UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ALEXANDRE GUILHERME NICCO COELHO

Apoio à Gerência de Recursos em ODE

Vitória
2007

ALEXANDRE GUILHERME NICCO COELHO

Apoio à Gerência de Recursos em ODE

Monografia apresentada à Universidade Federal do Espírito Santo como requisito parcial para obtenção do título de Bacharel em Ciência da Computação, na área de concentração de Sistemas de Informação, sob orientação do professor Ricardo de Almeida Falbo.

Vitória
2007

ALEXANDRE GUILHERME NICCO COELHO

Apoio à Gerência de Recursos em ODE

Aprovada em 28 de Fevereiro de 2007.

COMISSÃO EXAMINADORA

Prof. Ricardo de Almeida Falbo, D.Sc.

Orientador

Prof. Davidson Cury, D.Sc.

Profa. Monalessa Perini Barcellos, M.Sc.

Agradeço primeiramente a Deus, por tudo que tem me dado. Ao professor Falbo, não só por ter me concedido todo apoio necessário para a realização deste trabalho, mas também pela sua **extraordinária** amizade. A minha família, pela educação, sustento e conhecimento que me deram durante toda a minha vida. À equipe do LabES (Bruno Segrini, Aline, Vítor, Thiago, Gabriel, André, Bruno Nandolpho, etc) por todas as informações que me passaram. Aos meus amigos, colegas, parentes, enfim, todos aqueles que sempre desejaram o melhor para a minha vida. Eu, verdadeiramente, agradeço.

RESUMO

Uma das áreas que se destaca na gerência de projetos é a gestão de recursos. Em geral, as organizações sempre precisam gerenciar seus recursos, sejam eles recursos humanos, de software, de hardware, enfim, recursos que são necessários a projetos.

Para prover uma melhor gestão, o gerenciamento de recursos pode ser feito com apoio automatizado, ou seja, ferramentas de software que podem informatizar, pelo menos parcialmente, o processo de gestão de recursos em uma organização. Nesse contexto, é fundamental que, dentre as ferramentas contidas nesse meio, existam funcionalidades que apóiem, sobretudo, a alocação de recursos.

O principal objetivo deste trabalho foi desenvolver e melhorar funcionalidades que automatizam o processo de alocação de recursos do ambiente de desenvolvimento de software, ODE. Para realizar tais tarefas, duas ferramentas do ambiente ODE foram evoluídas, a ferramenta de Apóio à Alocação de Recursos e a ferramenta de Agenda, sendo estas apresentadas no decorrer deste documento.

SUMÁRIO

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

Capítulo 1

Introdução

A gerência de projetos inicia-se com um conjunto de atividades conhecido como planejamento de projeto. Antes de começar um projeto, o gerente de projeto e a equipe de software precisam estimar o trabalho a ser feito, os recursos necessários, além do tempo que será preciso para a realização do projeto (PRESSMAN, 2002).

Estimar os recursos necessários para realizar o esforço do desenvolvimento é uma tarefa de grande importância. Quando se trata de recursos, estes englobam pessoas, software e hardware.

No caso de recursos humanos, vários fatores devem de ser considerados, dentre eles a competência para realizar a atividade para a qual está sendo alocado. Com isso, é necessário analisar as competências de cada membro da equipe para poder realizar a alocação de recursos. Além disso, outros fatores externos importantes para a formação de equipe, como liderança, relacionamento inter-pessoal entre outros, são igualmente considerados para a alocação de recursos a atividades.

No que se refere a software, deve-se pensar em ferramentas de software, tais como ferramentas CASE ou software de infra-estrutura (por exemplo, um sistema operacional), além de itens de software passíveis de reutilização no desenvolvimento, tais como bibliotecas de interface ou uma camada de persistência.

Em todos os casos, sejam eles recursos humanos, de software e de hardware, é necessário analisar a disponibilidade do recurso. Assim, é importante definir a partir de que data o recurso será necessário, por quanto tempo será necessário, além da quantidade de horas por dia necessária durante esse tempo (FALBO, 2002).

Dada a complexidade envolvida, é importante oferecer apoio automatizado a essa atividade, com o objetivo de facilitar, organizar e estruturar o trabalho relacionado à alocação de

recursos em um projeto. Como essa tarefa é altamente interligada a outras, não só de planejamento, mas também do restante do processo, o ideal é que esse apoio automatizado se dê de forma integrada, de forma que se tenha um ambiente com várias ferramentas integradas, isto é, Ambientes de Desenvolvimento de Software (ADSs), que visam a integrar técnicas, métodos e ferramentas para auxiliarem o Engenheiro de Software no desenvolvimento de sistemas.

1.1 Motivação

Atualmente, no ambiente ODE (*Ontonlogy-based software Development Environment*) (FALBO et al., 2004a), existem duas ferramentas que estão relacionadas à alocação de recursos. Uma delas tem como objetivo definir uma equipe de um projeto e, após definida a equipe, auxiliar a alocação de recursos humanos dessa equipe, bem como recursos de software, a atividades de um projeto. A outra é a ferramenta de agenda. Essa tem como objetivo lembrar os desenvolvedores acerca das atividades às quais estão alocados dentro de projetos gerenciados no ambiente ODE.

Com uma análise crítica feita nessas ferramentas, foi possível encontrar melhorias consideráveis para a evolução das mesmas. Assim, este trabalho tem como foco a criação e reestruturação dessas ferramentas do ambiente ODE, para que os recursos de um projeto possam ser melhor gerenciados e organizados.

1.2 Contexto e Objetivos do Trabalho

Este trabalho foi desenvolvido no contexto do Projeto ODE no Laboratório de Engenharia de Software (LabES), localizado no Departamento de Informática (DI) da Universidade Federal do Espírito Santo (UFES). Atualmente, o projeto é desenvolvido por alunos de graduação e de pós-graduação do Departamento de Informática da UFES.

ODE é um Ambiente de Desenvolvimento de Software (ADS) centrado em processo e baseado em ontologias. Nele existem ferramentas que apóiam diversas atividades da gerência de projetos, dentre elas ferramentas de acompanhamento de projetos (DAL MORO et al., 2005), definição de processos (BERTOLLO et al., 2006), gerência de riscos (FALBO et al., 2004b),

realização de estimativas (CARVALHO et al., 2006) e alocação de recursos (SCHWAMBACH, 2002).

Conforme citado anteriormente, no que se refere à parte de alocação de recursos, existem duas ferramentas relacionadas, uma que propriamente permite ao gerente de projeto definir a equipe e as alocações a atividades de um projeto (SCHWAMBACH, 2002) e outra que auxilia, na forma de agenda, a orientar cada desenvolvedor a respeito de suas alocações a atividades (PEZZIN, 2002).

Apesar de haver essas ferramentas de apoio à alocação de recursos, há uma necessidade de que elas sejam aperfeiçoadas para atender às evoluções do ambiente. É justamente nesse contexto que este trabalho foi realizado, com o objetivo principal de evoluir as ferramentas atuais de alocação de recursos e de agenda existentes em ODE.

As novas ferramentas, agora denominadas AlocaODE e AgendaODE, permitem um maior controle sobre a parte que trata de alocação de recursos do ambiente, sendo possível controlar com mais detalhes o andamento das alocações de recursos humanos, alocar recursos de hardware, visualizar uma agenda mais completa para cada desenvolvedor, permitindo desde registrar esforço até selecionar ferramentas de software disponíveis para o trabalho, além de reestruturações quanto à usabilidade das mesmas.

1.3 Metodologia

A metodologia adotada para o desenvolvimento deste trabalho iniciou-se com uma revisão bibliográfica, incluindo estudos relacionados ao contexto do ambiente ODE, bem como análises feitas para a reestruturação das ferramentas de apoio à alocação de recursos e agenda já existentes.

Na revisão bibliográfica foram estudados trabalhos acadêmicos, de graduação e de iniciação científica, *sites* na Internet, livros e notas de aula acerca dos assuntos relacionados à gerência de projetos, Ambientes de Desenvolvimento de Software, alocação de recursos, gestão de pessoas em projetos de software, ferramentas de agenda e componentes gráficos para construção de interfaces *desktop* em Java.

Logo após a revisão bibliográfica, definiu-se como escopo do projeto a reestruturação das ferramentas de alocação de recursos e agenda, de modo a melhorar todo o processo de

alocação de recursos do ambiente ODE, desde a criação de novas funcionalidades até a melhoria da usabilidade das mesmas.

Com o escopo definido, iniciou-se o processo de desenvolvimento das novas ferramentas, passando pelas atividades de especificação de requisitos, análise, projeto, implementação e testes. Paralelamente, foi-se elaborando esta monografia.

1.4 Organização do Trabalho

Este trabalho está organizado em seis capítulos e três anexos. Além deste capítulo, a Introdução, existem ainda os capítulos e anexos descritos a seguir.

O Capítulo 2 – *Apoio Automatizado à Gerência de Projetos* – aborda os temas gerência de projetos e automatização do processo de gerência de projetos. Além disso, apresenta informações específicas para este trabalho.

O Capítulo 3 – *O Processo de Software Adotado* – apresenta o processo de software adotado neste trabalho, tratando sucintamente o ciclo de vida e as principais atividades realizadas no processo.

O Capítulo 4 – *Especificação de Requisitos e Análise* – apresenta a especificação de requisitos funcionais e a modelagem de análise para as duas ferramentas desenvolvidas neste trabalho, sendo que, para a definição dos requisitos funcionais, utilizou-se Diagramas de Casos de Uso e suas descrições. Já para a parte de análise, foram elaborados diagramas de classes e de estados, utilizando a UML. Por fim, o capítulo apresenta os requisitos não funcionais considerados para ambas as ferramentas.

O Capítulo 5 – *Projeto, Implementação e Testes* – apresenta os modelos de projeto construídos para as ferramentas, considerando aspectos tecnológicos. Além disso, o capítulo mostra aspectos relacionados à implementação, apresentando algumas interfaces das ferramentas desenvolvidas, e testes.

O Capítulo 6 – *Considerações Finais* – apresenta as conclusões feitas sobre o desenvolvimento do trabalho, além de discutir perspectivas e propostas para trabalhos futuros.

O Anexo A – *Documentação da Ferramenta de Alocação de Recursos* – apresenta os documentos completos de especificação de requisitos funcionais, análise e projeto da ferramenta de Alocação de Recursos.

O Anexo B – *Documentação da Ferramenta de Agenda* – apresenta os documentos completos de especificação de requisitos funcionais, análise e projeto ferramenta de Agenda.

O Anexo C – *Documentação da Interface Baseada em Fluxo de Tarefas do Ambiente ODE* – apresenta de forma detalhada o projeto da interface baseada em fluxo de tarefas desenvolvida para o ambiente ODE.

Capítulo 2

Apoio Automatizado à Gerência de Projetos

No desenvolvimento de software, a necessidade de gerenciamento é uma importante distinção entre desenvolvimento profissional de software e o desenvolvimento em nível amador. É extremamente necessário o gerenciamento de projetos de software, pois a Engenharia de Software profissional enfrenta restrições tanto de orçamento como de prazo.

Os gerentes de projetos de software são responsáveis por planejar e programar todo o desenvolvimento de um sistema. Eles supervisionam o projeto para assegurar que seja realizado de acordo com os padrões requeridos e monitoram o processo para verificar se o projeto está dentro do prazo e do orçamento estabelecidos (SOMMERVILLE, 2003).

Dentre todas as responsabilidades designadas ao gerente de projeto, uma que se destaca é o processo de estimar os recursos necessários para a realização do projeto, sejam eles recursos humanos ou recursos de apoio, tais como ferramentas de software e equipamentos de hardware.

Para a realização do projeto, é necessário que o gerente adquira, de alguma forma, os recursos estimados para o projeto. Um aspecto importante nessa etapa é a seleção do pessoal para trabalhar no projeto que, geralmente, envolve a definição de uma equipe, composta por pessoas com determinadas habilidades, competências e experiências, para a realização das atividades de todo o projeto.

Este capítulo apresenta as diversas áreas estudadas e contempladas neste trabalho. A Seção 2.1 apresenta uma discussão sobre Gerência de Projetos e as áreas de conhecimento do Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos – O Guia PMBOK (*Project Management Body Of Knowledge*) (PMI, 2004). A Seção 2.2 aborda a área de conhecimento de gerência de tempo. A Seção 2.3 trata com mais detalhe a área de conhecimento de gerência de recursos humanos. A Seção 2.4 trata da automatização do processo de software. A Seção 2.5 apresenta o ambiente ODE e suas ferramentas. Finalmente, a Seção 2.6 abre uma discussão sobre a automatização da gerência de projetos em ODE.

2.1 Gerência de Projetos

A disciplina gerência de projeto nasceu na indústria bélica e aeroespacial americana na década de 1960. Logo depois, foi adotada na construção civil e todas as outras áreas de engenharia. Atualmente, seu conceito passou a ser entendido e aplicado em diferentes setores da economia, incluindo a política (MARTINS, 2005).

Devido ao crescente interesse por essa disciplina, surgiu o *Project Management Institute* (PMI) com o objetivo de regularizar e distribuir o conhecimento aplicado na gerência de projetos. O PMI é uma entidade sem fins lucrativos que congrega os profissionais de áreas relacionadas à Gerência de Projetos. Sua missão é promover o profissionalismo e desenvolver o "estado da arte" na gestão de projetos (MARTINS, 2005), especificando um conjunto de procedimentos que visam padronizar a teoria do gerenciamento de projetos. Essa teoria de gestão de projetos definida pelo PMI está registrada em um documento conhecido como Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos (*Project Management Body of Knowledge* - PMBOK), que serve de orientação para os profissionais da área.

No PMBOK, foram definidas nove áreas de conhecimento relacionadas à gerência de projetos (PMI, 2004):

1. **Gerência de Integração**: inclui os processos e as atividades necessárias para identificar, definir, combinar, unificar e coordenar os diversos processos e atividades de gerenciamento de projetos dentro dos grupos de processos de gerenciamento de projetos. A integração inclui características de unificação, consolidação, articulação e ações integradoras que são essenciais para o término do projeto, para atender com sucesso às necessidades do cliente e de outras partes interessadas e para gerenciar as expectativas. A integração, no contexto do gerenciamento de um projeto, consiste em fazer escolhas sobre em que pontos concentrar recursos e esforços, antecipando possíveis problemas, tratando-os antes de se tornarem críticos, e coordenando o trabalho, visando ao bem geral do projeto. O esforço de integração também envolve fazer compensações entre objetivos e alternativas conflitantes.
2. **Gerência de Escopo:** envolve os processos necessários para garantir que o projeto inclua todo o trabalho necessário, e somente ele, para terminar o projeto com sucesso. O

gerenciamento do escopo do projeto trata principalmente da definição e controle do que está e do que não está incluído no projeto.

3. **Gerência de Tempo:** inclui os processos necessários para concluir o projeto no prazo. Os processos contemplados no gerenciamento do tempo são:
    - **Definição de atividades:** identificação das atividades específicas do cronograma que precisam ser realizadas para produzir as várias entregas do projeto;
    - **Seqüenciamento de atividades**: identificação e documentação das dependências entre as atividades do cronograma;
    - **Estimativa de recursos das atividades:** estimativa do tipo e das quantidades de recursos necessários para realizar cada atividade do cronograma;
    - **Estimativa de duração das atividades:** estimativa do número de períodos de trabalho que serão necessários para terminar as atividades individuais do cronograma;
    - **Desenvolvimento do cronograma**: análise dos recursos necessários, restrições do cronograma, durações e seqüências de atividades para criar o cronograma do projeto;
    - **Controle do cronograma:** controle de mudanças no cronograma do projeto.

4. **Gerência de Custos:** inclui os processos envolvidos no planejamento, estimativa, definição do orçamento e controle de custos, de modo que seja possível terminar o projeto dentro do orçamento aprovado.
5. **Gerência da Qualidade:** Os processos de gerenciamento da qualidade do projeto incluem todas as atividades da organização executora que determinam as responsabilidades, os objetivos e as políticas de qualidade, de modo que o projeto atenda às necessidades que motivaram sua realização. Eles implementam o sistema de gerência da qualidade por meio de políticas, procedimentos e processos de planejamento da qualidade, garantia da qualidade e controle da qualidade, com atividades de melhoria contínua dos processos conduzidas do início ao fim, conforme necessário.
6. **Gerência de Recursos Humanos:** inclui os processos que organizam e gerenciam a equipe do projeto. Nessa área de conhecimento, os processos contemplados são:

- **Planejamento de recursos humanos:** envolve a identificação e documentação de funções, responsabilidades e relações hierárquicas do projeto, além da criação do plano de gerenciamento de pessoal;
- **Contratar ou mobilizar a equipe do projeto:** refere-se à obtenção dos recursos humanos necessários para terminar o projeto;
- **Desenvolver a equipe do projeto:** trata da melhoria de competências e interação de membros da equipe para aprimorar o desempenho do projeto;
- **Gerenciar a equipe do projeto:** envolve o acompanhamento do desempenho de membros da equipe, fornecimento de feedback, resolução de problemas e coordenação de mudanças para melhorar o desempenho do projeto.

7. **Gerência de Comunicação:** é a área de conhecimento que emprega os processos necessários para garantir a geração, coleta, distribuição, armazenamento, recuperação e destinação final das informações sobre o projeto de forma oportuna e adequada.
8. **Gerência de Riscos:** inclui os processos que tratam da identificação, análise, respostas, monitoramento e controle e planejamento do gerenciamento de riscos em um projeto. A maioria desses processos é atualizada durante todo o projeto. Os objetivos do gerenciamento de riscos do projeto são aumentar a probabilidade e o impacto dos eventos positivos e diminuir a probabilidade e o impacto dos eventos adversos ao projeto.
9. **Gerência de Aquisições:** inclui os processos para comprar ou adquirir os produtos, serviços ou resultados necessários de fora da equipe do projeto. Além disso, são contemplados os processos de gerenciamento de contratos e de controle de mudanças necessários para administrar os contratos ou pedidos de compra emitidos por membros autorizados da equipe do projeto. O gerenciamento de aquisições do projeto também inclui a administração de qualquer contrato emitido por uma organização externa (o comprador) que está adquirindo o projeto da organização executora (o fornecedor) e a administração de obrigações contratuais estabelecidas para a equipe do projeto pelo contrato.

Este trabalho é focado nas disciplinas de gerência de tempo e gerência de recursos, notadamente nesta última.

2.2 Gerência de Tempo, Custos e Recursos

O tempo é um recurso fundamental e, sendo suficiente, uma equipe de desenvolvimento de software pode desenvolver um produto de alta qualidade, projetando-o cuidadosamente e o testando por completo.

Contudo, nem sempre se tem o tempo necessário disponível. As pressões externas do mercado fazem com que os produtos sejam vendidos antes mesmo que os concorrentes o vendam, ou até mesmo com que serviços sejam oferecidos no tempo exigido pelos clientes. Além disso, pressões da coordenação obrigam a entregar um produto enquanto outros ainda estão sendo desenvolvidos, de acordo com os cronogramas de integração e testes estabelecidos. Com isso, é importante ter um entendimento sobre as relações entre custos, cronograma e qualidade para planejar o desenvolvimento e a manutenção, sem prejudicar funções ou a qualidade (PFLEEGER, 2004).

Um dos aspectos importantes no planejamento e gerenciamento de projetos é a compreensão de quanto o projeto provavelmente custará. O orçamento do projeto é feito com base em vários tipos de custos: instalações, pessoal, métodos e ferramentas, sendo que os custos de instalações incluem hardware, espaço físico, mobília, telefones, modens, sistemas de aquecimento ou ar-condicionado, cabos, discos, papel, caneta, copiadoras e demais itens que completam o ambiente em que os desenvolvedores trabalharão. Para alguns projetos, esse ambiente já pode existir, o que torna mais fácil a compreensão e a estimativa de recursos (PFLEEGER, 2004).

Além de estimar o esforço requerido e o custo para o desenvolvimento de um software, os gerentes de projetos também devem estimar quantas pessoas serão necessárias para trabalhar no projeto e quanto tempo levará para desenvolvê-lo. Esse tempo é conhecido como prazo do projeto (SOMMERVILLE, 2003).

Além disso, outros custos de projeto envolvem a aquisição de software ou ferramentas de apoio ao desenvolvimento, conhecidas como ferramentas CASE (*Computer-Aided Software Engeneering*). Essas ferramentas podem ser exigências do cliente ou até mesmo fazerem parte do processo de desenvolvimento de software padrão da empresa.

A Figura 2.1 mostra os recursos envolvidos no desenvolvimento de software como uma pirâmide (PRESSMAN, 2002).

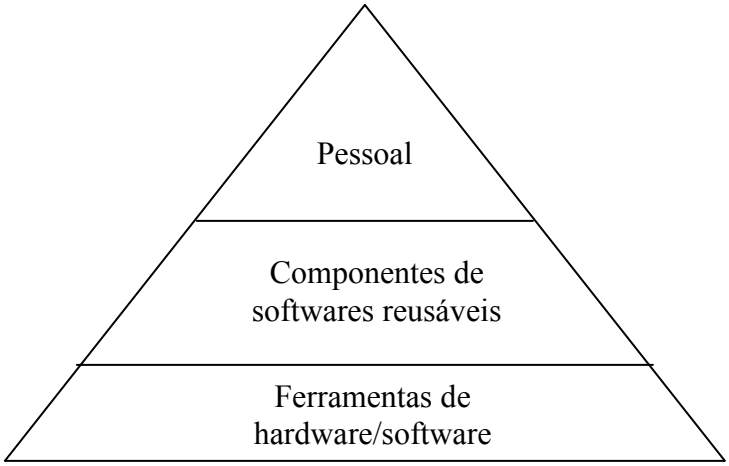

Figura 2.1 – Recursos de desenvolvimento como uma pirâmide

Em geral, o componente de maior custo em um projeto de software é o esforço. O gerente de projeto deve determinar quantas pessoas/dias serão necessárias para realizar o projeto. O esforço é, claramente, um componente com maior grau de incerteza. Estimativas de custos, prazos e esforço devem ser feitas o quanto antes e acompanhadas durante o ciclo de vida de um projeto, pois elas implicam na distribuição de recursos e na viabilidade do projeto. À medida que os aspectos do projeto se modificam, as estimativas devem ser verificadas, podendo ser aprimoradas, com base em informações sobre o andamento do projeto (PFLEEGER, 2004).

2.3 Gerência de Recursos Humanos

Para definir o cronograma do projeto e estimar o esforço e os custos associados, é necessário saber quantas pessoas trabalharão no projeto, quais tarefas serão realizadas por elas e que habilidades e experiência devem ter para realizarem seu trabalho de forma eficiente (PFLEEGER, 2004).

Esta seção trata justamente sobre a importância da gerência de recursos humanos em um projeto de software. As Sub-seções 2.3.1 e 2.3.2 apresentam uma discussão sobre os recursos humanos necessários para um projeto de software e o conceito de tarefas em um projeto, respectivamente.

2.3.1 Pessoal Necessário para o Projeto

Para cada projeto de software, sempre existem atividades ou tarefas necessárias, independentemente do modelo adotado. Para desenvolver um projeto é necessário definir uma equipe para a realização dessas atividades, observando as habilidades, competências e experiências de cada pessoa envolvida.

Várias características podem afetar a capacidade de um indivíduo para realizar o seu trabalho produtivamente (PFLEEGER, 2004), a saber:

- Capacidade para desempenhar o trabalho;
- Interesse no trabalho;
- Experiência com aplicações semelhantes;
- Experiência com ferramentas ou linguagem semelhantes;
- Experiência com técnicas semelhantes;
- Experiência com ambientes de desenvolvimento semelhantes;
- Treinamento;
- Capacidade para se comunicar com as pessoas;
- Capacidade para compartilhar responsabilidades com as pessoas; e
- Habilidades de gerenciamento;

A escolha do pessoal destinado ao projeto pode ter um grande impacto, não somente para a estimativa de tempo, mas também no sucesso do projeto. Segundo (SOMMERVILLE, 2003), três tipos de informação devem ser avaliados na seleção do pessoal para um projeto, a saber:

- A formação e a experiência fornecida pelos candidatos (*curriculum vitae*);
- A informação obtida pela entrevista dos candidatos, e
- As recomendações obtidas por pessoas que tenham trabalhado ou que conheçam os candidatos.

Para entender o desempenho de cada profissional, é necessário conhecer sua capacidade para realizar o trabalho em questão. Alguns deles têm habilidade na visualização “do quadro completo”, mas, por um lado, podem não gostar de se dedicar aos pequenos detalhes, se forem definidos para trabalhar em uma pequena parte de um grande projeto. Essas pessoas podem ser mais adequadas para a concepção do sistema e ou na realização de testes do que atuando nas

etapas de projeto (*design*) ou na codificação do sistema. Em certas ocasiões, a habilidade está relacionada a se sentir "à vontade", o que geralmente faz com que as pessoas tenham mais confiança em sua capacidade para desempenhar certa tarefa, aumentando, assim, a produtividade.

O interesse no trabalho também pode determinar se uma pessoa terá sucesso ou não em um projeto. Embora realize bem um trabalho específico, um profissional pode estar mais interessado em tentar alguma experiência nova do que realizar tarefas que já fez muitas vezes. Assim, a novidade no trabalho pode ser um fator que gera interesse. Por outro lado, existem pessoas que sempre preferem trabalhar com o que já têm experiência do que se arriscar com tipos de tarefas que nunca realizaram. Qualquer que seja a pessoa escolhida para realizar uma tarefa, o importante é que ela esteja bastante motivada para realizá-la, independentemente da razão.

Considerando capacidade e interesses iguais, duas pessoas podem se diferenciar quanto à experiência e ao treinamento que têm com o domínio da aplicação. Com isso, a seleção do pessoal do projeto não envolve somente a capacidade e as habilidades individuais, mas também a experiência e o treinamento (PFLEEGER, 2004).

O processo de software, junto com todo o projeto, é constituído por participantes que podem ser classificados da seguinte forma (PRESSMAN, 2002):

- **Gerentes Seniores:** têm como responsabilidade definir os aspectos de negócio que freqüentemente têm uma influência significativa no projeto.
- **Gerentes de Projeto (técnicos):** têm o papel de controlar, planejar, organizar e controlar os profissionais que atuam no desenvolvimento do projeto de software.
- **Desenvolvedores:** possuem aptidões técnicas necessárias para a construção de um produto ou aplicação.
- **Clientes:** definem os requisitos para o software submetido à engenharia. Também podem ser pessoas interessadas superficialmente no resultado.
- **Usuários finais:** interagem com o sistema depois que o mesmo é liberado para ser utilizado.

2.3.2 Tarefas de um Projeto

Em um projeto de software centenas de pequenas tarefas devem ser executadas para atingir uma meta maior. Algumas dessas tarefas não se situam no fluxo principal do projeto e

podem ser executadas sem preocupação com o impacto na data de finalização do projeto. Por outro lado, existem tarefas que estão no caminho crítico do projeto, ou seja, se sofrerem atrasos, a data de finalização do projeto pode ser ameaçada.

O gerente de projeto deve definir as tarefas do projeto, construir uma rede de dependências entre elas, identificar quais são as críticas e, por fim, acompanhar a execução das mesmas. Assim, é necessário que o gerente tenha um cronograma para monitorar o progresso e controlar o projeto (PRESSMAN, 2002).

A elaboração do cronograma do projeto é uma atividade que distribui o esforço estimado pela duração total planejada do projeto, dividindo esse esforço por tarefas específicas de Engenharia de Software. É válido notar que o cronograma evolui com o tempo. Logo no início do planejamento do projeto, um cronograma macroscópico é desenvolvido, no qual são identificadas as principais atividades de Engenharia de Software e as funções do produto a que se aplicam. À medida que o projeto deslancha, cada entrada no cronograma macroscópico é refinada em um cronograma mais detalhado. Nele, tarefas necessárias para executar uma atividade são identificadas e programadas (PRESSMAN, 2002).

Como em outras áreas, alguns princípios básicos orientam a construção de um cronograma de um projeto de software (PRESSMAN, 2002):

- Compartilhamento: o projeto deve ser detalhado em um conjunto de tarefas e atividades gerenciáveis. Para tanto, o produto e o processo são decompostos.
- Interdependência: a interdependência de cada atividade ou tarefa deve ser determinada, identificando quais tarefas podem ser executadas seqüencialmente e quais podem ser executadas paralelamente. Algumas tarefas não podem iniciar até que o trabalho produzido por outra esteja disponível. Outras atividades podem ser executadas independentemente.
- Atribuição de tempo: para cada tarefa do cronograma deve ser atribuído certo número de unidades de trabalho (p. ex.: pessoas-dias de esforço). Além disso, datas de início e término devem ser definidas para as tarefas, de acordo com suas interdependências e com o fato do trabalho poder ser conduzido em tempo integral ou parcial.
- Validação do esforço: todo projeto tem certa quantidade de membros na equipe. À medida que a atribuição do tempo ocorre, o gerente deve garantir que em nenhuma

fase seja programado mais trabalho do que a quantidade de recursos humanos disponível.

- Responsabilidades definidas: cada tarefa definida no cronograma deve ser atribuída a uma pessoa da equipe.
- Resultados definidos: cada tarefa definida no cronograma deve ter um resultado que, normalmente, é um produto de trabalho ou parte dele.
- Marcos de referência definidos: cada tarefa ou grupo de tarefas deve ser associado a um marco de referência no projeto. Um marco é alcançado quando um ou mais produtos resultante do trabalho tiverem sido revisados quanto à qualidade e tiverem sido aprovados.

2.4 Ambientes de Desenvolvimento de Software

Nos primórdios da Engenharia de Software, todas as informações relacionadas ao andamento de um projeto eram guardadas em documentos escritos em papel, como planilhas e relatórios, o que aumentava a probabilidade de ocorrência de erros e dificultava o acesso aos documentos.

À medida que os projetos ganharam tamanho e se tornaram mais complexos, tornou-se inviável essa forma não automatizada de controlar documentos escritos em papéis que, de certa forma, passou a consumir um tempo considerável da jornada de trabalho do desenvolvedor.

Com base nesses problemas, surgiram ferramentas com intuito de fornecer ao engenheiro de software apoio automatizado às atividades manuais e aperfeiçoar o conhecimento de Engenharia de Software. Essas ferramentas são conhecidas como ferramentas *CASE* (*Computer Aided-Software Engineering*). Além disso, elas podem melhorar o apoio à automatização das atividades do processo de um projeto, uma vez que, com auxílio dessas ferramentas, o esforço de obtenção, manutenção e melhoria da qualidade dos processos é bastante reduzido (PRESSMAN, 2002).

Ferramentas *CASE*, geralmente, são construídas com o objetivo de apoiar o engenheiro de software em uma atividade específica do processo de software. Com a grande variedade de ferramentas *CASE*, é interessante que haja uma integração entre as mesmas com o propósito de

reduzir mais o trabalho e o tempo gasto pelo desenvolvedor, pois, apesar de serem ferramentas específicas para cada atividade, dados de saída de uma podem ser dados de entrada para outra. Com isso, é de grande importância que se tenha um sistema que forneça uma estrutura para a integração das ferramentas, de modo garantir a transferência de dados entre elas, proporcionando maior agilidade e qualidade a todo o processo de desenvolvimento (PRESSMAN, 2002).

Com isso, surgiram os Ambientes de Desenvolvimento de Software (ADSs) com o objetivo de fornecer essa base de integração, permitindo (PRESSMAN, 2002):

- Transferência de dados de uma ferramenta para a outra e de um passo de Engenharia de Software para o seguinte;
- Redução no trabalho necessário para realizar atividades de apoio, como gerência de configuração, garantia de qualidade e produção de documentos;
- Maior controle sobre o projeto devido a um melhor planejamento, monitoração e comunicação; e
- Maior gerenciamento sobre as pessoas da equipe que estão trabalhando em um grande projeto.

ADSs também impõem desafios significativos. A integração exige representações consistentes de informação de Engenharia de Software, interfaces com um formato padronizado entre as ferramentas, comunicação homogênea entre o engenheiro de software e cada ferramenta e, finalmente, é importante permitir se mover entre várias plataformas de hardware e sistemas operacionais (PRESSMAN, 2002).

Para definir integração, dentro do contexto de Engenharia de Software, é necessário estabelecer um conjunto de requisitos, a saber (PRESSMAN, 2002):

- Possibilitar o compartilhamento de informação de Engenharia de Software entre todas as ferramentas do ambiente;
- Possuir controle de versão e gestão global de configuração para cada informação de Engenharia de Software;
- Ter acesso direto, não seqüencial a todas as ferramentas do ambiente;
- Permitir automatização para o modelo de processo escolhido, integrando ferramentas CASE e itens de configuração de software;
- Fornecer uma interface consistente aos usuários de cada ferramenta;
- Facilitar a comunicação entre os engenheiros de software;

- Coletar métricas de gestão e de técnicas que podem ser utilizadas para a melhoria do processo e do produto.

2.5 O Ambiente ODE

ODE (*Ontology-based software Development Environment*) (FALBO et al., 2003) é um ambiente de desenvolvimento de software centrado em processo, baseado em um conjunto de ontologias relacionadas do domínio de Engenharia de Software. Esse ambiente vem sendo desenvolvido no Laboratório de Engenharia de Software (LabES) do Departamento de Informática (DI) da Universidade Federal do Espírito Santo (UFES), utilizando tecnologias livres como a linguagem de programação Java, o sistema de gerenciamento de banco de dados PostgreSQL, o *framework* de persistência de dados *Hibernate* e o sistema operacional *Linux*.

A idéia principal do projeto ODE é a seguinte: se as ferramentas de um ADS são construídas baseadas em ontologias, a integração das mesmas é facilitada, pois os conceitos envolvidos estão formalmente definidos nas ontologias. Essa é uma das principais características que diferencia ODE de outros ADSs: sua base ontológica. As ontologias reduzem confusões terminológicas e conceituais, principalmente em ADSs, facilitando a comunicação e o entendimento global entre pessoas com diferentes necessidades e pontos de vista. Além disso, a padronização de conceitos devido à utilização de ontologias facilita a comunicação entre as ferramentas que compõem o ambiente (FALBO et al., 2005).

Atualmente, o ambiente ODE possui uma infra-estrutura que permite controlar projetos de software e definir seus respectivos processos. Integradas a ele, existem várias ferramentas de apoio à construção, gerência e avaliação de qualidade. Dentre as atividades apoiadas, podem-se citar as seguintes: estimativas, análise de riscos, modelagem de análise e projeto, documentação, definição e acompanhamento de processo, gerenciamento de recursos, gerenciamento de requisitos, planejamento e acompanhamento de projeto e controle da qualidade.

2.6 Apoio Automatizado à Gerência de Projetos em ODE

Para que as tarefas do planejamento de um projeto sejam realizadas de forma eficaz, é importante que se tenha gerentes experientes que saibam lidar com a execução das mesmas,

dentro de um ambiente com diversas variáveis que um projeto pode envolver, tais como o prazo e o custo. Além disso, o gerente de projetos deve se preocupar com o acompanhamento do projeto, avaliando e atualizando diversos artefatos gerados e dados relativos à fase de planejamento.

Apesar de permitir uma melhor administração dos projetos, nem sempre é possível contar com profissionais de gerência experientes, pois profissionais com esse perfil podem ser poucos no mercado e, geralmente, há um custo muito alto para contratá-los.

Com isso, a automatização da gerência de projetos é importante para auxiliar os gerentes de projetos, com experiência ou não, na realização de suas tarefas, proporcionando uma menor carga de trabalho e agilizando um processo contínuo de gerência.

Atualmente, o ambiente ODE conta com diversas ferramentas para automatizar o processo de gerência de projetos:

- Ferramenta de Apoio à Definição de Processos de Software (BERTOLLO *et al*., 2006): permite a definição de processos padrão e, a partir desses, instanciar um processo específico para cada projeto;
- EstimaODE: permite a realização de estimativas, contando com ferramentas de apoio à Análise de Pontos de Função e à Análise de Pontos de Caso de Uso (ARANTES, 2006);
- GeRIS: ferramenta de apoio ao gerenciamento de riscos, na qual os riscos de um projeto podem ser identificados, avaliados, planejados e monitorados (FALBO *et al.*, 2004);
- ControlPRO: essa ferramenta permite o acompanhamento de projetos de forma ampla. Nela é possível visualizar o andamento das atividades de um processo (DAL MORO *et al*., 2005);
- ReqODE → ferramenta de apoio à gerência de requisitos (MARTINS *et al*., 2006);
- Ferramenta de Apoio à Alocação de Recursos (SCHWAMBACH, 2002): permite a definição de equipe e alocações de recursos humanos e de ferramentas de software em atividades de um projeto.
- Ferramenta Agenda (PEZZIN, 2002): auxilia a orientar cada desenvolvedor a respeito de suas alocações a atividades.

Apesar de ODE ser um sistema com muitas ferramentas gerenciais integradas, é importante que esteja em constante evolução, proporcionando melhorias nas mesmas. Neste trabalho, foram avaliadas duas ferramentas do ambiente que poderiam ser evoluídas: a ferramenta de apoio à alocação de recursos e a ferramenta de agenda.

Essas duas ferramentas foram umas das primeiras a serem construídas para o ambiente. A ferramenta de apoio à alocação de recursos permitia ao gerente de projetos definir uma equipe para cada projeto, bem como alocar recursos humanos e de software para as atividades de um processo. Já a ferramenta de agenda permitia que o desenvolvedor visualizasse as atividades às quais estava alocado. É fácil notar que a ferramenta de agenda está fortemente ligada à ferramenta de alocação de recursos.

Ao avaliar a ferramenta de apoio à alocação de recursos, vários pontos foram identificados para evolução, a saber:

- Ao se definir uma equipe para um projeto, é necessário que essa seja criada de acordo com as especialidades necessárias para a realização das atividades do processo, ou seja, a equipe deve ser composta por pessoas que tenham as especialidades requeridas pelas atividades e que estejam ativas no ambiente ODE.
- Para prover um maior controle sobre as alocações de recursos humanos, é necessário que se acompanhem os seus estados, para que o gerente de projeto possa ter uma visualização mais completa do andamento das alocações de recursos humanos e, por conseguinte, da realização das atividades, permitindo uma avaliação das mesmas.
- Além de alocar recursos humanos e de software, é também necessário alocar recursos de hardware em atividades de um projeto.
- Para proporcionar uma maior usabilidade e produtividade, as alocações de recursos devem ser parcialmente automatizadas, levando-se em conta os papéis dos recursos requeridos, estabelecidos durante a definição do processo de software do projeto, e os recursos alocados ao projeto.

Já na avaliação da ferramenta de agenda, foi possível encontrar também vários pontos considerados como potenciais melhorias para a evolução da mesma, sendo eles:

- Orientar melhor o desenvolvedor dentro do ambiente ODE, mostrando as suas alocações nos projetos, além de apresentar o andamento das mesmas, tais como o estado e o percentual de conclusão de cada alocação.
- Selecionar, a partir da ferramenta agenda, ferramentas de software disponíveis para realização das atividades às quais está alocado o desenvolvedor.
- Adicionar novas funcionalidades de uma agenda eletrônica básica, tal como cadastrar compromissos, anotações e contatos.

O ambiente ODE é um sistema de várias ferramentas, geralmente com diversas interfaces gráficas para cada uma e, em alguns casos, não intuitivas o bastante. Com isso, usuários iniciantes do ambiente normalmente têm dificuldade de utilizar algumas das ferramentas.

Baseado nesse aspecto, uma grande melhoria notada, tanto na ferramenta de alocação de recursos quanto na de agenda, é o desenvolvimento de uma interface mais amigável, para prover uma melhor usabilidade e interatividade das mesmas.

Capítulo 3

O Processo de Software Adotado

Para se construir um produto ou um software, independente de qual natureza for, é necessário seguir uma série de etapas, ou seja, um tipo de guia que ajude a chegar a um resultado de qualidade, dentro de um tempo previsto. Quando se trata de desenvolvimento de software, esse "guia" é denominado processo de software (FALBO, 2005).

Este capítulo apresenta o processo de software adotado para o desenvolvimento deste trabalho. Para a realização do mesmo, foi utilizado o paradigma orientado a objetos.

3.1 Processo de Software

Um processo de software pode ser definido como um conjunto de atividades, métodos, práticas e transformações que guiam pessoas no desenvolvimento de um software (FUGGETTA, 2000).

Os processos de software, por muitas vezes, são decompostos em subprocessos, como, por exemplo, processo de desenvolvimento, processo de garantia de qualidade, processo de gerência de projetos etc. Esses processos são compostos por atividades que também podem ser decompostas em sub-atividades. Para cada atividade é importante saber quais são suas pré-atividades (atividades que devem precedê-las), os artefatos de entrada (insumos) e saída (produtos), os recursos necessários (humanos, software, hardware, entre outros) e os procedimentos (métodos, modelos de documento, técnicas etc) a serem utilizados para a sua realização (FALBO, 2005). A Figura 3.1 esquematiza os elementos que compõem um processo de software.

