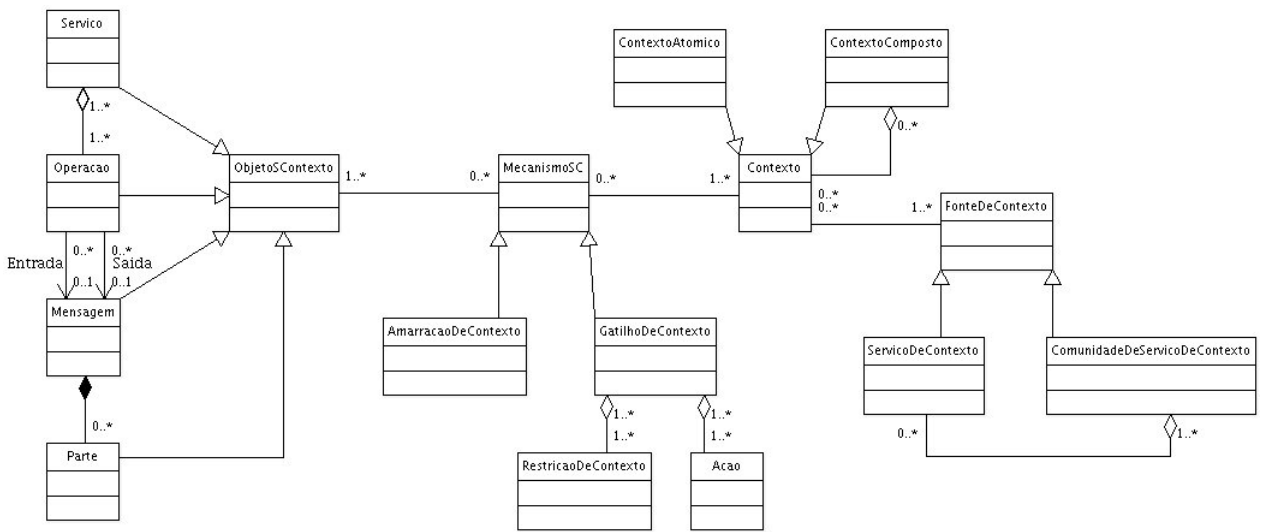


Figura 18: Alteração no meta-modelo de ContextUML

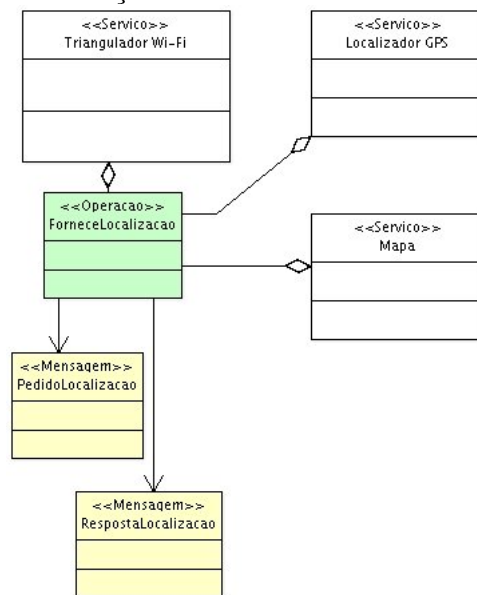


Figura 19: Operação fornecida por múltiplos serviços

No mundo real, são comuns exemplos de um mesmo serviço sendo oferecido por diversos meios de acesso. Como mostrado na figura 19, uma mesma funcionalidade que fornece localização pode ser oferecida por diversos Serviços, o que corresponde ao meta-modelo da ContextUML modificado mostrado na figura 18.

Através da identificação explícita de uma entidade Contexto, dos Serviços Sensíveis ao Contexto e dos Mecanismos de Sensibilidade ao Contexto (Amarração e Gatilho de Contexto) é possível representar a dependência de um serviço em relação a um determinado contexto.

Não é necessário que durante a implementação das aplicações em um ambiente, exista um elemento Contexto. O contexto pode ser somente um tipo de informação, que pode ser fornecido por um ou diversos serviços no ambiente e é representado explicitamente na modelagem como uma entidade separada. A indicação da origem de um contexto é útil para fazer a identificação dos elementos do sistema que podem alterar o mesmo. Uma sugestão de como denotar essa relação é mostrada na figura 20. A relação de dependência entre o contexto Localização e objeto tangível Usuário indica que Localização pode sofrer alguma alteração devido há alguma ação de Usuário, como por exemplo, se o usuário se locomover. A dependência também poderia ser entre o contexto e um serviço ou operação.

A utilização de uma estrutura hierárquica de contenção de objetos no ambiente é uma maneira rápida e fácil de identificar que partes do ambiente modelado são sujeitas a alterações no seu estado ou funcionamento em função de uma mudança de estado ou ação de outro objeto (uma alteração em um objeto vai quase que certamente causar mudanças nos objetos nele contidos, mas não necessariamente em objetos no mesmo nível ou em níveis superiores).

Na figura 21, duas lojas são localizadas no segundo piso, porém de diferentes *Shopping Centers*, modeladas em um diagrama de objetos. O diagrama faz sentido somente se quem o estiver utilizando souber da existência dos dois *Shopping Centers* porém quer trabalhar somente no nível hierárquico das lojas.

A hierarquia, ao menos em estruturas com poucos níveis, não muito complexas e onde a estrutura permaneça constante (Objetos podem se deslocar pelo ambiente, assim, um consumidor andando em um *shopping center* pode estar em um momento localizado na hierarquia abaixo de loja e em outro momento fora das lojas), pode ser feita com a utilização de estereótipos, indicando o escopo do objeto em cada nível da hierarquia. O diagrama resultante é mais limpo e legível do que se a hierarquia fosse estabelecida através de relacionamentos.

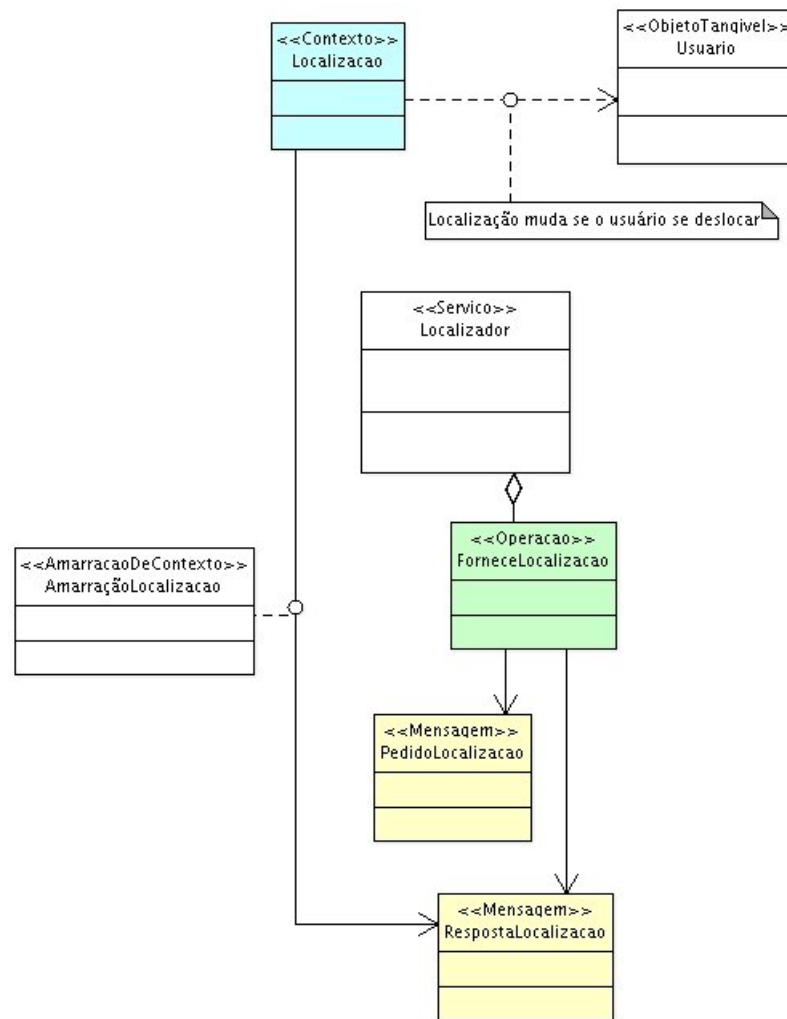


Figura 20: Contexto alterado por mudança no ambiente

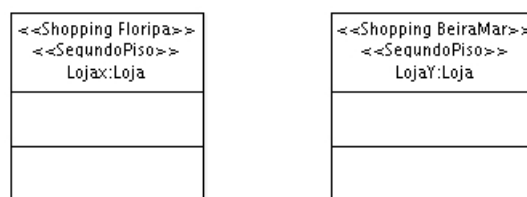


Figura 21: Hierarquia dos Objetos

4.3 Visão Dinâmica

Para tratar a relação entre os contextos e as ações que podem ser influenciadas por eles, pode ser utilizado o diagrama de atividades.

Existem duas possibilidades diferentes de relacionamentos que envolvem contexto e um serviço, a alteração do funcionamento de um serviço em função do contexto e a alteração

do contexto em função do funcionamento serviço do ambiente.

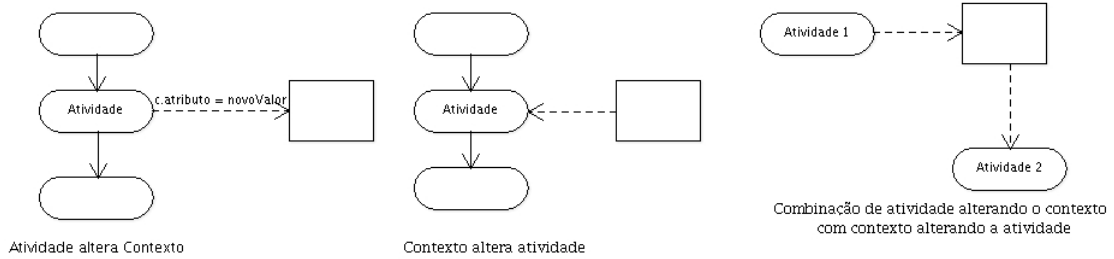


Figura 22: Exemplo de atividade alterando contexto e de contexto alterando a atividade (7)

O sentido do relacionamento entre um Contexto e um Serviço, no diagrama de atividade, é suficiente para identificar se é a ação sendo executada depende de um determinado contexto ou se o contexto vai ser influenciado pela ação, como mostrado na figura 22. A representação explícita do contexto no diagrama não significa que na implementação resultante deva existir um objeto correspondente ou serviços (um ou vários) que fornecem as informações deste (estas informações podem ser fornecidas pelo usuário).