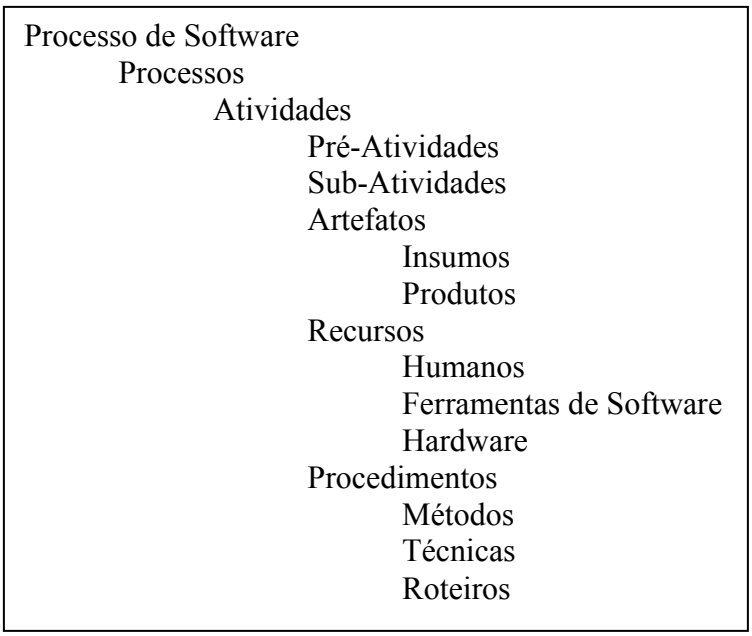

Figura 3.1 - Elementos que compõem um Processo de Software (FALBO, 2005)

Para a definição de um processo de software, vários fatores devem ser considerados. No centro da arquitetura de um processo estão as atividades-chave, a saber: especificação de requisitos, análise, projeto, implementação e testes, sendo essas a base para a construção de um processo de software. Entretanto, a definição de um processo envolve a escolha de um modelo de ciclo de vida, o detalhamento de suas macro-atividades, os métodos, as técnicas e roteiros para sua realização, bem com os recursos e insumos necessários e produtos a serem produzidos.

A próxima seção discute a escolha do modelo de ciclo de vida adotado para a realização deste trabalho.

3.2 O Modelo de Ciclo de Vida Adotado

Um modelo de ciclo de vida, conhecido também como modelo de processo, pode ser visto como uma representação abstrata de um esqueleto de um processo de software, incluindo atividades-chave, a ordem de precedência entre ela e, opcionalmente, artefatos necessários e produzidos.

Mesmo que processos de software devam ser definidos individualmente, em geral, um ciclo de vida de um software envolve, no mínimo, as seguintes atividades (FALBO, 2005):

- Planejamento: tem como objetivo fornecer uma estrutura que possibilite estimar custos, recursos e prazos de um projeto. Tendo o escopo de software estabelecido com seus respectivos requisitos esboçados, um plano de projeto é elaborado, configurando o processo a ser usado no desenvolvimento do sistema. É importante que, ao final de cada uma das fases do desenvolvimento (especificação de requisitos, análise, projeto, implementação e testes), o planejamento seja revisto e detalhado;
- Especificação de Requisitos e Análise: é a fase na qual os requisitos do sistema são levantados e modelados. O escopo deve ser refinado e os requisitos mais bem definidos. Após ter os requisitos especificados, o engenheiro de software tem de entender o domínio do problema, além da funcionalidade e do comportamento esperado. Com isso, os requisitos levantados pelo sistema devem ser modelados, avaliados e documentados, de forma que fique claro o que o sistema deve fazer (e não como fazê-lo);
- Projeto: nesta fase é necessário incorporar os requisitos tecnológicos aos requisitos essenciais do sistema, modelados na fase anterior, de forma que a plataforma de implementação também seja levada em consideração. Essa fase envolve basicamente duas etapas: projeto arquitetural e projeto detalhado. O projeto arquitetural define a arquitetura geral do software com base no modelo construído na fase de análise. Além de descrever a estrutura de nível mais alto da aplicação, essa arquitetura deve identificar seus principais componentes. Já o projeto detalhado tem como objetivo detalhar o projeto do software para cada componente identificado na etapa anterior. Assim, os componentes de software devem ser sucessivamente refinados em níveis maiores detalhamento, até que possam ser codificados e testados;
- Implementação: é a fase na qual o projeto é traduzido para uma forma que possa ser executado por uma máquina. Cada unidade do projeto detalhado deve ser implementada;
- Testes: nessa fase, diversos níveis de testes podem ser incluídos (teste de unidade, teste de integração e teste de sistema). Primeiramente, cada unidade de software

implementada é testada. Em seguida, as unidades vão sendo integradas sucessivamente até se obter o sistema. Finalmente, todo o sistema é testado;

- Entrega e Implantação: depois do software ser testado, este deve ser colocado em produção. Assim, é também necessário treinar os usuários, configurar o ambiente de produção e, até mesmo, converter base de dados. Nesta fase é feito o teste de aceitação. Se o sistema atender às capacidades requeridas, ele pode ser aceito e a operação iniciada;
- Operação: é a fase na qual os usuários utilizam o software no ambiente de produção;
- Manutenção: é a fase na qual o software sofrerá mudanças após ter sido entregue aos usuários, seja porque erros foram encontrados, seja porque o software precisa ser adaptado a mudanças externas ou internas, seja porque o cliente necessita de novas funcionalidades no sistema. Muitas vezes, essa fase requer uma nova definição de um processo, em que cada uma das fases precedentes é re-aplicada no contexto de um software existente ao invés de um novo.

Neste trabalho as seguintes atividades etapas foram consideradas: planejamento, especificação de requisitos, análise, projeto, implementação e testes.

Na atividade de planejamento, foi definido um cronograma com as fases de desenvolvimento do software, divididas em tarefas, bem como a dependência entre as mesmas e o ciclo de vida utilizado. Para a escolha do modelo de ciclo de vida deste projeto, alguns fatores foram considerados:

- As ferramentas desenvolvidas para o ambiente ODE estão em constante evolução;
- Esse trabalho é a continuação de trabalhos realizados anteriormente, ou seja, é a reestruturação de ferramentas já existentes;

Com isso, o modelo de ciclo de vida adotado nesse projeto foi o modelo em espiral, ilustrado na Figura 3.2, que utiliza a idéia de gerar versões prototípicas, até que se tenha uma versão final madura o bastante para ser implantada e utilizada.

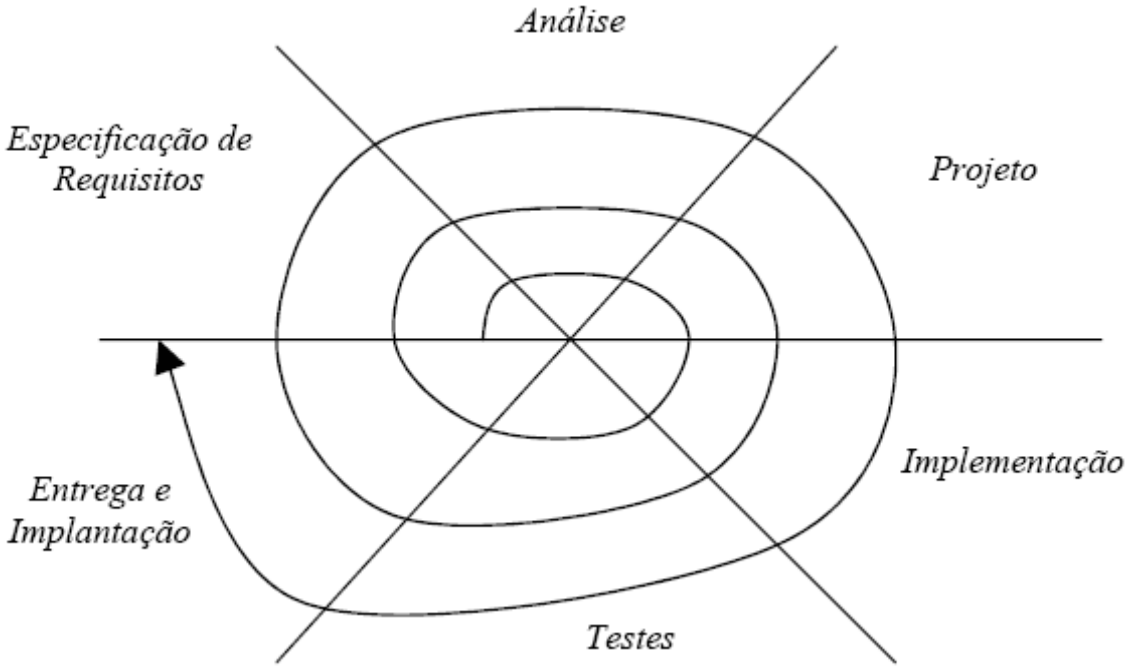

Figura 3.2 - Ciclo de vida espiral (FALBO, 2005)

3.3 As Principais Atividades do Processo

Na fase de planejamento, foi definido o escopo do software deste trabalho, bem como um cronograma para o desenvolvimento do mesmo.

Em seguida, na fase de especificação e análise de requisitos, foi levantado um entendimento geral do problema. Com isso, foram identificados os atores e os casos de uso do sistema com suas respectivas descrições, sendo que, para cada caso de uso foi apresentado um conjunto de cursos normais e cursos alternativos (quando necessários). Com base na modelagem e descrição dos casos de uso, foram elaborados diagramas de classes, bem como diagramas de estados para as classes com comportamento mais complexo, que permitem vários estados. Os resultados dessa fase são apresentados parcialmente no Capítulo 4, estando integralmente disponíveis nos anexos.

Na fase de projeto, discutida no Capítulo 5, o trabalho foi reorganizado em forma de componentes, de acordo com a arquitetura básica de ODE. Essa arquitetura é composta por cinco componentes, sendo eles:

- Componente do Domínio do Problema: conjunto de classes que modelam o domínio do problema, ou seja, representam informações do mundo real relevantes para o sistema;
- Componente de Gerência de Dados: conjunto de classes que se comunicam com a camada de persistência, tendo como objetivo facilitar a persistências dos objetos do domínio;
- Componente de Gerência de Tarefas: conjunto de classes que implementam os casos de uso definidos para o software;
- Componente de Interação Humana: conjunto de classes responsáveis pela interação do homem com a máquina, ou seja, pelas classes de interface com o usuário; e
- Componente de Controle de Interação: conjunto de classes responsáveis por fazer a comunicação entre as camadas de Gerência de Tarefas e de Interação Humana, permitindo um maior isolamento e reaproveitamento de código.

No projeto detalhado, para cada componente, foi elaborado um diagrama de classes, apresentando as classes correspondentes.

Com o projeto definido e todos os aspectos tecnológicos considerados, a implementação se torna muito mais fácil. Assim, a linguagem de programação Java, o *framework* de persistência *Hibernate* (BAUER *et al*., 2005) e o banco de dados PostgreSQL foram utilizados para a implementação do sistema.

Na fase de testes foram feitos vários testes com base nos casos de uso definidos para o sistema, principalmente com o intuito de eliminar erros de concorrência e persistência. Contudo, mesmo tendo realizado vários testes, é impossível afirmar que o sistema não possui falhas.

É válido considerar que, para a construção dos modelos gerados neste trabalho nas fases de análise e especificação de requisitos e projeto foi utilizada a Linguagem de Modelagem Unificada (*Unified Modeling Language* – UML) (BOOCH *et al.,* 2005).

Capítulo 4

Especificação e Análise de Requisitos

Um entendimento completo dos requisitos necessários para a construção do software é essencial para o sucesso de um esforço de desenvolvimento de software. A atividade de análise e especificação de requisitos é um processo de descoberta, refinamento, modelagem e especificação (FALBO, 2002).

O levantamento de requisitos tem como objetivo capturar os requisitos dos usuários para a construção do sistema, mostrando uma solução em alto nível que possa atender apenas às necessidades requeridas por eles. Para descrever as funcionalidades identificadas no sistema, foi utilizada a modelagem de casos de uso.

A análise dos requisitos é feita com base na especificação de requisitos gerada. Nessa fase, são modeladas as estruturas internas do sistema capazes de satisfazer os requisitos identificados. Quando se trata de Análise Orientada a Objetos, um modelo é criado representando o domínio do sistema, no qual os objetos são identificados com seus respectivos atributos, relacionamentos e comportamento.

Este capítulo apresenta parcialmente a especificação de requisitos funcionais e os modelos de análise elaborados para as ferramentas de Alocação de Recursos e de Agenda do ambiente ODE. Documentos completos são encontrados nos Anexos A e B, respectivamente.

A Seção 4.1 apresenta as especificações de requisitos funcionais e de análise da ferramenta de Alocação de Recursos. A Seção 4.2 apresenta as especificações de requisitos e de análise da ferramenta de Agenda. Finalmente, a Seção 4.3 apresenta os requisitos não-funcionais considerados neste trabalho.

4.1 A Ferramenta de Alocação de Recursos

Dentro de uma organização, pessoas são responsáveis por realizar atividades, geralmente com o apoio de algum outro tipo de recurso, tal como uma ferramenta de software ou um equipamento de hardware, que auxilie na realização das mesmas.

No ambiente ODE, para cada atividade, é necessário alocar pessoas para trabalhar na mesma, bem como ferramentas de software que estarão disponíveis para a realização do trabalho, além de outros recursos como hardware e sistemas de apoio.

Para ter um maior controle sobre as alocações de recursos humanos a atividades dentro do ambiente, é necessário saber o papel desempenhado pelo recurso, a dedicação em termos de tempo diário do recurso humano alocado à realização da atividade, o esforço total previsto para a alocação, as datas de início e fim previstas e efetivas da alocação, além do estado da mesma.

Os elementos que definem a ferramenta de Alocação de Recursos são tratados no pacote *AlocacaoRecurso*, cujo diagrama de casos de uso é apresentado na Figura 4.1. Os seguintes casos de uso foram considerados:

- **Definir Equipe do Projeto**: Este caso de uso permite que o gerente de projeto defina quais recursos humanos participarão da equipe de um projeto.
- **Definir Alocações de Recursos a Atividades**: Este caso de uso permite que o gerente de projeto faça alocações de recursos às atividades de um projeto.
- **Editar Alocação de Recurso Humano**: Este caso de uso permite que o gerente de projeto planeje detalhadamente as alocações de recursos humanos às atividades de um projeto.
- **Controlar Andamento de Alocação de Recurso Humano a Atividade**: Este caso de uso permite que o gerente de projeto cancele uma alocação de recurso humano a uma atividade, anule um cancelamento ou encerre a participação de um recurso humano na atividade. Por meio deste caso de uso, também o desenvolvedor pode solicitar o encerramento de sua participação na atividade.
- **Registrar Esforço Despendido:** Este caso de uso permite ao desenvolvedor registrar, alterar, consultar ou cancelar um esforço despendido em uma atividade

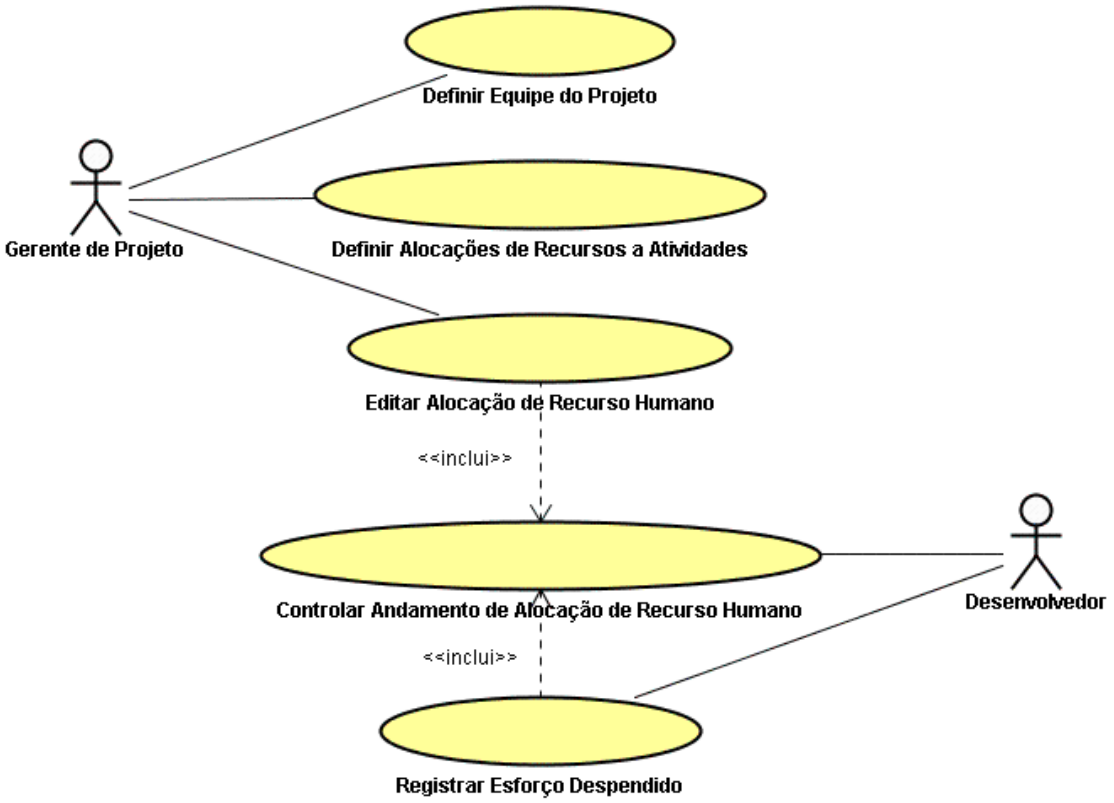

Figura 4.1 - Diagrama de Casos de Uso do pacote *AlocacaoRecurso*

Conforme apresentado na Figura 4.1, os atores dos casos de uso são usuários do ambiente ODE. O ator Desenvolvedor representa qualquer tipo de usuário. Já o ator Gerente de Projeto representa os profissionais que desempenham esse papel no ambiente.

A Figura 4.2 apresenta o diagrama de pacotes referente à parte do sistema que trata da ferramenta de Alocação de Recursos do ambiente ODE.

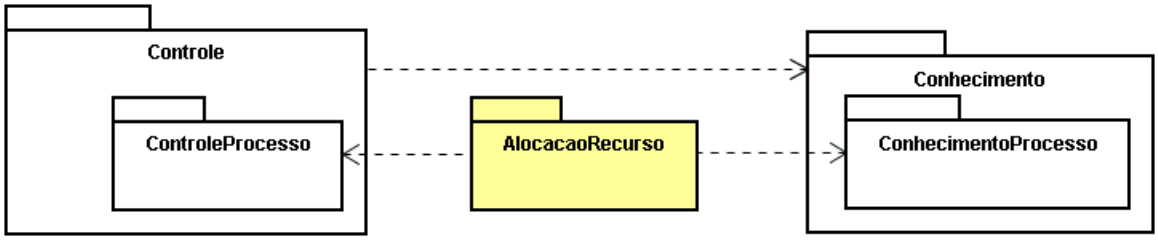

Figura 4.2 – Diagrama de Pacotes da ferramenta de Alocação de Recursos.

O pacote *Controle* é responsável pelo núcleo do ambiente ODE. É de sua responsabilidade as principais funcionalidades referentes ao gerenciamento do sistema, sendo essas o controle de projetos, a definição de processos e o controle de acesso às ferramentas do ambiente. Dentro desse pacote, existe o subpacote *ControleProcesso*, que contém as classes que controlam os processos de software no ambiente. De maneira geral, para cada projeto é definido um processo de projeto, que é decomposto em subprocessos, atividades e sub-atividades. Para cada atividade é possível definir, ainda, pré-atividades, recursos necessários, artefatos (que podem ser insumos ou produtos) e procedimentos a serem adotados.

O pacote *Conhecimento* é responsável pelo conhecimento contido no ambiente ODE, no qual são definidas classes que representam conceitos das ontologias utilizadas pelo sistema e que são usadas para falar sobre o universo de discurso do pacote *Controle* correspondente. Assim, o pacote *ConhecimentoProcesso*, que é um subpacote do pacote *Conhecimento*, trata do conhecimento do ambiente ODE no que se refere a processos de software e é utilizado pelo pacote *ControleProcesso*.

Na ferramenta de Definição de Processo do ambiente ODE, é possível definir os tipos de recursos humanos, de software e de hardware, necessários para a realização de cada atividade (representados como objetos da classe `KRecurso` do pacote *Conhecimento*). Depois que os tipos de recursos são definidos, pode-se, por meio da ferramenta de Alocação de Recursos, alocar recursos específicos (pessoas, ferramentas de software e equipamentos de hardware, respectivamente) para essas atividades.

O pacote *AlocacaoRecurso* é responsável pelo controle das alocações de recursos para cada atividade de um projeto. Após um projeto ter seu processo de software definido, é possível definir a equipe do projeto e alocar recursos para suas atividades. No que se refere à alocação de recursos humanos, é possível definir o papel que a pessoa (recurso humano) desempenhará na atividade, de acordo com os seus papéis desempenhados na equipe. Além disso, é possível alocar ferramentas de software e equipamentos de hardware para uma determinada atividade, de acordo com os tipos de recursos de software e de hardware requeridos pela mesma.

Para ter um maior controle sobre as alocações de recursos humanos a atividades, o gerente de projeto pode, ainda, controlar o andamento de uma alocação de recurso, cancelando-a, encerrando-a ou a retornando para andamento para que ajustes sejam feitos, indicando que algo ainda precisa ser feito pelo recurso humano naquela atividade. Além disso, é possível anular um

cancelamento de uma alocação de recurso humano. Já o desenvolvedor pode controlar o andamento de uma alocação, solicitando o encerramento de sua participação em alguma atividade, ao qual está alocado.

A Figura 4.3 apresenta o diagrama de classes do pacote *AlocacaoRecurso*, incluindo as classes utilizadas dos demais pacotes citados anteriormente.

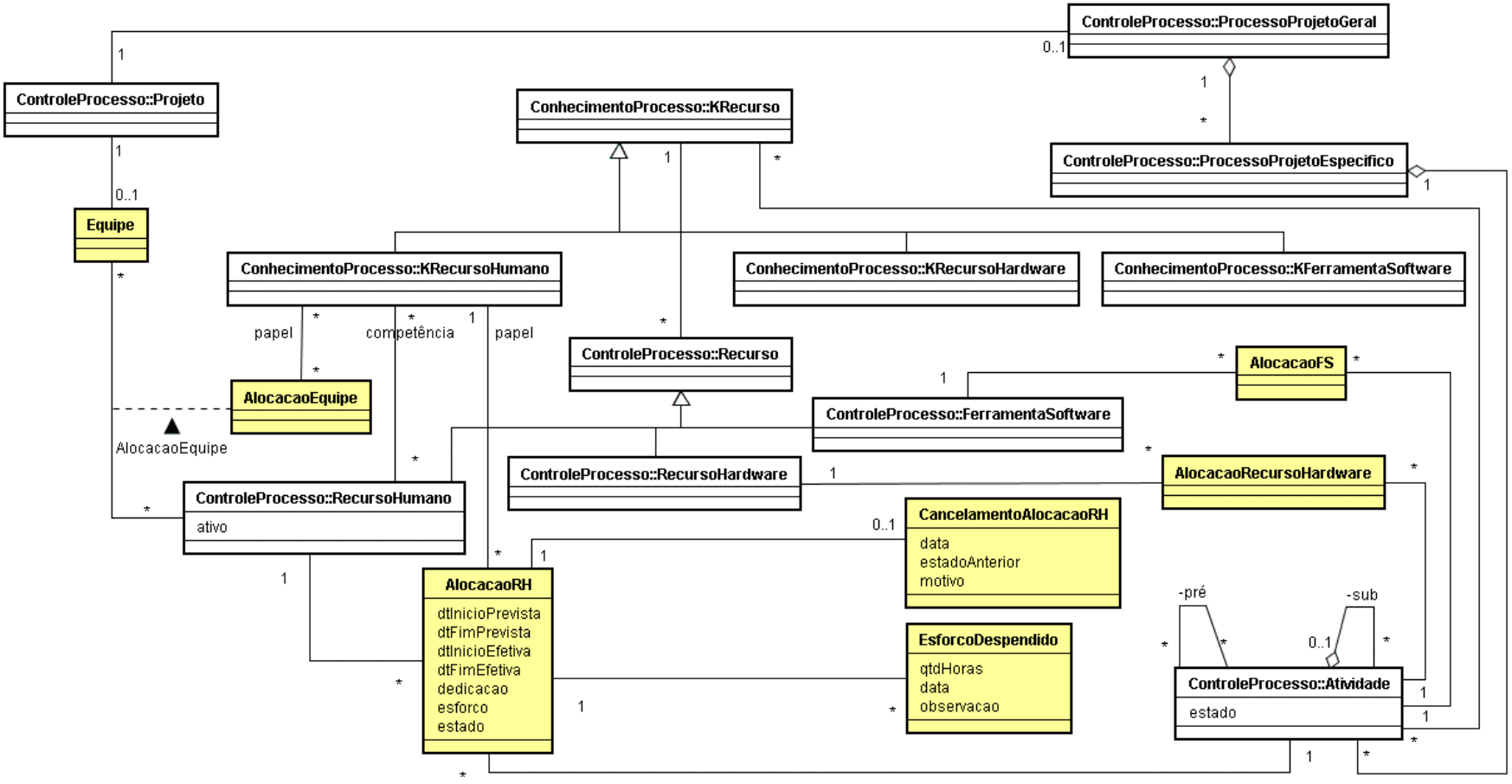

Figura 4.3 - Diagrama de Classes do pacote *AlocacaoRecurso*

A Figura 4.4 apresenta o diagrama o diagrama de estados dos objetos da classe *AlocacaoRH*. Quando uma alocação de recurso humano é definida para uma atividade ainda não iniciada, essa alocação é dita *Definida*. Após uma atividade ser iniciada, todas as alocações de recursos humanos da mesma passam a estar aptas a serem iniciadas, ou seja, o estado de cada uma delas passa a ser *Aguardando Início da Participação*. Quando o desenvolvedor registrar o primeiro esforço despendido na alocação a uma atividade, essa alocação passa a estar *Em Andamento.* Caso a alocação de recurso humano seja feita para uma atividade já em execução, a alocação inicia diretamente no estado *Aguardando Início da Participação*.

Após ter finalizado suas tarefas na alocação, o desenvolvedor pode solicitar o encerramento da sua participação na atividade. Então, o estado da alocação passa a ser *Em Avaliação para Encerramento.*

Finalmente, o gerente de projeto pode avaliar se uma alocação de recurso humano *Em Avaliação para Encerramento* pode ser efetivamente encerrada. Caso sim, a alocação é *Encerrada*. Caso contrário, ela passa para o estado *Em Andamento para Ajustes*, o que indica que a alocação deve continuar em andamento para que o recurso humano continue trabalhando na atividade para fazer alguns ajustes.

Vendo que sua alocação está com o estado *Em Andamento para Ajustes,* o desenvolvedor pode rever com o gerente de projeto o que ainda deve ser feito para encerrar sua participação na atividade. Depois de realizados os devidos ajustes, o desenvolvedor pode solicitar novamente o encerramento da sua participação na atividade.

Uma alocação de recurso também pode ser *Cancelada* se ainda não tiver sido concluída. Além disso, é possível que desse estado a alocação possa voltar ao seu estado anterior.

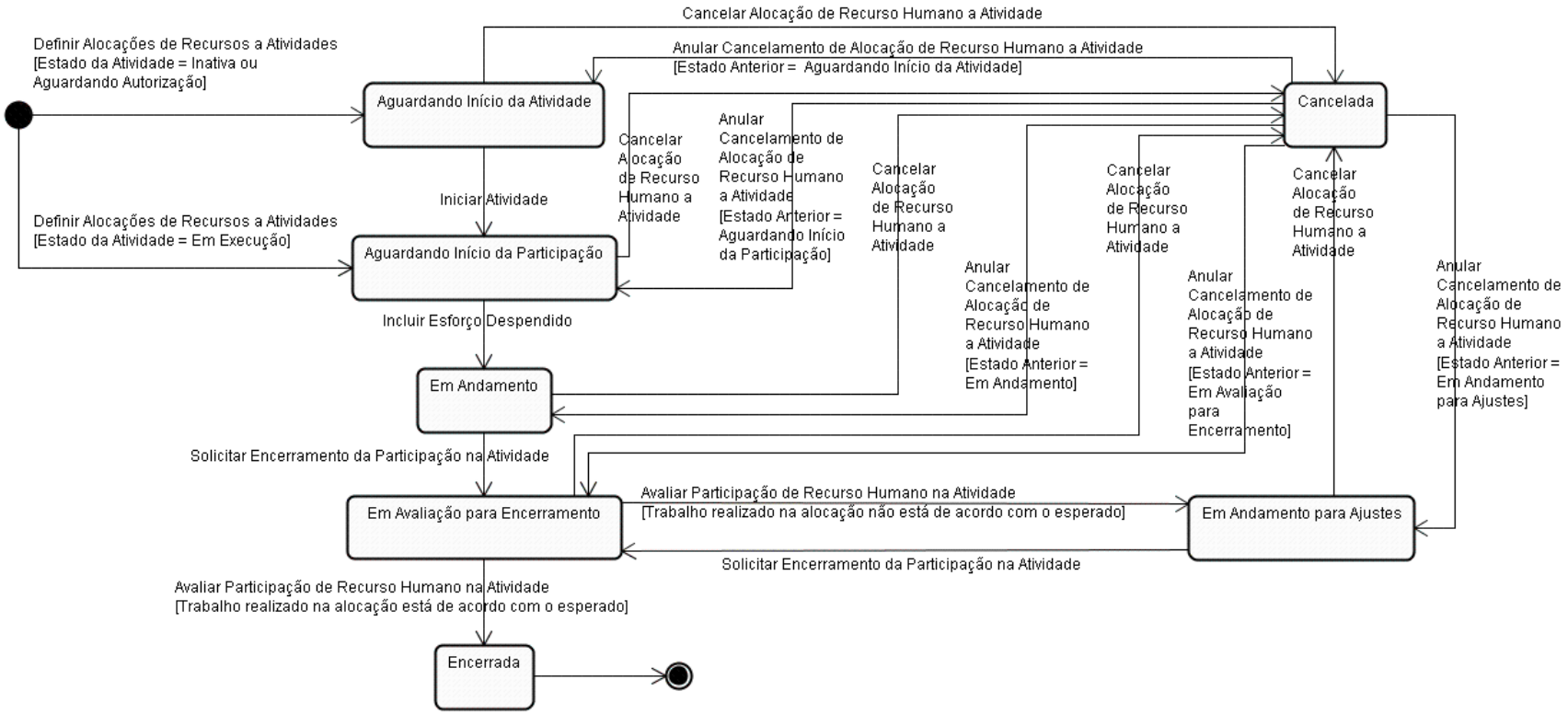

Figura 4.4 - Diagrama de Estado da classe `AlocacaoRH`

4.2 A Ferramenta de Agenda

Uma agenda deve conter informações capazes de ajudar o ser humano a lembrar compromissos, contatos e anotações. Ou seja, pode ser uma segunda forma para organizar e memorizar informações importantes para que sejam lembradas quando necessário.

No ambiente ODE é bastante interessante que cada desenvolvedor tenha uma agenda eletrônica, com o objetivo de orientá-lo dentro do sistema, informando-o a respeito das suas alocações a atividades, os compromissos que ele deve cumprir, além de outros recursos básicos que qualquer agenda deve ter.

Assim que o desenvolvedor fizer o *login* no ambiente, sua agenda será aberta e ele poderá ver as suas alocações a atividades, bem como os seus compromissos, suas anotações e seus contatos. Para cada atividade a que está alocado, é possível que ele registre esforços despendidos ou até mesmo selecione uma ferramenta de software para realizar seu trabalho.

A agenda no ambiente ODE é tratada no pacote *Agenda*, cujo diagrama de casos de uso é apresentado na Figura 4.5, contendo os seguintes casos de uso:

- **Visualizar Alocações de Recurso Humano a Atividades:** Este caso de uso permite que o desenvolvedor visualize suas alocações a atividades no ambiente ODE, de acordo com o projeto, o subprocesso e o andamento das mesmas.
- **Selecionar Ferramenta de Trabalho:** Este caso de uso permite que o desenvolvedor selecione uma ferramenta de software alocada à atividade com a qual deseja trabalhar.
- **Controlar Compromisso:** Este caso de uso permite ao desenvolvedor incluir, alterar, consultar ou excluir um compromisso na sua agenda.
- **Controlar Anotação:** Este caso de uso permite ao desenvolvedor incluir, alterar, consultar ou excluir uma anotação na sua agenda.
- **Cadastrar Contato**: Este caso de uso permite ao desenvolvedor incluir, alterar, consultar ou excluir um contato em sua agenda.

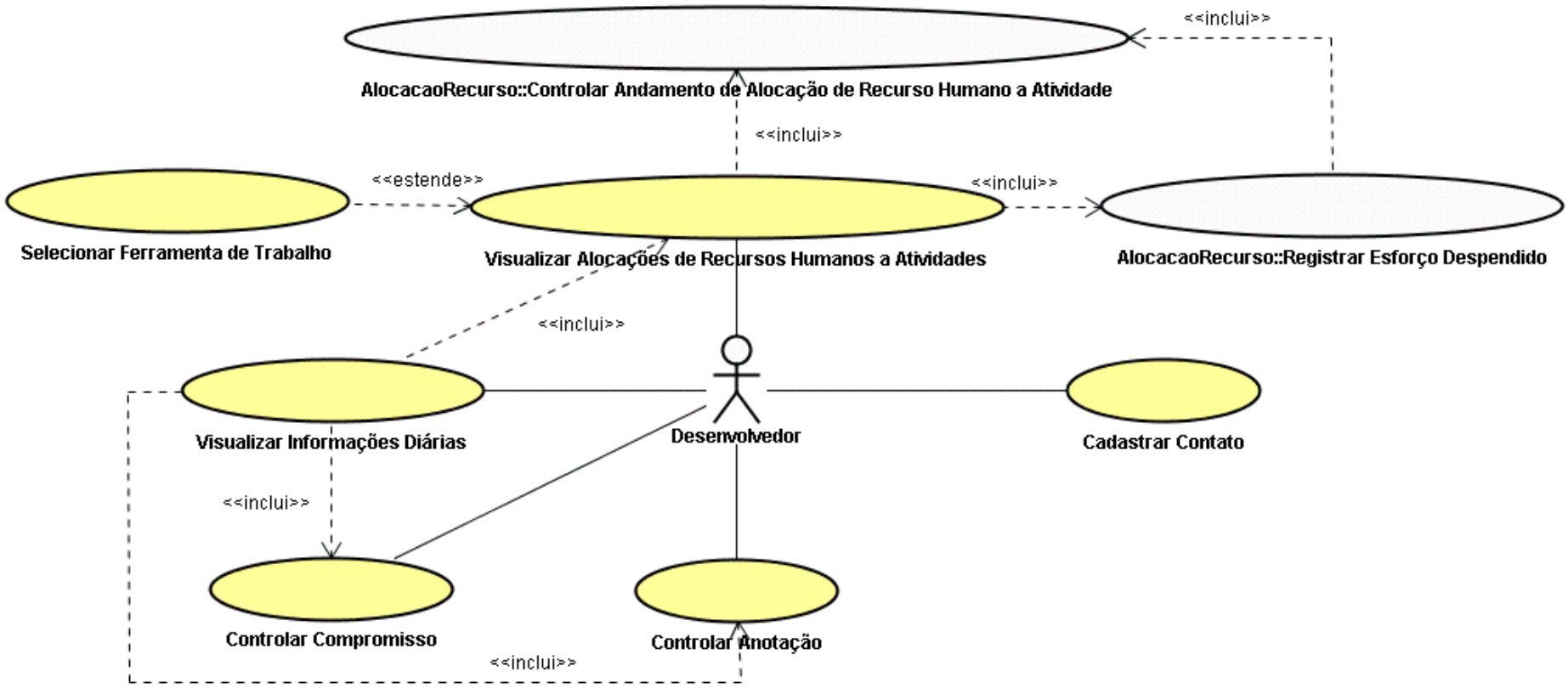

Figura 4.5 – Diagrama de Casos de Uso do Pacote *Agenda*

- **Visualizar Informações Diárias**: Este caso de uso permite que o desenvolvedor visualize as informações do dia corrente marcadas em sua agenda, sendo essas as suas alocações em andamento (normal ou para ajustes), seus compromissos no dia e suas anotações com grau de importância alto.

O diagrama de pacotes da Figura 4.6 mostra as dependências existentes entre os pacotes utilizados na modelagem de classes da ferramenta de Agenda.

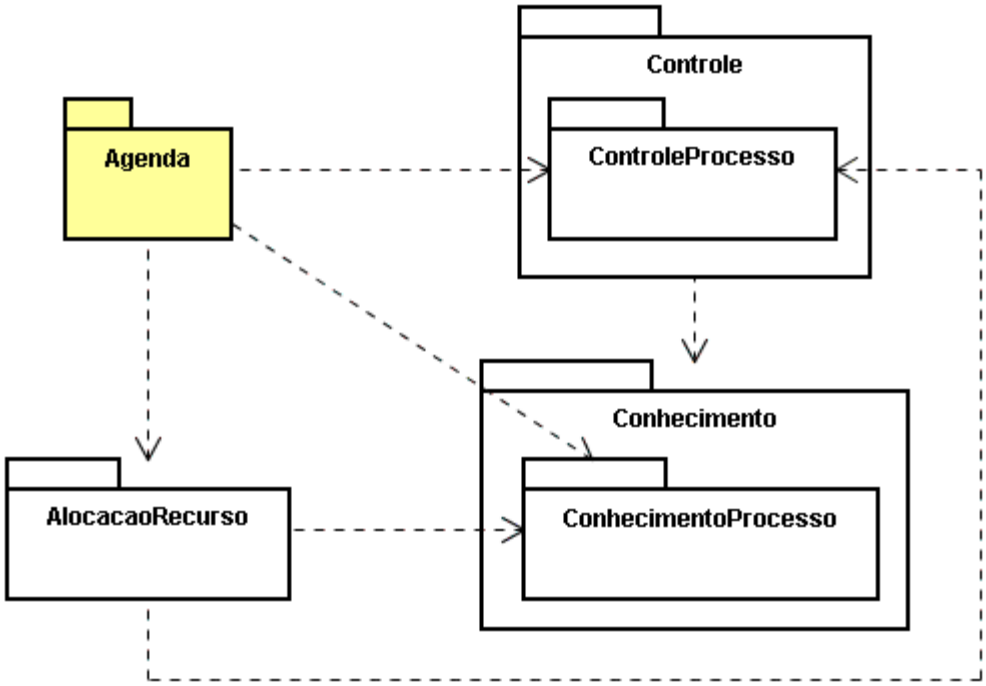

Figura 4.6 – Diagrama de Pacotes da ferramenta de Agenda.

O pacote *Agenda* é o principal pacote da ferramenta de Agenda e, tendo em vista que os demais pacotes foram apresentados na seção anterior, é o único discutido nesta seção. Esse pacote contém as classes `AgendaRH`, `Contato`, `Anotacao` e `Compromisso`, sendo que esta última é especializada para cada um dos tipos de compromisso definidos: eventual, diário, semanal ou mensal, conforme apresentado na Figura 4.7.

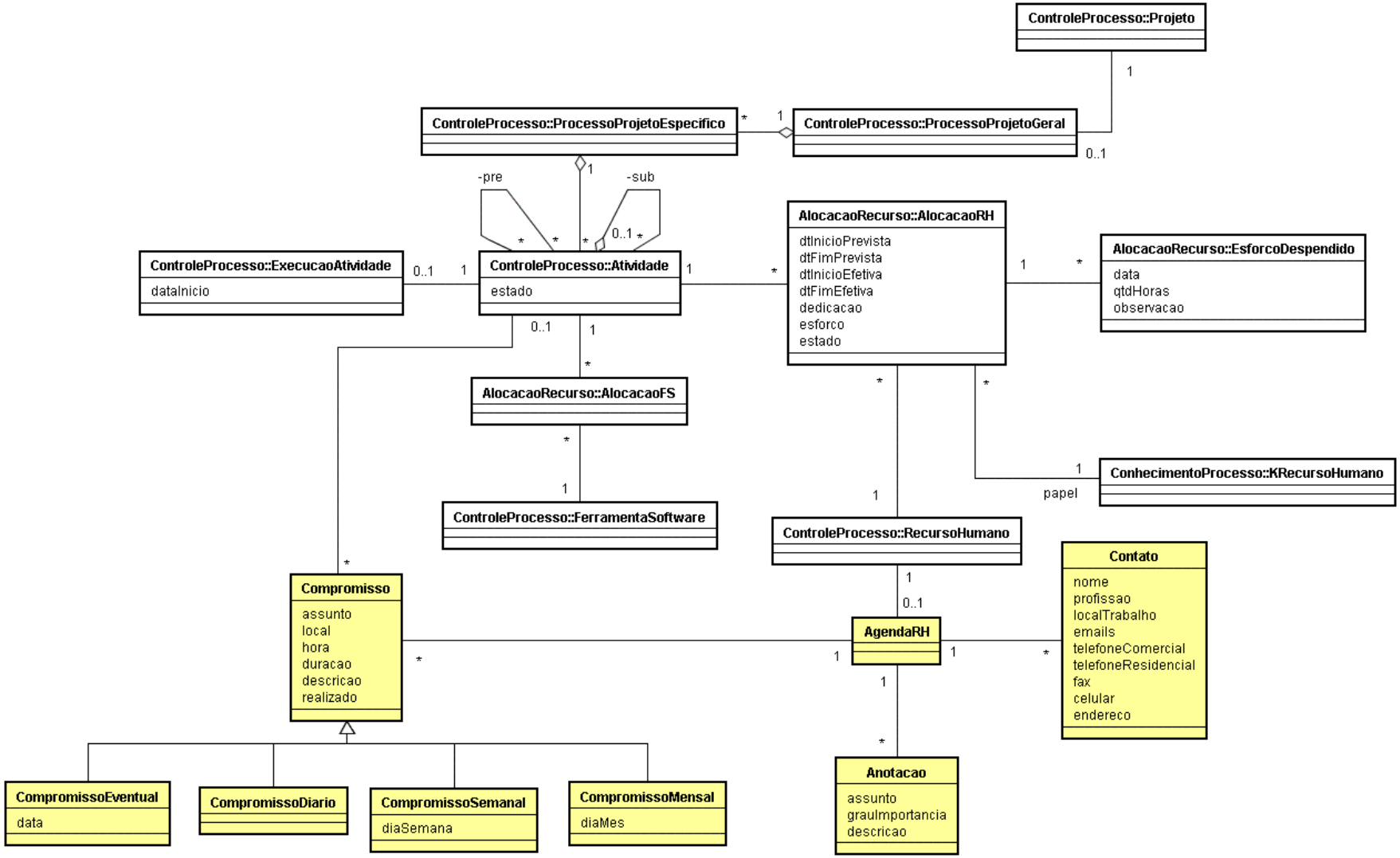

Figura 4.7 - Diagrama de Classes do pacote *Agenda*

Cada desenvolvedor pode visualizar sua agenda (classe `AgendaRH`), na qual é possível visualizar suas alocações a atividades. Essa visualização pode ser refinada pelo projeto, subprocesso e o estado da alocação. Ainda por meio da ferramenta de Agenda, é possível selecionar uma ferramenta de software alocada à atividade à qual o desenvolvedor está alocado, controlar compromissos (classe `Compromisso`), anotações (classe `Anotacao`) e contatos (classe `Contato`).

4.3 Especificação de Requisitos Não-Funcionais

Nesta seção, são apresentados os requisitos não-funcionais identificados como mais importantes para este trabalho. Os requisitos não-funcionais são tipicamente tratados na fase de projeto e devem atender alguns critérios de qualidade que um sistema precisa satisfazer, dentre eles (MEYER, 1997):

- Facilidade de manutenção (manutenibilidade).
- Uso otimizado dos recursos computacionais (desempenho);
- Funcionamento sob condições anormais (robustez ou escalabilidade);
- Facilidade de operar o sistema (facilidade de uso);
- Facilidade de integração com outros produtos de software (compatibilidade);
- Preservação da disponibilidade e da integridade das informações armazenadas (confiabilidade);
- Proteção contra acessos indevidos (segurança).

A maior parte dos critérios supracitados são importantes para o ambiente ODE e suas ferramentas. As ferramentas propostas neste trabalho não são exceção. Porém é importante ressaltar alguns critérios que se sobressaem perante os outros, a saber: manutenibilidade, facilidade de uso e compatibilidade.

4.3.1 Manutenibilidade

Manutenibilidade é um critério de extrema importância para qualquer software, pois mudanças são inevitáveis. Para se considerar esse requisito não funcional, nas ferramentas descritas neste trabalho foi utilizada uma arquitetura de software (detalhada na Seção 6.1), com o objetivo de permitir fazer manutenções nas interfaces das ferramentas, sem modificação alguma nas regras de negócio ou domínio do problema. Além disso, os padrões de documentação e programação definidos para o Projeto ODE foram sempre respeitados.

4.3.2 Facilidade de Uso

Um software que apresenta uma interface complexa e que não facilita o uso pelo cliente está fadado ao desuso, já que a quantidade de sistemas com um mesmo propósito cresce cada vez mais, permitindo que um usuário escolha um outro sistema com a interface mais amigável e que atenda às funcionalidades requisitadas.

Visando melhorar a usabilidade, neste trabalho foi utilizado um padrão de interfaces para as ferramentas que seguem um fluxo de trabalho bem definido, provendo um fluxo natural das funcionalidades encontradas em cada uma delas. Além disso, as funcionalidades implementadas são tratadas de forma mais clara, utilizando-se de mensagens explicativas, painéis e outros elementos de interfaces gráficas para facilitar o uso de cada ferramenta.

4.3.3 Compatibilidade

Todo ADS deve ter como principal propósito a integração das ferramentas que o compõem, provendo facilidade de troca de informações. Nas ferramentas descritas neste trabalho foi feito um esforço no sentido de prover uma alta integração, de forma que os dados compartilhados entre as ferramentas possam ser visualizados e até mesmo alterados tanto por uma quanto por outras.

Capítulo 5

Projeto, Implementação e Testes

Diferente da análise, nesta fase o sistema é modelado priorizando os aspectos tecnológicos a serem utilizados para que este possa ser materializado. Esses aspectos levam em conta, dentre outros, a linguagem de programação utilizada na implementação do sistema, características de interface com o usuário, a arquitetura do software e as formas de acesso e persistência de dados.

As ferramentas desenvolvidas neste trabalho utilizam os padrões e tecnologias definidos no Projeto ODE, sendo a implementação feita na linguagem de programação Java, acompanhada de um banco de dados relacional com o *framework* de persistência Hibernate (BAUER et al., 2005). Além disso, a arquitetura de software adotada no projeto é composta de cinco componentes, sendo eles: Componente de Domínio do Problema (Cdp), Componente de Gerência de Dados (Cgd), Componente de Gerência de Tarefas (Cgt), Componente de Interação Humana (Cih) e Componente de Controle de Interação (Cci).

Como este trabalho envolve a reestruturação de duas ferramentas, as ferramentas de apoio à alocação de recursos e de agenda, o projeto foi definido a partir de trabalhos existentes, adequando-os à nova arquitetura, além de se adicionar as novas funcionalidades especificadas no documento de especificação de requisitos. Por fim, buscou-se adequar essas ferramentas a interfaces que proporcionam maior usabilidade. Maiores detalhes sobre o projeto dessas ferramentas podem ser encontrados nos anexos A e B, respectivamente.

Este capítulo apresenta parte do projeto, implementação e testes das duas ferramentas desenvolvidas neste trabalho. A Seção 5.1 mostra a nova arquitetura de software proposta para o Projeto ODE. A Seção 5.2 discute sobre a nova camada de persistência utilizada e sobre os padrões que foram definidos para o bom funcionamento da mesma. A Seção 5.3 apresenta o utilitário de interface gráfica para ferramentas com fluxo de tarefas definido para o ambiente ODE e aplicado na ferramenta de Agenda. Nas Seções 5.4 e 5.5 são apresentados os projetos das ferramentas de Apoio a Alocação de Recursos e Agenda, respectivamente. A Seção 5.6 discute a

implementação e os testes realizados. Por fim, a Seção 5.7 comenta detalhadamente o funcionamento dessas ferramentas.

5.1 Arquitetura do Sistema

A primeira fase do projeto é a definição da arquitetura a ser utilizada, na qual são definidas a organização e as dependências dos componentes do sistema a ser desenvolvido, considerando os fatores técnicos (plataforma de implementação) e requisitos não funcionais.

Atualmente, o ambiente ODE encontra-se com parte de suas ferramentas utilizando uma arquitetura organizada em quatro componentes e outra com uma arquitetura organizada em cinco componentes.

Na arquitetura composta por quatro componentes os objetos do Componente de Interação Humana (Cih), responsáveis por definir as interfaces que exibem informações ao usuário ou capturam dados do mesmo, se comunicam diretamente com classes que realizam regras de negócio (Componente de Gerência de Tarefas - Cgt) extraídas dos casos de uso da especificação de requisitos. Ainda nessa arquitetura, o Componente de Domínio do Problema (Cdp) define as classes extraídas do domínio do problema da fase de análise e têm acesso às funcionalidades do mecanismo de persistência definidos nas classes do Componente de Gerência de Dados (Cgd). A Figura 5.1 apresenta essa arquitetura.

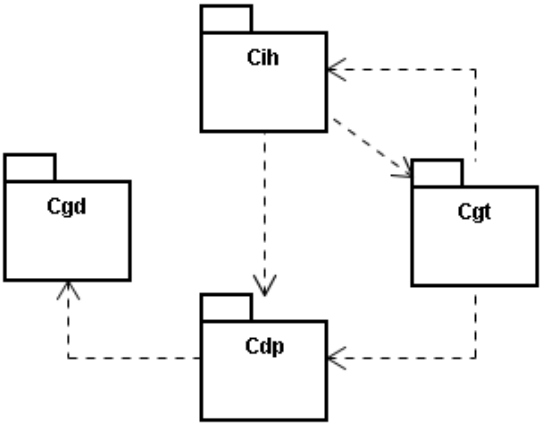

Figura 5.1 - Arquitetura de 4 Camadas

Apesar de algumas ferramentas do ambiente ODE utilizarem esse tipo de arquitetura, esta apresenta algumas desvantagens, sendo elas:

- O Cih fortemente acoplado com o Cgt prejudica a utilização de uma mesma interface ou funcionalidade em outra parte do ambiente;
- Manutenções nas interfaces provocam mudanças também no Cgt, devido ao grande acoplamento existente entre os dois;
- As classes do Cdp se tornam mais complexas por tratarem as funcionalidades do mecanismo de persistência, ou seja, nelas são adicionadas operações que são diretamente repassadas ao Cgd.

Com o objetivo de reduzir esses problemas, foi criada a arquitetura de cinco camadas, incluindo um novo componente, o Componente de Controle de Interação (Cci), visando a desacoplar o Cih do Cgt, permitindo que as funcionalidades do Cgt possam ser mais facilmente reutilizadas em outras partes do ambiente. Nessa arquitetura, o componente de interface se comunica com um controlador de interface (Cci) especializado para a mesma. Além disso, o Cdp possui apenas classes com métodos e atributos relacionados ao domínio, deixando que o Cgt faça a comunicação com o Cgd. A Figura 5.2 apresenta a nova arquitetura, sendo esta a utilizada neste trabalho.

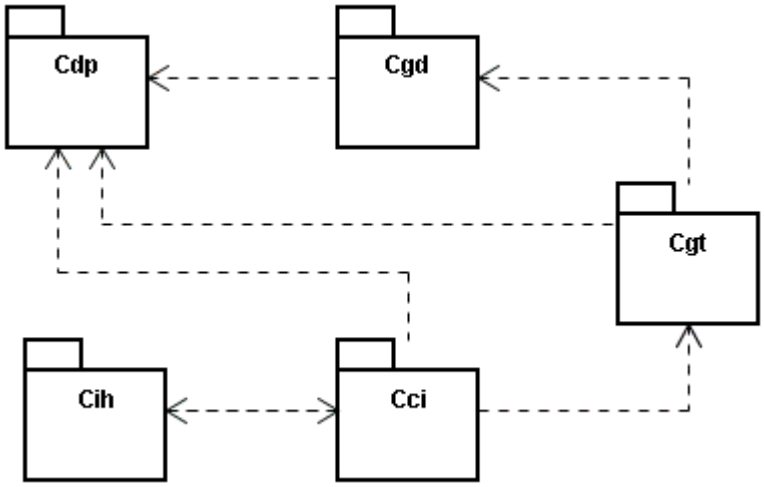

Figura 5.2 - Arquitetura de 5 Componentes de ODE

5.2 A Camada de Persistência de ODE

As classes do Componente de Gerência de Dados têm como objetivo abstrair o acesso ao banco de dados usado na implementação, dando maior flexibilidade e transparência ao sistema.

Para desenvolver a camada de persistência de ODE, são utilizados os padrões Objeto de Acesso a Dados (*Data Access Object* - DAO) (SUN, 2001) e Fábrica Abstrata (GAMMA et al., 1995), incluindo o uso do *framework* de persistência *Hibernate* (BAUER et al., 2005). O padrão DAO fornece uma interface flexível entre aplicações e os mecanismos de persistência (banco de dados, arquivo, memória etc) e, portanto, o padrão torna-se parte integrante da camada de persistência. A Figura 5.3 mostra as principais classes do utilitário de persistência de ODE.

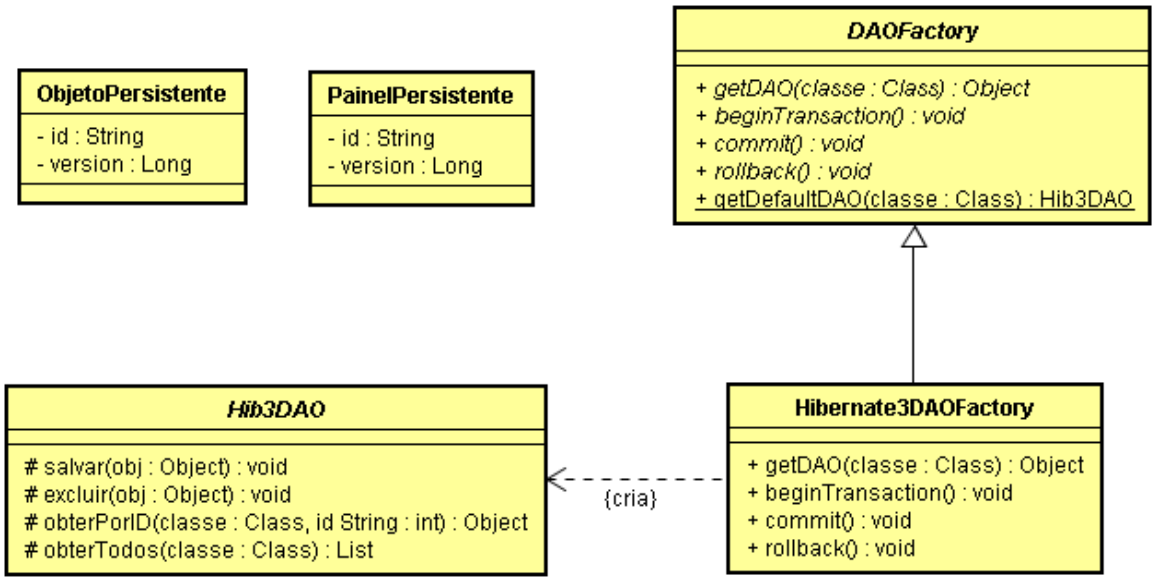

Figura 5.3 - Pacote *Utilitario::Persistencia::hibernate*

Classes de domínio e itens gráficos que devem ser persistidos devem herdar de `ObjetoPersistente` e `PainelPersistente`, respectivamente. Além disso, cada classe DAO relacionada deve herdar de `Hib3DAO`, que possui as funcionalidades básicas de acesso ao mecanismo de persistência, e deve implementar a interface DAO associada, provendo flexibilidade no momento da criação de novas operações especializadas para um certo elemento de domínio.

5.3 Utilitário de Interface Gráfica para Fluxos de Tarefas

Esta seção apresenta o utilitário de interface gráfica orientada a tarefas, desenvolvido neste trabalho para ser utilizado por ferramentas do ambiente ODE. A Figura 5.4 apresenta o diagrama de classes da interface. Como se pode perceber, em relação à arquitetura padrão do ambiente ODE, dois pacotes são tratados neste utilitário: o Componente de Interação Humana (CIH) e o Componente de Controle de Interação(CCI).

As classes `CtrlFluxoTarefas` e `PainelFluxoTarefas` representam, respectivamente, o controlador de interface desse utilitário e o painel responsável por exibir e controlar o fluxo de tarefas da interface. É importante lembrar que a classe `CtrlFluxoTarefas` herda da classe `AplFerramentaProjeto`, que por sua vez herda de `AplFerramenta`. Por isso, qualquer classe que herdar da classe `CtrlFluxoTarefas` precisará implementar os métodos abstratos da classe `AplFerramenta`.

As classes `PainelMensagem` e `PainelConteudo` servem para exibir a mensagem e o painel associados ao `BotaoSubTarefa`, respectivamente. A classe `PainelGuia` é responsável por exibir os botões das tarefas e subtarefas. Já o `PainelNavegador` é responsável por exibir os botões que permitem navegar pelas subtarefas.

Como a idéia principal deste utilitário de interface gráfica é controlar o fluxo de tarefas, as classes `BotaoTarefa` e `BotaoSubTarefa` representam, respectivamente, uma tarefa e uma subtarefa da interface.

A Figura 5.5 mostra alguns detalhes da interface. Os números na figura apontam para partes importantes da interface, conforme descrito a seguir:

1. Botões de Tarefas: representam as tarefas do processo sendo suportado pela interface, sendo que cada tarefa possui um conjunto de subtarefas.
2. Botões de Subtarefas: representam as subtarefas de uma tarefa, para as quais há painéis e mensagens específicos.
3. Painel de Mensagem: representa o local onde é exibida a mensagem associada à subtarefa selecionada.

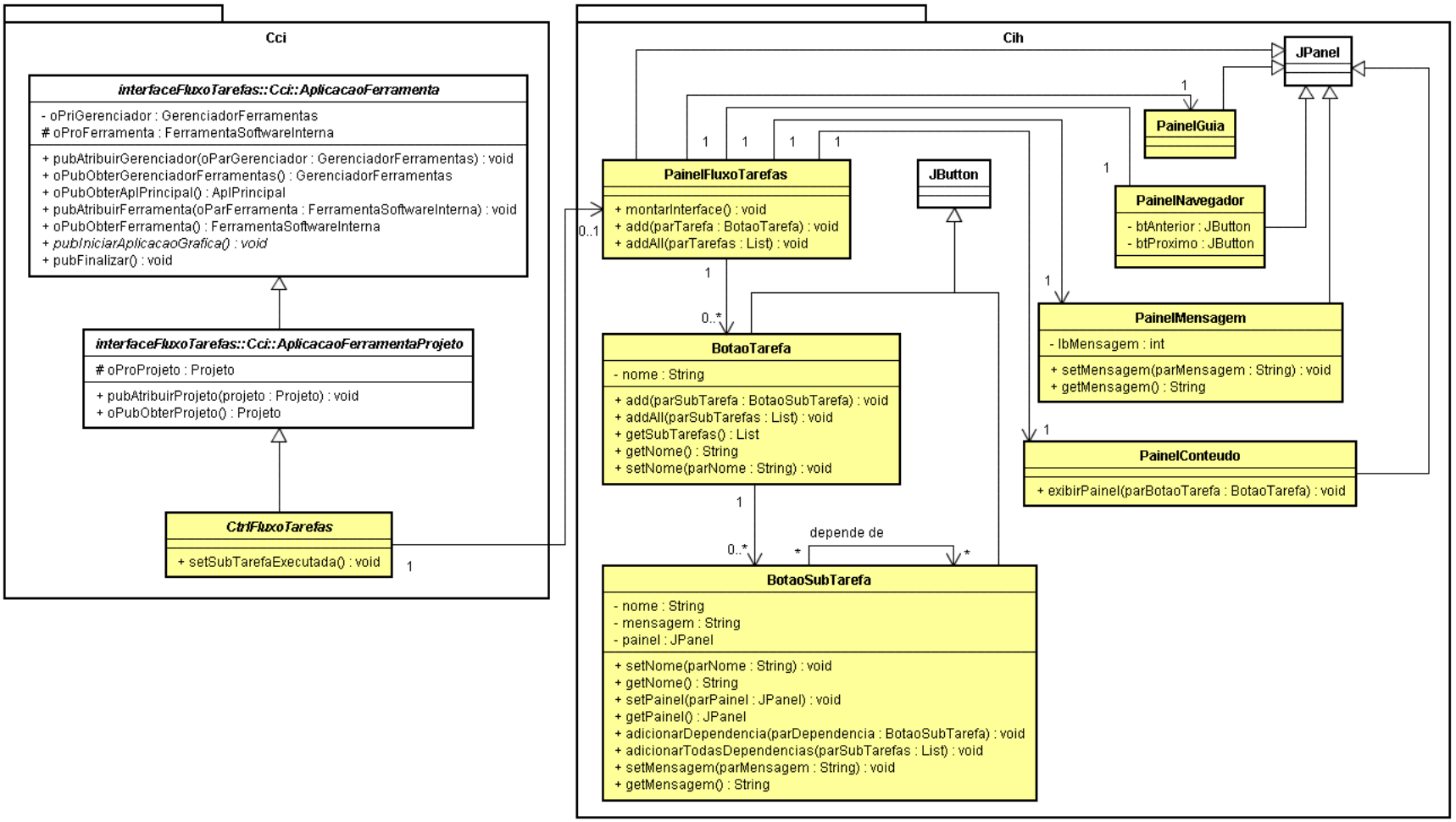

Figura 5.4 – Diagrama de classes do Utilitário de Interface

4. Painel Navegador: tem como funcionalidade navegar entre as subtarefas da interface. A navegação é feita entre todas as subtarefas habilitadas para serem executadas, desde a primeira subtarefa da primeira tarefa até a última subtarefa última tarefa.
5. Painel Conteúdo: representa o local onde é exibido o painel associado à subtarefa selecionada.

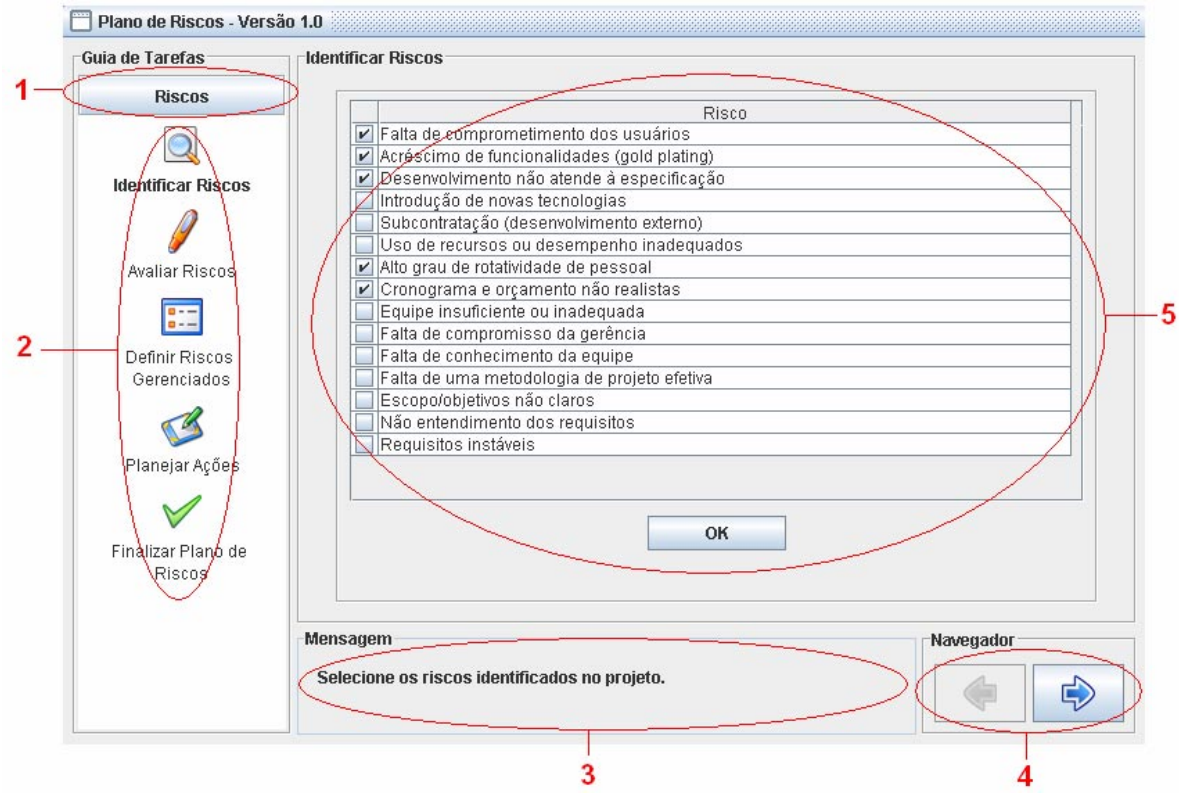

Figura 5.5 – Detalhes da Interface Implementada

Para maiores detalhes sobre o projeto, a estrutura e como utilizar esse utilitário de interface, veja Anexo C.

5.4 Ferramenta de Alocação de Recursos

O diagrama de pacotes da Figura 5.6 mostra as dependências existentes entre os pacotes físicos definidos para essa ferramenta. Esses pacotes espelham uma decomposição com base no domínio do problema e, portanto, o diagrama da Figura 5.6 é idêntico ao correspondente

diagrama da fase de análise. Contudo, cada um desses pacotes é decomposto segundo a arquitetura de cinco componentes, mostrada na Figura 5.2.

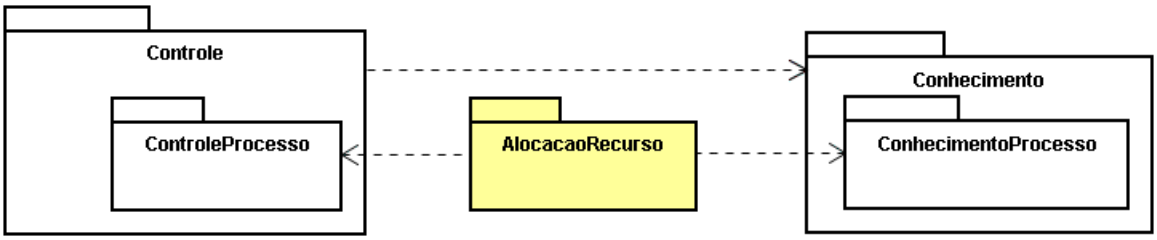

Figura 5.6 - Diagrama de pacotes

O pacote *AlocacaoRecurso* contém as classes que definem as funcionalidades relacionadas a alocações de recursos humanos, de software e de hardware no ambiente ODE.

5.4.1 Componente do Domínio do Problema (Cdp)

O diagrama de classes do Componente do Domínio do Problema do pacote *AlocacaoRecurso*, mostrado na Figura 5.7, foi originado a partir do modelo de análise, adequando-o à plataforma de implementação.

Em termos de projeto, poucas mudanças foram feitas em relação ao modelo de classes de análise. Além dos tipos de atributos e navegabilidades entre as classes inseridos no modelo, a classe AlocacaoEquipe, que era uma classe associativa no modelo de análise, agora foi transformada em uma classe regular para deixar o modelo mais próximo da implementação

Outra alteração que merece destaque é a criação das classes EstadoAlocacaoRH e EstadoAtividade, apresentadas no diagrama da Figura 5.8, para representarem, respectivamente, os estados dos objetos das classes AlocacaoRH e Atividade. É importante saber que essas classes não são persistentes e têm seus valores constantes, definidos pelos respectivos diagramas de estados, sendo que o diagrama de estados da classe Atividade pode ser encontrado em (SEGRINI, 2007).

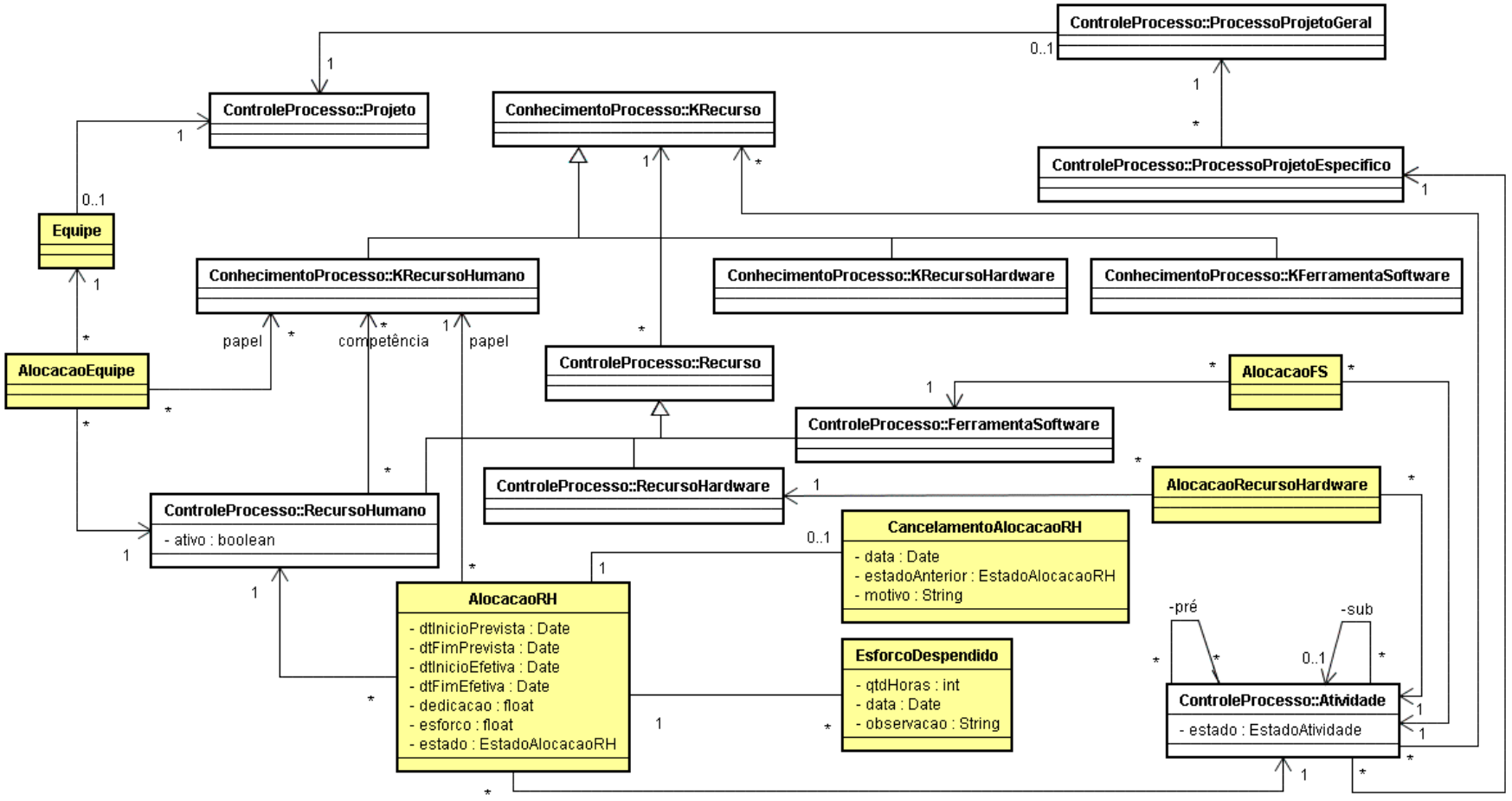

Figura 5.7 – Componente de Domínio do Problema do pacote *AlocacaoRecurso*

Para cada estado da classe `AlocacaoRH` mostrado no respectivo diagrama de estado, foi criada uma constante que, na linguagem em Java, é um atributo público (*public*, representado no diagrama pelo símbolo '+'), estático (*static*, representado no diagrama pelo sublinhado) e final, informando que o valor do atributo não pode ser alterado (não existe representação gráfica correspondente para indicar essa propriedade).

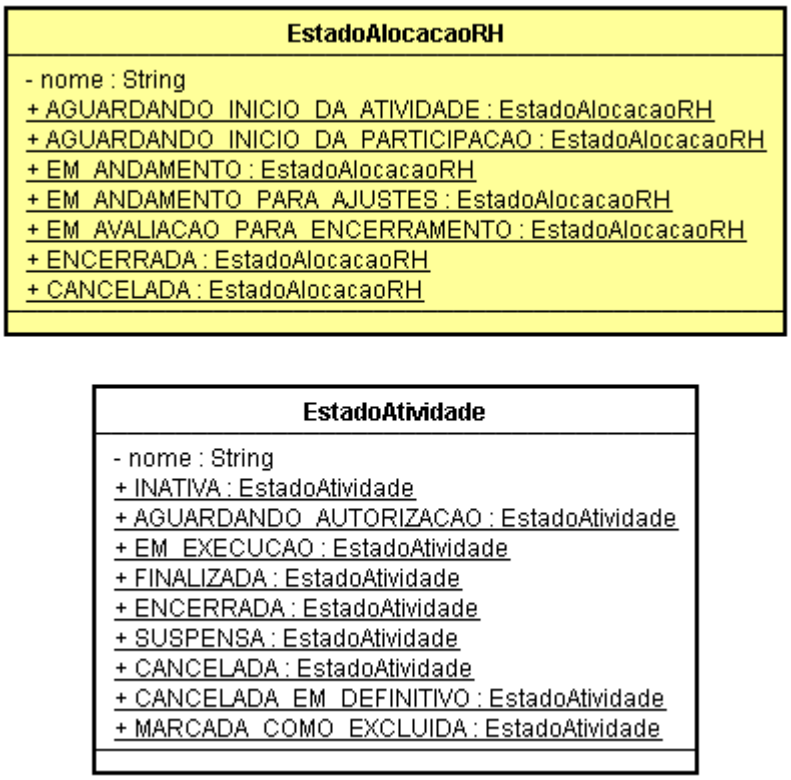

Figura 5.8 – Classes de Estados

5.4.2 Componente de Gerência de Dados (Cgd)

A Figura 5.9 apresenta o diagrama de classes do *Cgd* da ferramenta Alocação de Recursos.

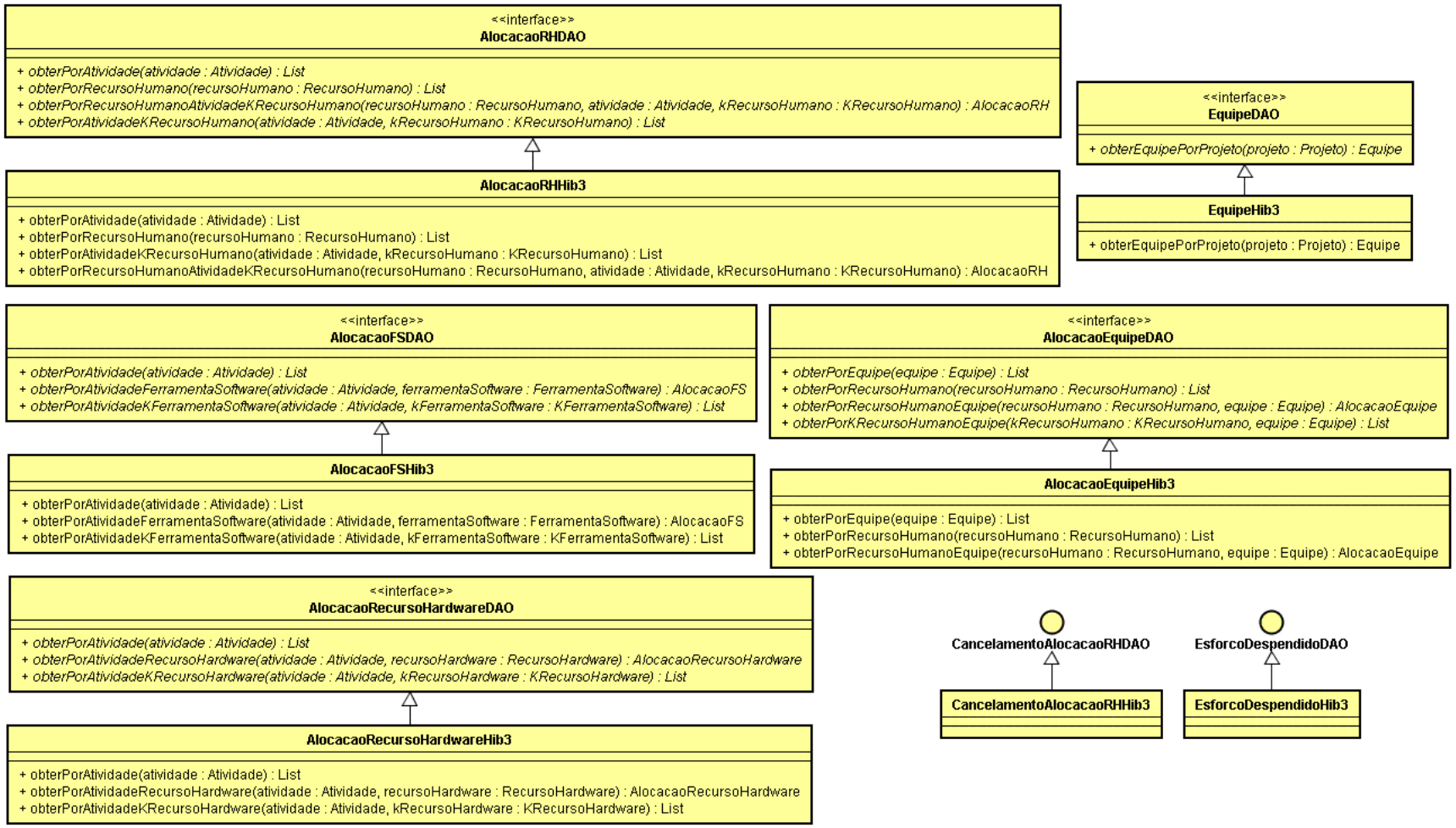

Figura 5.9 - Componente de Gerência de Dados do pacote *AlocacaoRecurso*

Ainda que não mostrado no diagrama, as classes apresentadas na Figura 5.9 herdam da classe abstrata `Hib3DAO` que implementa os métodos básicos de manipulação de objetos, a saber: *salvar*, *excluir*, *obterPorId* e *obterTodos*, conforme mostrado na Figura 5.3. Em algumas classes, foi necessário adicionar métodos auxiliares para recuperar objetos requeridos através de um ou mais objetos relacionados ao mesmo, tal como o método *obterPorRecursoHumano* da classe `AlocacaoRHDAO`, que retorna os objetos da classe `AlocacaoRH` que se relacionam com um objeto `RecursoHumano` específico.

5.4.3 Componente de Gerência de Tarefas (Cgt)

O Componente de Gerência de Tarefas desse pacote possui as classes `AplAlocacaoRecurso`, `AplControlarAlocacaoRH` e `AplRegistrarEsforcoDespendido`. A primeira define as funcionalidades requeridas para tratar os casos de uso "Definir Equipe do Projeto", "Definir Alocações de Recursos a Atividades" e "Editar Alocação de Recurso Humano". Já as classes `AplControlarAlocacaoRH` e `AplRegistrarEsforcoDespendido` tratam de controle do andamento das alocações de recursos humanos (caso de uso "Controlar Andamento de Alocação de Recurso Humano a Atividade") e do registro de esforços despendidos (caso de uso "Registrar Esforço Despendido"), respectivamente.

A Figura 5.10 mostra o Componente de Gerência de Tarefas desse pacote.

Figura 5.10 - Componente de Gerência de Tarefas do Pacote *AlocacaoRecurso*

Pode-se observar que, na figura acima, nenhuma classe do Cgt possui relacionamentos diretos com objetos dos pacotes Cdp e Cgd. Ou seja, nenhuma classe tem atributos que são objetos desses pacotes. Porém, objetos dos pacotes Cdp e Cgd são instanciados dentro dos métodos das classes do Cgt, o que explica os relacionamentos de dependência entre os pacotes Cdp, Cgd e Cgt, mostrados na Figura 5.2.

Além disso, as classes do Cgt possuem apenas métodos estáticos e, por isso, é necessário informar ao controlador se um método foi executado corretamente ou não. Para resolver tal problema, foram definidos tipos dos retornos padrão dos métodos das classes do Cgt do ambiente ODE, sendo eles números inteiros. Para maiores detalhes, vide documentação de projeto no Anexo A.

5.4.4 Componente de Interação Humana (Cih)

O Componente de Interação Humana do pacote *AlocacaoRecurso* comporta as interfaces utilizadas para a realização dos casos de uso definidos no documento de especificação de requisitos. A Figura 5.11 apresenta o diagrama de classes desse componente do pacote *AlocacaoRecurso.*

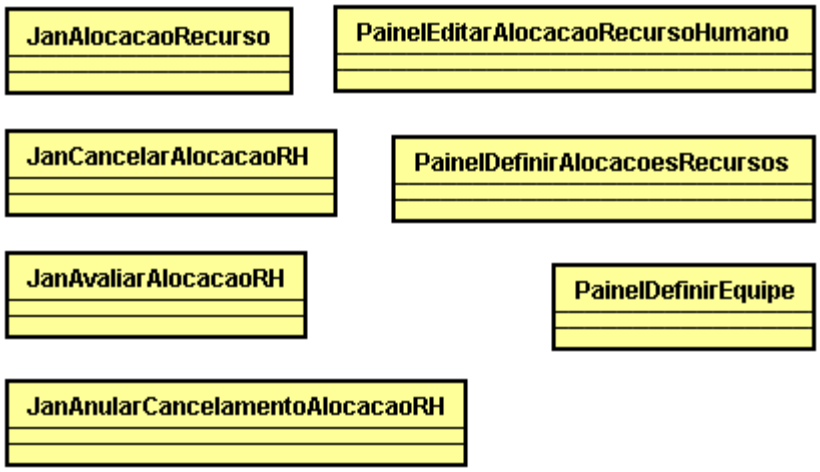

Figura 5.11 - Componente de Interface Humana do pacote *AlocacaoRecurso*

A ferramenta é composta de quatro janelas: `JanAlocacaoRecurso`, `JanAvaliarAlocacaoRH`, `JanCancelarAlocacaoRH` e `JanAnularCancelamentoAlocacaoRH`, sendo que `JanAlocacaoRecurso` é responsável por comportar os painéis `PainelDefinirEquipe`, `PainelDefinirAlocacoesRecursos` e `PainelEditarAlocacaoRecursoHumano`.

O `PainelDefinirEquipe` apresenta para o desenvolvedor os recursos humanos cadastrados no ambiente ODE com suas respectivas especialidades, para que uma equipe possa ser definida, conforme a descrição do caso de uso "Definir Equipe do Projeto". Já o `PainelDefinirAlocacoesRecursos` permite que o usuário faça alocações de recursos às atividades de um projeto, de acordo o caso de uso "Definir Alocações de Recursos a Atividades". O `PainelEditarAlocacaoRecursoHumano` apresenta informações de uma alocação de recurso selecionada para que esta possa ser editada, conforme o caso de uso "Editar Alocação de Recurso Humano".