Finalmente, a utilização de marcações de tempo (exemplificado em um diagrama de seqüência - veja figura 23) ajuda no controle da qualidade de serviço oferecida pelos serviços modelados (restrições de tempo máximo ou mesmo percepção de instantaneidade em uma tarefa são requisitos que agradam e são esperados pelos usuários).

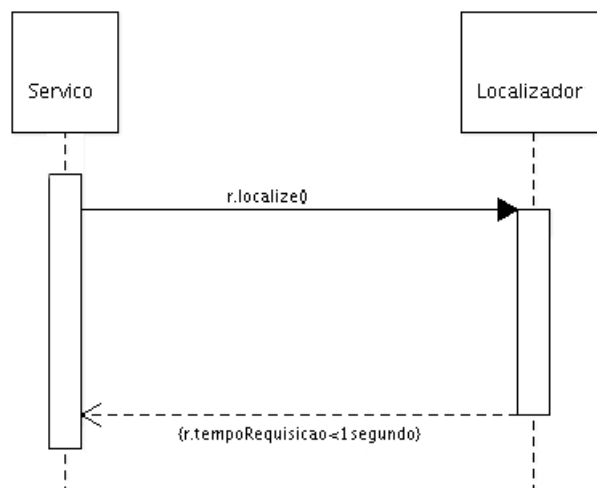


Figura 23: Marcação de tempo no diagrama. (9)

4.4 Deployment

Para esta perspectiva podem ser utilizadas soluções já apresentadas, como a utilização de um mapa para tratar a localização dos objetos tangíveis.

Um diagrama de *Deployment*, como mostrado na figura 24, pode ser usado para relacionar as diferentes funcionalidades existentes no ambiente e serviços, e também como podem ser acessados no ambiente.

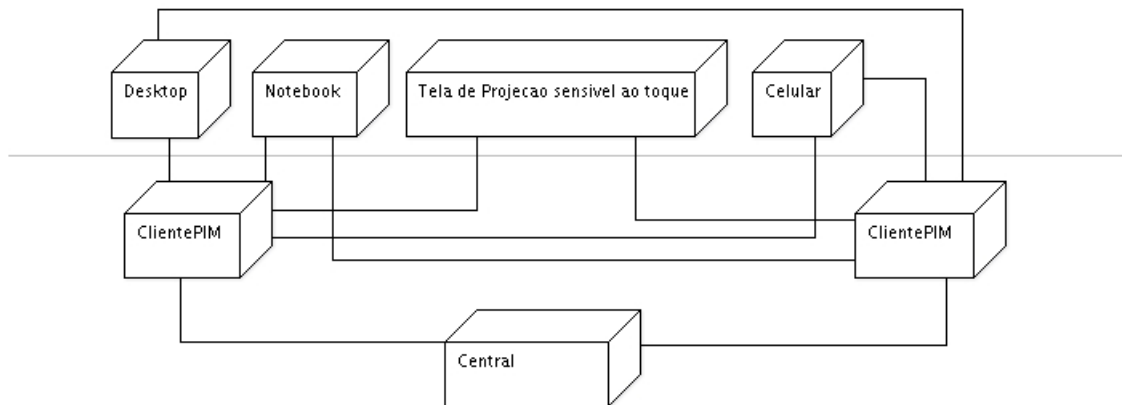


Figura 24: Serviços e como acessá-los (9)

4.5 Funcional

O objetivo dos *breakdowns* é identificar situações em que a integração da tecnologia no meio se faz perceber a ponto de desviar a atenção das tarefas sendo realizadas.

Eles podem ser identificados durante a elaboração dos casos de uso, (pelo menos parte dos *breakdowns*), para depois serem definidas estratégias a serem tomadas caso estes venham a ocorrer.

Para realizar a identificação de *breakdowns* deve se identificar falhas no comportamento esperado das aplicações no ambiente. A composição dos objetivos esperados e análise de como o cumprimento de um objetivo influencia na realização de outro é uma maneira de realizar essa identificação.

A organização e decomposição das metas do ambiente, mostrados na figura 25, permite, além da desambiguação dos requisitos não funcionais definidos para o ambiente (evitar que diferentes pessoas envolvidas no desenvolvimento entendam um mesmo requi-

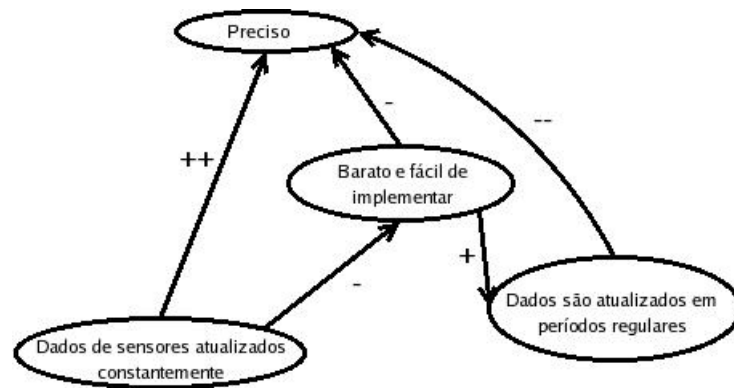


Figura 25: Decomposição dos objetivos: Previsão do funcionamento do sistema frente a novos serviços.

sito de maneiras diferentes), que o efeito da inserção de novos serviços no funcionamento do ambiente seja previsível (seja ao colaborar para alcançar as metas ou prejudicar o cumprimento dos requisitos). As arestas dos relacionamentos entre as metas são rotuladas com os símbolos +, -, ++ e --, que significam respectivamente que, uma meta contribui para a realização da outra, uma meta prejudica a realização da outra, uma meta garante a realização da outra e uma meta inviabiliza a realização da outra.

Para cada *breakdown* identificado, uma estratégia de tratamento pode ser definida. Um exemplo de *breakdown* é a aplicação interromper o usuário em busca de informações necessárias para a realização de uma tarefa. Buscar informações não é uma exceção do sistema, porém uma estratégia pode determinar que antes de interromper o usuário, o serviço realizando a tarefa propague no sistema, através dos diferentes escopos, sua necessidade por informações.

A figura 26 mostra seqüências de atividades que podem ocasionar falhas. Se quando um usuário A pede acesso ao ambiente e é liberado porém não entra no ambiente, o ambiente pode obter a informação falsa que ele está presente quando não está. Quando um usuário B tentar enviar uma mensagem para o usuário A, o sistema de mensagem funcionará como se A estivesse presente, porém no momento em que a Central do Serviço de Mensagem for rotear a mensagem, ele não encontrará o destino, o que pode causar a perda da mensagem enviada.

Junto da identificação da falha, uma estratégia pode ser definida, que pode por exemplo, notificar todos serviços que dependem de localização que o usuário não está presente no ambiente, ou solicitar que outro serviço realize a busca de A, para verificar se não houve falha no serviço de localização. As falhas e estratégias para propagá-las e tratá-las podem

ENTRADA NO AMBIENTE:

1. Usuário solicita entrada na sala
2. Controle de Entrada pede login e senha
3. Controle de Entrada manda para Autenticador verificar
4. Autenticador verifica e passa resposta para Entrada
 - 4a. Em caso de autorização Entrada libera a entrada. Continua em 5
 - 4b. Em caso de usuário ou senha errada Entrada não libera e pede para tentar novamente
5. Entrada avisa para o ambiente que Usuário X entrou no ambiente (para serviços que dependem da presença do usuário funcionarem)

ENVIO DE MENSAGEM

1. Usuário envia mensagem para destinatário através do Cliente do Serviço de Mensagem
2. Central do Serviço de Mensagem pergunta se usuário destino está presente
3. Autenticador responde
 - 3a. Em caso afirmativo envia a localização para a Central - Continua em 4
 - 3b. Em caso negativo retorna dizendo que não está - Vai para a sequencia ENVIAR MENSAGEM OFFLINE
4. Central pede a localização do usuário destino
5. Serviço de Localização responde
 - 5a. Serviço não encontrou o destinatário - BREAKDOWN - Estratégia de Propagação A
 - 5b. Serviço encontrou e avisa a localização - Continua em 6
6. Central recebe resposta e envia mensagem para o endereço destino.

ESTRATÉGIA A

1. Envia para Sensores um novo pedido de localização
2. Notifica serviço de manutenção de falha em um sensor de localização

Figura 26: Identificação de Falhas e Estratégia de Propagação da Exceção.

ser identificadas no diagrama de classes, estendendo a idéia de Gatilhos, onde um Gatilho de Falha é criado, que se relaciona com uma Falha (um contexto em que o funcionamento não ocorre como desejado) e uma Estratégia. Na figura 27 é identificada uma Falha, que quando ocorre ativa a EstratégiaA através do gatilho GatilhoLocalizacaoImprecisa. A mesma estratégia poderia ser usada para tratar outras falhas resultantes de imprecisão ou ausência de um dado necessário para o funcionamento.

No exemplo anterior, o Serviço de Mensagem precisa que as informações de localização sejam atualizadas continuamente, como mostrado na figura 25, assim, se um serviço que fornece localização se integrar a este sistema mas fornecer informações somente em intervalos de tempo, ele prejudicará o funcionamento.

Os *Breadkdowns* e metas ajudam a identificar estados que não são desejados mas podem vir a ocorrer. Estes problemas podem ser abordados por soluções genéricas para diversos problemas definidas através de estratégias de propagação de exceção e definição de escopos dos serviços existentes, o que permite que o desenvolvedor se preocupe com outros aspectos do desenvolvimento do sistema no tempo que seria gasto para definir mecanismos individuais de tratamento de exceção.