As janelas `JanAvaliarAlocacaoRH`, `JanCancelarAlocacaoRH` e `JanAnularCancelamentoAlocacaoRH` são responsáveis pela avaliação, cancelamento e anulação do cancelamento de uma alocação de recurso humano, relacionados, respectivamente, aos eventos de caso de uso *Avaliar Participação de Recurso Humano na Atividade, Cancelar Alocação de Recurso Humano a Atividade e Anular Cancelamento de Alocação de Recurso Humano a Atividade* do caso de uso *Controlar Andamento de Alocação de Recurso Humano a Atividade*.

Conforme discutido na próxima seção, é de responsabilidade dos controladores (Cci) a exibição das interfaces do sistema, o que explica a inexistência de relacionamentos entre as mesmas.

5.4.5 Componente de Controle de Interação (Cci)

O Componente de Controle de Interação tem como objetivo tratar a comunicação de dados entre as interfaces da ferramenta (Cih) e as classes de aplicação (Cgt), como mostra a Figura 5.12.

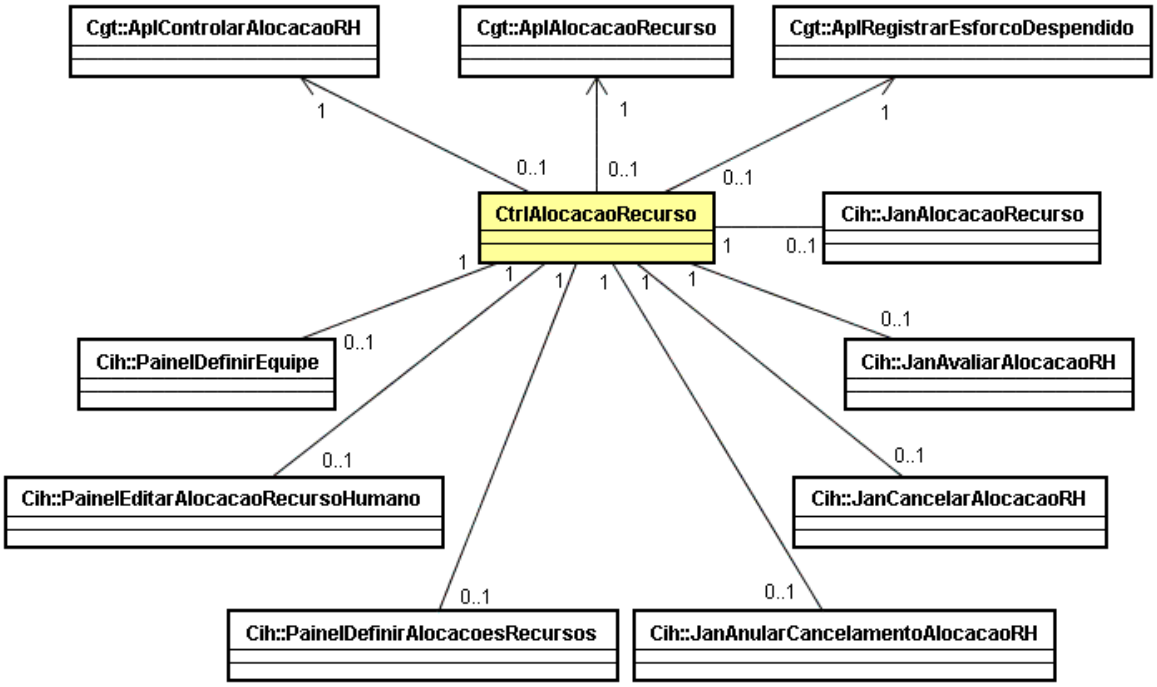

Figura 5.12 - Componente de Controle de Interação do Pacote *AlocacaoRecurso*

A navegabilidade no sentido de `CtrlAlocacaoRecurso` para as classes do Cgt permite que as funcionalidades da ferramenta também possam ser usadas por outras ferramentas do ambiente ODE e, caso exista necessidade de alterar a interface da ferramenta, apenas o controlador da mesma deve ser refeito de acordo com as necessidades exigidas pelas modificações.

O controlador conhece todas as operações necessárias para atender às requisições das interfaces. Quando uma interface necessita de alguma funcionalidade da classe de aplicação, o controlador faz uma requisição à aplicação. Em seguida, esta retorna um resultado para o controlador que o interpreta e atualiza as interfaces, caso haja necessidade.

Apesar dos métodos não estarem explícitos no diagrama da Figura 5.12, o controlador possui vários métodos cujas principais funcionalidades são de atualização das interfaces e de requisições às aplicações.

5.5 Ferramenta de Agenda

O diagrama de pacotes da Figura 5.13 mostra as dependências existentes entre os pacotes físicos definidos para essa ferramenta. Esses pacotes espelham uma decomposição com base no domínio do problema e, portanto, esse diagrama é idêntico ao correspondente diagrama da fase de análise. Contudo, cada um desses pacotes é decomposto segundo uma arquitetura de cinco componentes, mostrada na Figura 5.2.

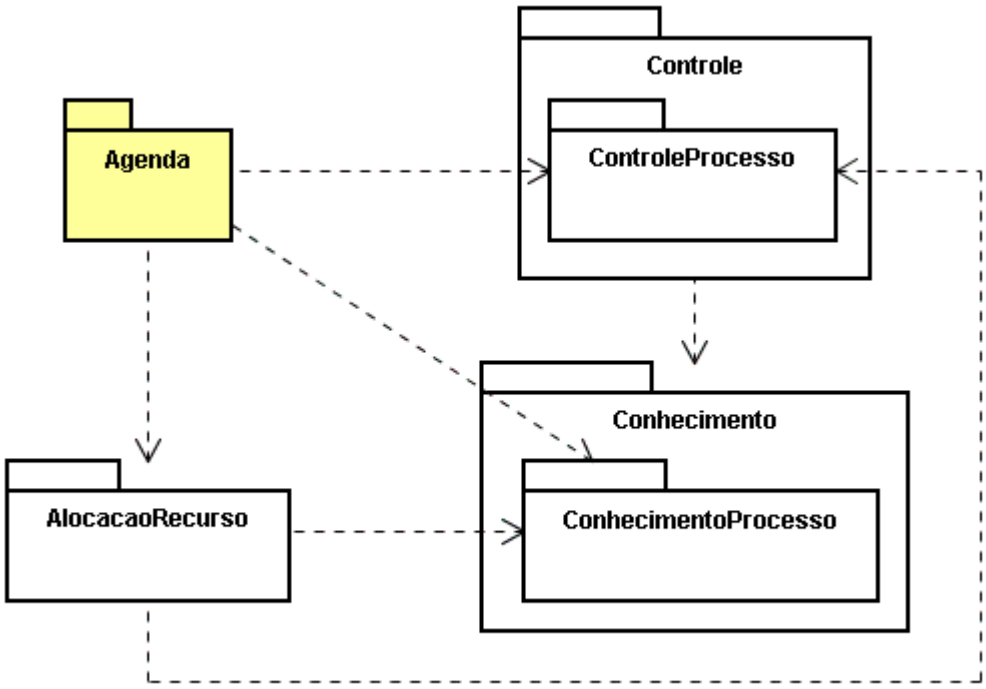

Figura 5.13 - Diagrama de pacotes

O pacote *Agenda* contém as classes que definem as funcionalidades relacionadas à agenda de um recurso humano no ambiente ODE.

5.5.1 – Componente do Domínio do Problema (Cdp)

O diagrama de classes do Componente do Domínio do Problema do pacote *Agenda* foi criado a partir do modelo de análise, adequando-o à plataforma de implementação, como mostra a Figura 5.14.

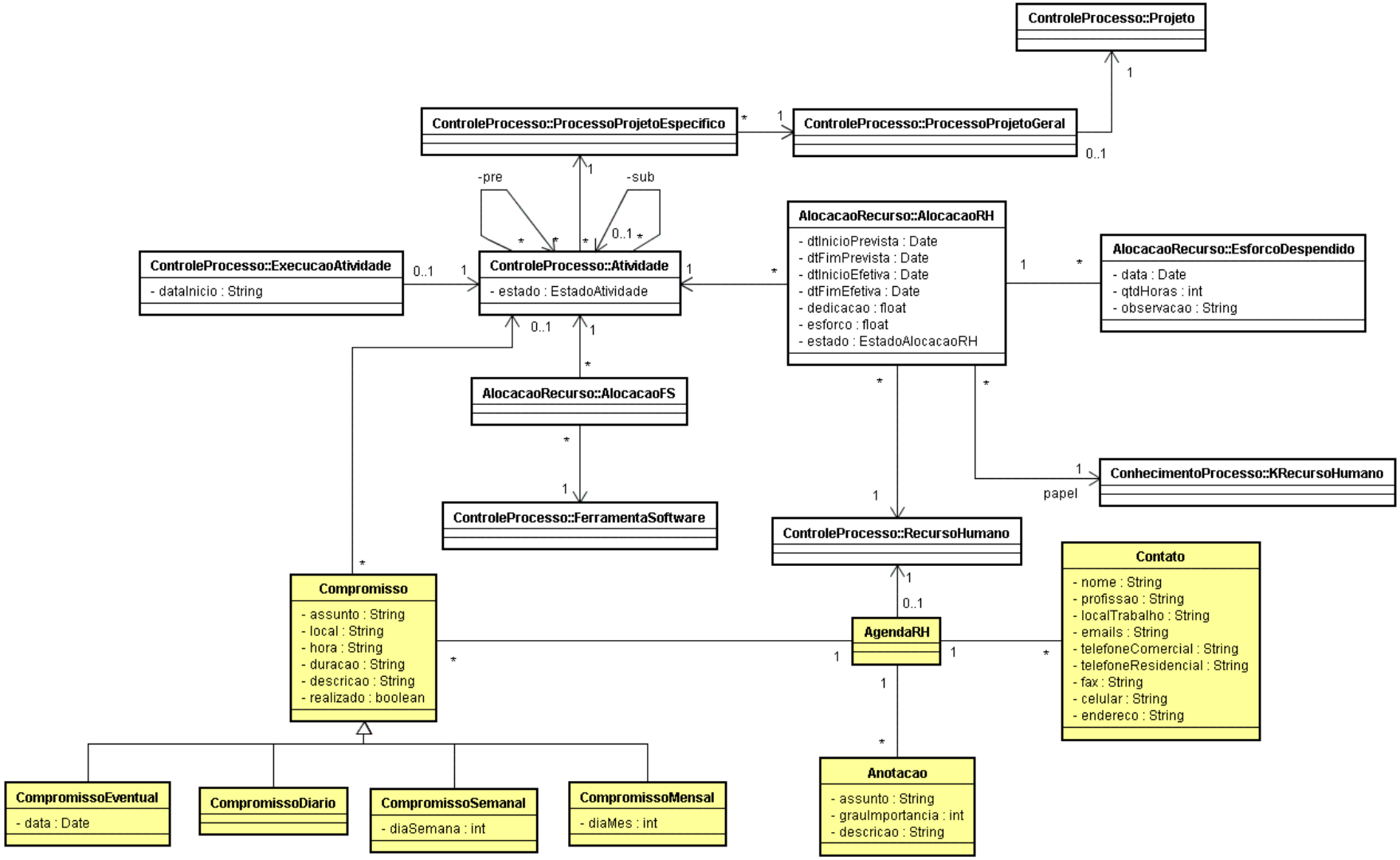

Figura 5.14 – Componente de Domínio do Problema do pacote *Agenda*

Em termos de projeto, poucas mudanças foram feitas em relação ao modelo de classes do documento de análise. Além dos tipos de atributos e das navegabilidades entre as classes inseridos no modelo, uma alteração que merece destaque é a criação das classes `EstadoAlocacaoRH` e `EstadoAtividade`, conforme discutido na Seção 5.4.1.

5.5.2 – Componente de Gerência de Dados (Cgd)

A Figura 5.15 apresenta o diagrama de classes do pacote Cgd da ferramenta Agenda.

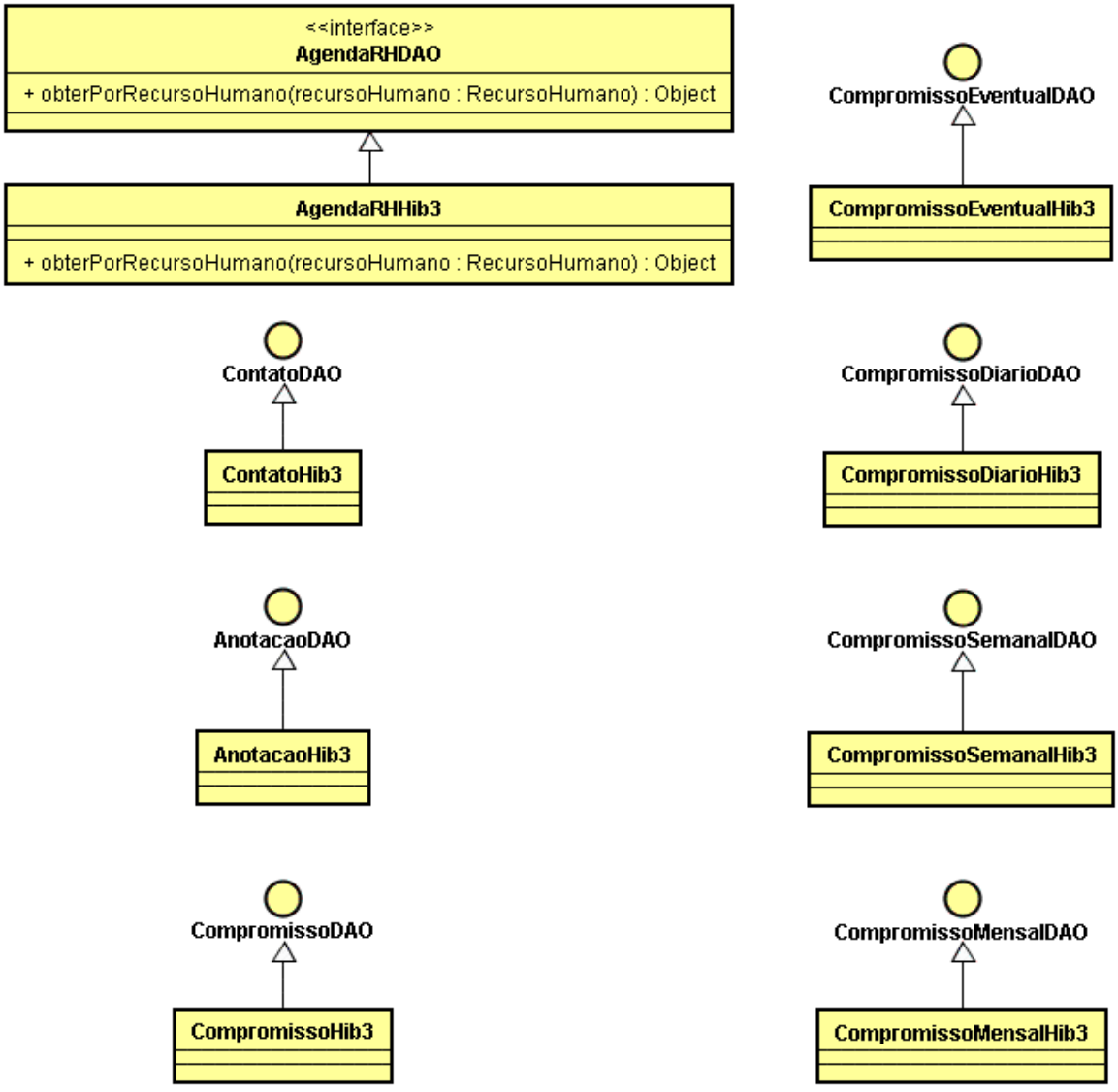

Figura 5.15 - Componente de Gerência de Dados do pacote *Agenda*

Conforme apontado anteriormente, todas as classes apresentadas na Figura 5.15, além de implementarem as interfaces mostradas, herdam da classe abstrata `Hib3DAO`. No caso da classe `AgendaRHHib3` foi necessário adicionar um método auxiliar para recuperar a agenda de um recurso humano (método *obterPorRecursoHumano*), que retorna o objeto da classe `AgendaRH` que se relacionam com um objeto `RecursoHumano` específico. Isso é necessário, porque na associação entre `AgendaRH` e `RecursoHumano` optou-se apenas pela navegabilidade no sentido `AgendaRH` → `RecursoHumano`, conforme mostrado na Figura 5.14.

5.5.3 – Componente de Gerência de Tarefas (Cgt)

O Componente de Gerência de Tarefas desse pacote possui as classes `AplAgenda`, `AplControlarCompromisso`, `AplCadastrarContato` e `AplControlarAnotacao`. A classe `AplAgenda` trata as funcionalidades relativas à seleção e visualização de alocações de recursos humanos (caso de uso "Visualizar Alocações de Recurso Humano a Atividades"), seleção da ferramenta de trabalho (caso de uso "Selecionar Ferramenta de Trabalho") e visualização das informações diárias do desenvolvedor (caso de uso "Visualizar Informações Diárias"). Já as classes `AplControlarCompromisso`, `AplCadastrarContato` e `AplControlarAnotacao`, são responsáveis, respectivamente, pelo cadastro de compromissos (caso de uso "Controlar Compromisso"), contatos (caso de uso "Cadastrar Contato") e anotações (caso de uso "Controlar Anotação") de um desenvolvedor. A Figura 5.16 mostra o diagrama de classes do Componente de Gerência de Tarefas desse pacote.

Vale destacar que as considerações feitas para esse componente no pacote *AlocacaoRH* no que se refere às dependências com os componentes Cdp e Cgd, bem como em relação aos retornos dos métodos, são análogas.

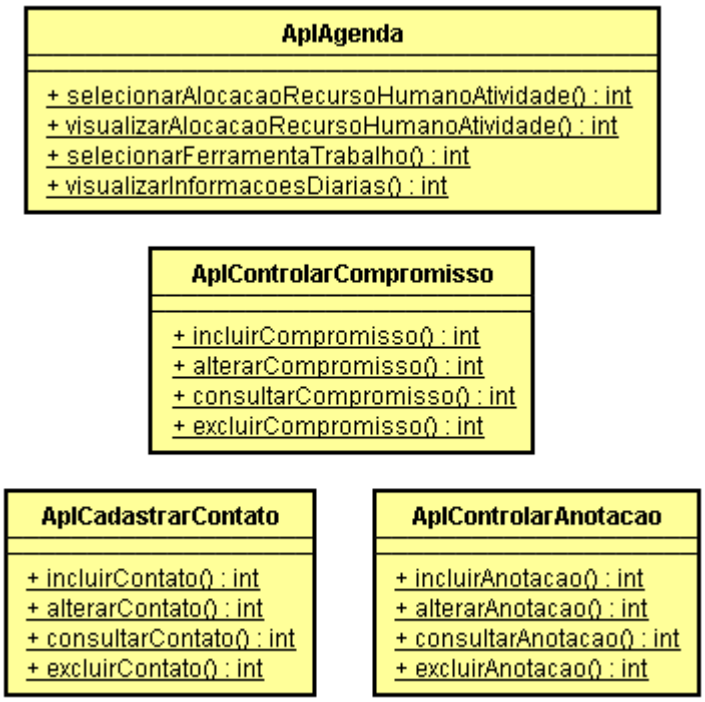

Figura 5.16 - Componente de Gerência de Tarefas do Pacote *Agenda*

5.5.4 – Componente de Interação Humana (Cih)

O Componente de Interação Humana do pacote *Agenda* define as interfaces que interagem com o usuário para a realização dos casos de uso definidos no documento de especificação de requisitos. Para a implementação desse componente foi utilizado o novo padrão de interface para ferramentas com fluxos de tarefas do ambiente ODE, apresentado na Seção 5.3. A Figura 5.17 mostra o componente de interação humana relativo ao pacote *Agenda.*

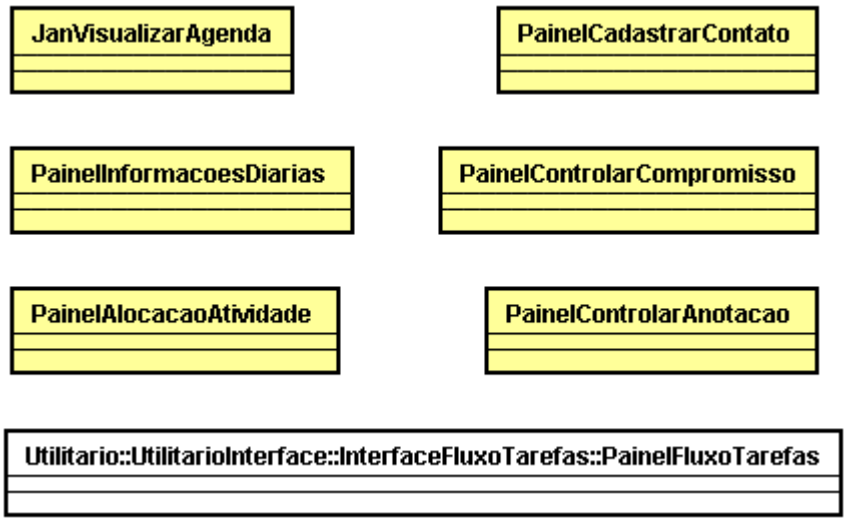

Figura 5.17 - Componente de Interface Humana do pacote *Agenda*

A ferramenta possui uma única janela (`JanVisualizarAgenda`) que comporta o `PainelFluxoTarefas`. Esse painel agrupa e organiza a ordem de exibição dos painéis em tela, conforme o novo padrão de interface desenvolvido para controlar o fluxo de tarefas executadas pelos usuários em uma ferramenta do ambiente ODE.

O `PainelInformacoesDiarias` apresenta para o desenvolvedor as principais informações diárias sobre suas alocações no ambiente ODE, seus compromissos e suas anotações, tratando do caso de uso *Visualizar Informações Diárias*. Já o `PainelAlocacaoAtividade` apresenta as alocações a atividades que o desenvolvedor possui no ambiente ODE, sendo ainda possível selecionar ferramentas de software disponíveis, registrar esforços despendidos e solicitar encerramento da participação na atividade, conforme a descrição dos casos de uso *Visualizar Alocações de Recurso Humano a Atividades*, *Selecionar Ferramenta de Trabalho* e *Registrar Esforço Despendido* (ferramenta *AlocaODE*) e o evento *Solicitar Encerramento da Participação na Atividade* do caso de uso *Controlar Andamento de Alocação de Recurso Humano a Atividade* (ferramenta *AlocaODE*) .

Os painéis `PainelControlarCompromisso`, `PainelControlarAnotacao` e `PainelCadastrarContato` mostram para o desenvolvedor funcionalidades relacionadas ao controle de compromissos, cadastro de contatos e cadastro de anotações, conforme a descrição dos casos de uso *Controlar Compromisso*, *Controlar Anotação* e *Cadastrar Contato.*

5.5.5 – Componente de Controle de Interação (Cci)

Conforme citado na Seção 5.4.5, o Componente de Controle de Interação tem como objetivo fazer a comunicação de dados entre os objetos da interface da ferramenta (Cih) e as classes de aplicação (Cgt), como mostra a Figura 5.18.

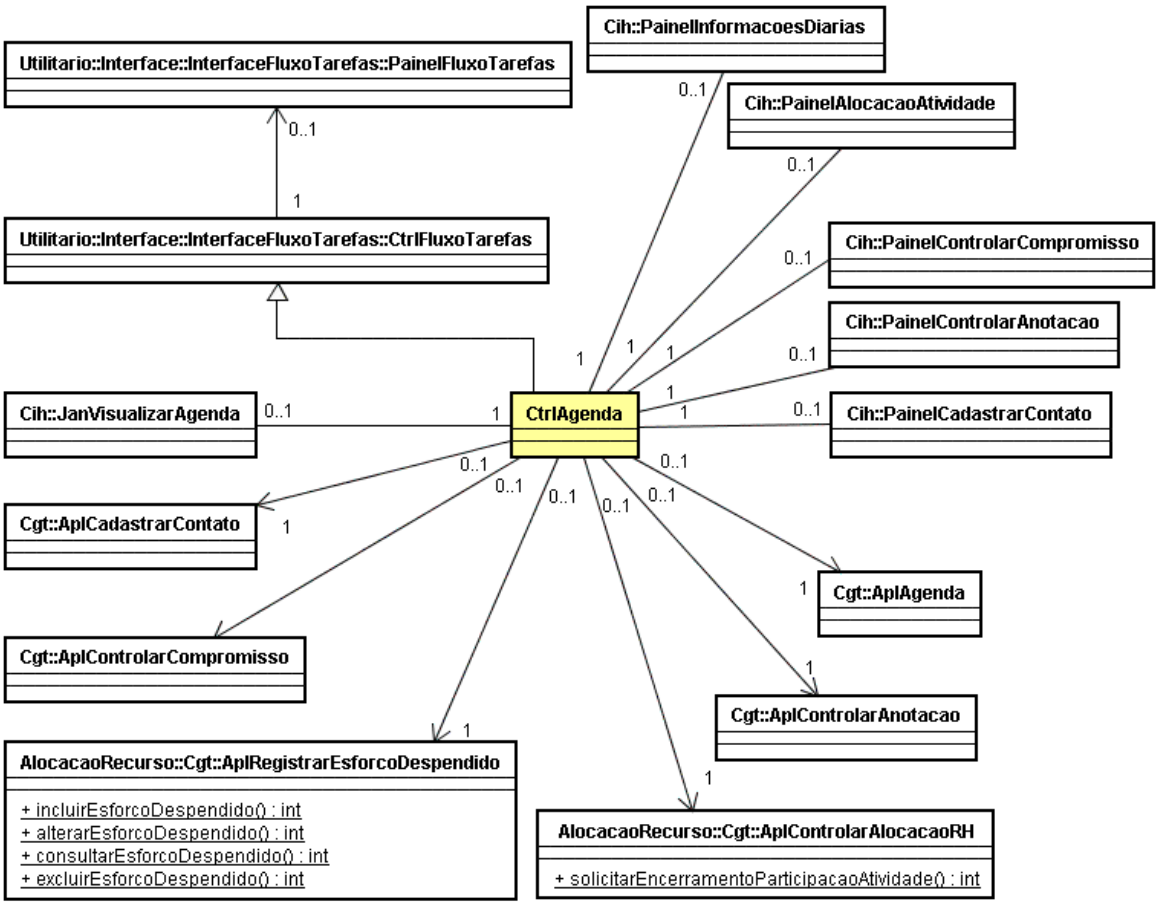

Figura 5.18 - Componente de Controle de Interação do Pacote *Agenda*

A navegabilidade no sentido de `CtrlAgenda` para as classes do Cgt, tanto da ferramenta Agenda quanto da ferramenta de Alocação de Recursos, é justamente feita para permitir que as funcionalidades da ferramenta possam ser utilizadas por outras ferramentas do ambiente ODE e, caso exista necessidade de modificar a interface, apenas o controlador deve ser reformulado.

Quando uma interface solicita a realização de uma funcionalidade de uma classe de aplicação (classe do Cgt), essa é feita por meio do controlador (`CtrlAgenda`), sendo que este

conhece todas as operações necessárias para atender às requisições das interfaces. Após a aplicação ser requisitada, a aplicação retorna um resultado para o controlador que o interpreta e atualiza as interfaces de acordo com as necessidades.

Conforme mostra a Figura 5.18, a classe `CtrlAgenda` herda de `CtrlFluxoTarefas`, que é o controlador definido para controlar fluxos de tarefas. Conforme discutido na Seção 5.3, ele possui um componente de interface, `PainelFluxoTarefas`, que implementa vários métodos para auxiliar o agrupamento e a ordem de exibição de outros painéis em tela.

Apesar dos métodos não estarem explícitos no diagrama da Figura 5.18, o controlador possui vários métodos, cujas principais funcionalidades são de atualização das interfaces e de requisições às aplicações.

5.6 Implementação e Testes

A fase de implementação corresponde à codificação do projeto do sistema. As classes do modelo de projeto devem ser implementadas com o objetivo de produzir os arquivos fontes que originarão um arquivo executável.

Primeiramente foi construída a ferramenta de Alocação de Recursos, devido à ferramenta de Agenda depender dela. De acordo com o diagrama de classes definido no projeto, as classes pertencentes ao pacote da ferramenta foram implementadas e mapeadas, com o auxilio da ferramenta XDoclet (XDOCLET, 2007). Após a implementação das classes, a parte que trata de persistência dos objetos no banco de dados foi construída de maneira superficial, apenas com os métodos *salvar*, *excluir* e *obterTodos*. Em seguida, foram implementados os casos de uso apresentados na especificação de requisitos e ,paralelamente, a camada de persistência foi incrementada com novos métodos de acordo com as necessidades dos casos de uso.

Com os casos de uso implementados, a camada de interação humana foi desenvolvida com base no projeto de interface. Nessa fase foram criados janelas e painéis, com todos os componentes gráficos necessários para prover a interação, bem como toda a organização dos mesmos em tela.

Finalmente, para interligar a camada de interação humana com a de gerência de tarefas, na qual foram implementados os casos de uso, a camada de controle de interação foi construída com

apenas uma classe representado o controlador, provendo todas as funcionalidades necessárias para acoplar as mesmas.

Após a construção da ferramenta de Alocação de Recurso, a ferramenta de Agenda foi construída de forma similar, de acordo com as suas respectivas necessidades. Finalmente, as ferramentas foram integradas, não só entre elas, mas também com todo o ambiente ODE.

Para buscar assegurar a qualidade das ferramentas produzidas, fez-se necessária a realização da atividade de testes, com o objetivo de identificar erros gerados durante as diversas fases do processo de desenvolvimento. Para realizar tal tarefa, foram produzidos casos de teste com o intuito de avaliar diferentes aspectos de cada ferramenta.

Foram feitos testes para todos os eventos dos casos de usos apresentados na Especificação de Requisitos. Para realizá-los, foram gerados casos de teste para cada evento, de acordo com a descrição do evento. Com isso, os testes foram efetuados avaliando os elementos do domínio do problema, os elementos de interface e a execução dos eventos dos casos de uso.

Além de casos de testes específicos para cada evento dos casos de uso, as ferramentas foram testadas quanto à concorrência de transações e a persistência de dados. Quanto à concorrência, os testes foram feitos com duas instâncias abertas do ambiente ODE utilizando as ferramentas deste trabalho e outras relacionadas a ele. Sendo assim, as ferramentas eram testadas acessando os mesmos dados com o objetivo de encontrar e eliminar erros de concorrência. Quanto à persistência, os testes foram feitos com apenas uma instância do ambiente ODE, sendo que todos métodos definidos na camada de persistência foram testados, verificando o estado dos objetos no banco de dados quando necessário.

5.7 Apresentação das Ferramentas Desenvolvidas

Nesta seção são apresentadas as ferramentas desenvolvidas. Para isso, são mostrados os passos principais na utilização das mesmas.

5.7.1 – Ferramenta de Alocação de Recursos

A janela principal da ferramenta AlocaODE (*JanAlocacaoRecurso*), mostrada na Figura 5.19, permite definir uma equipe e alocar recursos em um projeto.

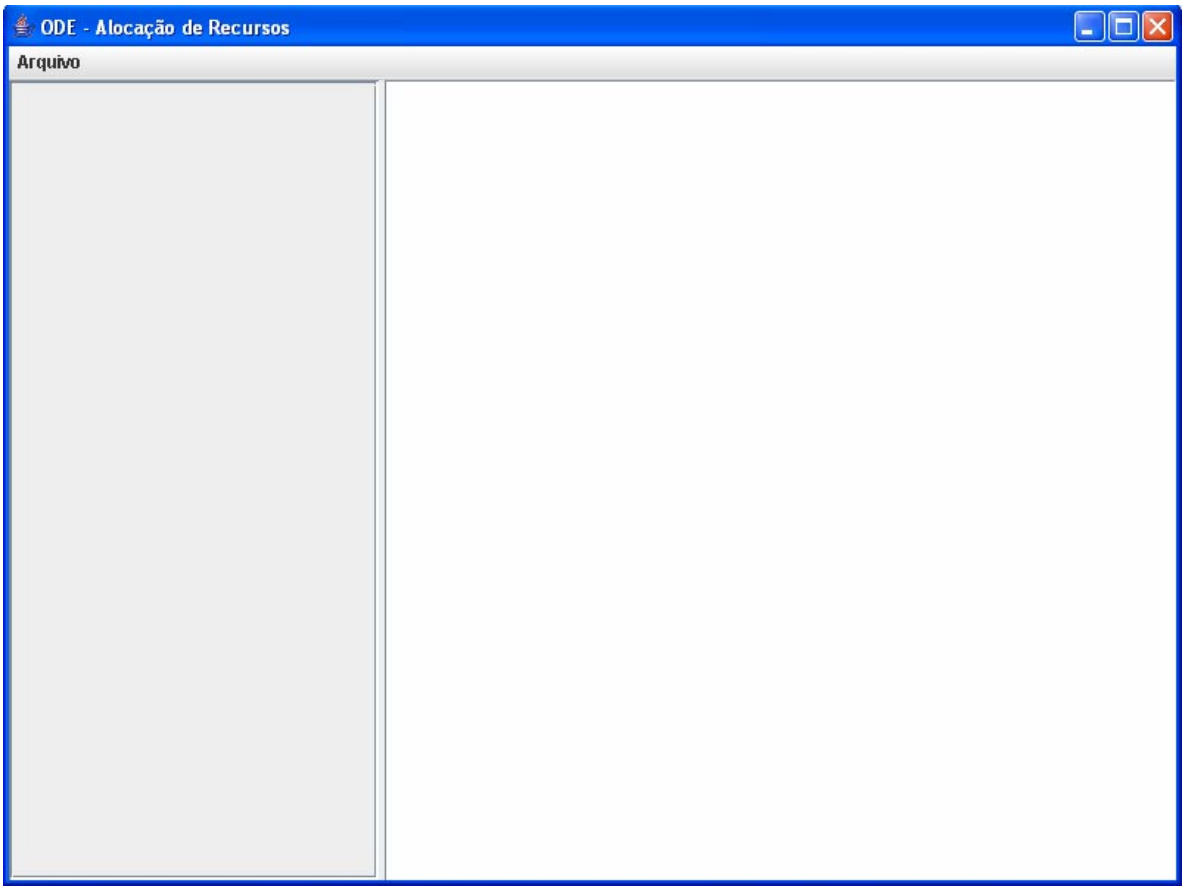

Figura 5.19 – Janela Principal da Ferramenta (*JanAlocacaoRecurso*)

Ao clicar no item de menu *Definir Equipe*, encontrado no menu *Arquivo*, a janela principal exibe no lado esquerdo uma árvore contendo os papéis necessários para a execução das atividades do projeto (Figura 5.20).

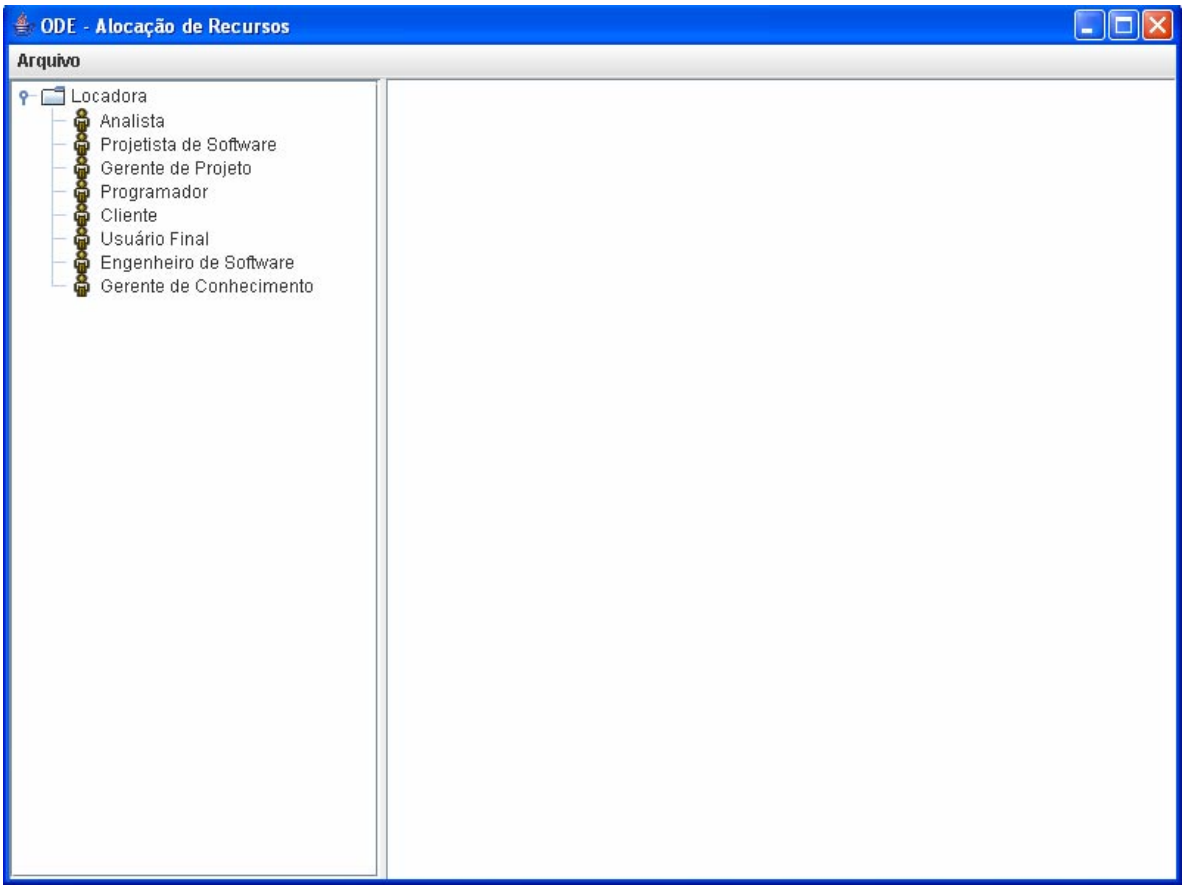

Figura 5.20 – Definir Equipe do Projeto

Ao selecionar um papel na árvore, a janela principal exibe no lado direito um painel (*PainelDefinirEquipe*) contendo uma tabela com todos os recursos humanos cadastrados no ambiente ODE que possuem especialidade para exercer o papel selecionado, para que a partir desses, faça-se uma seleção e, por fim, definam-se os recursos humanos que participarão da equipe exercendo o papel selecionado, como mostra a Figura 5.21. Sendo assim, para definir uma equipe, deve-se fazer esse procedimento para cada papel apresentado na árvore.

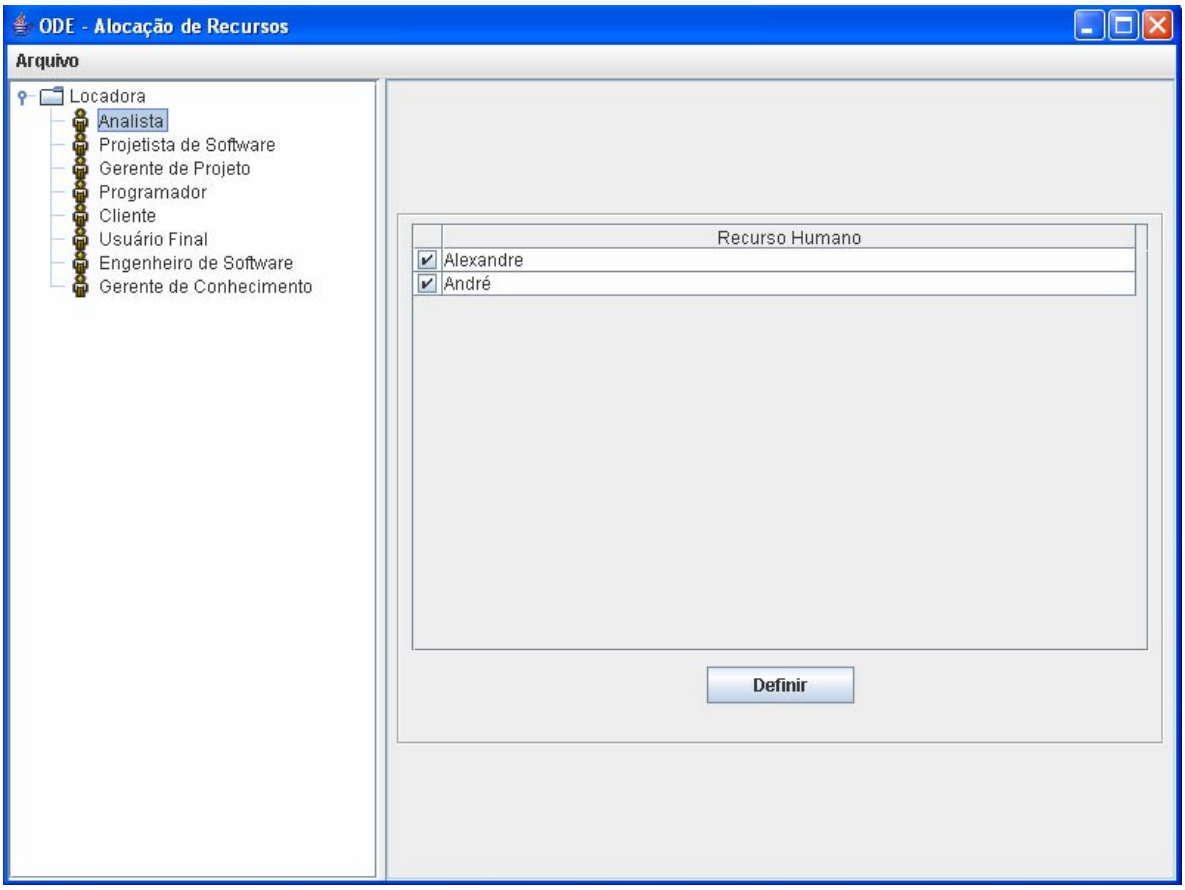

Figura 5.21 – Selecionar e Definir Recursos Humanos da Equipe

Além do item de menu *Definir Equipe*, há o item de menu *Alocar Recursos*, também no menu *Arquivo*, que permite definir as alocações de recursos humanos, de software e de hardware de um projeto. Ao clicar nesse item de menu, a janela principal exibe no seu lado esquerdo uma árvore contendo todas as atividades do projeto, com todos os tipos de recursos humanos, de software e de hardware requeridos por cada uma delas, bem como todas as alocações já definidas para estas (Figura 5.22).

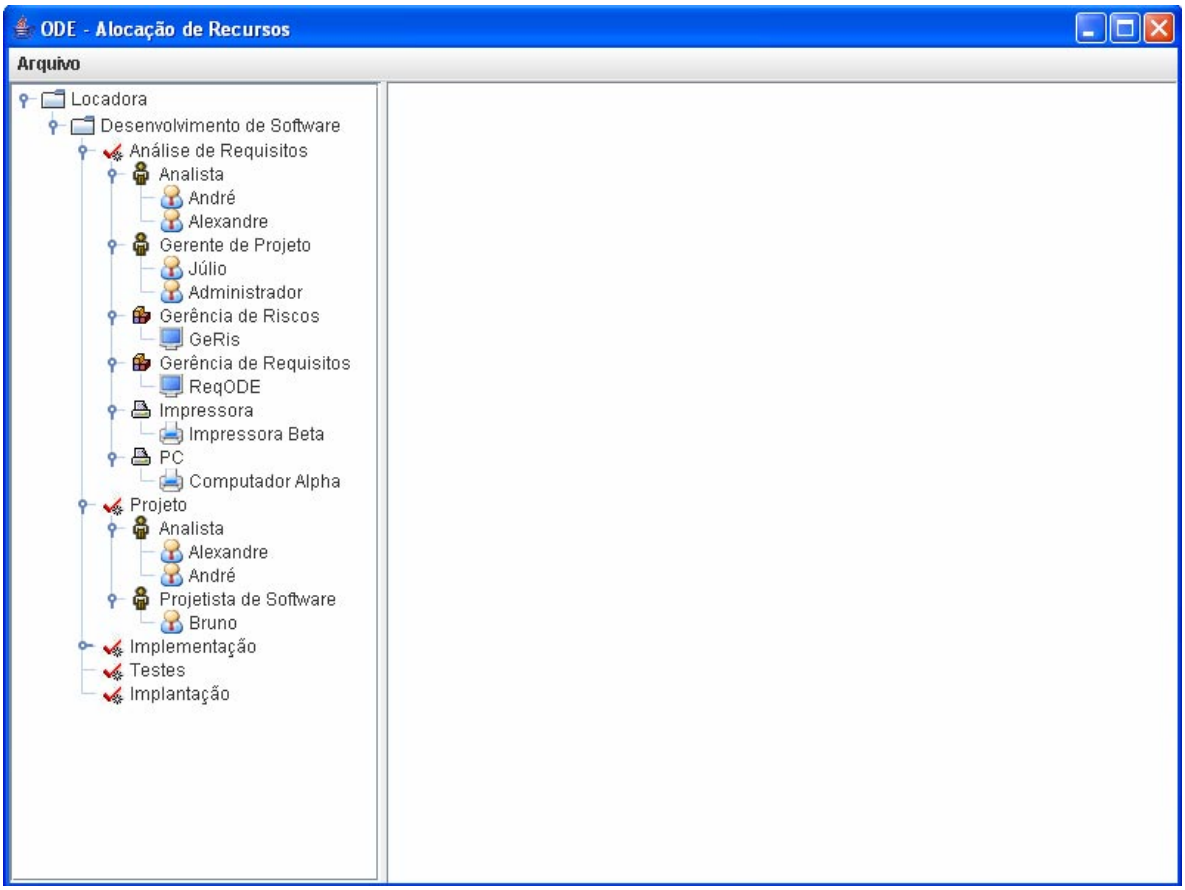

Figura 5.22 – Alocar Recursos

Ao selecionar qualquer tipo de recurso requerido por uma atividade, a janela principal exibe no seu lado direito o painel *PainelDefinirAlocacoesRecursos* que mostra uma tabela com todos os recursos do tipo selecionado, como mostra a Figura 5.23. Assim, é possível selecionar e definir alocações de acordo com cada tipo de recurso requerido pelas atividades. Além disso, se o recurso requerido for um recurso humano, só são exibidos no lado direito da janela principal os membros da equipe do projeto.

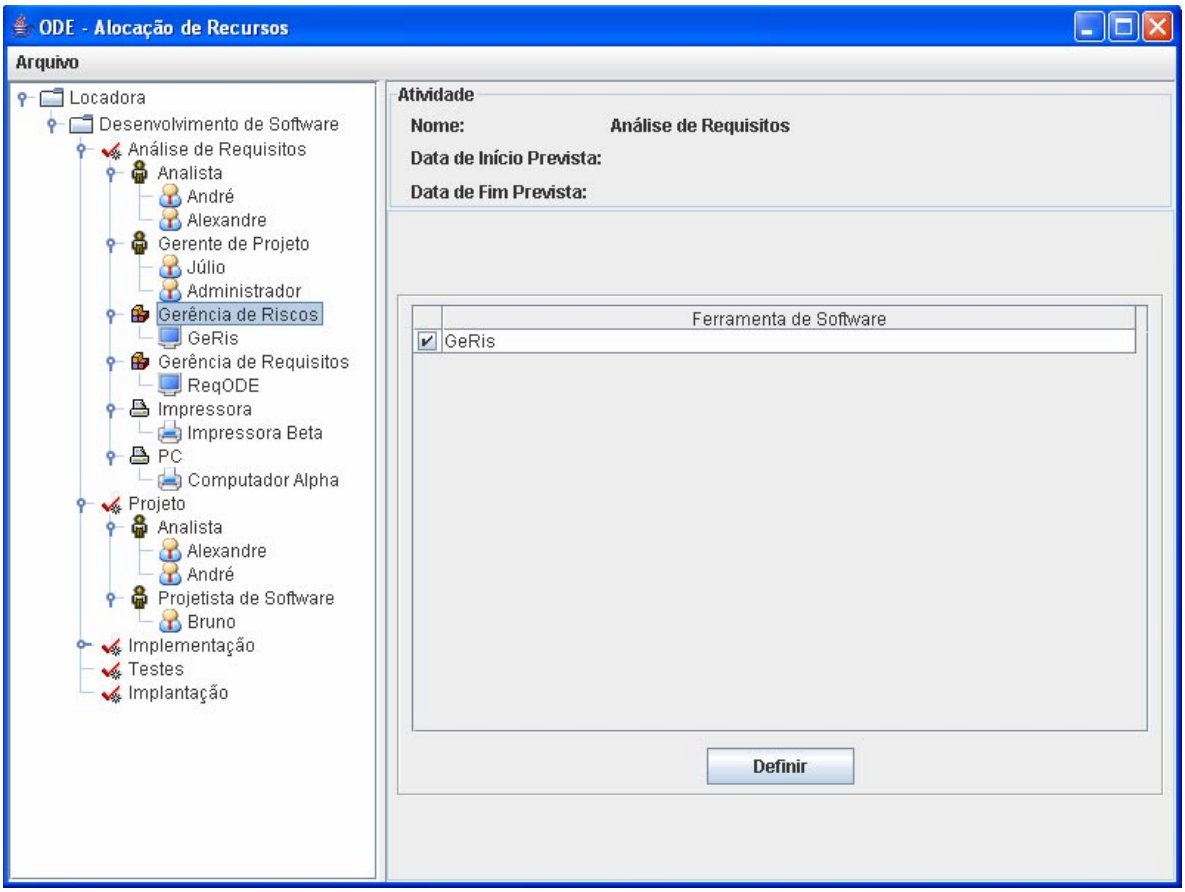

Figura 5.23 – Definir Alocações de Recursos a Atividades

Conforme apresentado na Figura 5.24, se uma alocação de recurso humano for selecionada na árvore, a janela principal exibe no seu lado direito um painel (*PainelEditarAlocacaoRecursoHumano*) com os dados referentes à alocação de recurso humano selecionada, para que esta possa ser editada e até mesmo controlada.

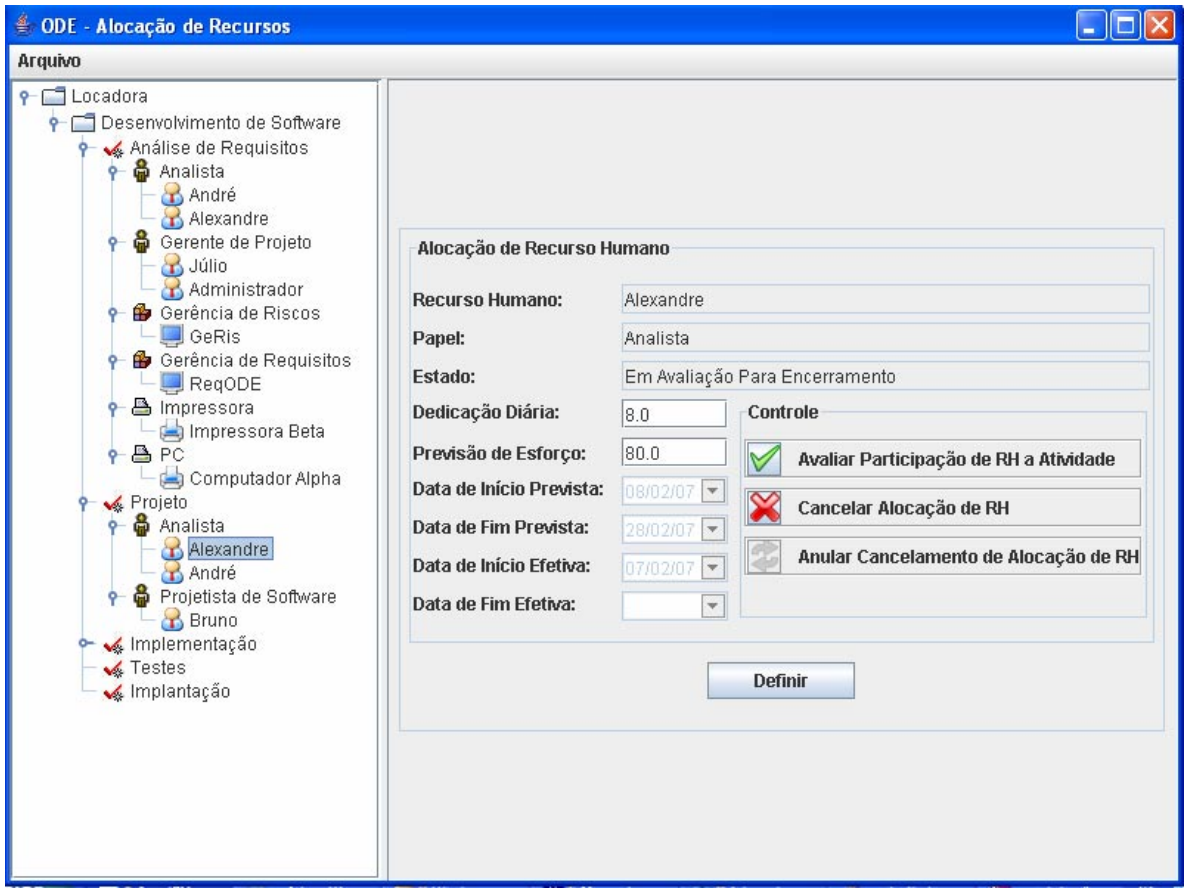

Figura 5.24 – Editar Alocações de Recursos Humanos a Atividades

Para controlar a alocação de recurso humano selecionada, é necessário acionar um dos seguintes botões: *Avaliar Participação de RH a Atividade*, *Cancelar Alocação de RH* e *Anular Cancelamento de Alocação de RH*. Caso o botão *Avaliar Participação de RH a Atividade* seja acionado, a janela *JanAvaliarAlocacaoRH* é exibida, conforme mostra a Figura 5.25. Caso o botão *Cancelar Alocação de RH* seja acionado, a janela *JanCancelarAlocacaoRH* é exibida (Figura 5.26). Por fim, se o botão *Anular Cancelamento de Alocação de RH* for acionado, a janela *JanAnularCancelamentoAlocacaoRH* é exibida (Figura 5.27).

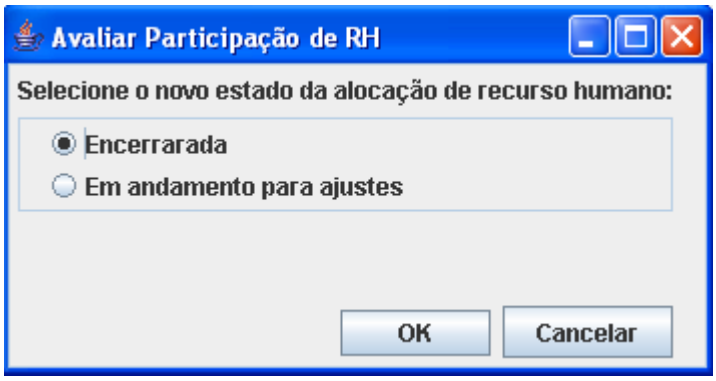

Figura 5.25 – Avaliar Participação de Recurso Humano

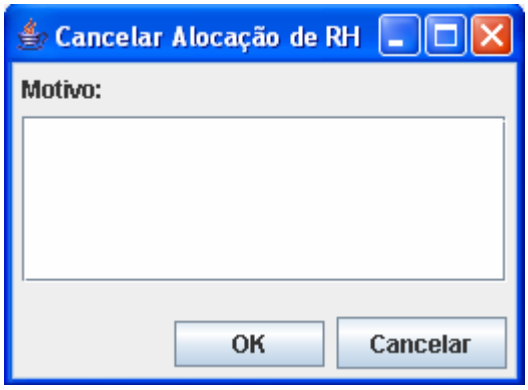

Figura 5.26 – Cancelar Alocação de Recurso Humano

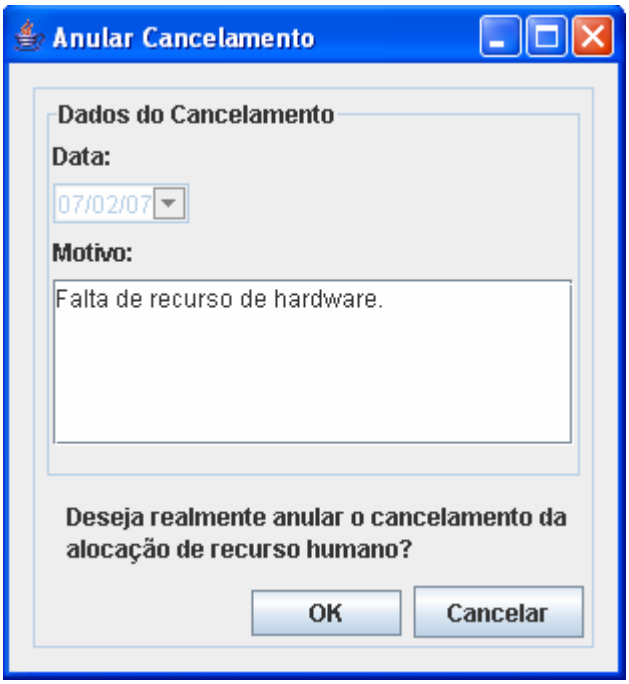

Figura 5.27 – Anular Cancelamento de Alocação de Recurso Humano

Vale a pena destacar que as árvores citadas na apresentação desta ferramenta são componentes gráficos auxiliares pertencentes à janela *JanAlocacaoRecurso*.

5.5.2 – Ferramenta de Agenda

A janela principal da ferramenta AgendaODE (*JanVisualizarAgenda*), mostrada na Figura 5.28, permite visualizar a agenda de um desenvolvedor. O lado esquerdo dessa janela contém um guia, contendo várias opções e, de acordo com a opção selecionada, o lado direito exibe o conteúdo relacionado a esta. Por padrão, a primeira opção do guia, *Informações Diárias*, é selecionada. Com isso, o painel *PainelInformacoesDiarias*, que contém as informações diárias do desenvolvedor, é exibido no lado direito da janela principal. A partir desse painel é possível visualizar uma alocação do recurso humano a uma atividade, consultar um compromisso e até mesmo consultar uma informação, dando apenas um clique duplo no item desejado.

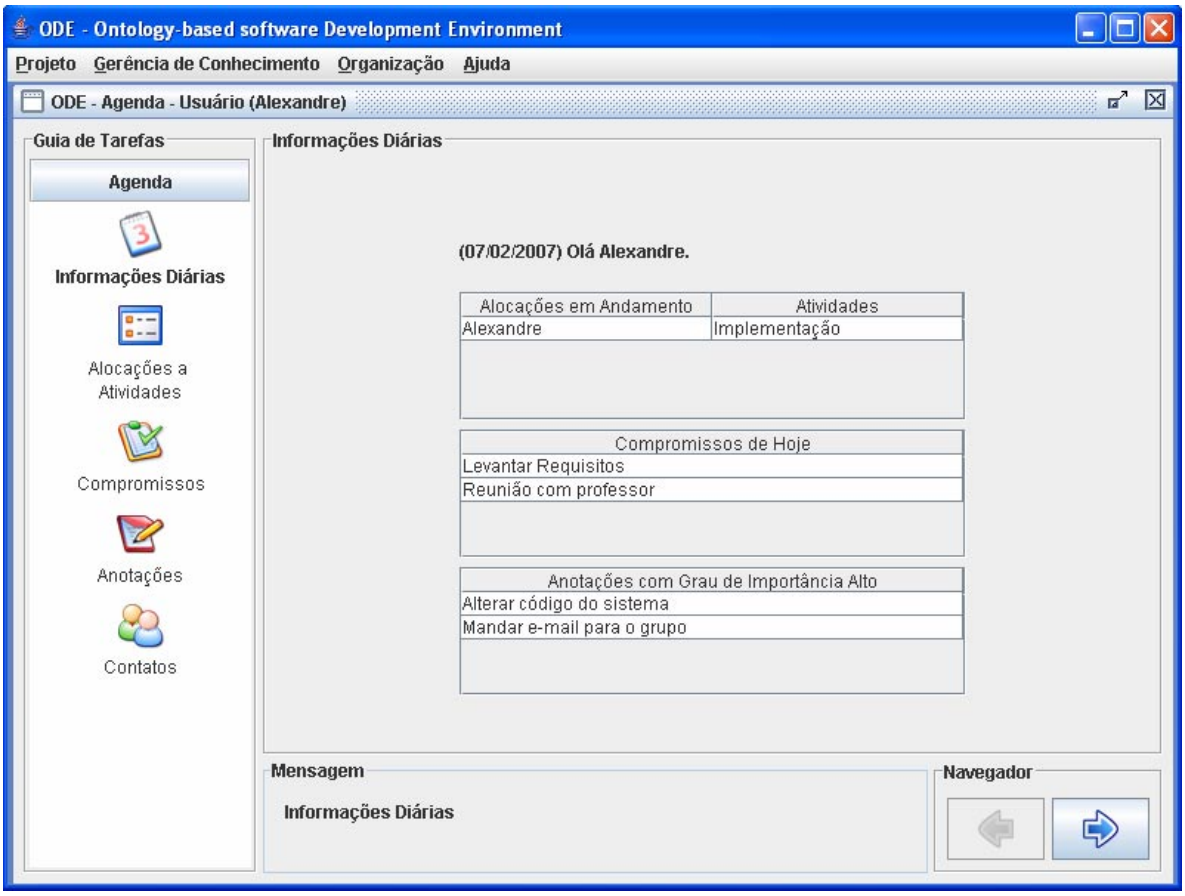

Figura 5.28 – Visualizar Informações Diárias

Caso a opção Visualizar Alocações seja selecionada, é exibido no lado direito da janela principal o painel *PainelVisualizarAlocacoesRHAtividades*. Esse painel apresenta as alocações do desenvolvedor ativo no ambiente, permitindo que este selecione uma para visualizar as informações (Figura 5.29).

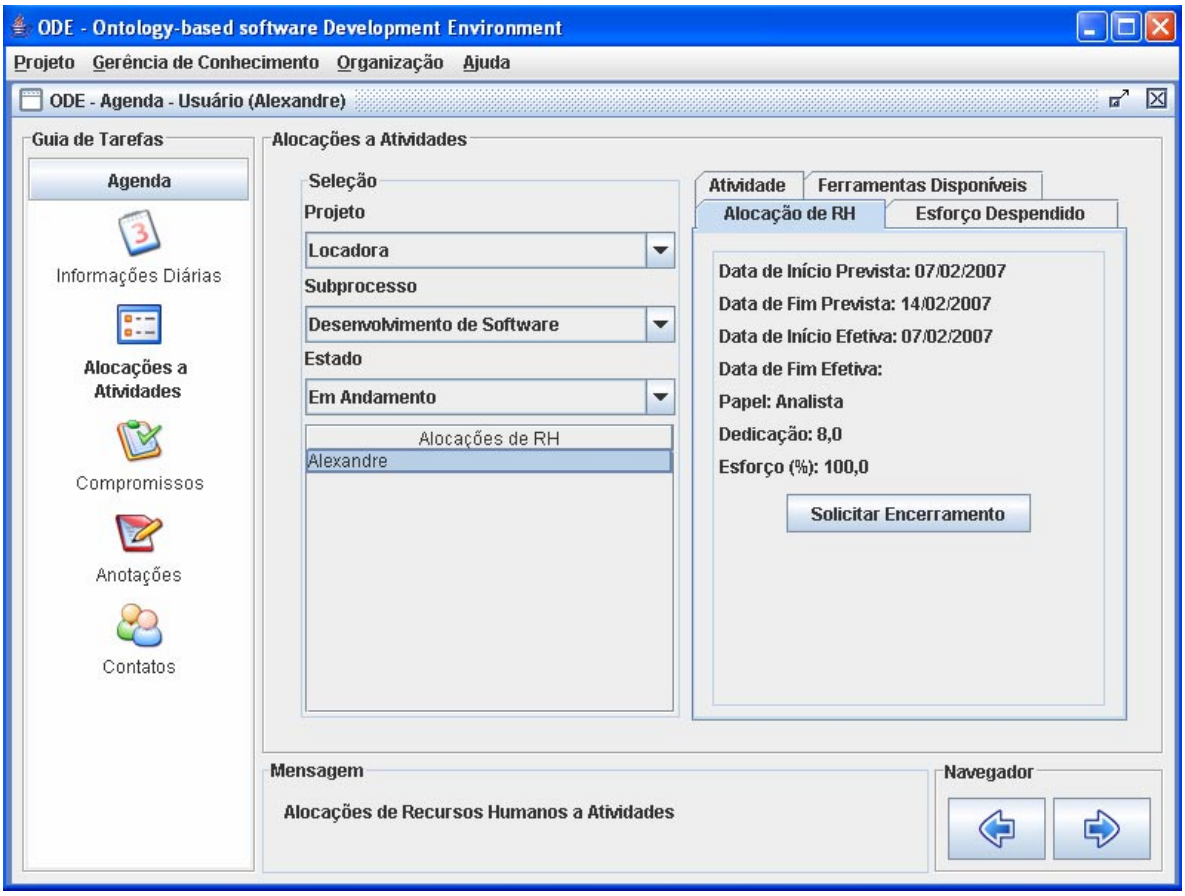

Figura 5.29 – Visualizar Alocações de Recursos Humanos a Atividades

A tabela *Alocações de RH* exibe as alocações do recurso humano de acordo com o projeto, o subprocesso e o estado da alocação selecionados. Nessa tabela é possível selecionar uma alocação de recurso humano para que os dados da mesma sejam exibidos no conjunto de abas ao lado. A aba *Alocação RH* contém informações da alocação de recurso humano selecionada na tabela e um botão (*Solicitar Encerramento*) que permite solicitar o encerramento da participação do desenvolvedor na atividade, conforme mostra a Figura 5.29.

A aba *Atividade* exibe dados da atividade da alocação de recurso humano selecionada, bem como os recursos humanos alocados nela (Figura 5.30). Já a aba *Esforço Despendido* exibe uma tabela com os esforços despendidos na alocação selecionada, sendo possível incluir, alterar, consultar e excluir um esforço despendido (Figura 5.31). Por fim, como mostra a Figura 5.32, a aba *Ferramentas Disponíveis* permite o acesso às ferramentas de software disponíveis para o trabalho.

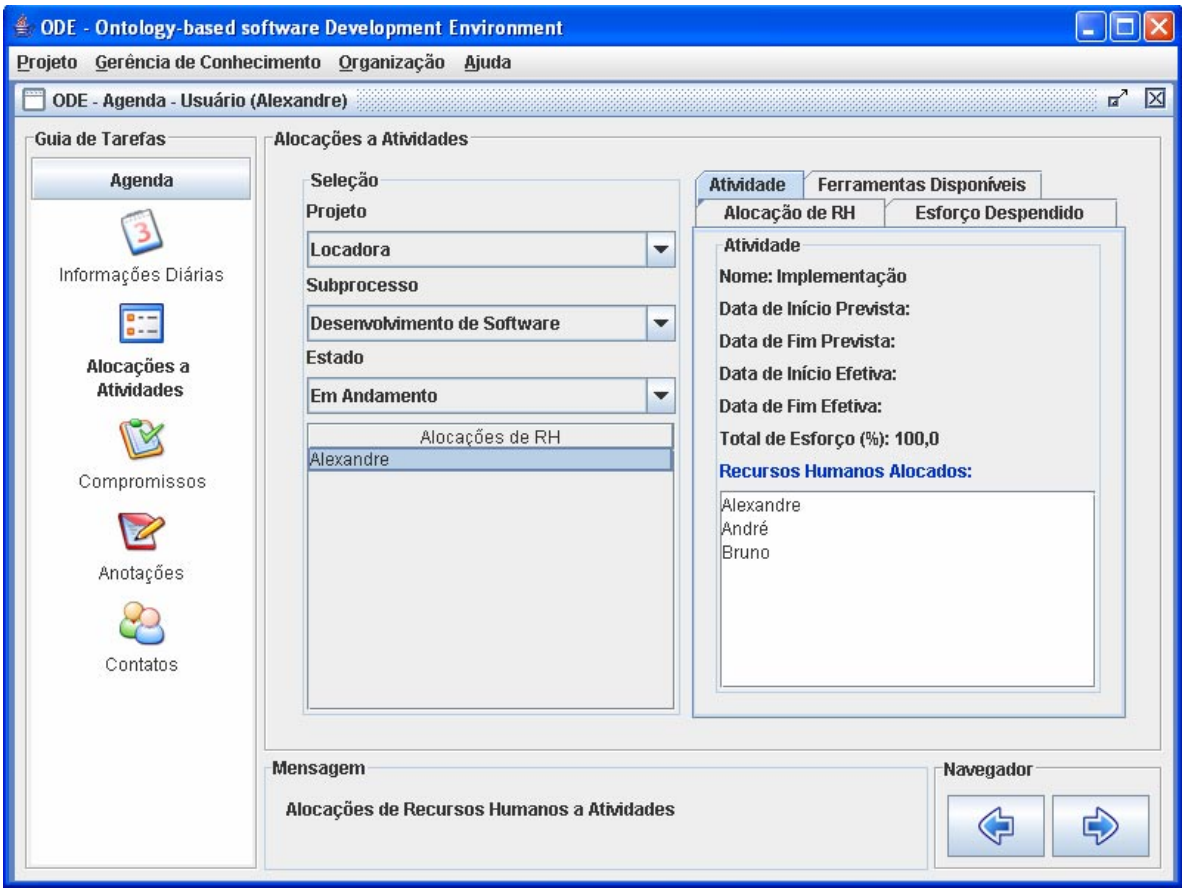

Figura 5.30 – Visualizar Informações da Atividade

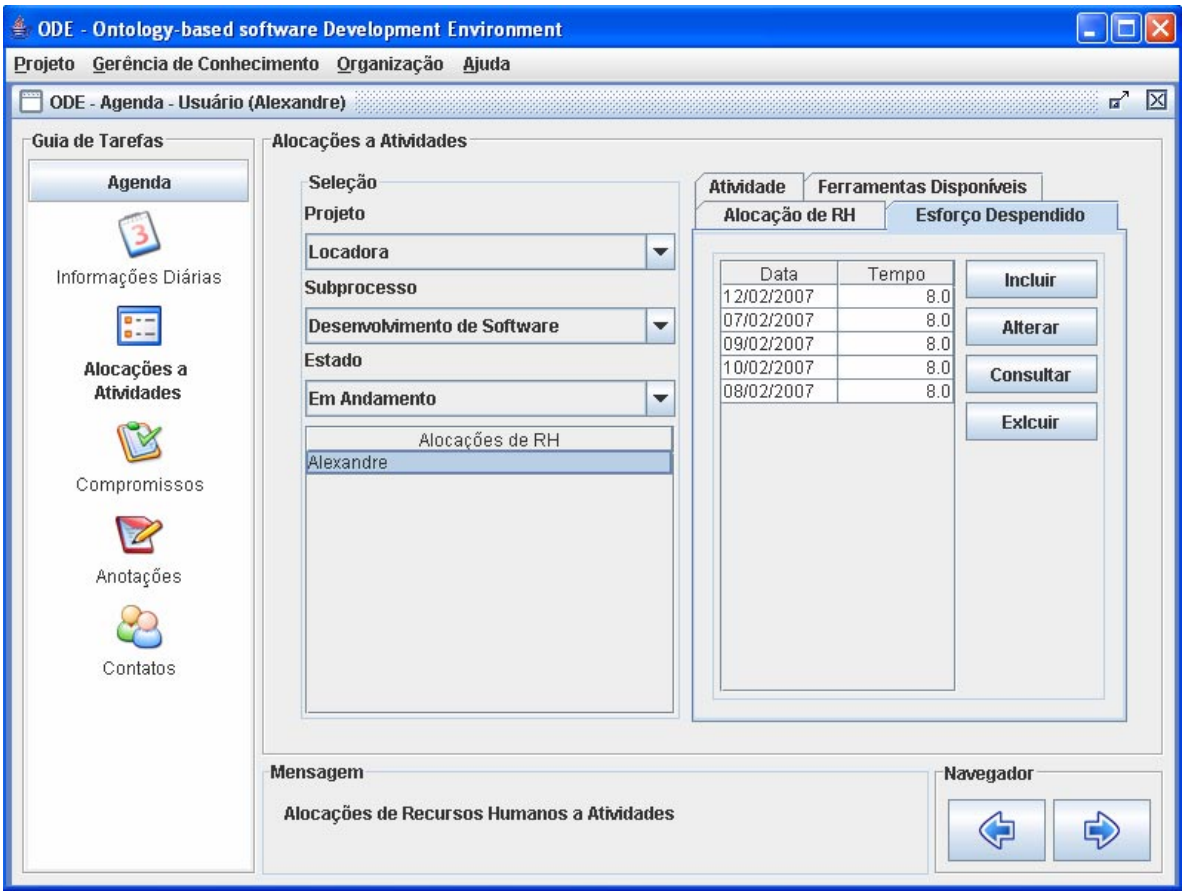

Figura 5.31 – Registrar Esforço Despendido

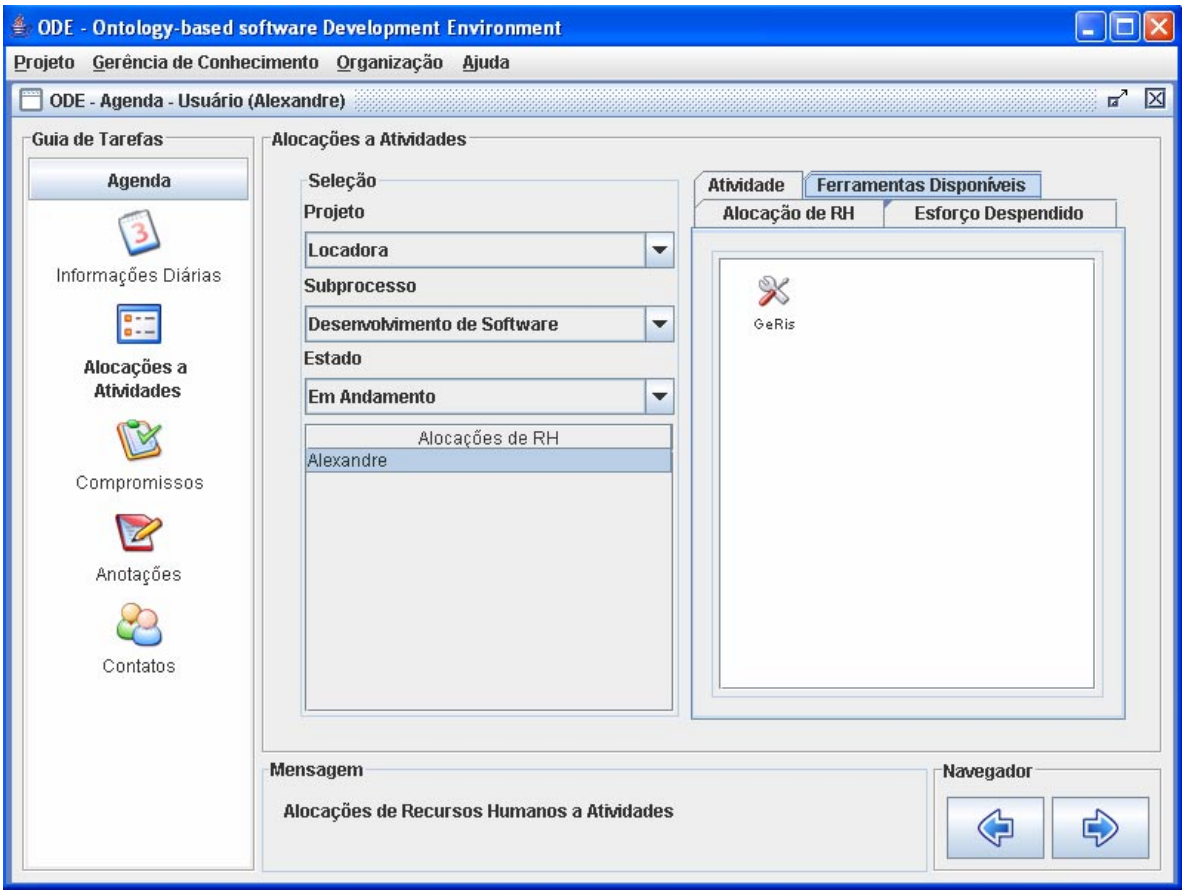

Figura 5.32 – Selecionar Ferramenta de Trabalho

Caso a opção *Compromissos* seja selecionada, o painel *PainelControlarComprimisso* é exibido no lado direito da janela principal, contendo todos os compromissos do desenvolvedor ativo (Figura 5.33).

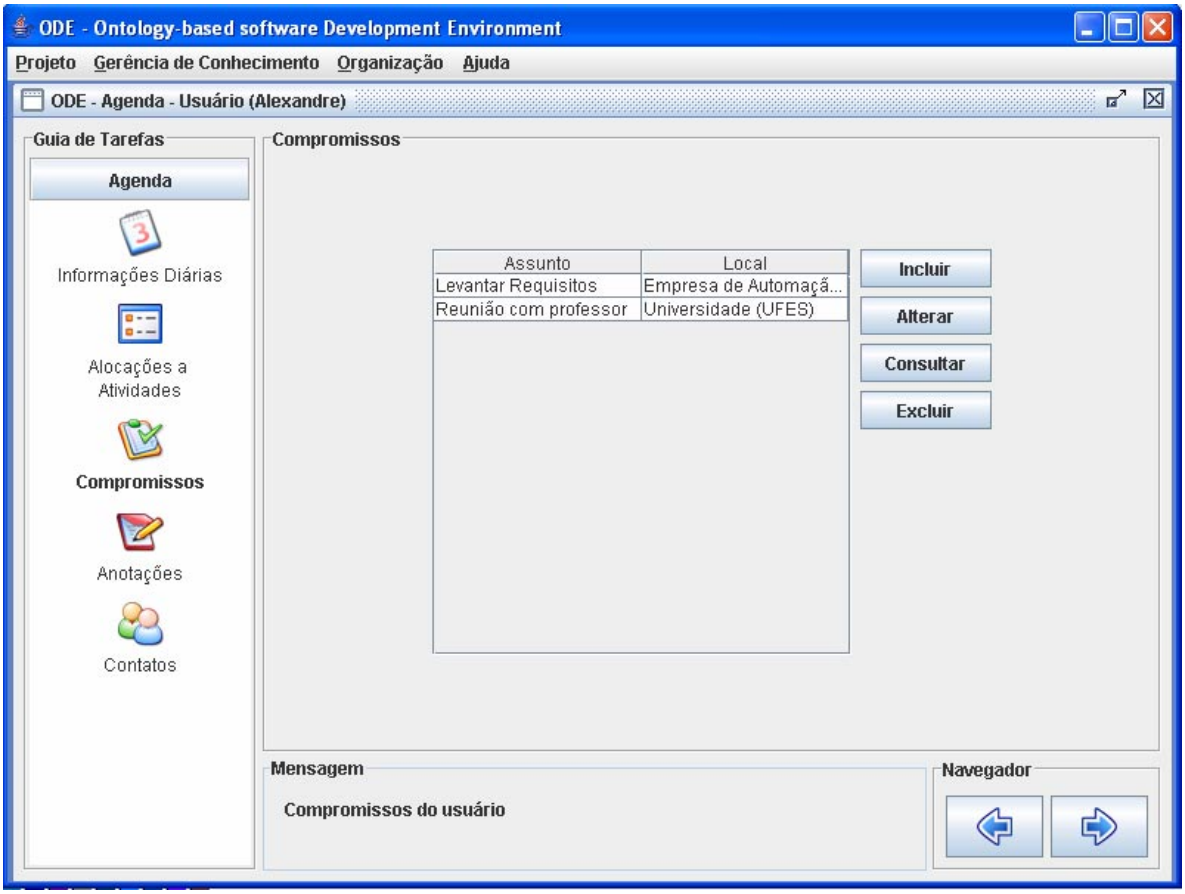

Figura 5.33 – Controlar Compromisso

Caso a opção *Anotações* seja selecionada, o painel *PainelControlarAnotacao* é exibido contendo todos as anotações do desenvolvedor (Figura 5.34). Esse painel foi construído com objetivo de tornar o uso o mais próximo do mundo real. Quando se fala em anotações, logo se pensa em um quadro com vários bilhetes nele pregados. A idéia desse painel é justamente essa. O painel branco representa o quadro e uma anotação é representada por um ícone contendo o assunto da mesma. Cada cor de uma anotação nesse painel representa um grau de importância. Quanto mais próximo de vermelho, maior o grau é o grau de importância. Para incluir uma anotação, basta dar um clique duplo no painel branco. Para alterar, consultar ou excluir uma anotação, basta clicar na mesma.

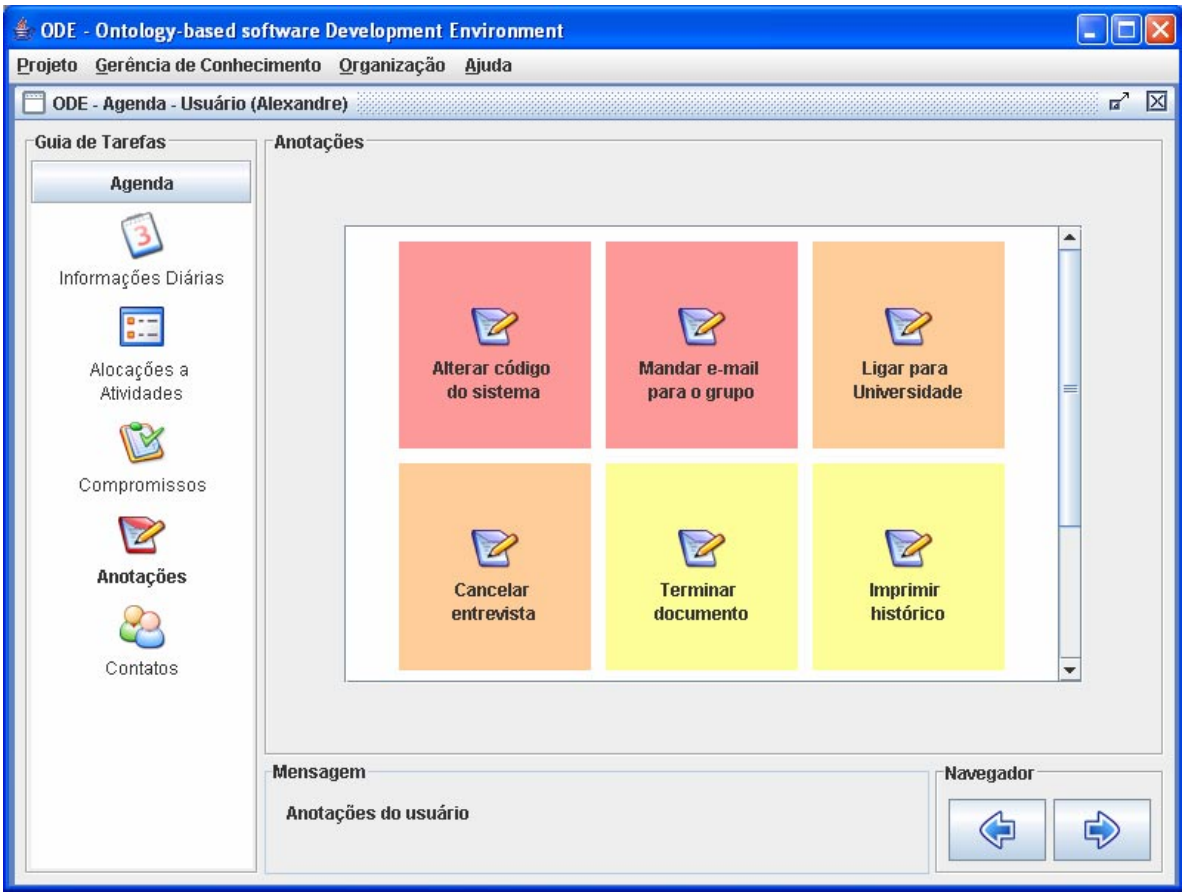

Figura 5.34 – Controlar Anotação

Caso a opção *Contatos* seja selecionada, o painel *PainelCadastrarContato* é exibido mostrando os contatos do desenvolvedor (Figura 5.35).