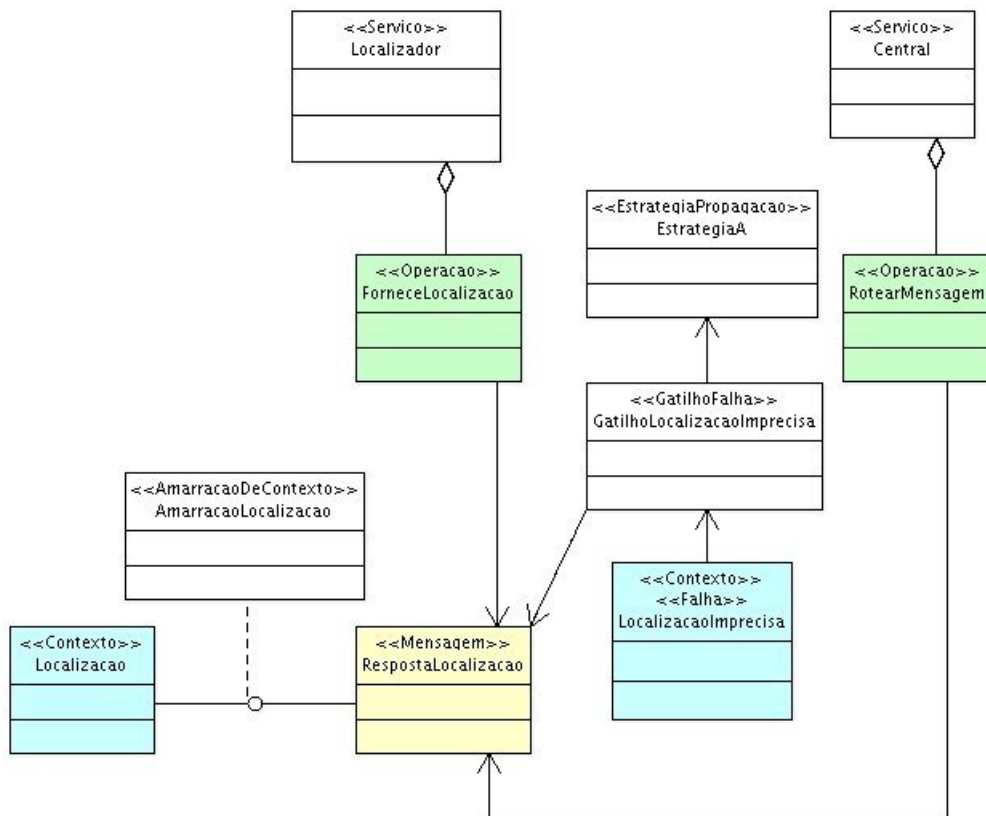


Figura 27: Identificação de Falhas e Estratégia de Propagação da Exceção no diagrama de classes.

4.6 Resumo

Nesta seção é apresentado um resumo da proposta feita, listando os passos que devem ser incorporados ao desenvolvimento da modelagem realizada, a fim de melhorar a qualidade da modelagem de ambientes de aplicações de computação ubíqua.

A organização em diferentes etapas, apresentadas nesta seção, não significa que os passos apresentados devam ser executados na mesma seqüência, serve somente para organizar melhor o conteúdo apresentado.

1. Elaboração de um grafo contendo as metas esperadas do sistema para identificar possíveis requisitos conflitantes e posterior detecção de comportamentos emergentes utilizando a notação +, -, ++ e - para relacionar como os diferentes objetivos se relacionam. Os objetivos devem ser obtidos dos usuários que utilizarão a aplicação.
2. Identificação de todos os objetos envolvidos nas aplicações existentes no ambiente. Devem ser identificados tanto objetos virtuais, como objetos do mundo físico e obje-

tos tangíveis. No mapa espacial, os objetos devem ser identificados com a utilização de marcações circulares sobre a sua representação e no diagrama de classes devem ser estereotipados como `ObjetosTangíveis`, se for o caso. Os serviços existentes no ambiente devem ser representados por esses objetos.

3. Mapeamento, se necessário, da localização dos objetos envolvidos, identificação de obstáculos (portas, paredes), áreas de funcionamento, localização espacial (é um ambiente contido em outro ambiente?) no mapa espacial. Hierarquias de contenção podem ser utilizadas para organizar objetos que tenham uma organização hierárquica e dos quais não é necessário saber a localização exata para o funcionamento correto da aplicação. A organização hierárquica pode ser realizada no diagrama de classes utilizando estereótipos.
4. Identificação das informações do ambiente que determinem o funcionamento dos serviços para listar os diferentes contextos que devem ser considerados. Devem ser analisadas as funcionalidades de cada serviço para verificar quais dados são necessários para a sua realização. Todos os dados devem ser representados através de um contexto em que devem estar inseridos, sendo que o contexto é representado no diagrama de classes como uma classe estereotipada como `Contexto`, acrescidos no diagrama de classes definido no passo 2.
5. Agrupamento dos serviços existentes no projeto do ambiente em escopos (conjuntos de serviços que possam prover um mesmo tipo de funcionalidade). Os escopos serão utilizados mais adiante e podem ser organizados simplesmente listando em uma anotação os serviços que o compõem.
6. Identificação de *breakdowns* e definição de estratégias de propagação para os mesmos, utilizando os escopos definidos anteriormente. A identificação pode ser feita durante a elaboração dos casos de uso. Se originalmente não seriam elaborados os casos de uso, eles devem ser feitos.
7. Durante a criação dos modelos dinâmicos, certificar-se de anotar no modelo restrições de tempo de resposta para as ações ou outro tipo de restrição necessária para garantir a qualidade da informação tratada e conseqüentemente do serviço provido. Caso não seja utilizado nenhum recurso para poder fazer a marcação das restrições de tempo e elas forem necessárias, um diagrama de atividade ou de sequência deve ser elaborado.

4.7 Comparação com MARVIn

A tabela 2, mostrada a seguir, é a mesma utilizada anteriormente (tabela 1), com a adição de uma coluna para incluir a proposta apresentada neste capítulo.

Tabela 2: Tabela comparativa final

fator	Trabalhos						
	MARVIn	CUML	InfServ	eLer	Tango	Obj	Escopo
A - Objetos físico-digitais	<i>SIM. Objetos Tangíveis identificados por estereótipos ou marcas no mapa de localização.</i>	Não faz referência	SIM. Ao propor modelagem baseada na distribuição física da informação e capacidade de processamento, considera que objetos podem ser tanto elementos físicos como digitais.	Não faz referência	SIM. Utiliza conceito de Objetos Tangíveis.	Não faz referência	Não faz referência
B - Localização	<i>SIM. Utilização de uma diagrama de localização (mapa) e estereótipos identificando a relação hierárquica entre objetos em um ambiente.</i>	Não faz referência	Não faz referência ¹ .	Não faz referência	SIM. Localização é atribuída como atributos dos Objetos Tangíveis. Diagrama espacial (um mapa), com localização geográfica precisa.	Não faz referência	SIM. Escopos definem relações de contenção entre objetos no ambiente. Não permite muita precisão.
C - Informação	<i>SIM. Utilização de objetos Operação e Mensagem (ContextUML). Anotações como marcações de tempo para garantir a qualidade da informação.</i>	SIM. Utiliza Mensagens e Mecanismos de atribuição para identificar os diferentes tipos de informação e onde é utilizada (quais serviços).	SIM. Informação pode ser estática ou dinâmica, que pode ser discreta ou contínua.	SIM. Através da solução dos itens E e F .	Chama atenção que a Modelagem do Espaço Informativo e Conceitual é essencial. Proposta é bastante teórica e não propõe muitas soluções.	Não faz referência	Não faz referência
D - Serviço (funcionalidade)	<i>SIM. Utilização de objetos Serviço (ContextUML).</i>	SIM. Baseia a modelagem ao redor dos serviços presentes no ambiente	SIM. Baseia a modelagem ao redor dos serviços presentes no ambiente	SIM. Baseia a modelagem ao redor dos serviços presentes no ambiente	SIM. Baseia a modelagem ao redor dos serviços presentes no ambiente, providos pelos objetos tangíveis.	Não faz referência	Não faz referência

¹Sugere modelagem em função da localização mas faz poucas contribuições por isso não foi considerado

E - Contexto altera funcionamento	<i>SIM. Gatilhos de Contexto e Amarração de Contexto (ContextUML) e indicação de quais contextos influenciam o sistema em determinadas ações ou estados</i>	SIM. Diferencia alterações constantes de alterações ocasionais com utilização de gatilhos de contexto e amarração de contexto. Identifica explicitamente um elemento Contexto.	NÃO, pois não identifica explicitamente um elemento Contexto.	SIM. O atual contexto em que uma aplicação funciona determina, ou ao menos influencia, o resultado das ações realizadas que dependem deste mesmo contexto.	NÃO. Não identifica explicitamente um elemento Contexto. Somente insere o Objeto Tangível em um Habitat, que fornece menos informações sobre o ambiente do que a utilização de diferentes contextos.	Não faz referência	Não faz referência
F - Funcionamento altera contexto	<i>SIM. Metamodelo ContextUML modificado para identificar que objetos podem alterar os contextos e indicação de quais ações ou estados influenciam o contexto.</i>	Não representa esse tipo de acontecimento	Não identifica explicitamente um elemento contexto	Assim como o contexto pode influenciar uma ação, o resultado de ações realizadas pode ocasionar alterações em um contexto.	Não identifica explicitamente um elemento Contexto. O habitat é habitado por objetos tangíveis e é o contexto para as interações.	Não faz referência	Não faz referência
G - Utilidade / utilitário	<i>SIM. Metamodelo ContextUML modificado para permitir que uma mesma Operação seja de diversos Serviços.</i>	SIM. Diferencia Objeto Sensível ao Contexto, Serviço e Operação (funcionalidade)	SIM. A funcionalidade, é o componente ubíquo, disponibilizado ao meio através de serviços que funcionam somente como interfaces com o ambiente.	Não faz referência	SIM. Distinção semelhante pode ser feita no caso de funcionalidades proporcionadas por objetos que existem somente no mundo digital mas que são acessados por Objetos Tangíveis (presentes no mundo físico).	Não faz referência	Não faz referência

H - Identificação e tratamento de comportamento emergente	<i>SIM. Identificação de breakdowns e utilização de escopos e estratégias de propagação de exceção.</i>	Não faz referência	SIM. A modelagem da manipulação e armazenamento da informação provê uma base útil para análise de fatores humano e da qualidade de informação no ambiente	Não faz referência	Não faz referência	SIM. De-compõe os requisitos e resolve ambigüidades. Facilita identificação de sinergia ou conflito para realização das metas.	SIM. A definição de comportamentos genéricos para a propagação de exceções, facilitando o tratamento de exceções não previstas no desenvolvimento.
---	---	--------------------	---	--------------------	--------------------	--	--

5 Exemplo de uma aplicação em um ambiente

O serviço Mensageiro Instantâneo, que gerencia a comunicação entre usuários de um ambiente foi utilizado em diversos momentos ao longo do trabalho para ilustrar as propostas dos trabalhos estudados e alterações sugeridas nestes trabalhos.