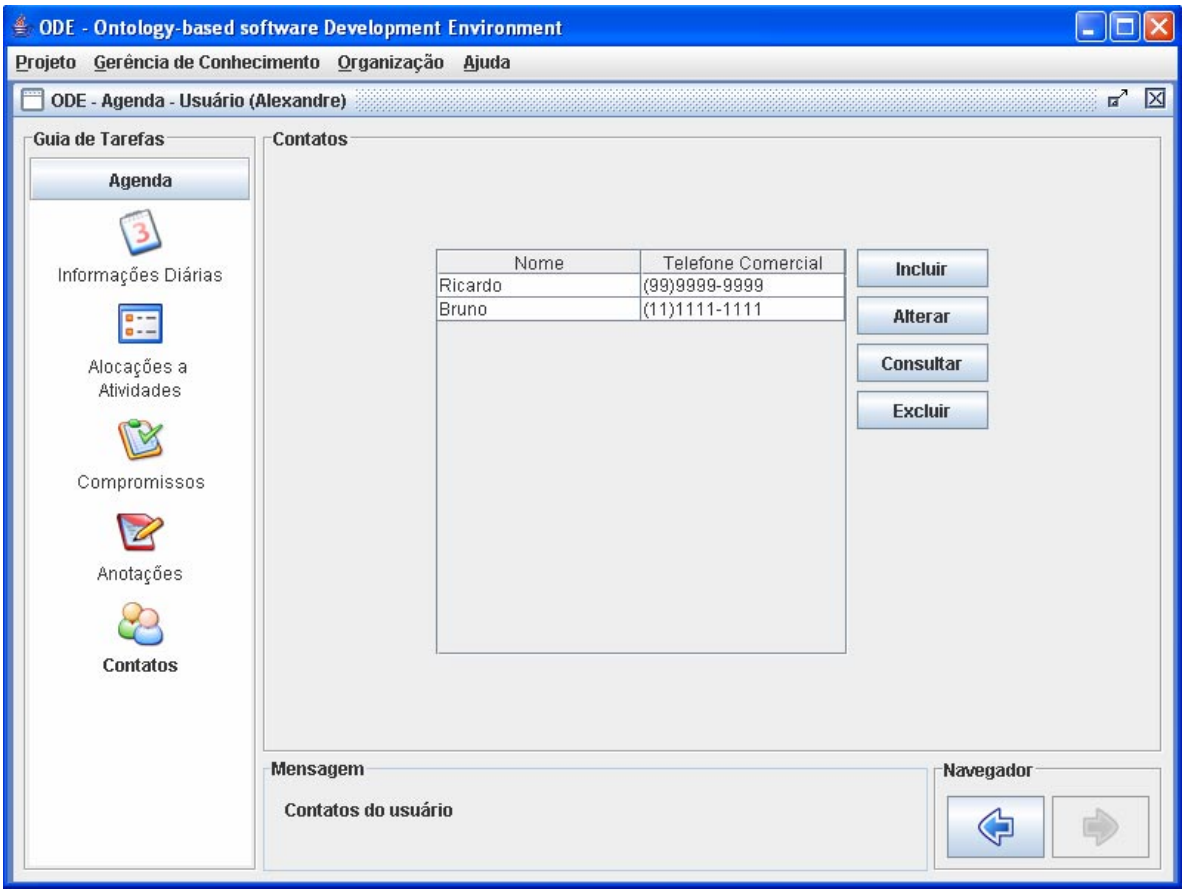

Figura 5.35 – Cadastrar Contato

Vale a pena destacar que tanto no painel *PainelControlarComprimisso* quanto no painel *PainelCadastrarContato* é possível realizar operações de incluir, alterar, consultar e excluir.

Capítulo 6

Conclusões e Perspectivas Futuras

Este capítulo apresenta as conclusões finais e perspectivas futuras referentes a este trabalho. A Seção 6.1 trata das conclusões finais do trabalho. Já a Seção 6.2 apresenta algumas oportunidades de trabalhos futuros, sejam eles evoluções das ferramentas ou até mesmo novos trabalhos que podem ser feitos dentro do Projeto ODE, sendo todas essas melhorias observadas ao longo do desenvolvimento deste trabalho.

6.1 Conclusões

A automatização das atividades relacionadas à gerência de projetos proporciona um maior aumento da produtividade e até mesmo da qualidade dos produtos de software desenvolvidos em uma organização.

Com o objetivo de melhorar o apoio automatizado provido por ODE à gerência de projetos, neste trabalho foram desenvolvidas ou aperfeiçoadas algumas ferramentas, a saber:

- Ferramenta de Apoio à Alocação de Recursos: a evolução dessa ferramenta passou a permitir a definição de uma equipe de um projeto de acordo com as especialidades requeridas pelas atividades do mesmo. Além disso, é possível definir detalhadamente os recursos humanos a serem alocados a cada atividade, acompanhar o andamento de cada alocação de recurso humano e alocar ferramentas de software, bem como recursos hardware, de acordo com os tipos recursos requeridos pelas atividades do projeto.
- Ferramenta Agenda: tem como objetivo orientar o desenvolvedor dentro do ambiente ODE. Nessa evolução foram adicionadas funcionalidades básicas de uma agenda, permitindo que o desenvolvedor controle compromissos, anotações e contatos. Além disso, agora é possível

que o desenvolvedor visualize suas alocações, podendo, a partir dessas, registrar esforços despendidos, selecionar ferramentas de software disponíveis para trabalho e até mesmo solicitar o encerramento de sua participação em alguma atividade.

Este trabalho permitiu o aprendizado de vários assuntos, dentre eles: (i) linguagem de programação Java; (ii) utilização de camadas de persistência utilizando o *framework* Hibernate; (iii) gerenciamento de projetos, com ênfase na parte que trata de alocação de recursos e (iv) utilização de ferramentas para o desenvolvimento deste trabalho, entre elas sistemas de gerenciamento de banco de dados e ferramentas de modelagem baseadas na notação UML.

Além desses aprendizados, muitos conceitos foram aplicados durante o desenvolvimento deste trabalho, dentre eles especificação de requisitos, análise e projetos de sistemas, envolvendo a documentação dos mesmos, implementação e testes de sistemas. Assim, o desenvolvimento deste trabalho foi uma excelente oportunidade para aplicação dos conhecimentos adquiridos durante o curso de Ciência da Computação com a ênfase em Sistemas de Informação.

6.2 Perspectivas Futuras

Quando se trata de desenvolvimento de software, é muito comum e até mesmo necessário que a finalização de um trabalho implique na construção e manutenção de outros, fazendo com que um software ou um conjunto deles cresça visando atender melhor as necessidades dos clientes.

Apesar das ferramentas desenvolvidas neste trabalho estarem em um nível de funcionalidade e utilização bem aceitável, novas oportunidades foram percebidas, com o objetivo de evoluir e proporcionar outras melhorias nessas ferramentas. A seguir, são apresentadas algumas propostas de trabalhos futuros:

- Na Ferramenta de Alocação de Recursos, cada recurso poderia ser mais bem acompanhado. Poderia haver um maior controle sobre a disponibilidade dos recursos, no qual o sistema não permitisse que um intervalo de um recurso

coincidisse com outro intervalo do mesmo recurso, inclusive considerando diversos projetos;

- Como a alocação de recursos não envolve somente recursos, mas também tempo e custo, ferramentas de cronograma e gerenciamento de custos seriam importantes e se integrariam muito bem com a ferramenta de Alocação de Recursos.
- Na ferramenta de Agenda, novas funcionalidades podem ser adicionadas, como o gerenciamento da caixa postal do desenvolvedor.
- Não só nas ferramentas desenvolvidas nesse trabalho, mas em várias outras ferramentas do ambiente ODE, poderiam ser desenvolvidas versões para serem utilizadas via *web*.

Referências Bibliográficas

BAUER, C., KING, G. Hibernate em Ação. Tradução 1ª edição. Ciência Moderna, 2005.

BERTOLLO, G., SEGRINI, B., FALBO, R. A. Definição de Processos de Software em um Ambiente de Desenvolvimento de Software Baseado em Ontologias. In: V Simpósio Brasileiro de Qualidade de Software, 2006, Vila Velha, Brasil. Anais...2006. p. 72-86.

BOOCH, G., RUMBAUGH, J, JACOBSON, I., UML: Guia do Usuário. 2. ed., Editora Campus, 2005.

CARVALHO, V. A., FALBO, R. A., OLIVEIRA, L. O., "EstimaODE: Apoio a Estimativas de Tamanho e Esforço no Ambiente de Desenvolvimento ODE", Anais do V Simpósio Brasileiro de Qualidade de Software, Vitória, Brasil, pp. 12-26, 2006.

DAL MORO, R., NARDI, J.C., FALBO, R. A., "ControlPro: Uma Ferramenta de Acompanhamento de Projetos Integrada a um Ambiente de Desenvolvimento de Software", Anais da XII Sessão de Ferramentas do Simpósio Brasileiro de Engenharia de Software, SBES'2005, Uberlândia, Brasil, Outubro 2005.

FALBO, R.A., RUY, F.B., MORO, R.D., "Using Ontologies to Add Semantics to a Software Engineering Environment". 17th International Conference on Software Engineering and Knowledge Engineering, SEKE'2005, p. 151 - 156, Taipei, China, July 2005.

FALBO, R.A., RUY, F.B., PEZZIN, J., DAL MORO R., "Ontologias e Ambientes de Desenvolvimento de Software Semânticos". Actas de las IV Jornadas Iberoamericanas de Ingeniería del Software e Ingeniería Del Conocimiento, JIISIC'2004, Volumen I, pp. 277-292, Madrid, España, Noviembre 2004a.

FALBO, R.A., RUY, F.B., BERTOLLO, G., TOGNERI, D.F., "*Learning How to Manage Risks Using Organizational Knowledge*". Proceedings of the 6th International Workshop on Advances in Learning Software Organizations, LSO'2004, pp. 7-18, Banff, Canada, June 2004b.

FALBO, R. A., NATALI, A. C. C., MIAN, P. G., BERTOLLO, G., RUY, F. B. ODE: Ontology-based software Development Environment. In: IX Argentine Congress on Computer Science, 2003, La Plata, Argentina. Proceedings…2003. p. 1124-1135.

FALBO, R.A., Engenharia de Software: Notas de Aula 2005**,** Documento Eletrônico disponível em <http://www.inf.ufes.br/~falbo/download/aulas/es-g/2005-2/NotasDeAula.pdf>. Acesso em 12/02/2007.

FUGGETTA, A., Software Process: A Roadmap, Proc. of The Future of Software Engineering, ICSE ’2000 , Limerick, Irlanda, 2000.

GAMMA, E., HELM R., JOHNSON R., VLISSIDES, J., *Design Patterns - Elements of Reusable Object-oriented Software***.** Addison-Wesley Professional Computing Series, 1995.

MARTINS, A. F., NARDI, J. C., FALBO, R. A. ReqODE: Uma Ferramenta de Apoio à Engenharia de Requisitos Integrada ao Ambiente ODE. In: XX Simpósio Brasileiro de Engenharia de Software, 2006, Florianópolis, Brasil. Anais da XIII Sessão de Ferramentas do SBES 2006, 2006. p. 31-36.

MARTINS, J.C.C., *Gerenciando projetos de desenvolvimento de software com PMI, RUP e UML*. 2a edição, Rio de Janeiro: Brasport, 2005.

MEYER, B., *Object-Oriented Software Construction*, Second Edition, Prentice Hall, 1997.

PEZZIN, J., Apoio à Visualização do Processo de Software no Contexto de um Ambiente de Desenvolvimento de Software. Monografia (Graduação em Ciência da Computação) Universidade Federal do Espírito Santo, 2002.

PFLEEGER, S.L. Engenharia de Software: teoria e prática. Tradução Dino Franklin. 2. ed. São Paulo: Prentice Hall, 2004.

PMI, PMBOK: Um Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos. 3ª edição. 2004.

PRESSMAN, R. S., *Engenharia de Software*. 5ª edição. Rio de Janeiro: McGrawHill, 2002.

SCHWAMBACH, M.M., Apoio Automatizado ao Planejamento e Acompanhamento de Projetos de Software. Monografia (Graduação em Ciência da Computação) Universidade Federal do Espírito Santo, 2002.

SOMMERVILLE, I., Engenharia de Software. Tradução André Maurício de Andrade Ribeiro. 6. ed. São Paulo: Addison Wesley, 2003.

SUN Developer Network, "*Data Access Object*", 2001. Disponível em: <java.sun.com/j2ee/patterns/DataAccessObject.html >. Acesso em: 12/02/2007.

Anexo A

Documentação da Evolução da Ferramenta de Apoio a Alocação de Recursos

Este documento tem como objetivo apresentar informações sobre a versão atual da ferramenta de Apoio a Alocação de Recursos, abrangendo todos as fases do ciclo de vida de desenvolvimento.

Inicialmente é apresentada a especificação de requisitos da ferramenta (Seção A.1), detalhando apenas os requisitos funcionais, já que os requisitos não-funcionais foram tratados na Seção 4.3. Em seguida, a Seção A.2 define a fase de análise da ferramenta, apresentando uma visão do domínio do problema, desconsiderando aspectos tecnológicos. Finalmente a Seção A.3 discute o projeto dessa versão da ferramenta.

A.1 Especificação de Requisitos

Esta seção apresenta a especificação de requisitos para a construção da ferramenta de agenda do ambiente ODE. A Seção B.1.1 contém uma breve descrição do sistema (descrição do mini-mundo). Em seguida, na Seção B.1.2, são apresentados os diagramas de casos de uso associados às descrições de cada caso de uso.

A.1.1 Descrição do Mini-Mundo

Dentro de uma organização, pessoas são responsáveis por realizar atividades, geralmente com o apoio de algum outro tipo de recurso, tal como uma ferramenta de software ou um equipamento de hardware, que auxilie na realização de tal atividade.

No ambiente ODE, para cada atividade é necessário alocar pessoas para trabalhar na mesma, bem como ferramentas de software que estarão disponíveis para a realização do trabalho, além de outros recursos como hardware e sistemas de apoio.

Para ter um maior controle sobre as alocações de recursos humanos a atividades dentro do ambiente, é necessário saber o papel desempenhado pelo recurso, a dedicação em termos de tempo diário do recurso humano alocado à realização da atividade, o esforço total previsto para a alocação, as datas de início e fim previstas e efetivas da alocação, além do estado da mesma.

A.1.2 Modelo de Casos de Uso

A Figura A.1 mostra o diagrama de pacotes da ferramenta de apoio à Alocação de Recursos no que se refere às funcionalidades providas.

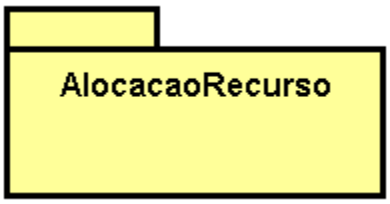

Figura A.1 - Diagrama de Pacotes

O pacote *AlocacaoRecurso* trata das funcionalidades que o gerente de projeto pode encontrar na ferramenta de alocação, como definir equipe de projeto, alocar recursos humanos, de software e de hardware, e controlar alocações de recursos humanos. Além disso, há funcionalidades relacionadas à alocação de recursos, mas que são disponibilizadas em outras ferramentas do ambiente ODE, tais como as que permitem registrar esforço despendido em uma alocação e solicitar o encerramento de uma participação em uma atividade.

A Figura A.2 mostra o diagrama de casos de uso do pacote *AlocacaoRecurso*. Para cada atividade definida em um processo de um projeto, é possível alocar recursos humanos, de software ou de hardware. Antes de alocar recursos humanos, é necessário que o gerente de projeto defina uma equipe para trabalhar no projeto.

Para ter um maior controle das atividades, o gerente de projeto pode controlar o andamento das alocações de recursos humanos a atividades, cabendo a ele confirmar o encerramento da mesma.

Por meio das ferramentas de Agenda e de Acompanhamento de Projeto do ambiente ODE, o desenvolvedor pode registrar esforços despendidos em uma alocação e até mesmo solicitar o encerramento de sua participação em uma atividade.

A seguir, os casos de uso propostos são descritos.

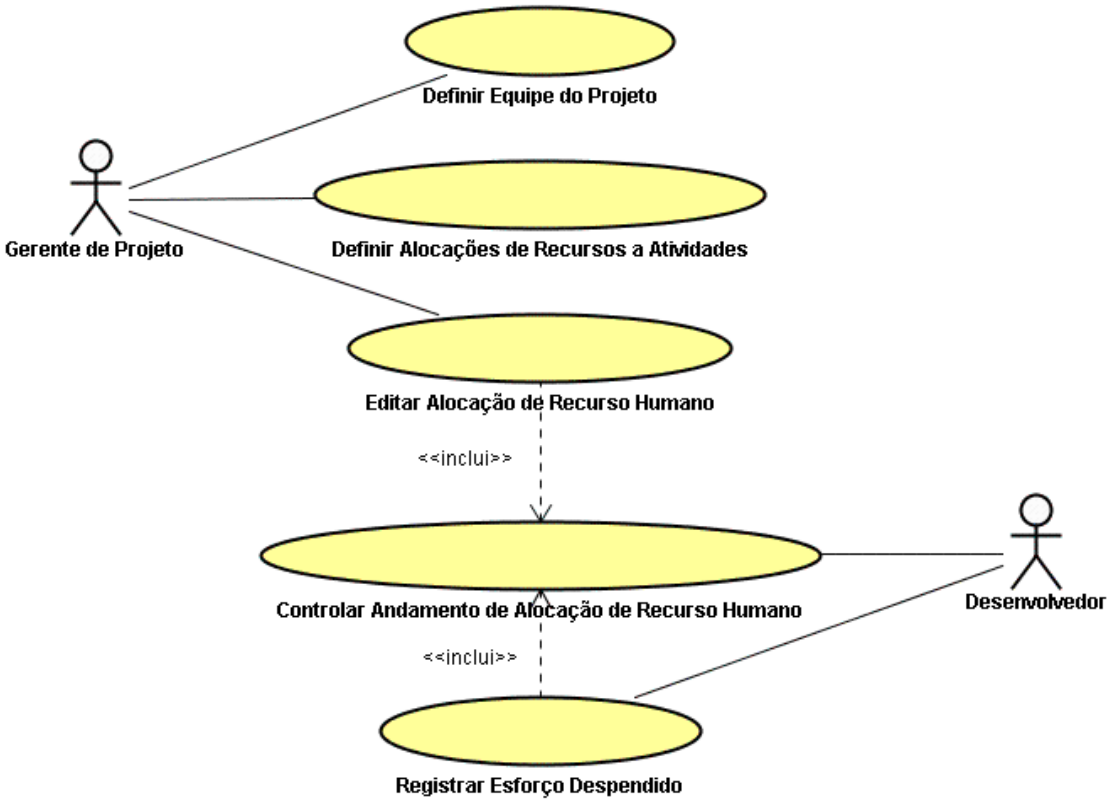

Figura A.2 - Diagrama de Casos de Uso do pacote *AlocacaoRecurso*.

Projeto: AlocaODE – Ferramenta de Apoio à Alocação de Recursos de ODE
Caso de Uso: Definir Equipe do Projeto
Descrição: Este caso de uso permite que o gerente de projeto defina quais recursos humanos participarão da equipe de um projeto.

Pré-Condição 1: O gerente de projeto deve ter aberto um projeto.
Pré-Condição 2: Os recursos humanos que participarão da equipe do projeto devem estar ativos no ambiente ODE.

Curso Normal:

O gerente de projeto seleciona os recursos humanos que vão compor a equipe do projeto, indicando o papel que cada um deles vai desempenhar no projeto, de acordo com as especialidades requeridas pelas atividades do mesmo. Caso os dados sejam válidos, a composição da equipe é registrada.

Curso Alternativo:

Remoção de recursos humanos que estão alocados a alguma atividade: Uma mensagem de erro é exibida, informando que não é possível remover recursos humanos que estão alocados a alguma atividade do projeto.

Classes: RecursoHumano, KRecursoHumano, Atividade, ProcessoProjetoGeral, ProcessosProjetoEspecifico, Projeto, Equipe e AlocacaoEquipe.

Projeto: AlocaODE – Ferramenta de Apoio à Alocação de Recursos de ODE
Caso de Uso: Definir Alocações de Recursos a Atividades
Descrição: Este caso de uso permite que o gerente de projeto faça alocações de recursos às atividades de um projeto.

Pré-Condição: O gerente de projeto deve ter aberto um projeto.

Cursos Normais:

Definir Alocações de Recursos a uma Atividade

O gerente de projeto seleciona a atividade para a qual deseja fazer alocações de recursos. Em seguida, são apresentados o nome e as datas de início e fim previstas da atividade, os tipos de recursos requeridos por elas e as alocações de recursos à atividade já realizadas. O gerente, então, seleciona os recursos a serem alocados para a atividade selecionada e as alocações são registradas.

Definir Alocações de Recursos a Atividades Automaticamente

Caso haja atividades do projeto que requeiram recursos e em cada uma delas exista somente um recurso que pode ser alocado conforme os recursos requeridos pelas mesmas definidos durante a definição de processo, uma mensagem é exibida perguntando se o desenvolvedor deseja automatizar o processo de definição, pois existem recursos requeridos que podem ser alocados automaticamente. Caso ele confirme, os recursos são alocados automaticamente para cada uma dessas atividades.

Classes: AlocacaoRH, AlocacaoFS, RecursoHumano, KRecursoHumano, FerramentaSoftware, KFerramentaSofware, RecursoHardware, KRecursoHardware, Atividade, ProcessoProjetoGeral, ProcessosProjetoEspecífico, Projeto, Equipe, AlocacaoEquipe.

Projeto: AlocaODE – Ferramenta de Apoio à Alocação de Recursos de ODE
Caso de Uso: Editar Alocação de Recurso Humano
Descrição: Este caso de uso permite que o gerente de projeto planeje detalhadamente as alocações de recursos humanos às atividades de um projeto.

Pré-Condição: O gerente de projeto deve ter aberto um projeto.

Curso Normal:

O gerente de projeto seleciona a alocação de recurso humano a atividade que deseja editar. Em seguida, o nome do recurso humano, o papel desempenhado pelo mesmo e os dados da alocação são exibidos, sendo que o gerente pode alterar as datas de início e fim previstas, a dedicação diária e a previsão do esforço do recurso humano a ser despendido na atividade. Os dados são validados e a alteração é registrada. Caso o gerente de projeto queira avaliar a participação do recurso humano da alocação na atividade, o evento "Avaliar Participação do Recurso Humano na Atividade" do caso de uso "Controlar Andamento de Alocação de Recurso Humano" pode ser realizado. Caso deseje cancelar uma alocação, o evento "Cancelar Alocação de Recurso Humano a Atividade" do caso de uso "Controlar Andamento de Alocação de Recurso Humano" pode ser realizado, e caso deseje anular o cancelamento de uma alocação, o evento "Anular Cancelamento de Alocação de Recurso Humano a Atividade" do mesmo caso de uso também pode ser realizado.

Curso Alternativo:

Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Alterar dados da alocação depois de sua participação ter sido iniciada: Os dados de datas de início e fim previstas, a dedicação diária e o esforço a ser despendido na atividade estarão habilitados para alteração caso a alocação tenha sido iniciada. Caso contrário, somente a dedicação diária e o esforço a ser despendido poderão ser alterados.

Classes: AlocacaoRH, RecursoHumano, KRecursoHumano, ProcessoProjetoGeral, ProcessosProjetoEspecifico, Projeto, Atividade.

Projeto: AlocaODE – Ferramenta de Apoio à Alocação de Recursos de ODE
Caso de Uso: Controlar Andamento de Alocação de Recurso Humano a Atividade
Descrição: Este caso de uso permite que o gerente de projeto cancele uma alocação de recurso humano a uma atividade, anule um cancelamento ou encerre a participação de um recurso humano na atividade. Por meio deste caso de uso também, o desenvolvedor pode solicitar o encerramento de sua participação na atividade.

Cursos Normais:

Solicitar Encerramento da Participação na Atividade

O desenvolvedor solicita o encerramento da sua participação em uma atividade. Uma mensagem de confirmação é exibida e, caso confirmada, a alocação é registrada como aguardando autorização para ser encerrada.

Avaliar Participação de Recurso Humano na Atividade

O gerente de projeto seleciona uma alocação de recurso humano a atividade que esteja em avaliação para encerramento e indica se essa alocação pode ser efetivamente encerrada ou se deve retornar a ficar em andamento para ajustes no trabalho feito. No primeiro caso, a alocação é encerrada e a data de fim efetiva da alocação é registrada como a data atual.

Cancelar Alocação de Recurso Humano a Atividade

Pré-Condição: A alocação de recurso humano selecionada não pode estar encerrada.

O gerente de projeto informa a alocação de recurso humano a atividade que deseja cancelar e o motivo do cancelamento. Os dados são validados e a data do cancelamento é registrada como sendo a data atual e o estado anterior da alocação é registrado, sendo a alocação cancelada.

Anular Cancelamento de Alocação de Recurso Humano a Atividade

Pré-Condição: A alocação de recurso humano a atividade selecionada deve estar cancelada.

A alocação de recurso humano a atividade passa para o estado anterior registrado no momento do cancelamento.

Curso Alternativo:

Cancelar Alocação de Recurso Humano a Atividade

Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes: Atividade, AlocacaoRH, RecursoHumano, CancelamentoAlocacaoRH

Projeto: AlocaODE – Ferramenta de Apoio à Alocação de Recursos de ODE

Caso de Uso: Registrar Esforço Despendido

Descrição: Este caso de uso permite ao desenvolvedor registrar, alterar, consultar ou cancelar um esforço despendido em uma atividade.

Pré-Condição 1: O desenvolvedor deve ter selecionado uma alocação de recurso humano.

Pré-Condição 2: A alocação do desenvolvedor à atividade deve estar aguardando início da participação ou em andamento (normal ou para ajustes).

Cursos Normais:

Incluir Esforço Despendido:

O desenvolvedor informa a data, a quantidade de horas e a observação do esforço despendido na atividade da alocação de recurso humano selecionada. Os dados são validados e o esforço despendido é registrado. Caso o desenvolvedor deseje solicitar o encerramento da sua participação na atividade, o evento "Solicitar Encerramento da Participação na Atividade" do caso de uso "Controlar Andamento de Alocação de Recurso Humano a Atividade" é realizado. Caso seja o primeiro esforço despendido incluído na alocação, uma mensagem é exibida informando que a participação na atividade foi iniciada na data informada ao incluir o primeiro esforço e a data de início efetiva da alocação também é registrada de acordo com a data informada no primeiro esforço.

Alterar Esforço Despendido:

O desenvolvedor informa o esforço despendido que deseja alterar. Em seguida informa os dados a serem alterados. Os dados são validados e registrados.

Consultar Esforço Despendido:

O desenvolvedor informa o esforço despendido que deseja consultar. Os dados desse esforço despendido são apresentados.

Excluir Esforço Despendido:

O desenvolvedor informa o esforço despendido que deseja excluir. Uma solicitação de confirmação é exibida e, caso confirmada, o esforço despendido é excluído.

Cursos Alternativos:

Incluir/Alterar Esforço Despendido:

Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes: AlocacaoRH, EsforcoDespendido.

A.2 Análise

Esta seção apresenta a especificação de análise para a ferramenta Alocação de Recursos do ambiente ODE. Para realizar essa atividade, foi utilizado o método da Análise Orientada a Objetos, juntamente com a notação da UML. A Seção A.2.1 apresenta o modelo de classes, no qual são definidos os diagramas de pacotes e os diagramas de classes. Já a Seção A.2.2 mostra a modelagem comportamental, apresentando alguns dos aspectos dinâmicos do sistema, representados por meio de um diagrama de estados.

A.2.1 Modelo de Classes

O diagrama de pacotes da Figura A.3 apresenta as dependências existentes entre os pacotes contemplados nessa ferramenta.

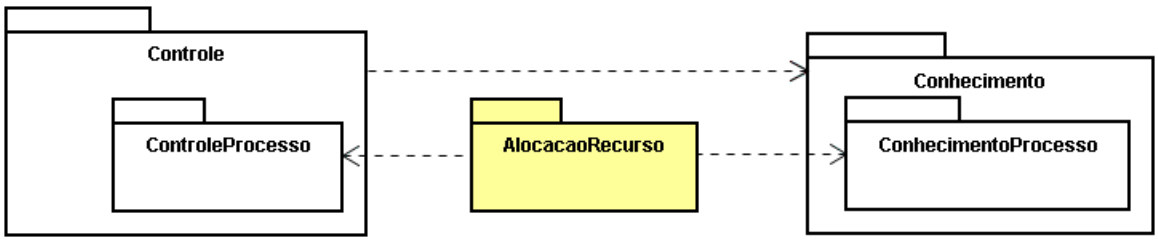

Figura A.3 - Diagrama de pacotes

O pacote *Controle* é responsável pelo núcleo do ambiente ODE. É de sua responsabilidade as principais funcionalidades referentes ao gerenciamento do sistema, sendo essas o controle de projetos, a definição de processos e o controle de acesso às ferramentas do ambiente. Dentro desse pacote, existe um subpacote, o pacote *ControleProcesso*, que contém as classes que controlam os processos de software no ambiente. De maneira geral, para cada projeto é definido um processo de projeto, que é decomposto em subprocessos, atividades e sub-atividades. Para cada atividade é ainda possível definir pré-atividades, recursos necessários, artefatos que podem ser insumos ou produtos e procedimentos a serem adotados.

O pacote *Conhecimento* é responsável pelo conhecimento contido no ambiente ODE, no qual são definidas classes que representam conceitos das ontologias utilizadas pelo sistema e que são usadas para falar sobre o universo de discurso do pacote *Controle* correspondente. Assim, o pacote *ConhecimentoProcesso*, que é um subpacote do pacote *Conhecimento*, trata do conhecimento do ambiente ODE no que se refere a processos de software e é utilizado pelo pacote *ControleProcesso*.

Na ferramenta de Definição de Processo do ambiente ODE, é possível definir os tipos de recursos humanos, de software e de hardware, necessários para a realização de cada atividade (representados como objetos da classe `KRecurso` do pacote *Conhecimento*). Depois que os tipos de recursos são definidos, pode-se, por meio da ferramenta de Alocação de Recursos, alocar recursos específicos (pessoas, ferramentas de software e equipamentos de hardware, respectivamente) para essas atividades.

O pacote *AlocacaoRecurso* é responsável pelo controle das alocações de recursos para cada atividade de um projeto. Após um projeto ter seu processo de software definido, é possível definir a equipe do projeto e alocar recursos para suas atividades. No que se refere à alocação de recursos humanos, é possível definir o papel que a pessoa (recurso humano) desempenhará na atividade, de acordo com os seus papeis desempenhados na equipe. Além disso, é possível alocar ferramentas de software e equipamentos de hardware para uma determinada atividade, de acordo com os tipos de recursos de software e de hardware requeridos pela mesma.

Para ter um maior controle sobre as alocações de recursos humanos a atividades, o gerente de projeto pode, ainda, controlar o andamento de uma alocação de recurso, cancelando-a, encerrando-a ou retornando-a para andamento para que ajustes sejam feitos, indicando que algo ainda precisa ser feito pelo recurso humano naquela atividade. Além disso, é possível anular um cancelamento de uma alocação de recurso humano. Já o desenvolvedor pode controlar o andamento de uma alocação, solicitando o encerramento de sua participação em alguma atividade, ao qual está alocado.

A Figura A.4 mostra o diagrama de classes desse pacote.

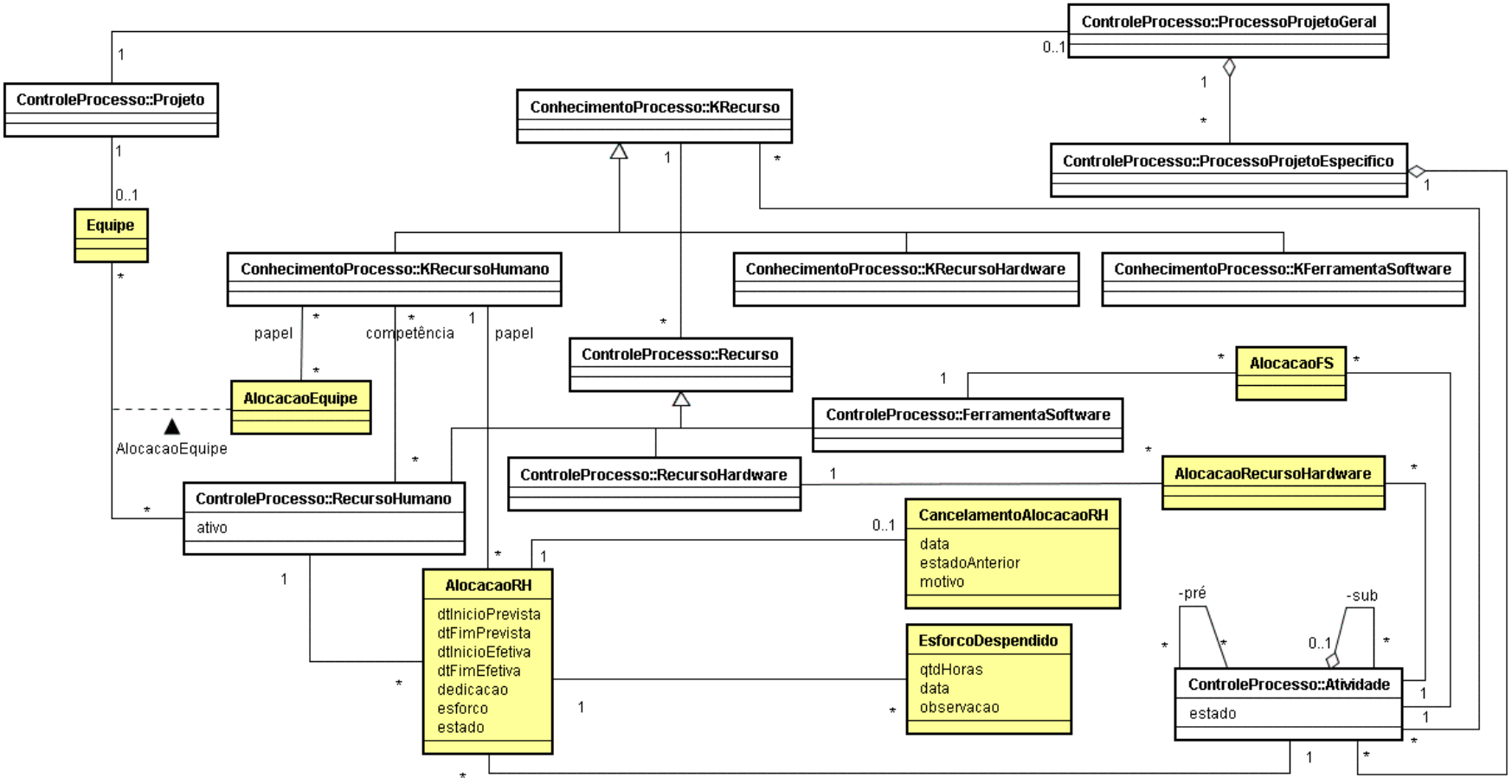

Figura A.4 - Diagrama de Classes do pacote *AlocacaoRecurso*

A.2.2 Modelo Comportamental

Os diagramas de classes apenas representam objetos estáticos de um modelo de objetos. Entretanto, além desses diagramas, muitas vezes é necessário analisar o seu comportamento dinamicamente.

Um modelo de comportamento mostra uma perspectiva dinâmica da aplicação, indicando como o sistema vai responder a eventos externos. Para modelar o comportamento da ferramenta de agenda, foi utilizado um diagrama de estados, apresentado na Sub-seção A.2.2.1

Vale a pena destacar que nenhum diagrama de seqüência foi criado, pois todos os casos usos com seus respectivos eventos especificados neste trabalho são fáceis de serem compreendidos e implementados.

A.2.2.1 Diagrama de Estados

Os diagramas de estados mostram as seqüências de estados pelos quais um objeto pode passar ao longo de sua vida, em reposta a estímulos recebidos, juntamente com suas respostas e ações (FALBO, 2002). A Figura A.5 apresenta o diagrama de estados elaborado para a classe `AlocacaoRH`.

Quando uma alocação de recurso humano é definida para uma atividade ainda não iniciada, essa alocação está apenas *Definida*. Após uma atividade ser iniciada, todas as alocações de recursos humanos da mesma passam a estar aptas a serem iniciadas, ou seja, o estado de cada uma delas passa a ser *Aguardando Início da Participação*. Quando o desenvolvedor registrar o primeiro esforço despendido na alocação a uma atividade, essa alocação passa a estar *Em Andamento.* Caso a alocação de recurso humano seja definida para uma atividade em execução, a alocação já passa a ter seu estado como *Aguardando Início da Participação*.

Após ter finalizado suas tarefas na alocação, o desenvolvedor solicita o encerramento da sua participação na atividade. Então, o estado da alocação passa a ser *Em Avaliação para Encerramento.*

Finalmente, o gerente de projeto pode avaliar se uma alocação de recurso humano *Em Avaliação para Encerramento* pode ser encerrada. Caso sim, a alocação é *Encerrada*. Caso contrário, ela passa para o estado *Em Andamento para Ajustes*, o que indica que a alocação deve

continuar em andamento para que o recurso humano continue trabalhando na atividade para fazer alguns ajustes.

Vendo que sua alocação está com o estado *Em Andamento para Ajustes,* o desenvolvedor pode rever com o gerente de projeto o que ainda deve ser feito para encerrar sua participação na atividade. Depois de realizados os devidos ajustes, o desenvolvedor pode solicitar novamente o encerramento da sua participação na atividade.

Uma alocação de recurso também pode ser *Cancelada* se ainda não tiver sido concluída. Além disso, é possível que deste estado a alocação possa voltar ao seu estado anterior, como mostra a Figura A.5.

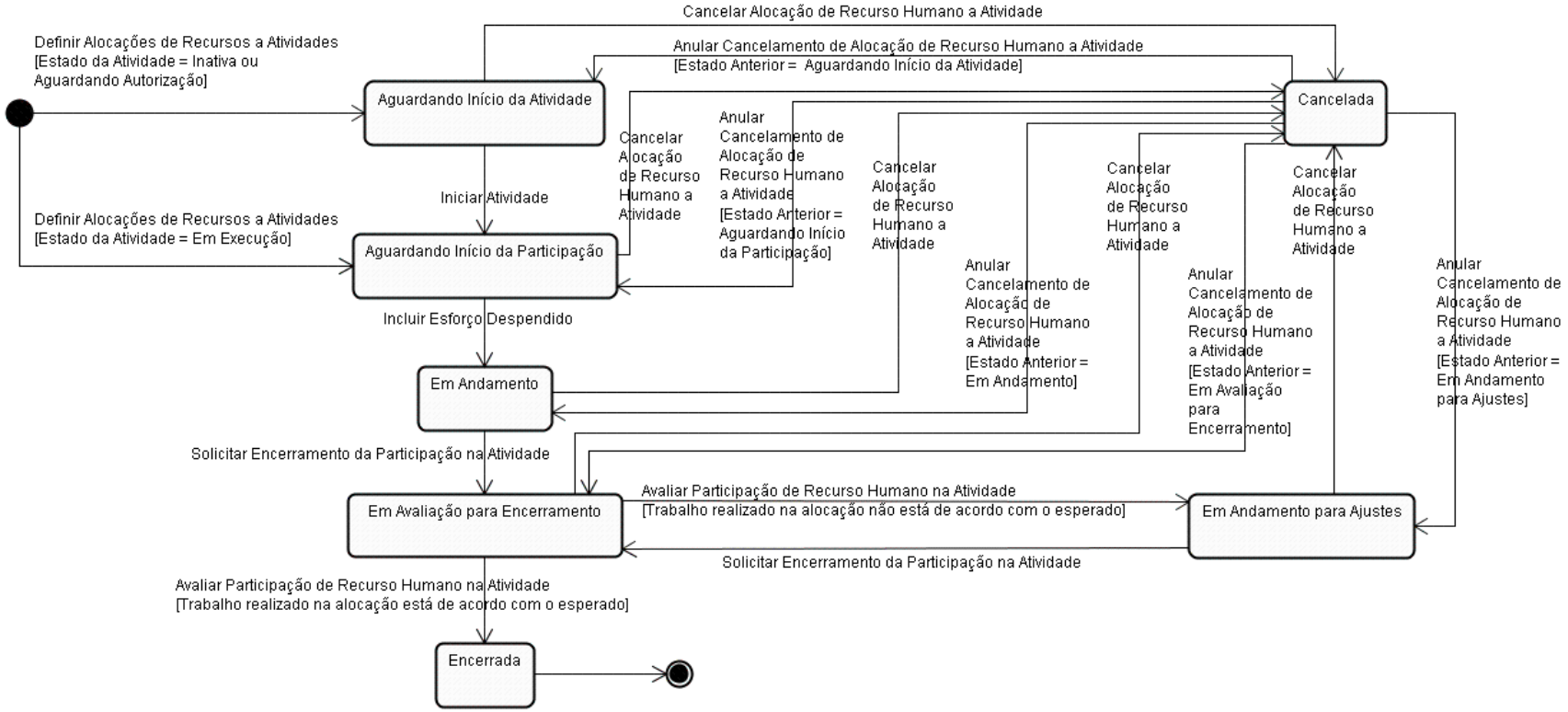

Figura A.5 - Diagrama de estado da classe `AlocacaoRH`.

A.3 Projeto

A.3.1 Introdução

Este documento define a especificação de projeto da ferramenta de apoio à alocação de recursos do ambiente ODE. Para a realização dessa atividade é necessário saber a plataforma de implementação. O ambiente ODE e por conseguinte a sua ferramenta de apoio à alocação de recursos são implementados usando a linguagem de programação Java, o banco de dados relacional PostgreSQL e o *framework* de persistência de objetos Hibernate.

Essa atividade foi elaborada em duas etapas: Projeto Arquitetural e Projeto Detalhado. A Seção A.3.2 apresenta o Projeto Arquitetural, definindo os diagramas de pacotes. A Seção A.3.3 apresenta o Projeto Detalhado para a ferramenta, mostrando os diagramas de classes de cada componente da arquitetura do pacote principal da ferramenta de apoio à alocação de recursos. A notação da UML foi utilizada.

A.3.2. Projeto Arquitetural

O diagrama de pacotes da Figura 1 mostra as dependências existentes entre os pacotes físicos definidos para essa ferramenta. Esses pacotes espelham uma decomposição com base no domínio do problema e, portanto, o diagrama da Figura A.6 é idêntico ao correspondente diagrama da fase de análise. Contudo, cada um desses pacotes é decomposto segundo uma arquitetura de cinco componentes, mostrada na Figura A.7.

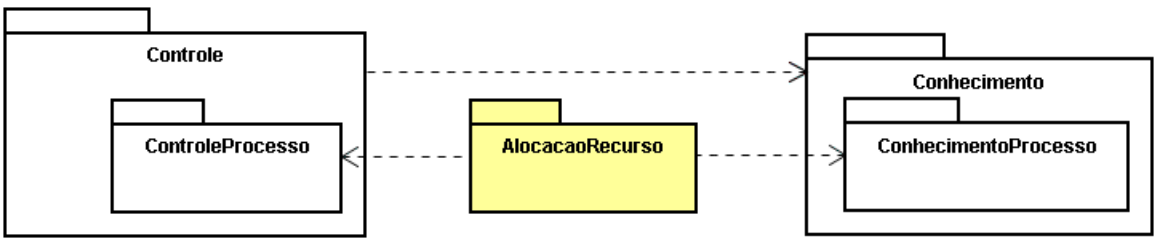

Figura A.6 - Diagrama de pacotes

O Componente de Domínio do Problema (Cdp), como o próprio nome indica, envolve as classes relativas a abstrações do domínio do problema e está diretamente relacionado com os modelos de análise.

O Componente de Gerência de Dados (Cgd) é responsável pela persistência dos objetos, tipicamente do Cdp e, por isso, depende deste.

O Componente de Gerência de Tarefas (Cgt) trata das funcionalidades da ferramenta e, portanto, está diretamente relacionado aos casos de uso descritos no Documento de Especificação de Requisitos. Para a realização dos casos de uso, é necessário interagir com os objetos do domínio do problema (Cdp) e com o Cgd para recuperar e armazenar esses objetos.

O Componente de Interação Humana (Cih) é responsável pelas interfaces com o usuário e, para isolá-las do restante do sistema, há o Componente de Controle de Interação (Cci) que separa o Cih dos pacotes contendo lógica de negócio, a saber Cdp e Cgt.

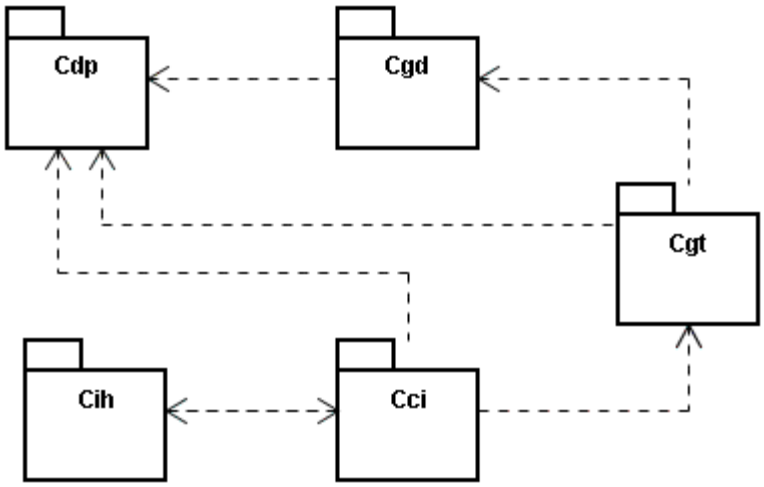

Figura A.7 - Arquitetura de Software Adotada

A.3.3 Pacote AlocacaoRecurso

O pacote *AlocacaoRecurso* contém as classes que definem as funcionalidades relacionadas a alocações de recursos humanos, de software e de hardware no ambiente ODE.

A.3.3.1 – Componente do Domínio do Problema (Cdp)

O diagrama de classes do Componente do Domínio do Problema do pacote *AlocacaoRecurso*, mostrado na Figura A.8, foi originado a partir do modelo de análise, adequando-o à plataforma de implementação.

Em termos de projeto, poucas mudanças foram feitas em relação ao modelo de classes do documento de análise. Além dos tipos de atributos e das navegabilidades entre as classes inseridos no modelo, a classe AlocacaoEquipe, que era associativa no modelo de análise, agora foi transformada em uma classe regular para deixar o modelo mais próximo da implementação.

Outra alteração que merece destaque é a criação das classes `EstadoAlocacaoRH` e `EstadoAtividade`, apresentadas no diagrama da Figura A.9, para representarem, respectivamente, os estados dos objetos das classes `AlocacaoRH` e `Atividade`. É importante saber que essas classes não são persistentes e têm seus valores constantes, definidos pelos respectivos diagramas de estados, sendo que o diagrama de estados da classe `Atividade` pode ser encontrado em (SEGRINI, 2007).

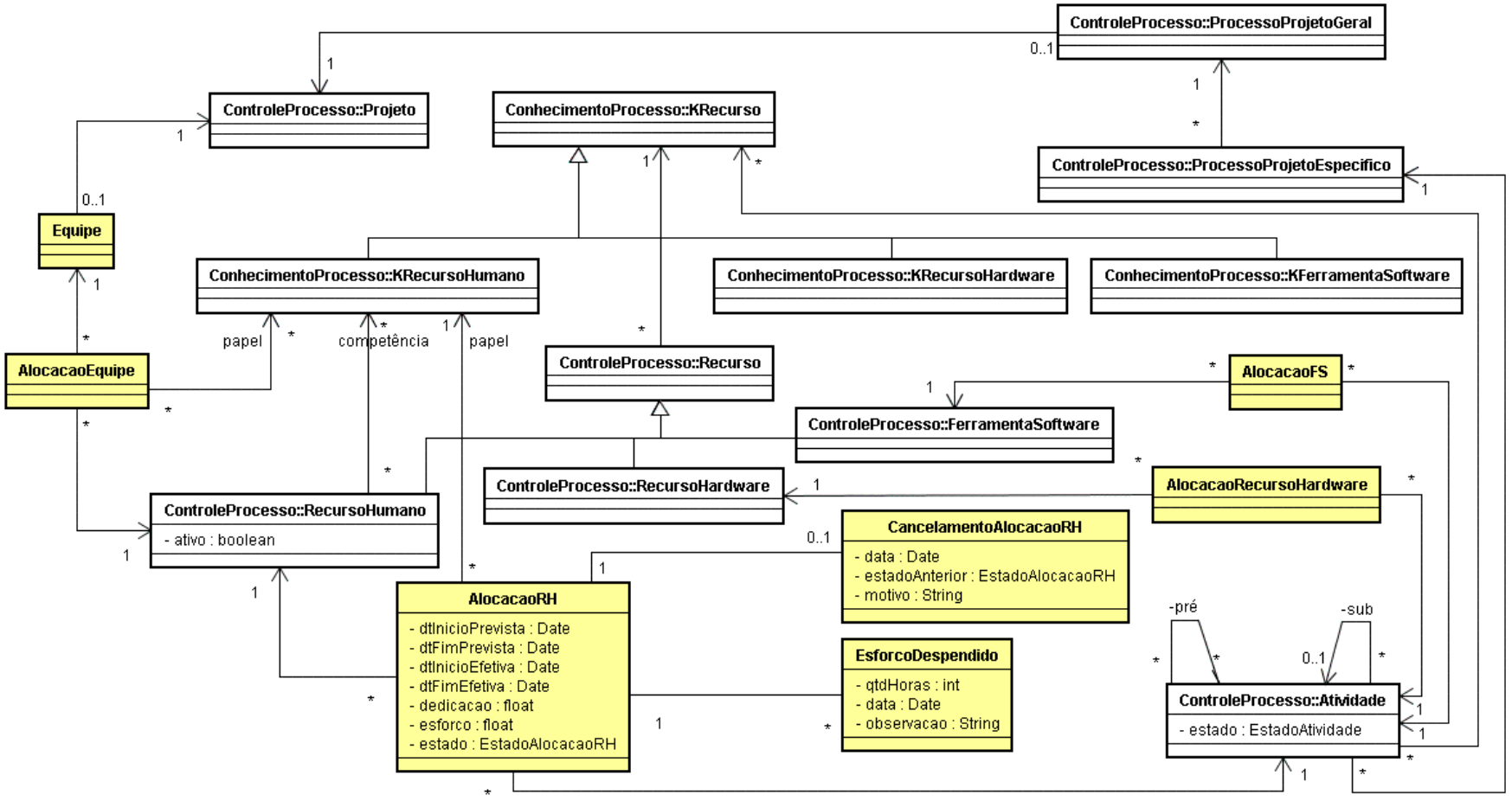

Figura A.8 – Componente de Domínio do Problema do pacote *AlocacaoRecurso*

Para cada estado da classe `AlocacaoRH` mostrado no respectivo diagrama de estado, foi criada uma constante que, na linguagem em Java, é um atributo público (*public*, representado no diagrama pelo símbolo '+'), estático (*static*, representado no diagrama pelo sublinhado) e final, informando que o valor do atributo não pode ser alterado (não existe representação gráfica correspondente para indicar essa propriedade).

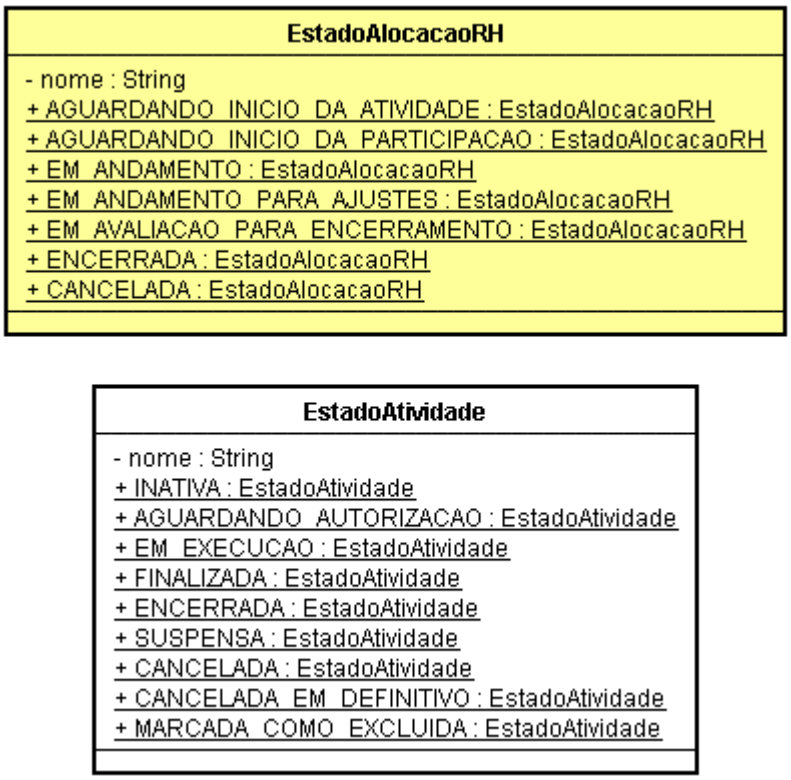

Figura A.9 - Classes de Estados

A.3.3.2 – Componente de Gerência de Dados (Cgd)

As classes do Componente de Gerência de Dados têm como objetivo abstrair o acesso ao banco de dados usado na implementação, dando maior flexibilidade e transparência ao sistema.

Para desenvolver a camada de persistência de ODE, são utilizados os padrões Objeto de Acesso a Dados (*Data Access Object* - DAO) (SUN, 2001) e Fábrica Abstrata (GAMMA et al., 1995), incluindo o uso do *framework* de persistência Hibernate (BAUER et al., 2005). O padrão

DAO fornece uma interface flexível entre aplicações e os mecanismos de persistência (banco de dados, arquivo, memória etc) e, portanto, o padrão torna-se parte integrante da camada de persistência. A Figura A.10 mostra as principais classes do utilitário de persistência de ODE.

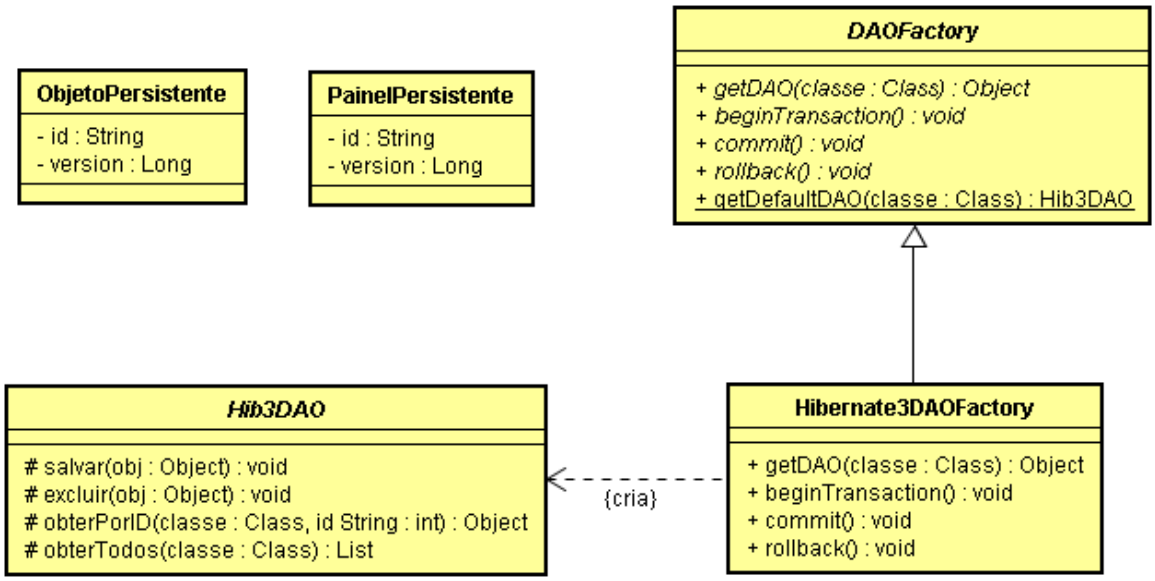

Figura A.10 - Pacote *Utilitario::Persistencia::hibernate*

Classes de domínio e itens gráficos que devem ser persistidos devem herdar de `ObjetoPersistente` e `PainelPersistente`, respectivamente. Além disso, cada classe DAO relacionada deve herdar de `Hib3DAO`, que possui as funcionalidades básicas de acesso ao mecanismo de persistência, e deve implementar a interface DAO associada, provendo flexibilidade no momento da criação de novas operações especializadas para um certo elemento de domínio.

A Figura A.11 apresenta o diagrama de classes do pacote *Cgd* da ferramenta Alocação de Recursos.

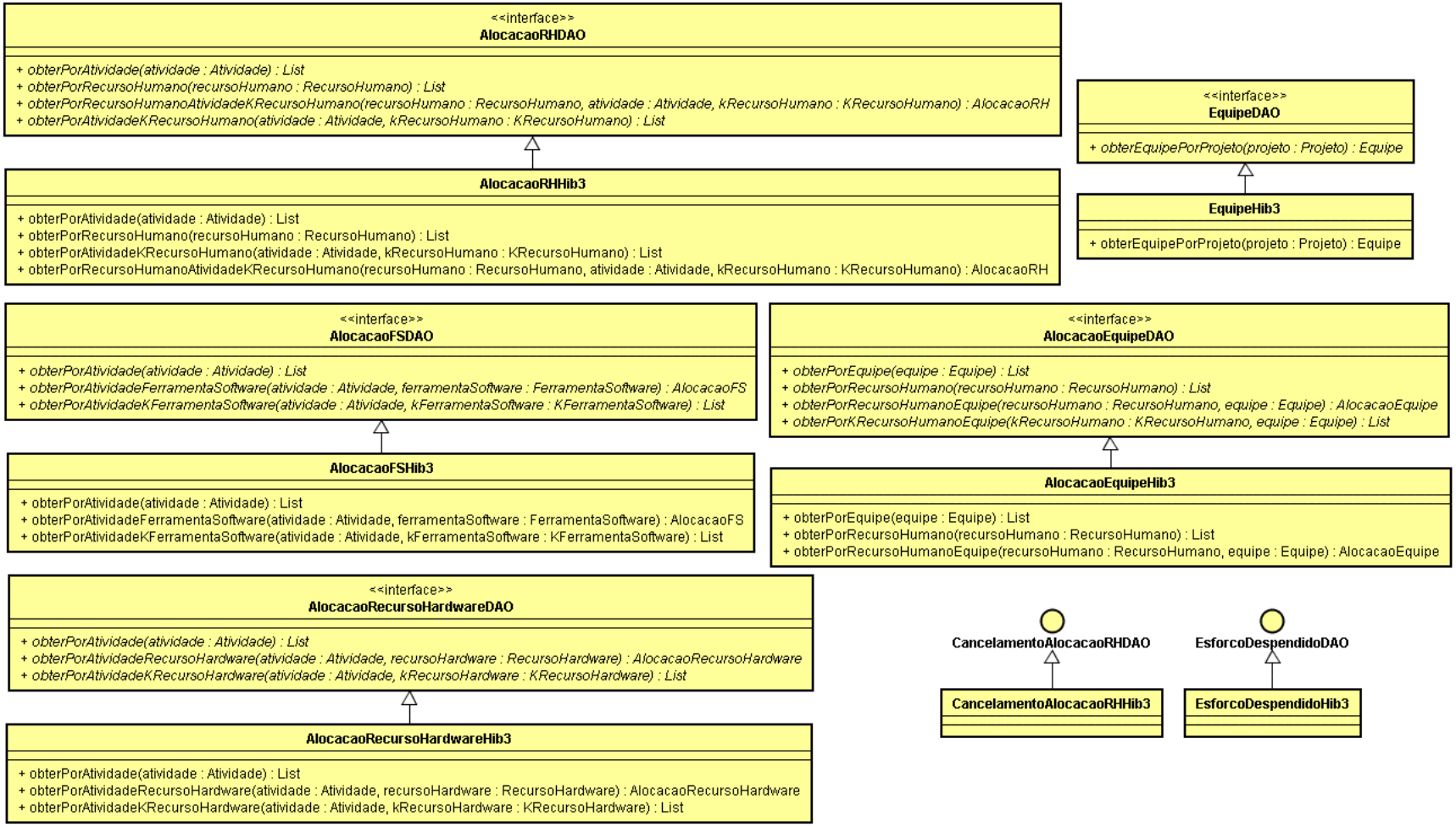

Figura A.11 - Componente de Gerência de Dados do pacote *AlocacaoRecurso*

Ainda que não mostrado no diagrama, as classes apresentadas na Figura A.11 herdam da classe abstrata `Hibernate3DAO` que implementa os métodos básicos de manipulação de objetos, a saber: *salvar*, *excluir*, *obterPorId* e *obterTodos*, conforme mostrado na Figura A.10 Em algumas classes, foi necessário adicionar métodos auxiliares para recuperar objetos requeridos através de um ou mais objetos relacionados ao mesmo, tal como o método *obterPorRecursoHumano* da classe `AlocacaoRHDAO` que retorna os objetos da classe `AlocacaoRH` que se relacionam com um objeto `RecursoHumano` específico.

A.3.3.3 – Componente de Gerência de Tarefas (Cgt)

O Componente de Gerência de Tarefas desse pacote possui as classes `AplAlocacaoRecurso`, `AplControlarAlocacaoRH` e `AplRegistrarEsforcoDespendido`. A primeira define as funcionalidades requeridas para tratar os casos de uso "Definir Equipe do Projeto", "Definir Alocações de Recursos a Atividades" e "Editar Alocação de Recurso Humano". Já as classes `AplControlarAlocacaoRH` e `AplRegistrarEsforcoDespendido` tratam de controle do andamento das alocações de recursos humanos (caso de uso "Controlar Andamento de Alocação de Recurso Humano a Atividade") e do registro de esforços despendidos (caso de uso "Registrar Esforço Despendido"), respectivamente.

A Figura A.12 mostra o Componente de Gerência de Tarefas desse pacote.

Figura A.12 - Componente de Gerência de Tarefas do Pacote *AlocacaoRecurso*

Pode-se observar que, na figura acima, nenhuma classe do Cgt possui relacionamentos diretos com objetos dos pacotes Cdp e Cgd. Ou seja, nenhuma classe tem atributos que são objetos desses pacotes. Porém, objetos dos pacotes Cdp e Cgd são instanciados dentro dos métodos das classes do Cgt, o que explica os relacionamentos de dependência entre os pacotes Cdp, Cgd e Cgt, mostrados na Figura A.7.

Além disso, as classes do Cgt possuem apenas métodos estáticos e, por isso, é necessário informar ao controlador se um método foi executado corretamente ou não. Para resolver tal problema, foram definidos tipos dos retornos padrão dos métodos das classes do Cgt do ambiente ODE, sendo eles números inteiros, representados por constantes, conforme apresentado na Tabela A.T.1.

Tabela A.T.1 – Tabela de Constantes de Retorno Padrão das Aplicações de ODE

Constante	**Valor**	**Descrição**
SUCESSO	0	Indica que a operação foi realizada com sucesso.
ERRO_PERSISTENCIA	1	Indica que ocorreu algum erro ao tentar atualizar dados.
ERRO_CONCORRENCIA	2	Indica um erro decorrente do acesso concorrente aos dados.
ERRO_INESPERADO	3	Indica erros que não são tratados na aplicação. Essa definição é importante, pois, mesmo sem saber o tipo de erro ocorrido, é possível tratá-lo, evitando que ele acarrete maiores problemas para o usuário, como a indisponibilidade do sistema.

Além dos tipos de retornos padrão do ambiente, cada ferramenta desenvolvida pode necessitar de definir tipos de retornos específicos para mesma. No caso da ferramenta deste trabalho, foi necessário definir alguns tipos de retornos específicos, mostrados na Tabela A.T.2.

Tabela A.T.2 –Constantes de Retorno Específicas para a ferramenta de Alocação de Recursos

Constante	Valor	Descrição
EXISTE_ALOCACOES_EQUIPES_NAO_EXCLUIDAS	4	Indica que não foi possível remover recursos humanos da equipe, pois esses estão alocados a atividades do projeto.
EXISTE_ALOCACOES_RH_NAO_EXCLUIDAS	5	Indica que não foi possível remover algumas alocações de recurso humano, pois essas já foram iniciadas.

A.3.3.4 – Componente de Interação Humana (Cih)

O Componente de Interação Humana do pacote *AlocacaoRecurso* comporta as interfaces utilizadas para a realização dos casos de uso definidos no documento de especificação de requisitos. A Figura A.13 apresenta o diagrama de classes desse componente do pacote *AlocacaoRecurso.*

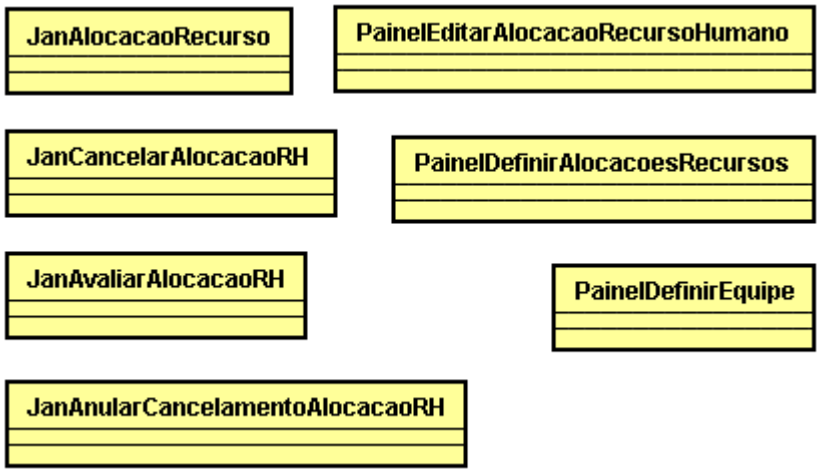

Figura A.13 - Componente de Interface Humana do pacote *AlocacaoRecurso*

A ferramenta é composta de quatro janelas: `JanAlocacaoRecurso`, `JanAvaliarAlocacaoRH`, `JanCancelarAlocacaoRH` e `JanAnularCancelamentoAlocacaoRH`, sendo que `JanAlocacaoRecurso` é responsável por comportar os painéis `PainelDefinirEquipe`, `PainelDefinirAlocacoesRecursos` e `PainelEditarAlocacaoRecursoHumano`.

O `PainelDefinirEquipe` apresenta para o desenvolvedor os recursos humanos cadastrados no ambiente ODE com suas respectivas especialidades, para que uma equipe possa ser definida, conforme a descrição do caso de uso "Definir Equipe do Projeto". Já o `PainelDefinirAlocacoesRecursos` permite que o usuário faça alocações de recursos às atividades de um projeto, de acordo o caso de uso "Definir Alocações de Recursos a Atividades". O `PainelEditarAlocacaoRecursoHumano` apresenta informações de uma alocação de recurso selecionada para que esta possa ser editada, conforme o caso de uso "Editar Alocação de Recurso Humano".

As janelas `JanAvaliarAlocacaoRH`, `JanCancelarAlocacaoRH` e `JanAnularCancelamentoAlocacaoRH` são responsáveis pela avaliação, cancelamento e anulação do cancelamento de uma alocação de recurso humano, relacionados, respectivamente, aos eventos de caso de uso *Avaliar Participação de Recurso Humano na Atividade, Cancelar Alocação de Recurso Humano a Atividade e Anular Cancelamento de Alocação de Recurso*

Humano a Atividade do caso de uso *Controlar Andamento de Alocação de Recurso Humano a Atividade*.

Conforme apresentado na próxima seção, o controlador é responsável pela exibição de todas as interfaces do sistema, o que explica a inexistência de relacionamentos entre as mesmas.

A.3.3.5 – Componente de Controle de Interação (Cci)

O Componente de Controle de Interação tem como objetivo tratar a comunicação de dados entre as interfaces da ferramenta e sua aplicação, como mostra a Figura A.14.

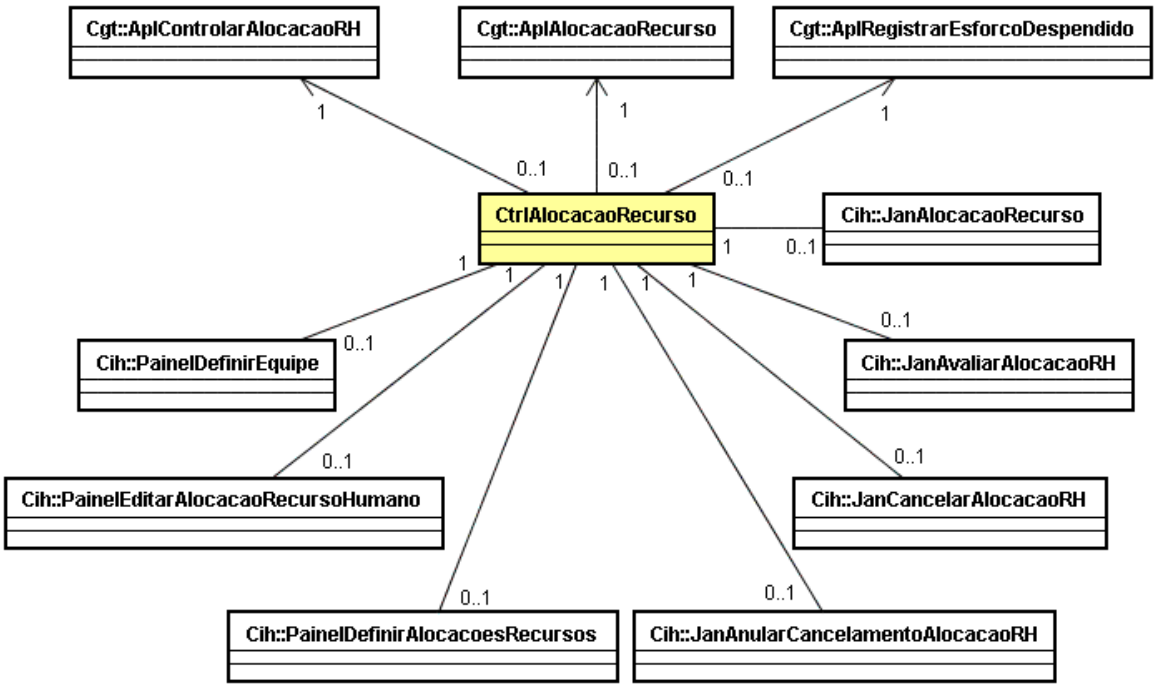

Figura A.14 - Componente de Controle de Interação do Pacote *AlocacaoRecurso*

A navegabilidade no sentido de `CtrlAlocacaoRecurso` as classes do Cgt permite que as funcionalidades da ferramenta também possam ser usadas por outras ferramentas do ambiente ODE e, caso exista necessidade de alterar a interface da ferramenta, apenas o controlador da mesma deve ser refeito de acordo com as necessidades exigidas pelas modificações.

O controlador conhece todas as operações necessárias para atender às requisições das interfaces. Quando uma interface necessita de alguma funcionalidade da classe de aplicação, o

controlador faz uma requisição à aplicação. Em seguida, esta retorna um resultado para o controlador que o interpreta e atualiza as interfaces, caso haja necessidades.

Apesar dos métodos não estarem explícitos no diagrama da Figura A.14, o controlador possui vários métodos cujas principais funcionalidades são de atualização das interfaces e de requisições às aplicações.

Anexo B

Documentação da Evolução da Ferramenta de Agenda

Este documento tem como objetivo apresentar informações sobre a versão atual da ferramenta de Agenda, abrangendo todos as fases do ciclo de vida de desenvolvimento.

Inicialmente é apresentada a especificação de requisitos da ferramenta (Seção B.1), detalhando apenas os requisitos funcionais, já que os requisitos não-funcionais foram tratados na Seção 4.3. Em seguida, a Seção B.2 define a fase de análise da ferramenta, apresentando uma visão do domínio do problema, desconsiderando aspectos tecnológicos. Finalmente a Seção B.3 discute o projeto dessa versão da ferramenta.

B.1 Especificação de Requisitos

Este documento apresenta a especificação de requisitos para a construção da ferramenta de agenda do ambiente ODE. A Seção B.1.1 contém uma breve descrição do sistema (descrição do mini-mundo). Em seguida, na Seção B.1.2, são apresentados os diagramas de casos de uso associados às descrições de cada caso de uso.

B.1.1 Descrição do Mini-Mundo

Às vezes é difícil memorizarmos muitas informações em nossas mentes, apesar do ser humano ter uma grande capacidade para realizar tal tarefa. Normalmente, estamos preocupados com tantas coisas, que é possível que nossa memória falhe e nos faça esquecer algo muito importante.

Uma agenda pode conter informações capazes de ajudar o ser humano a lembrar compromissos, contatos e anotações, ou seja, pode ser uma segunda forma para organizar e memorizar informações importantes para que sejam lembradas quando necessário.

No ambiente ODE é bastante interessante que cada desenvolvedor tenha uma agenda eletrônica, com o objetivo de orientá-lo dentro do sistema, informando-o à respeito das suas alocações a atividades dentro do ambiente, os compromissos que ele deve cumprir, além de outros recursos básicos que qualquer agenda deve ter.

Assim que o desenvolvedor fizer o *login* no ambiente, sua agenda será aberta e ele poderá ver as suas alocações a atividades, bem como os seus compromissos, suas anotações e seus contatos. Para cada atividade a que está alocado, é possível que ele registre esforços despendidos ou até mesmo selecione uma ferramenta de software para realizar seu trabalho.

B.1.2 Modelo de Casos de Uso

A Figura B.1 mostra o diagrama de pacotes da ferramenta *AgendaODE*, no que se refere às funcionalidades a serem providas.

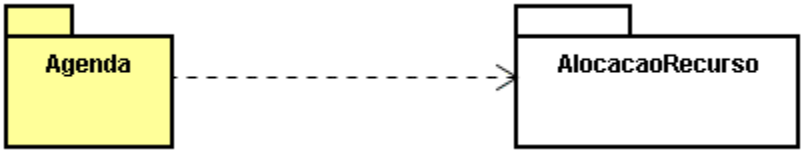

Figura B.1 - Diagrama de Pacotes

O pacote *Agenda* trata de todas as funcionalidades que o desenvolvedor pode encontrar na ferramenta, com exceção das funcionalidades que alteram os dados de uma alocação de recurso humano a atividade, sendo essas tratadas pelo pacote *AlocacaoRecurso* que define as funcionalidades relacionadas a alocações de recursos humanos, de software e de hardware no ambiente ODE.

A Figura B.2 mostra o diagrama de casos de uso do pacote *Agenda*. A seguir, os casos de uso propostos nesse pacote são descritos. Os casos de uso do pacote *AlocacaoRecurso* estão descritos no documento de especificação de requisitos da ferramenta de alocação de recursos.

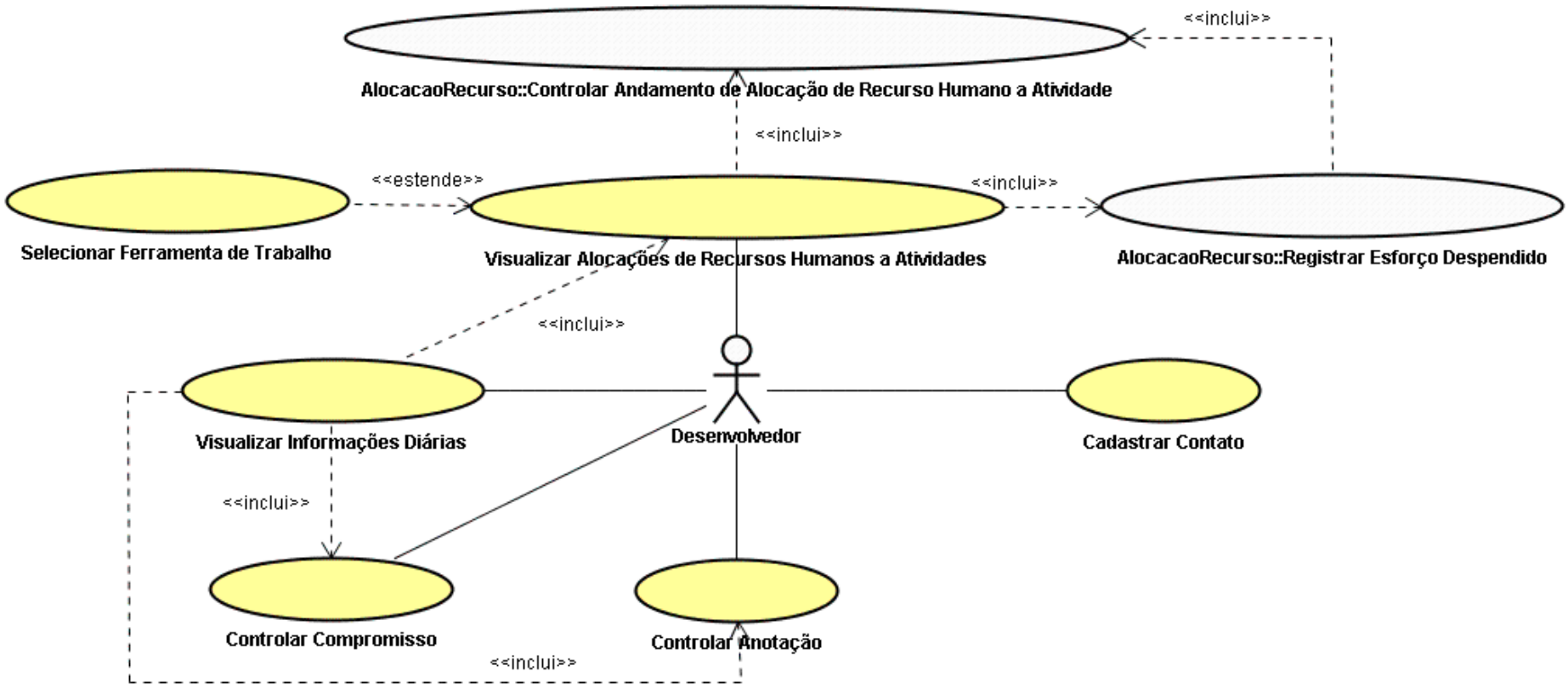

Figura B.2 - Diagrama de Casos de Uso do pacote *Agenda*

Projeto: *AgendaODE* (pacote *Agenda*)

Caso de Uso: Visualizar Alocações de Recurso Humano a Atividades

Descrição: Este caso de uso permite que o desenvolvedor visualize suas alocações a atividades no ambiente ODE, de acordo com o projeto, o subprocesso e o andamento das alocações selecionadas.

Cursos Normais:

Selecionar Alocação de Recurso Humano a Atividade

O desenvolvedor pode selecionar um projeto em que deseja trabalhar, bem como um subprocesso desse projeto, dentre aqueles aos quais está alocado. A seguir, ele deve selecionar uma alocação a uma atividade, podendo indicar se deseja escolher dentre todas suas alocações ou se deseja escolher uma alocação pelo estado da mesma (aguardando início da participação, em andamento, em andamento para ajustes, todas em andamento, em avaliação para encerramento, cancelada ou encerrada. Em seguida, o evento "Visualizar Alocação de Recurso Humano a Atividade" deste caso de uso é realizado.

Visualizar Alocação de Recurso Humano a Atividade

O nome, o estado, o percentual total de esforço registrado e as datas de início e fim previstas e efetivas da atividade relacionada à alocação selecionada são apresentados, bem como os recursos humanos alocados à mesma. Além disso, são disponibilizadas as ferramentas de software alocadas para essa atividade e são apresentados a data de início prevista, a data de fim prevista, a data de início efetiva, a data de fim efetiva, a dedicação, o estado, o papel desempenhado e o percentual de esforço registrado na alocação. Caso o desenvolvedor deseje registrar um esforço despendido para a alocação, o caso de uso "Registrar Esforço Despendido" é realizado. Caso deseje trabalhar com uma das ferramentas de software disponíveis, o caso de uso "Selecionar Ferramenta de Trabalho" pode ser realizado. Caso deseje solicitar o encerramento da sua participação na atividade, o evento "Solicitar Encerramento da Participação na Atividade" do caso de uso "Controlar Andamento da Alocação de Recurso Humano a Atividade" também pode ser realizado.

Classes: Projeto, ProcessoProjetoEspecifico, Atividade, AlocacaoRH, AlocacaoFS, EsforcoDespendido, RecursoHumano e FerramentaSoftware.

Projeto: *AgendaODE* (pacote *Agenda*)
Caso de Uso: Selecionar Ferramenta de Trabalho
Descrição: Este caso de uso permite que o desenvolvedor selecione uma ferramenta de software, alocada à atividade sendo exibida, que deseja trabalhar.

Pré-Condição 1: O desenvolvedor deve ter selecionado uma alocação de recurso humano a atividade.
Pré-Condição 2: A alocação selecionada deve estar aguardando o início da participação ou em andamento (normal ou para ajustes).
Pré-Condição 3: A atividade da alocação selecionada deve ter pelo menos uma ferramenta alocada a ela.

Curso Normal:

O desenvolvedor seleciona uma dentre as ferramentas de software disponíveis para a atividade da alocação selecionada e a ferramenta é iniciada.

Classes: Atividade, AlocacaoRH, AlocacaoFS e FerramentaSoftware.

Projeto: *AgendaODE* (pacote *Agenda*)

Caso de Uso: Controlar Compromisso

Descrição: Este caso de uso permite ao desenvolvedor incluir, alterar, consultar ou excluir um compromisso na sua agenda.