O cenário do Mensageiro Instantâneo é um cenário real. Diversos serviços foram desenvolvidos e ao trabalharem juntos fornecem ao usuário do ambiente o serviço de mensagens. Porém, a modelagem apresentada até agora, que ilustra os exemplos, foi feita a partir de engenharia reversa da implementação desenvolvida anteriormente, sem a devida atenção aos aspectos da modelagem de aplicações para computação ubíqua.

Neste capítulo, o modelo proposto a partir dos diversos trabalhos estudados será utilizado para modelar uma ferramenta de agenda que deve organizar os compromissos de um grupo de usuários.

5.1 Agenda

O serviço escolhido é uma aplicação de agendamento de compromissos e atividades, que gerencia tanto a agenda do grupo de usuários, utilizando as agendas pessoais dos mesmos, e as corrige se necessária alguma alteração para atender ao grupo.

A Agenda pode ser utilizada remotamente, não sendo necessário estar em um ambiente físico em que convive o grupo do qual ela deve gerenciar os compromissos. Quando o usuário estiver em um espaço físico designado como ambiente de trabalho do grupo, ela deve poder derivar intenções dos usuários utilizando informações de localização física.

Quando um usuário desejar, este pode solicitar o agendamento de um compromisso do grupo. É suposto que atividades coletivas do grupo são mais importantes que compromissos pessoais, fora exceções (uma consulta ao médico marcada em um horário conflitante,

com 6 meses de antecedência). A agenda deve verificar a disponibilidade de horário de todos envolvidos no compromisso e atualizar as respectivas agendas pessoais. Em caso de conflito de horários, estes devem ser solucionados da maneira mais automatizada possível. Dados insuficientes para permitir o funcionamento do sistema devem ser obtidos sem que os usuários percebam a sua falta.

Toda a parte de comunicação existente no sistema pode ser deixada sob responsabilidade do Mensageiro Instantâneo já apresentado.

A seguir estão separados os passos sugeridos no resumo apresentado no capítulo anterior. Somente foram tratadas ações relativas à agendamento de compromissos, não sua recuperação.

5.2 Passo 1

- Elaboração de um grafo contendo os objetivos esperados do sistema para identificar possíveis requisitos conflitantes e posterior detecção de comportamentos emergentes utilizando a notação +, -, ++ e – para relacionar como os diferentes objetivos se relacionam. Os objetivos devem ser obtidos dos usuários que utilizarão a aplicação.

Conforme proposto, o primeiro passo é a elaboração de um grafo de conceitos e características que podem surgir no funcionamento das aplicações no ambiente, provenientes das diferentes possibilidades de cooperação entre os serviços existentes e os serviços que podem vir a ocupar o ambiente.

Para poder elaborar o grafo antes é necessário decidir como a aplicação irá funcionar. Na figura 28 é apresentado um pequeno caso de uso referente ao agendamento de um compromisso. Mais adiante, com os resultados dos outros passos, o caso de uso ficará mais elaborado.

```

AGENDAMENTO DE COMPROMISSOS
1.Usuário solicita agendar compromisso de grupo na Agenda
2.Agenda solicita dados do compromisso e envolvidos
3.Agenda obtém os dados necessários através das agendas pessoais dos usuários envolvidos
4.Agenda verifica conflitos
  4a.Se houverem conflitos tenta resolvê-los e segue para 5
  4b.Se não houver conflitos segue direto para 5
5.Agenda confirma agendamento e notifica envolvidos
  
```

Figura 28: Agendamento de Compromisso

Agora deve-se analisar o comportamento esperado. Um sistema de agenda que espera indeterminadamente pela resposta de todos envolvidos em um compromisso, para que este seja agendado, prejudica a velocidade de operação.

Para que a Agenda tome decisões de forma pró-ativa, é necessário que dados atualizados estejam disponíveis, o que sugere dispositivos que atualizem sua saída de dados constantemente, o que é mais caro do que um sistema que atualiza os dados periodicamente.

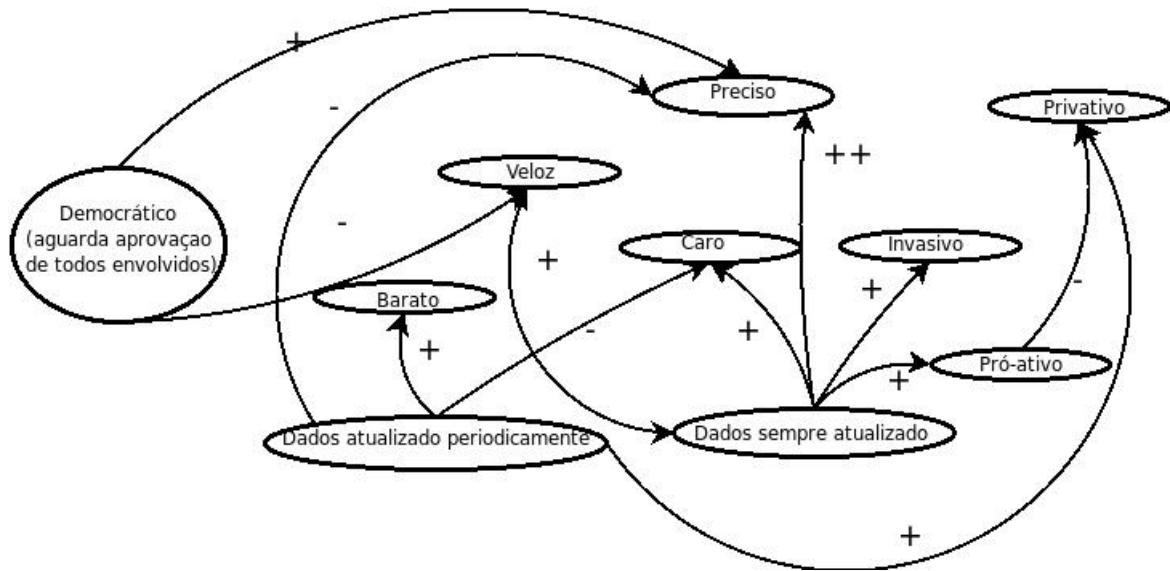


Figura 29: Grafo dos objetivos da Agenda

Ao verificar a figura 29 é fácil de perceber que agendas pessoais que liberam o máximo possível de dados para consulta pela Agenda vão se integrar melhor ao ambiente, pois vão possibilitar que o sistema seja mais pró-ativo, mesmo que às custas da perda de privacidade dos usuários. Com relação ainda à privacidade, se os usuários permitirem que sejam monitorados no espaço físico, outras possibilidades de pró-atividade do sistema podem surgir.

Sempre que um novo dispositivo ou aplicação for integrado à Agenda, a detecção de comportamentos emergentes será facilitada, pois já estão listadas as características mais/menos desejadas no ambiente e a forma como elas se relacionam.

5.3 Passo 2

- Identificação de todos os objetos envolvidos nas aplicações existentes no ambiente. Devem ser identificados tanto objetos virtuais, como objetos do mundo físico e objetos tangíveis. No mapa espacial, os objetos devem ser identificados com a utilização de marcações circulares sobre a sua representação e no diagrama de classes devem ser estereotipados como `ObjetosTangíveis`, se for o caso. Os serviços existentes no

ambiente devem ser representados por esses objetos.

Para o funcionamento de uma agenda virtual, os componentes básicos necessários são os usuários da agenda e a própria agenda.

Nas funcionalidades desejadas para a agenda, espera-se que ela possa deduzir informações sobre agendamento de reuniões utilizando a disposição física dos usuários ao redor de objetos que caracterizam um contexto de reunião. Assim, objetos de mobília relacionados com contextos específicos (uma rede pendurada caracteriza um momento de lazer, não de trabalho, uma mesa de reunião com pessoas sentadas ao redor ao mesmo tempo caracteriza uma reunião em andamento) são identificados, no caso, a mesa.

A Agenda deve conciliar os compromissos coletivos com os compromissos individuais e vice-versa, logo, ela precisa de dados de compromissos pessoais, provenientes de algum tipo de agenda pessoal.

É necessária comunicação entre usuários e a Agenda e entre os diversos serviços que a compõem. A comunicação pode ser realizada pelo Mensageiro Instantâneo já apresentado, poupando a identificação dos objetos envolvidos sob este aspecto. Basta verificar se a atual situação do Mensageiro Instantâneo, ao ser analisada com o grafo de objetivos do passo 1 (figura 29), atende aos objetivos definidos, condição que é cumprida.

Ao listar os objetos identificados, esses são:

- Sistema e Comunicação
- Usuário
- Mobília (mesa)
- Agenda pessoal
- Agenda coletiva
- Sensor (ver passo 3)

Na figura 30 estão alguns dos objetos, identificados quando na condição de Objeto Tangível. A mesa é identificada no passo 3.

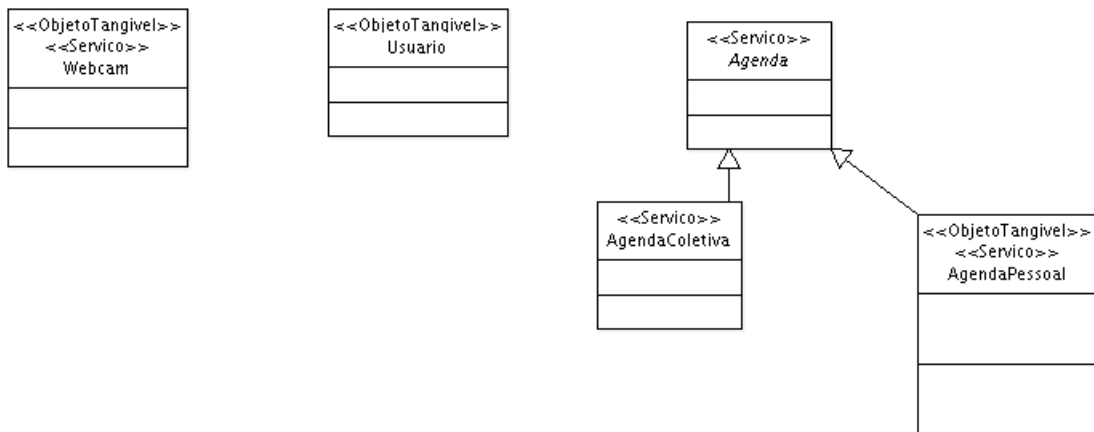


Figura 30: Identificação dos Objetos Tangíveis.

5.4 Passo 3

- Mapeamento (se necessário) da localização dos objetos envolvidos, identificação de obstáculos (portas, paredes), áreas de funcionamento, localização espacial (é um ambiente contido em outro ambiente?) no mapa espacial. Hierarquias de contenção podem ser utilizadas para organizar objetos que tenham uma organização hierárquica e dos quais não é necessário saber a localização exata para o funcionamento correto da aplicação. A organização hierárquica pode ser realizada no diagrama de classes utilizando estereótipos.

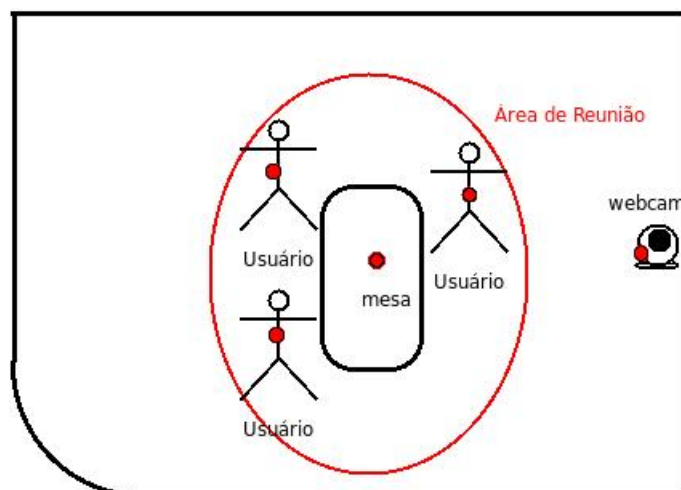


Figura 31: Delimitação de área física.

Caso um usuário solicite o agendamento de uma reunião, mas não informe os envolvidos, uma dedução possível é que a nova reunião deverá envolver quem está participando

da reunião atual. Se existir uma área onde reuniões geralmente são feitas, algum serviço disponível no ambiente poderia verificar quem está presente naquele momento na área de reunião, como ilustrado na figura 31 e retornar os nomes dos mesmos.

O reconhecimento e localização dos usuários no espaço deve ser feito por algum tipo de sensor (webcam, sinal *bluetooth* como telefone celular dos usuários, radio-freqüência, etc), cada um com um tipo de limitação. Durante o desenvolvimento não há como saber quais tipos de mecanismos podem ser usados ou quais ainda vão vir a ser inventados, porém no mapa podem constar objetos físicos que bloqueariam o campo de visão de uma câmera ou interromperiam um sinal de rádio, por exemplo.

O sensor foi um objeto não identificado no passo 2, mas que devido a representação dos aspectos físicos fez perceber sua necessidade. Ao aproveitar a implementação do Mensageiro Instantâneo, é provável que o ambiente já disponha de um sensor desse tipo, mas não é garantido, assim, um item 'sensor' pode ser adicionado ao passo anterior.

Se o desenvolvedor analisar o mapa do espaço físico do ambiente e fizer com que os serviços identifiquem a área de reunião não pelas suas coordenadas geográficas na sala, mas pela proximidade com um objeto físico no ambiente (mesa de reunião), se o objeto for deslocado, o sistema ainda funcionará corretamente e não haverá necessidade de envolvimento humano para fazer uma correção

5.5 Passo 4

- Identificação das informações do ambiente que determinem o funcionamento dos serviços para listar os diferentes contextos que devem ser considerados. Devem ser analisadas as funcionalidades de cada serviço para verificar que dados que são necessários para a sua realização. Todos os dados devem ser representados através de um contexto em que devem estar inseridos, sendo que o contexto é representado no diagrama de classes como uma classe estereotipada como Contexto.

Uma agenda trabalha basicamente com compromissos, horários e datas.

Compromissos são vinculados às pessoas, individualmente ou em grupo, e uma atividade. A atividade acontece no horário e local definido e as pessoas vinculadas ao compromisso devem comparecer naquele horário, na data correta.

Data é uma informação fácil de se conseguir, porém o horário possui fatores que devem ser considerados, como fuso-horários e horário de verão. A Agenda deve ter exatidão sufi-

ciente para impedir que usuários percam compromissos por culpa de um horário fornecido errado ao sistema ou diferença de horário mal calculada.

A agenda também deve ser capaz de se comunicar com todos envolvidos nos compromissos, para verificar conflitos de horário e alertar compromissos que devem ser atendidos, o que exige dados como número de telefone, endereço e e-mail.

Ao listar as informações que circulam no ambiente, essas são:

- Compromissos (atividades)
- Identificações pessoais
- Dados de contato (telefone, e-mail, endereço)
- Horário
- Fuso-Horário
- Data
- Localização

A análise das informações do ambiente pode ser utilizada para refinar o grafo de objetivos do passo 1, caso seja necessário.

Ao saber as informações que trafegam no ambiente é mais fácil definir os objetos Contexto que devem ser representados nos diagramas do modelo. Na figura 32 são mostrados os contextos Localização e Urgência, relativo às questões de tempo do agendamento. A amarração entre o contexto de localização e o pedido de agendamento ocorre devido à funcionalidade que faz com que, a partir da localização física, a agenda possa deduzir quem está envolvido no compromisso sendo agendado.

5.6 Passo 5

- Agrupamento dos serviços existentes no projeto do ambiente em escopos (conjuntos de serviços que possam prover um mesmo tipo de funcionalidade). Os escopos serão utilizados mais adiante e podem ser organizados simplesmente listando em uma anotação os serviços que o compõem.

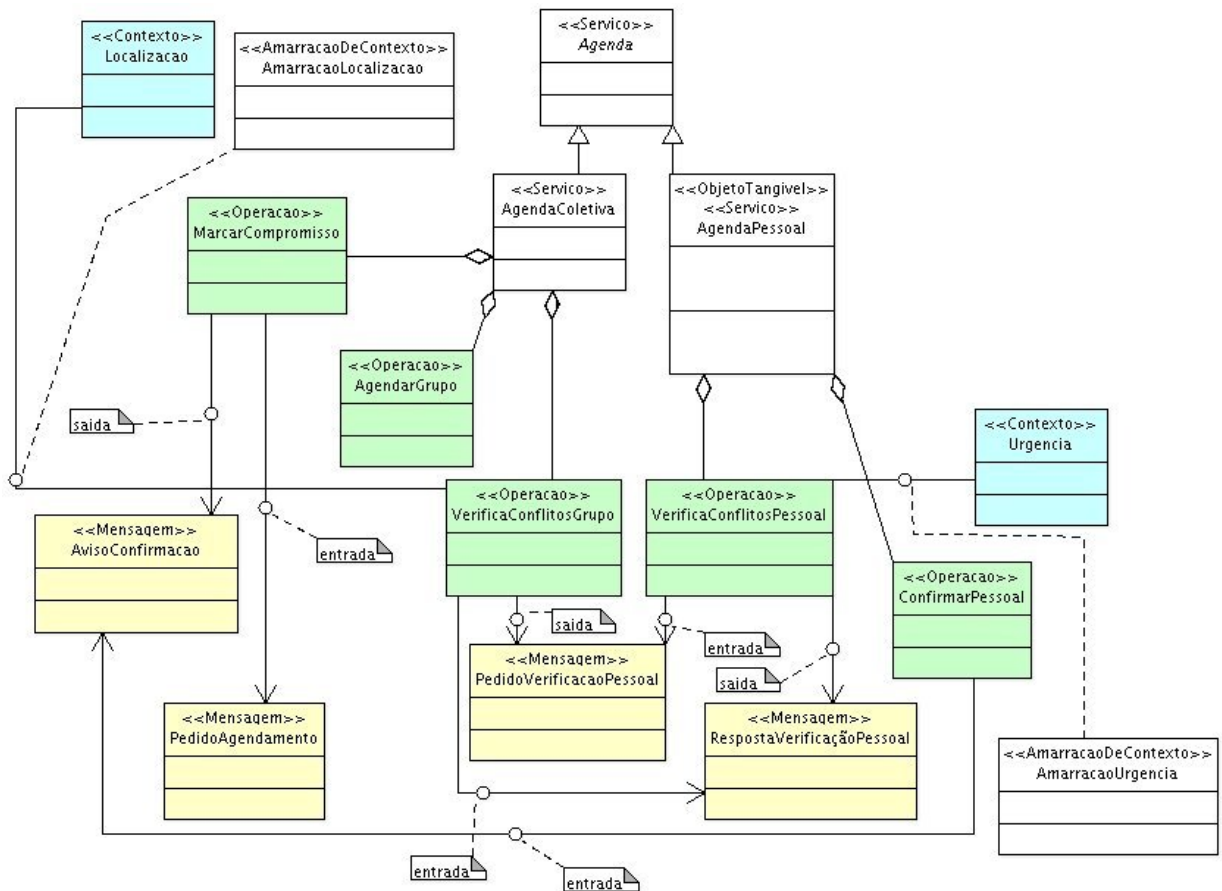


Figura 32: Contextos no diagrama de classes

Neste passo é útil analisar também os recursos já presentes no ambiente, no caso, os que compõem o Mensageiro Instantâneo. Uma possibilidade é organizar os escopos de acordo com os tipos de informação que são tratados no ambiente (figura 33):

- Reconhecimento/Identificação: Sensor de Localização utilizado pelo Mensageiro Instantâneo, Webcam (com algum recurso de identificação facial por exemplo) prevista para ser utilizada pela Agenda no ambiente físico de trabalho, *bluetooth* dos celulares dos usuários junto com os dados de números de celular da agenda.
- Dados pessoais (exemplo: é necessário obter alguma informação como e-mail de um usuário que não está disponível no sistema, para alertá-lo que foi incluído em uma reunião): Agendas dos celulares dos usuários, aplicativos dos computadores presentes no ambiente (lista de contatos do outlook-express, eudora, thunderbird, etc), serviço 102-online.

Outros escopos já presentes na elaboração do modelo do Mensageiro Instantâneo podem ser incluídos.

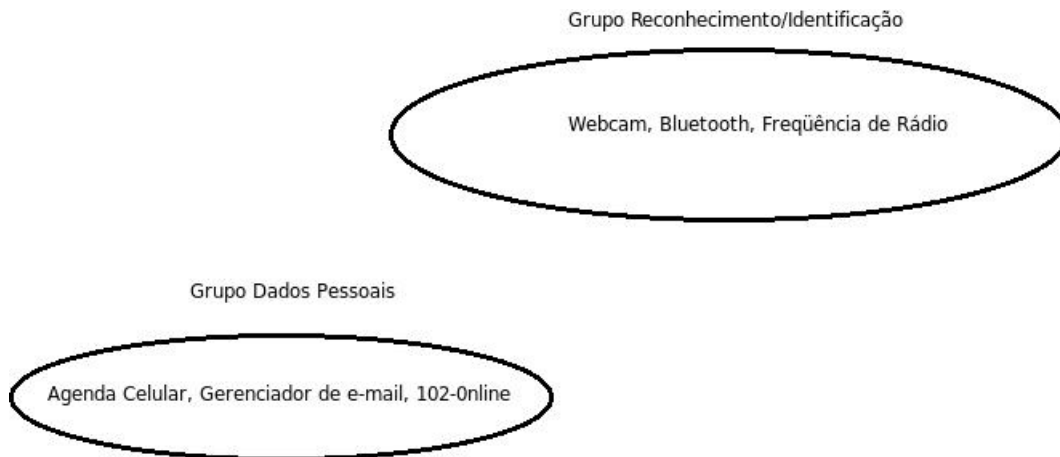


Figura 33: Serviços possíveis no ambiente agrupados em escopo

5.7 Passo 6

- Identificação de *breakdowns* e definição de estratégias de propagação para os mesmos, utilizando os escopos definidos anteriormente. A identificação pode ser feita durante a elaboração dos casos de uso. Se originalmente não foram elaborados os casos de uso, eles devem ser feitos.

Para detectar as falhas(*breakdowns*) é necessário identificar situações indesejadas no comportamento esperado do sistema.

AGENDAMENTO DE COMPROMISSO

- 1.Usuário solicita agendar compromisso de grupo na Agenda
- 2.Agenda solicita dados do compromissos e envolvidos
- 3.Agenda aguarda resposta
 - 3a. A agenda obtém os dados necessários. Continua em 4
 - 3b. A agenda obtém os dados do compromissos mas não dos envolvidos - BREAKDOWN - Estratégia de Propagação PropDadosInsuficientes
 - 3c. A agenda não obtém os dados do compromisso. Volta para 2
- 4.Agenda solicita informações de horários possivelmente conflitantes para as agendas pessoais dos envolvidos
5. Agenda aguarda resposta
 - 5a. Agenda recebe resposta em XX segundos. Continua em 6
 - 5b. Resposta de usuário(s) não retorna em XX segundos - BREAKDOWN - Estratégia de Propagação PropDadosInsuficientes
- 6.Faz o agendamento
 - 6a. Se não houver conflitos de horário confirma o agendamento e atualiza agendas pessoais dos envolvidos
 - 6b. Se houver conflitos que podem ser ajustados mudando agendas pessoais, confirma o compromisso e atualiza agendas pessoais
 - 6c. Se houver conflitos que não podem ser ajustados mudando as agendas pessoais, busca novo horário possível mais próximo

ESTRATÉGIA PropDadosInsuficientes

- 1.Envia a solicitação de dados novamente, porém em um nível mais abrangente (ex.:busca por telefone residencial vira busca por telefone)
- 2.Volta a aguardar a resposta

Figura 34: Identificação de falhas no processo de agendamento e estratégia de propagação.

Na figura 34 foram identificadas duas situações em que acontecem *breakdowns*, porém as duas falhas são causadas pelo mesmo problema, dados não fornecidos ou não fornecidos

em tempo útil. O tratamento para ambos os casos pode ser o mesmo. Ao identificar o contexto em que os dados estão sendo buscados é possível solicitar a mesma busca, porém com um escopo mais abrangente, para incluir fontes que anteriormente não participaram da busca original.

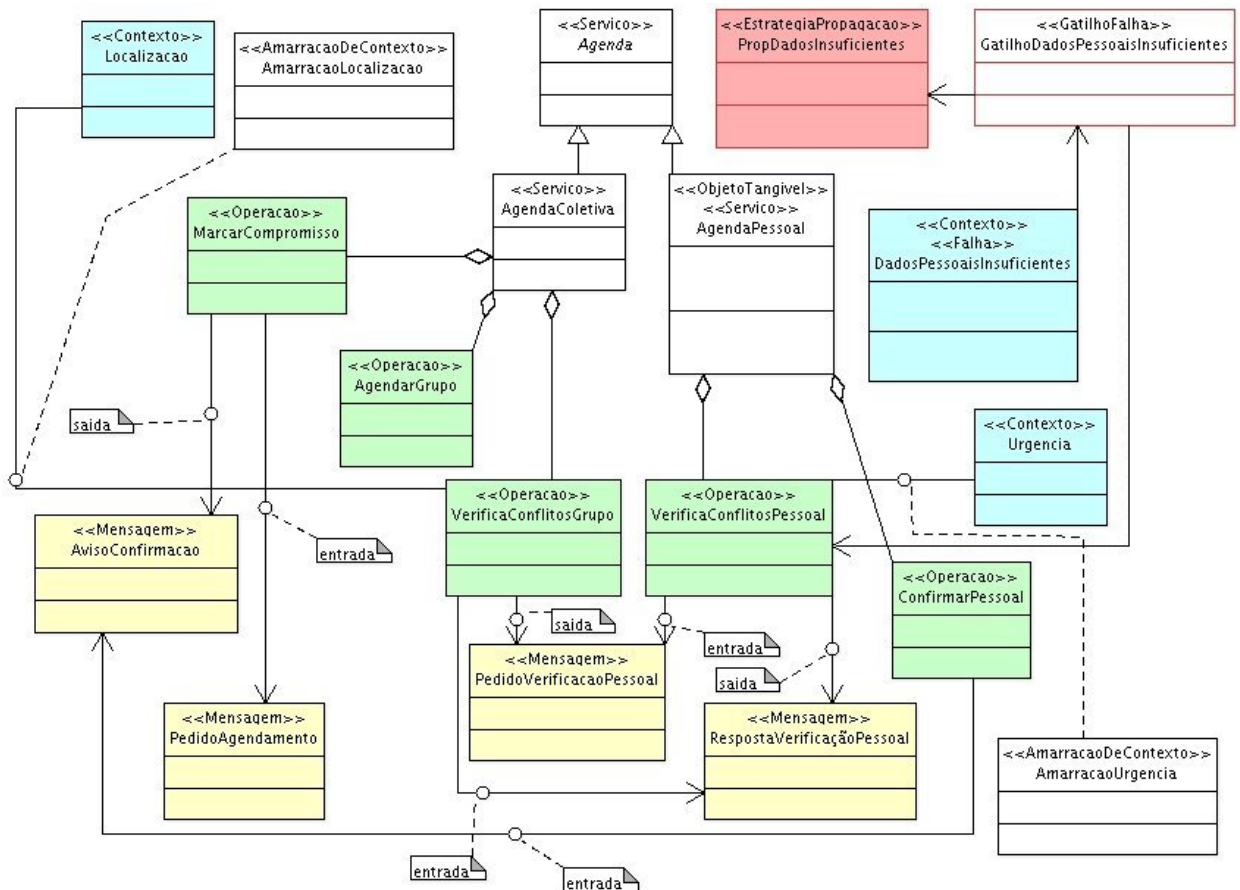


Figura 35: Relacionamento entre Serviços, Operações e Contextos da Agenda.

Durante a elaboração do diagrama de classes são identificadas as operações do sistema que tem falhas identificadas em seu funcionamento, a causa da falha e o tratamento que a operação deve conhecer para realizar quando necessário. A figura 35 mostra em destaque a falha e estratégia de propagação mostradas na figura 36. As informações sobre as falhas foram mostradas em figuras diferentes para evitar que devido ao espaço limitado para a sua apresentação, o diagrama se tornasse difícil de ser visualizado.

A mesma estratégia é reutilizada para a falha quando o usuário não fornece todas as informações necessárias. Na figura 37 há um serviço Webcam que pode fazer a identificação por reconhecimento facial dos usuários. Caso o usuário não forneça o nome dos envolvidos no compromisso a agenda pode usar a estratégia de propagação definida e solicitar a identificação para os serviços disponíveis no momento e o pedido seria atendido

pela Webcam, sem necessidade de interromper o usuário para solicitar entrada de dados.

No diagrama da figura 37, a amarração entre o contexto de localização e a operação de verificação de conflitos foi substituída pela falha e a estratégia de propagação.

Na figura 38 está ilustrado o comportamento da estratégia de propagação. Quando houverem dados insuficientes, eles devem ser novamente solicitados para outros serviços capazes de provê-los (escopo do grupo de dados pessoais). Caso os dados ainda não possam ser obtidos (como no caso de o usuário não fornecer os nomes dos envolvidos na reunião sendo agendada) a exceção é propagada para o grupo reconhecimento/identificação. Ele irá retornar a identificação para o tratador invocado no primeiro escopo, que por sua vez retornará a resposta, sem que o usuário perceba a falha no funcionamento.

5.8 Passo 7

- Durante a criação dos modelos dinâmicos, certificar-se de anotar no modelo restrições de tempo de resposta para as ações ou outro tipo de restrição necessária para garantir a qualidade da informação tratada e conseqüentemente do serviço provido. Caso não seja utilizado nenhum recurso para poder fazer a marcação das restrições de tempo e elas forem necessárias, um diagrama de atividade ou de sequência deve ser elaborado.

A Agenda depende da resposta para as buscas de dados no ambiente ser rápida o bastante para não perturbar o usuário que deseja marcar um compromisso.

No diagrama de atividades, a atividade descrita no item 5b de "Agendamento de Compromissos" da figura 34, tem a limitação de tempo para a resposta, que causa uma falha. A restrição de tempo é identificada no comentário relacionado com a ação "Agendas Pessoais Verificam Conflitos", conforme pode ser visto na figura 39.

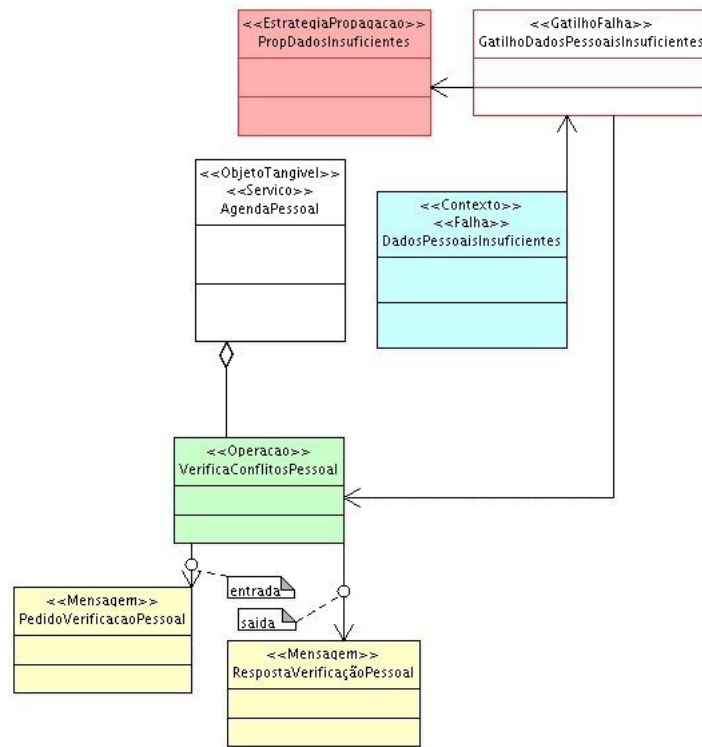


Figura 36: Falha causada por dados insuficientes ou demora na resposta.

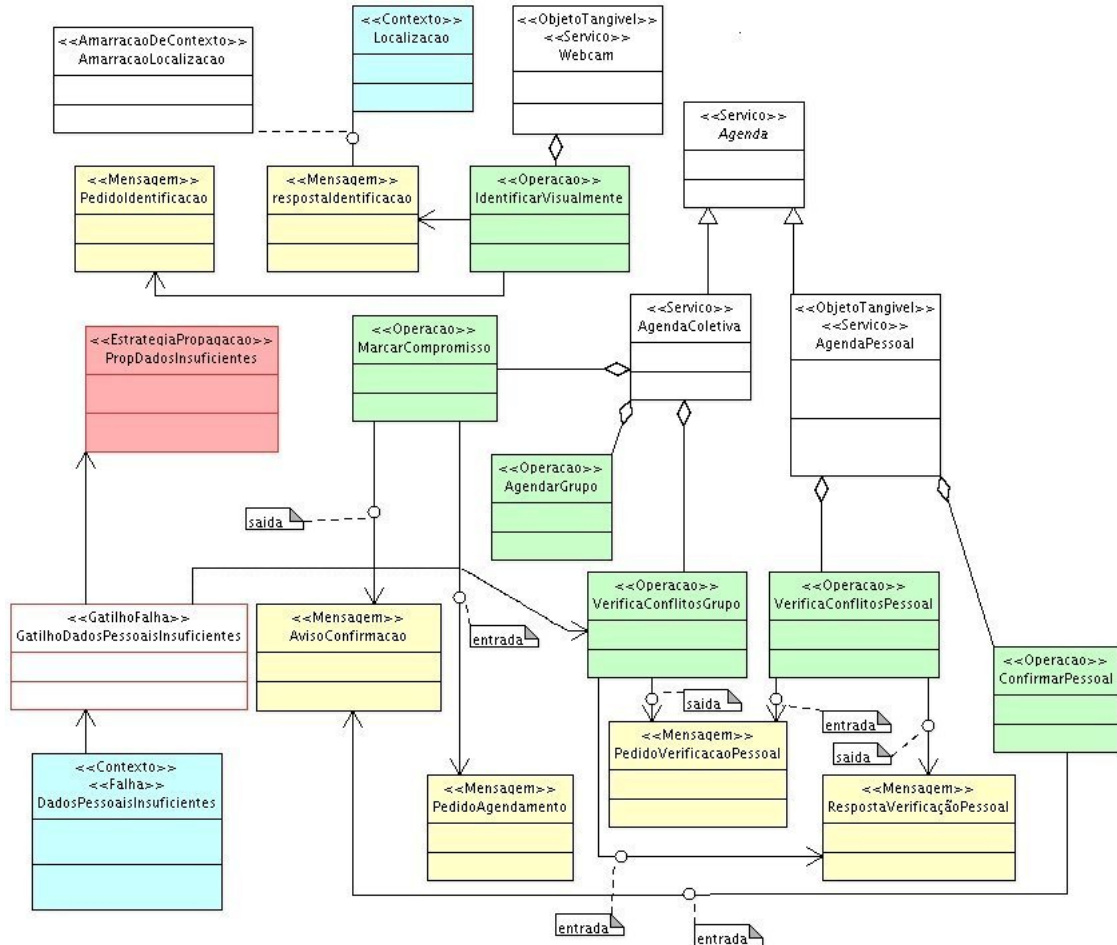


Figura 37: Outra situação em que é aplicada a mesma estratégia.

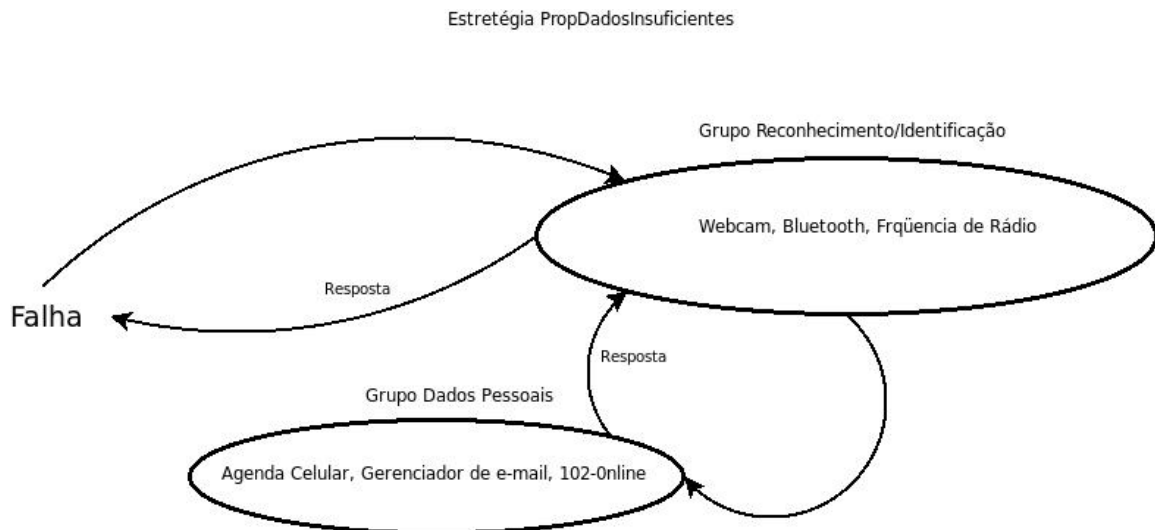


Figura 38: Estratgia de Propagaao

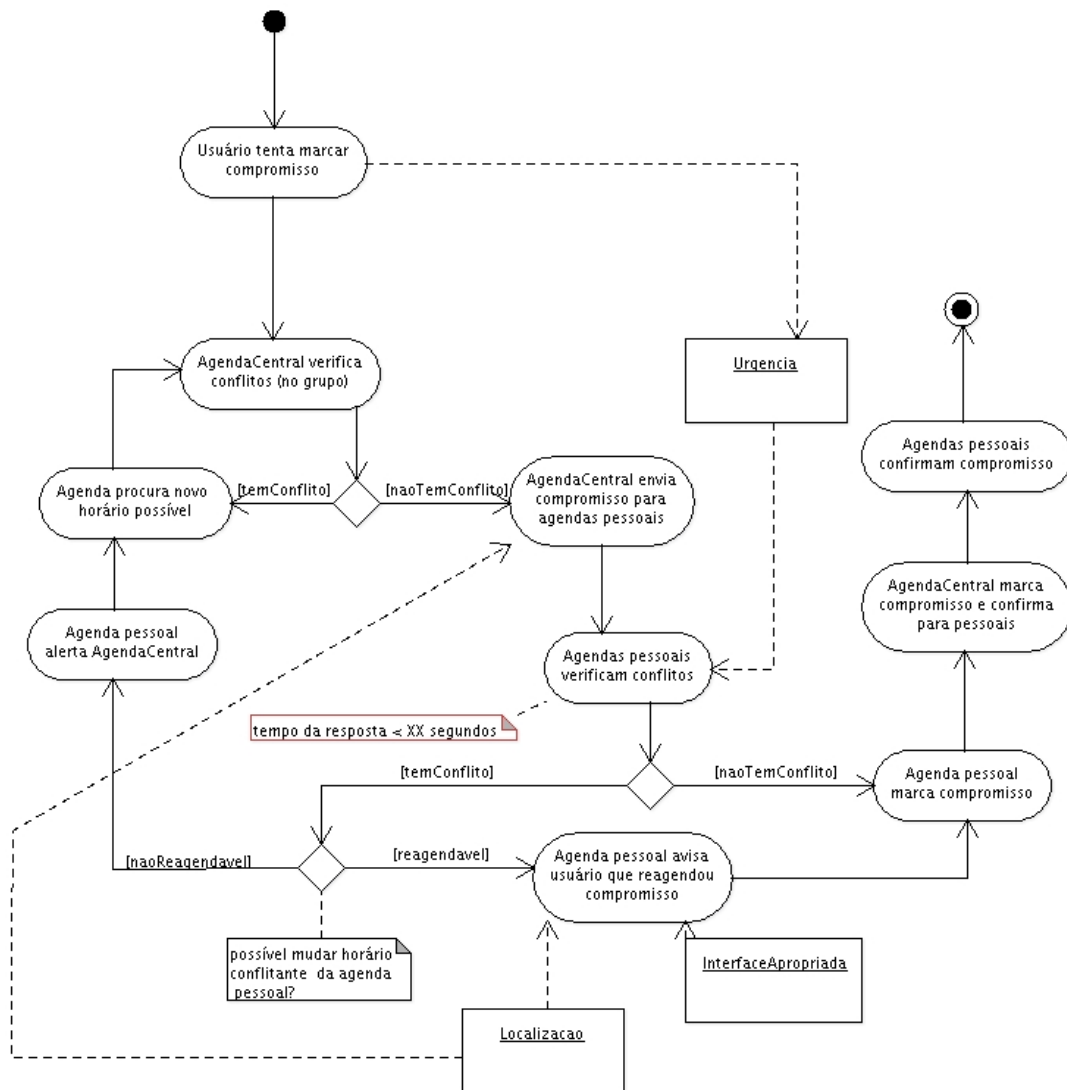


Figura 39: Tempo limite para resposta.

6 Conclusão e Trabalhos Futuros

O objetivo deste trabalho foi buscar soluções para deficiências referentes à modelagem de aplicações dependentes de aspectos físicos do ambiente, sensibilidade a contexto ou que podem ter seu funcionamento alterado devido a mudanças no ambiente físico ou virtual em que se localizam, acrescentando a modelagem destas características aos métodos de desenvolvimento tradicionais. Para tanto, foram pesquisadas publicações de diversas áreas, como sistemas distribuídos, engenharia de software, sistemas embutidos, web-semântica, etc, para encontrar material de referência para ser utilizado como base para o seu desenvolvimento. Dentre os trabalhos estudados, ContextUML serviu de base para o desenvolvimento de uma proposta, que incorporou elementos de outras publicações.

A diferenciação entre objetos que existem no mundo físico e também no mundo digital e que podem alterar o funcionamento das aplicações existentes no ambiente, quando alterados no mundo físico, de outros que existem somente no mundo digital, é feita através do conceito de Objetos Tangíveis (14).

A relação entre Serviço e Operação de ContextUML foi alterada, permitindo que diversos serviços ofereçam uma mesma funcionalidade. A relação de amarração e de gatilho entre um serviços e um contexto foi mantida, porém foi acrescentado a possibilidade da relação identificar uma falha no ambiente, descrita por Ingstrup (10), situação em que os mecanismos que provêm a funcionalidade se fazem perceber, seja por não atenderem adequadamente a um serviço requisitado ou por pelo serviço requisitado não existir no ambiente.

Como as aplicações devem ser capazes de se adaptar ao ambiente conforme ele varia, é inviável listar todos os requisitos que um usuário espera do serviço, pelo simples fato de que muitas funcionalidades na qual um serviço contribui, podem surgir da colaboração de outras aplicações. A utilização do framework NFR (4) foi incorporada na proposta elaborada neste trabalho, para relacionar as diversas metas pretendidas para o ambiente e como a realização de uma interfere na realização das outras, o que permite prever como o

ambiente irá reagir quando uma mudança, como por exemplo, a adição de um novo serviço, for realizada. Ao saber como o novo serviço contribui ou prejudica a realização de uma meta, é possível analisar o grafo com as metas e visualizar como que as características do novo serviço vão influenciar o ambiente.

Se o desenvolvedor for capaz de modelar em alguns cenários, como que os diferentes serviços funcionam em conjunto e quais situações seriam indesejáveis para um usuário, ele pode definir estratégias de tratamento para estas situações e criar soluções genéricas que podem atender a situações que ainda não foram previstas. As características físicas do ambiente podem ser modeladas com um recurso simples como um mapa aproximado do ambiente, uma solução eficiente para fornecer um meio de visualizar o espaço físico envolvido no sistema.

Enquanto individualmente os trabalhos estudados atendiam somente a alguns dos requisitos definidos no capítulo 2.1, ao reunir características de diversos trabalhos de diferentes áreas, foi possível atender todos requisitos listados e aos objetivos esperados deste trabalho, com exceção da verificação da eficiência do modelo proposto, que fica como sugestão de trabalho futuro.

6.1 Trabalhos Futuros

Uma das dificuldades no desenvolvimento de ambientes de computação ubíqua é não saber o que vai se encontrar no ambiente. Este foi desenvolvido individualmente, assim realizar a implementação do exemplo do serviço Agenda e de todos serviços que a compõem não representaria o resultado da utilização dos recursos de modelagem apresentados no trabalho, que seria obtido em um cenário em que os envolvidos no desenvolvimento não têm conhecimento do funcionamento de todo o sistema, ficando como sugestão de um trabalho futuro.

Além disso, para ter certeza da eficácia do modelo proposto é necessário que mais aplicações sejam modeladas e implementadas, para que sejam realizadas correções e adições ao modelo conforme for necessário.

A modelagem obtida ao se utilizar a proposta elaborada neste trabalho, é a modelagem de todo o ambiente, de como os serviços presentes se relacionam e dos contextos que determinam seus funcionamentos. Porém, ela não aborda aspectos do projeto. A elaboração de um modelo de projeto para auxiliar a implementação dos serviços, operações e os outros componentes, por exemplo, como implementar as estratégias de propagação de

exceção sensíveis ao contexto, utilizadas para tratar as falhas (*breakdowns*) identificadas, é necessária para viabilizar a utilização prática dos resultados do trabalho.

Para que a visão de computação ubíqua, não intrusiva, seja efetivamente realizada, há necessidade de trabalhar com informações de contexto a fim de permitir que os diversos serviços presentes em um ambiente possam executar sem a necessidade de interferência de usuários, seja para invocá-los ou corrigir o funcionamento. Para um computador derivar informações de contexto é necessário que ele compreenda os dados presentes, para depois derivar e representar o contexto. A recuperação e representação de contexto são linhas de pesquisa que podem contribuir e dar continuidade ao trabalho, que abordou a utilização e modelagem da dependência de um serviço em relação a um contexto, mas ignorou a modelagem do contexto em si.

Referências

- 1 *Tutorial 3 SBES: Desenvolvimento de Software Orientado a Agentes.*
- 2 *Take me with you! A Case Study of Context-Aware Application integrating Cyber and Physical Spaces*, Mar. 2004.
- 3 Douglas Adams. *O Guia do Mochileiro das Galáxias*. Editora Sextante.
- 4 Lawrence Chung and Brian A. Nixon. Dealing with non-functional requirements: Three experimental studies of a process-oriented approach. In *ICSE17*, pages 25–37. ACM, apr 1995.
- 5 Gianpaolo Cugola, Elisabetta Di Nitto, and Alfonso Fuggetta. The jedi event-based infrastructure and its application to the development of the opss wfms. *IEEE Transactions on Software Engineering*, 27(9):827–850, Sep 2001.
- 6 Karla Damasceno, Nélio Cacho, Alessandro Garcia, and Carlos Lucena. Tratamento de exceções sensível ao contexto. In *Anais do XX Simpósio Brasileiro de Engenharia de Software (SBES)*.
- 7 Michael Derntl and Karin Hummel. Modeling context-aware e-learning scenarios, 2005.
- 8 Anind K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5:4–7, 2001.
- 9 G.J. Doherty and M.D. Wilson. Modelling ubiquitous computing applications. *Proceedings of International Workshop on Continuity in Future Computing Systems*, 2001.
- 10 Mads Ingstrup. Modeling for pervasive computing: Organizational aspects of system analysis and designs. Master’s thesis, University of Southern Denmark, 2003.
- 11 Brad Johanson, Armando Fox, and Terry Winograd. The interactive workspace project: Experiences with ubiquitous computing rooms. *IEEE Pervasive Computing*, Apr 2002.
- 12 Bradley Johanson. Application coordination infrastructure for ubiquitous computing rooms. Master’s thesis, Stanford, Dec 2002.
- 13 Peter Marwedel. *Embedded System Design*. Kluwer Academic Publishers, 2003.
- 14 Daniel Chien-Meng May. Tango: Designing for the digitally pervasive world. Master’s thesis, University of Southern Denmark, 2003.

- 15 Thomas Pederson. Towards a unified model of simple physical and virtual environments, 2003.
- 16 Monique Casagrande Pizzetti. Uma abordagem estratégica de integração da tecnologia rfid, 2007.
- 17 M. Satyanarayanan. Pervasive computing: Vision and challenges. *IEEE Personal Communications*, 2001.
- 18 Quan Z. Sheng and Boualem Benatallah. Contextuml: A uml-based modelling language for model-driven development of context-aware web services. *Proceedings of the International Conference of Mobile Business*, 2005.
- 19 Leon A. Watts, Joule Coutaz, David Thevenin, Emmanuel Dubois, Mieke Massink, and Gavin Doherty. Environmental interactive systems: principles of systematic digital-physical fusion. Technical report, CLIPS-IMAG, France and Instituto CNUCE, Italy and Chilton Didcot, UK, 2000.
- 20 Mark Weiser. The world is not a desktop. *Interactions*, pages 7–8, Jan 1994.