Cursos Normais:

Incluir Compromisso:

O desenvolvedor informa o assunto, o local, a hora, a duração, a descrição, o tipo do compromisso (eventual, diário, semanal ou mensal), a atividade relacionada, caso exista, e se o mesmo foi realizado. Caso o tipo seja eventual, ele deve informar também a data do compromisso. Caso seja semanal, ele deve informar o dia da semana. Caso seja mensal, ele deve informar o dia do mês. Os dados são validados e um novo compromisso é criado**.**

Alterar Compromisso:

O desenvolvedor informa o compromisso que deseja alterar. Em seguida informa os dados a serem alterados. Os dados são validados e registrados.

Consultar Compromisso:

O desenvolvedor informa o compromisso que deseja consultar. Os dados do compromisso selecionado são apresentados.

Excluir Compromisso:

O desenvolvedor informa o compromisso que deseja excluir. Uma solicitação de confirmação é exibida e, caso confirmada, o compromisso é excluído.

Cursos Alternativos:

Incluir/Alterar Compromisso:

- O intervalo do compromisso a ser incluído/alterado coincide com outro intervalo de um compromisso cadastrado: Uma mensagem é exibida, informando que há

conflito de horário com outro compromisso e se o desenvolvedor deseja incluir/alterar mesmo assim. Caso deseje, os dados são validados e registrados.

- Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes: Atividade, AgendaRH, RecursoHumano, Compromisso, CompromissoEventual, CompromissoDiario, CompromissoSemanal e CompromissoMensal.

Projeto: *AgendaODE* (pacote *Agenda*)

Caso de Uso: Controlar Anotação

Descrição: Este caso de uso permite ao desenvolvedor incluir, alterar, consultar ou excluir uma anotação na sua agenda.

Cursos Normais:

Incluir Anotação:

O desenvolvedor informa o assunto, o grau de importância (baixo, médio, alto) e a descrição da anotação. Os dados são validados e uma nova anotação é criada.

Alterar Anotação:

O desenvolvedor informa a anotação que deseja alterar. Em seguida informa os dados a serem alterados. Os dados são validados e registrados.

Consultar Anotação:

O desenvolvedor informa a anotação que deseja consultar. Os dados da anotação selecionada são apresentados.

Excluir Anotação:

O desenvolvedor informa a anotação que deseja excluir. Uma solicitação de confirmação é exibida e, caso confirmada, a anotação é excluída.

Curso Alternativo:

Incluir/Alterar Anotação:

Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes: Anotacao, AgendaRH, RecursoHumano

Projeto: *AgendaODE* (pacote *Agenda*)
Caso de Uso: Cadastrar Contato
Descrição: Este caso de uso permite ao desenvolvedor incluir, alterar, consultar ou excluir um contato em sua agenda.

Cursos Normais:

Incluir Contato:

O desenvolvedor informa o nome, a profissão, o local de trabalho, e-mails, os telefones (comercial, residencial e celular), o fax e o endereço do contato. Os dados são validados e um novo contato é criado.

Alterar Contato:

O desenvolvedor informa o contato que deseja alterar. Em seguida informa os dados a serem alterados. Os dados são validados e registrados.

Consultar Contato:

O desenvolvedor informa o contato que deseja consultar. Os dados do contato selecionado são apresentados.

Excluir Contato:

O desenvolvedor informa o contato que deseja excluir. Uma solicitação de confirmação é exibida e, caso confirmada, o contato é excluído.

Curso Alternativo:

Incluir/Alterar Contato:

Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes: Contato, AgendaRH, RecursoHumano

Projeto: *AgendaODE* (pacote *Agenda*)
Caso de Uso: Visualizar Informações Diárias
Descrição: Este caso de uso permite que o desenvolvedor visualize as informações do dia corrente marcadas em sua agenda, sendo essas as suas alocações em andamento (normal ou para ajustes), seus compromissos no dia e suas anotações com grau de importância alto.

Curso Normal:

O desenvolvedor solicita a visualização de suas informações diárias e são apresentados as suas alocações em andamento (normal ou para ajustes), os seus compromissos do dia e as suas anotações que possuem grau de importância alto. Caso deseje visualizar os dados de uma alocação, ele a informa e o evento "Visualizar Alocação de Recurso Humano a Atividade" do caso de uso "Visualizar Alocações de Recurso Humano a Atividades" é realizado, exibindo as informações da alocação. Caso deseje visualizar os dados de um compromisso, o desenvolvedor o informa e o evento "Consultar Compromisso" do caso de uso "Controlar Compromisso" é realizado, exibindo as informações do compromisso. Caso deseje visualizar os dados de uma anotação, ele a informa e o evento "Consultar Anotação" do caso de uso "Controlar Anotação" é realizado, exibindo as informações da anotação.

B.2 Análise

Este documento apresenta a especificação de análise para a ferramenta de Agenda do ambiente ODE. Para realizar essa atividade, foi utilizado o método da Análise Orientada a Objetos, juntamente com a notação UML. A Seção B.2.1 apresenta o modelo de classes, no qual são definidos os diagramas de pacotes e os diagramas de classes para a ferramenta.

B.2.1 Modelo de Classes

A ferramenta de Agenda tem como principal objetivo orientar um desenvolvedor do ambiente ODE dentro do sistema, apresentando suas alocações em determinadas atividades, bem como suas respectivas ferramentas de software disponíveis para realizar algum trabalho. Além dessa funcionalidade, a ferramenta contém recursos básicos que são encontrados na maioria dos sistemas de agenda, como tratar compromissos, anotações e contatos.

Após a identificação do desenvolvedor no sistema, a ferramenta de agenda é aberta, mostrando um breve resumo de suas informações mais importantes, como suas alocações em andamento (normal ou para ajustes), seus compromissos no dia e suas anotações com grau de importância alto, conforme a descrição do caso de uso “Visualizar Informações Diárias”. Em seguida, a ferramenta fica disponível para ser utilizada, exibindo as funcionalidades descritas na especificação de requisitos funcionais.

O diagrama de pacotes da Figura B.3 apresenta as dependências existentes entre os pacotes contemplados nessa ferramenta.

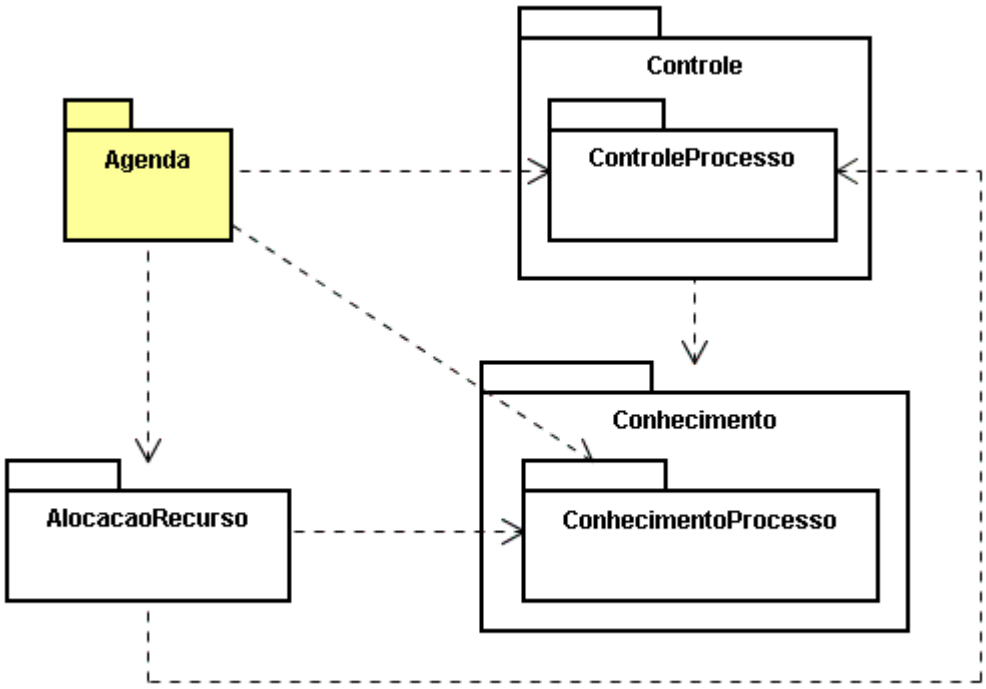

Figura B.3 - Diagrama de pacotes

O pacote *Controle* é responsável pelo núcleo do ambiente ODE. É de sua responsabilidade as principais funcionalidades referentes ao gerenciamento do sistema, sendo essas o controle de projetos, a definição de processos e o controle de acesso às ferramentas do ambiente. Dentro desse pacote, existe um subpacote, o pacote *ControleProcesso*, que contém as classes que controlam os processos de software no ambiente. De maneira geral, para cada projeto é definido um processo de projeto, que é decomposto em subprocessos, atividades e sub-atividades. Para cada atividade é ainda possível definir pré-atividades, recursos necessários, artefatos que podem ser insumos ou produtos e procedimentos a serem adotados.

O pacote *Conhecimento* é responsável pelo conhecimento contido no ambiente ODE, no qual são definidas classes que representam conceitos das ontologias utilizadas pelo sistema e que são usadas para falar sobre o universo de discurso do pacote *Controle* correspondente. Assim, o pacote *ConhecimentoProcesso*, que é um subpacote do pacote *Conhecimento*, trata do conhecimento do ambiente ODE no que se refere a processos de software e é utilizado pelo pacote *ControleProcesso*.

O pacote *AlocacaoRecurso* é responsável pelo controle das alocações de recursos para cada atividade de um projeto. Após um projeto ter seu processo de software definido, é possível alocar recursos humanos, de software e de hardware, para uma determinada atividade do mesmo.

O pacote *Agenda* é o principal pacote da ferramenta de Agenda e, portanto, é o único pacote cujo diagrama de classes é apresentado neste documento. Esse pacote contém as classes `AgendaRH`, `Contato`, `Anotacao` e `Compromisso`, sendo que esta última é especializada para cada um dos tipos de compromisso definidos: eventual, diário, semanal ou mensal, conforme apresentado na Figura B.4.

Na ferramenta de Definição de Processo do ambiente ODE, é possível definir os tipos de recursos, tanto humanos quanto de software, necessários para a realização de cada atividade. Depois que os tipos de recursos foram definidos, pode-se, por meio da ferramenta de Alocação de Recursos, alocar recursos específicos (pessoas e ferramentas de software, respectivamente). No que se refere à alocação de recursos, cria-se uma alocação de recurso humano, sendo que, para cada alocação, é possível definir a pessoa (o recurso humano) e o papel que a mesma vai exercer na alocação, de acordo com suas competências. Além disso, é possível alocar ferramentas de software para uma determinada atividade, de acordo com os tipos de recursos de software requeridos pela mesma.

Cada desenvolvedor pode visualizar sua agenda. Na ferramenta de agenda, é possível visualizar as alocações do desenvolvedor em determinadas atividades, sendo que essa visualização pode ser refinada pelo projeto, subprocesso e o estado da alocação. Ainda por meio da ferramenta de agenda, é possível selecionar uma ferramenta de software alocada à atividade que o desenvolvedor está alocado.

A classe `AgendaRH` representa a agenda de um recurso humano cadastrado no ambiente. A classe `Compromisso`, como o nome já diz, representa um compromisso que o desenvolvedor possui, podendo ser ele eventual, diário, semanal ou mensal. A classe `Anotacao` define um pequeno lembrete que pode ser visualizado ao abrir a agenda. Por fim, um desenvolvedor pode ter uma lista de contatos, representada pela classe `Contato`.

Vale destacar que, por ser uma ferramenta basicamente de controle, são utilizadas muitas classes de outros pacotes e poucas são definidas neste, como é possível perceber pela Figura B.4. Em relação aos atributos das classes dos pacotes *ControleProcesso* e *AlocacaoRecurso*, apenas aqueles relevantes para a ferramenta de agenda estão sendo mostrados. Para maiores informações

sobre esses modelos, vide os documentos de especificação de análise das ferramentas de Definição de Processos e Alocação de Recursos, respectivamente.

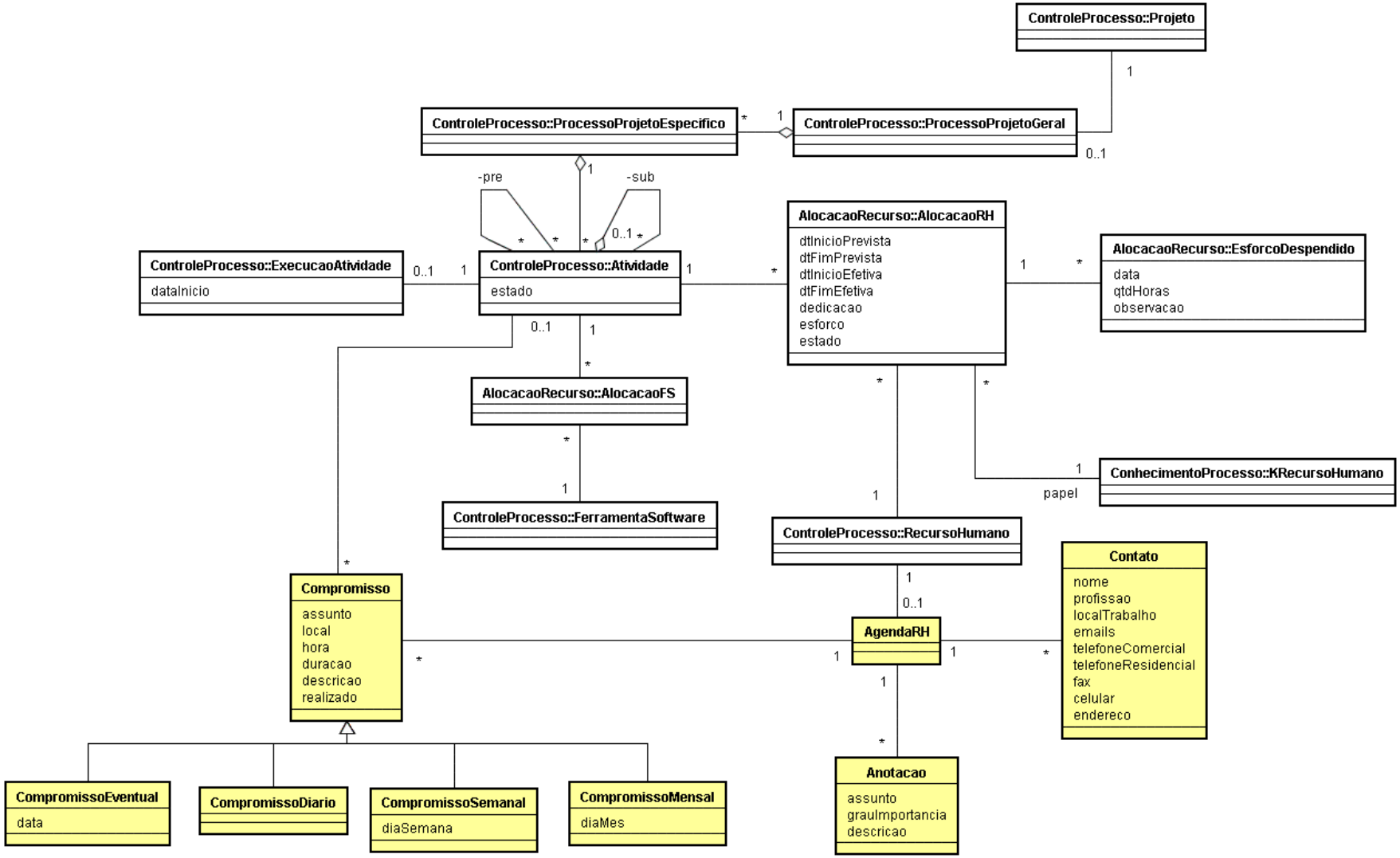

Figura B.4 - Diagrama de Classes do pacote *Agenda*

B.3 Projeto

B.3.1. Introdução

Este documento apresenta a especificação de projeto da ferramenta de Agenda do ambiente ODE. Para a realização dessa atividade é necessário saber a plataforma de implementação. O ambiente ODE e por conseguinte a sua ferramenta de Agenda são implementados usando a linguagem de programação Java, o banco de dados relacional PostgreSQL e o *framework* de persistência de objetos Hibernate.

Essa atividade foi elaborada em duas etapas: Projeto Arquitetural e Projeto Detalhado. A Seção B.3.2 apresenta o Projeto Arquitetural por meio de diagramas de pacotes. A Seção B.3.3 apresenta o Projeto Detalhado para a ferramenta, apresentando os diagramas de classes de cada componente da arquitetura do pacote principal da ferramenta de Agenda. A notação da UML foi utilizada.

B.3.2. Projeto Arquitetural

O diagrama de pacotes da Figura 1 mostra as dependências existentes entre os pacotes físicos definidos para essa ferramenta. Esses pacotes espelham uma decomposição com base no domínio do problema e, portanto, o diagrama da Figura B.5 é idêntico ao correspondente diagrama da fase de análise. Contudo, cada um desses pacotes é decomposto segundo uma arquitetura de cinco componentes, mostrada na Figura B.6.

O Componente de Domínio do Problema (Cdp), como o próprio nome indica, envolve as classes relativas a abstrações do domínio do problema e está diretamente relacionado com os modelos de análise.

O Componente de Gerência de Dados (Cgd) é responsável pela persistência dos objetos, tipicamente do Cdp e, por isso, depende deste.

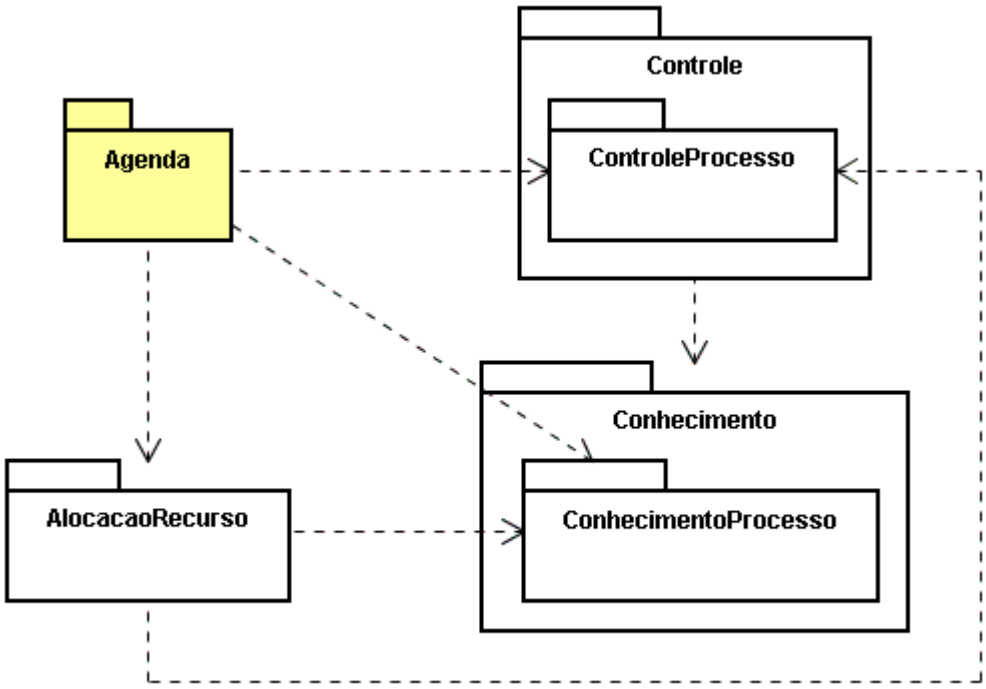

Figura B.5 - Diagrama de pacotes

O Componente de Gerência de Tarefas (Cgt) trata das funcionalidades da ferramenta e, portanto, está diretamente relacionado aos casos de uso descritos no Documento de Especificação de Requisitos. Para a realização dos casos de uso, é necessário interagir com os objetos do domínio do problema (Cdp) e com o Cgd para recuperar e armazenar esses objetos.

O Componente de Interação Humana (Cih) é responsável pelas interfaces com o usuário e, para isolá-las do restante do sistema, há o Componente de Controle de Interação (Cci) que separa o Cih dos pacotes contendo lógica de negócio, a saber Cdp e Cgt.

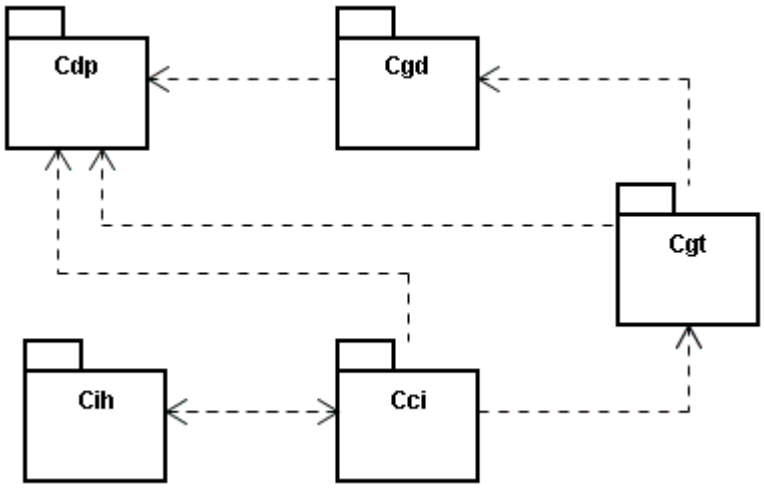

Figura B.6 - Arquitetura de Software Adotada

B.3.3. Pacote *Agenda*

O pacote *Agenda* contém as classes que definem as funcionalidades relacionadas à agenda de um recurso humano no ambiente ODE.

B.3.3.1 – Componente do Domínio do Problema (Cdp)

O diagrama de classes do Componente do Domínio do Problema do pacote *Agenda* foi criado a partir do modelo de análise, adequando-o à plataforma de implementação, como mostra a Figura B.7.

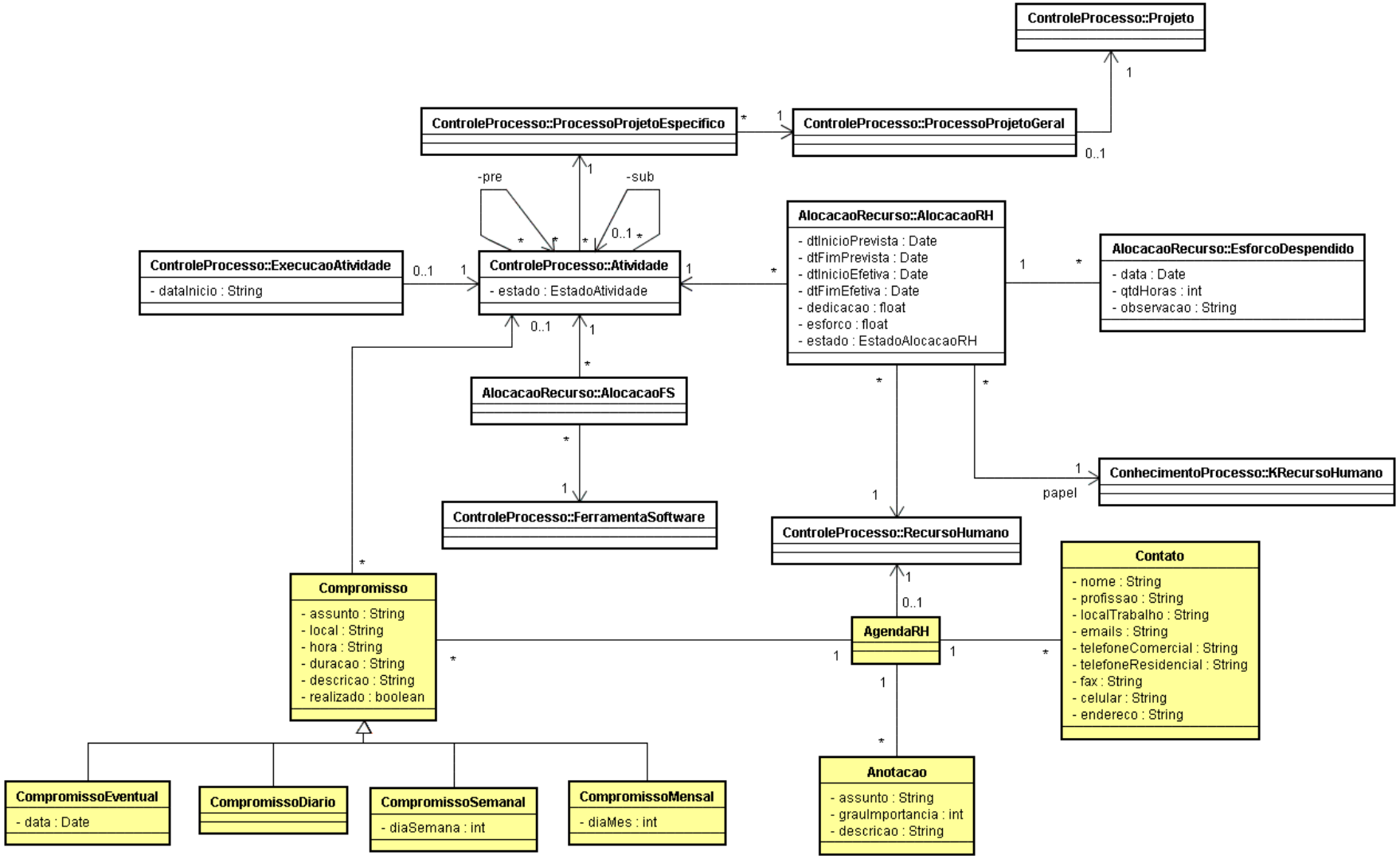

Figura B.7 – Componente de Domínio do Problema do pacote *Agenda*

Em termos de projeto, poucas mudanças foram feitas em relação ao modelo de classes do documento de análise. Além dos tipos de atributos e das navegabilidades entre as classes inseridos no modelo, uma alteração que merece destaque é a criação das classes `EstadoAlocacaoRH` e `EstadoAtividade`, apresentadas no diagrama da Figura B.8, para representarem, respectivamente, os estados dos objetos das classes `AlocacaoRH` e `Atividade`, conforme discutido no Documento de Especificação de Projeto da ferramenta de Alocação de Recursos. É importante saber que essas classes não são persistentes e têm seus valores constantes, definidos pelos respectivos diagramas de estados, sendo que o diagrama de estados da classe `Atividade` pode ser encontrado em (SEGRINI, 2007).

Para cada estado da classe `AlocacaoRH`, mostrado no respectivo diagrama de estado, foi criada uma constante que, na linguagem em Java, é um atributo público (*public*, representado no diagrama pelo símbolo '+'), estático (*static*, representado no diagrama pelo sublinhado) e final, informando que o valor do atributo não pode ser alterado (não existe representação gráfica correspondente para indicar essa propriedade).

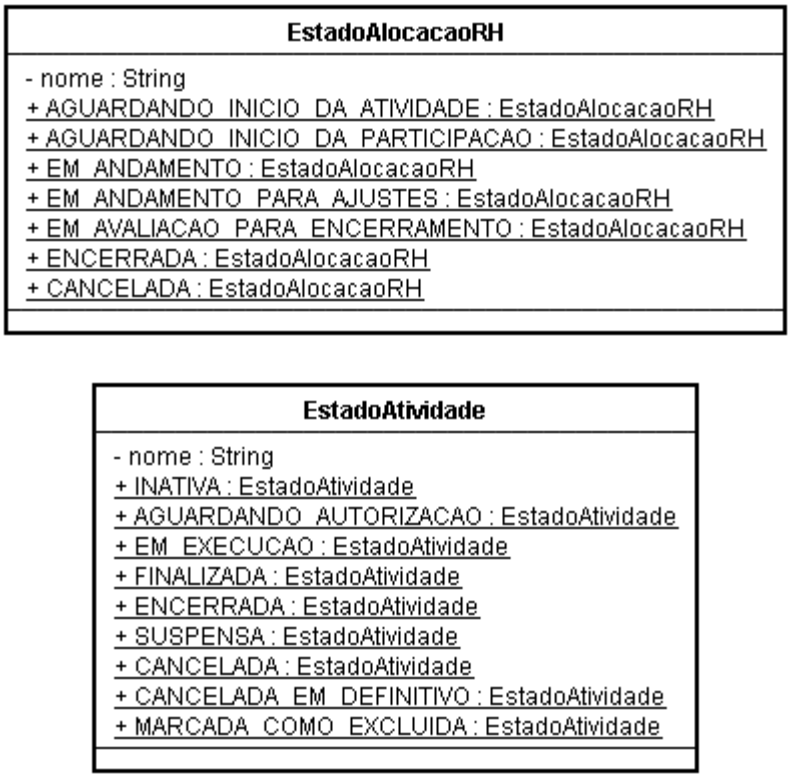

Figura B.8 – Classes de Estados

B.3.3.2 – Componente de Gerência de Dados (Cgd)

As classes do Componente de Gerência de Dados têm como objetivo abstrair o acesso ao banco de dados usado na implementação, dando maior flexibilidade e transparência ao sistema.

Para desenvolver a camada de persistência de ODE, são utilizados os padrões Objeto de Acesso a Dados (*Data Access Object* - DAO) (SUN, 2001) e Fábrica Abstrata (GAMMA et al., 1995), incluindo o uso do *framework* de persistência Hibernate (BAUER et al., 2005). O padrão DAO fornece uma interface flexível entre aplicações e os mecanismos de persistência (banco de dados, arquivo, memória etc) e, portanto, o padrão torna-se parte integrante da camada de persistência. A Figura B.9 mostra as classes de utilitário da persistência de ODE.

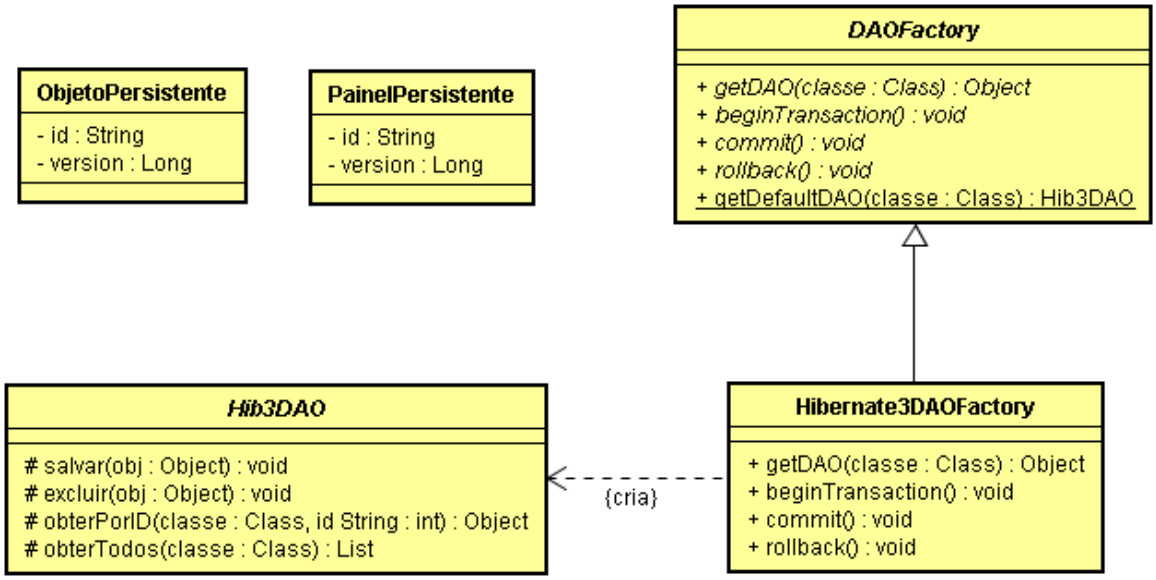

Figura B.9 - Pacote *Utilitario::Persistencia::hibernate*

Classes de domínio e itens gráficos que devem ser persistidos devem herdar de `ObjetoPersistente` e `PainelPersistente`, respectivamente. Além disso, cada classe DAO correspondente a uma classe do domínio do problema deve herdar da classe `Hib3DAO`, que implementa os métodos básicos de manipulação de objetos, a saber *salvar*, *excluir*, *obterPorId* e *obterTodos*, usando o Hibernate. A Figura B.10 apresenta o diagrama de classes do pacote Cgd da ferramenta Agenda.

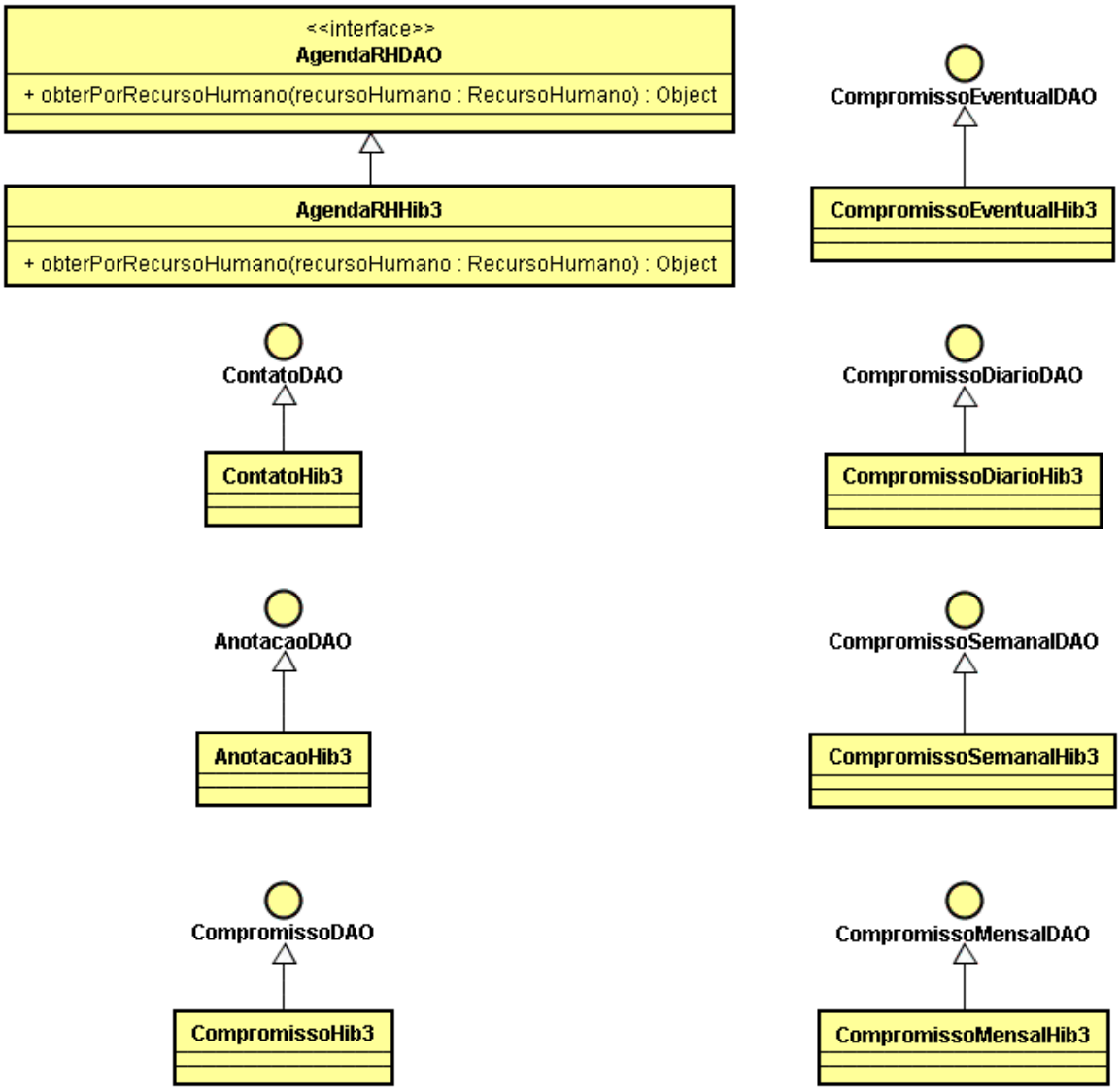

Figura B.10 - Componente de Gerência de Dados do pacote *Agenda*

Conforme apontado anteriormente, todas as classes apresentadas na Figura B.10, além de implementarem as interfaces mostradas, herdam da classe abstrata `Hib3DAO`. No caso da classe `AgendaRHHib3` foi necessário adicionar um método auxiliar para recuperar a agenda de um recurso humano (método *obterPorRecursoHumano*), que retorna o objeto da classe `AgendaRH` que se relacionam com um objeto `RecursoHumano` específico. Isso é necessário, porque na associação entre `AgendaRH` e `RecursoHumano` optou-se apenas pela navegabilidade no sentido `AgendaRH` → `RecursoHumano`, conforme mostrado na Figura B.7.

B.3.3.3 – Componente de Gerência de Tarefas (Cgt)

O Componente de Gerência de Tarefas desse pacote possui as classes `AplAgenda`, `AplControlarCompromisso`, `AplCadastrarContato` e `AplControlarAnotacao`. A classe `AplAgenda` trata as funcionalidades relativas à seleção e visualização de alocações de recursos humanos (caso de uso "Visualizar Alocações de Recurso Humano a Atividades"), seleção da ferramenta de trabalho (caso de uso "Selecionar Ferramenta de Trabalho") e visualização das informações diárias do desenvolvedor (caso de uso "Visualizar Informações Diárias"). Já as classes `AplControlarCompromisso`, `AplCadastrarContato` e `AplControlarAnotacao`, são responsáveis, respectivamente, pelo cadastro de compromissos (caso de uso "Controlar Compromisso"), contatos (caso de uso "Cadastrar Contato") e anotações (caso de uso "Controlar Anotação") de um desenvolvedor. A Figura B.11 mostra o diagrama de classes do Componente de Gerência de Tarefas desse pacote.

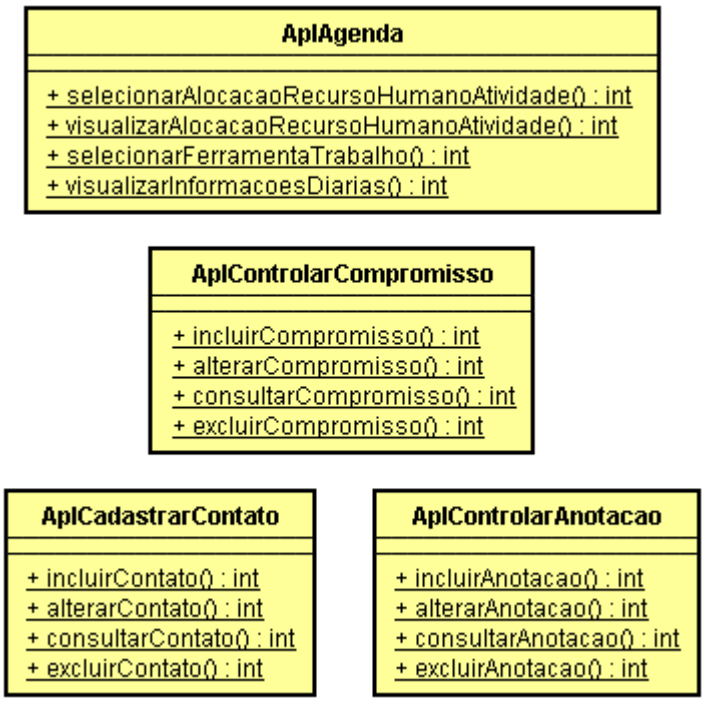

Figura B.11 - Componente de Gerência de Tarefas do Pacote *Agenda*

Pela Figura B.11, pode-se observar que nenhuma classe do Cgt possui relacionamentos diretos com objetos dos pacotes Cdp e Cgd. Porém, objetos dos pacotes

Cdp e Cgd são instanciados dentro dos métodos das classes do Cgt, o que explica o relacionamentos de dependência entre os pacotes Cdp, Cgd e Cgt mostrados na Figura B.6.

Além disso, as classes do Cgt possuem apenas métodos estáticos e, por isso, é necessário informar ao controlador se um método foi executado corretamente ou não. Para resolver tal problema, foram definidos tipos dos retornos padrão dos métodos das classes do Cgt do ambiente ODE, sendo eles números inteiros, representados por constantes, conforme apresentado na Tabela B.T.1.

Tabela B.T.1 – Tabela de Constantes de Retorno Padrão das Aplicações de ODE

Constante	**Valor**	**Descrição**
SUCESSO	0	Indica que a operação foi realizada com sucesso.
ERRO_PERSISTENCIA	1	Indica que ocorreu algum erro ao tentar atualizar dados.
ERRO_CONCORRENCIA	2	Indica um erro decorrente do acesso concorrente aos dados.
ERRO_INESPERADO	3	Indica erros que não são tratados na aplicação. Essa definição é importante, pois, mesmo sem saber o tipo de erro ocorrido, é possível tratá-lo, evitando que ele acarrete maiores problemas para o usuário, como a indisponibilidade do sistema.

Além dos tipos de retornos padrão do ambiente, cada ferramenta desenvolvida para o ambiente ODE pode necessitar de definir tipos de retornos específicos para mesma. No caso da ferramenta deste trabalho, não foi necessário definir nenhum outro tipo de retorno específico.

B.3.3.4 – Componente de Interação Humana (Cih)

O Componente de Interação Humana do pacote *Agenda* define as interfaces que interagem com o usuário para a realização dos casos de uso definidos no documento de especificação de requisitos. Para a implementação desse componente foram seguidos os novos padrões de interface para ferramentas internas do ambiente ODE. A Figura B.12 mostra o componente de interação humana relativo ao pacote *Agenda.*

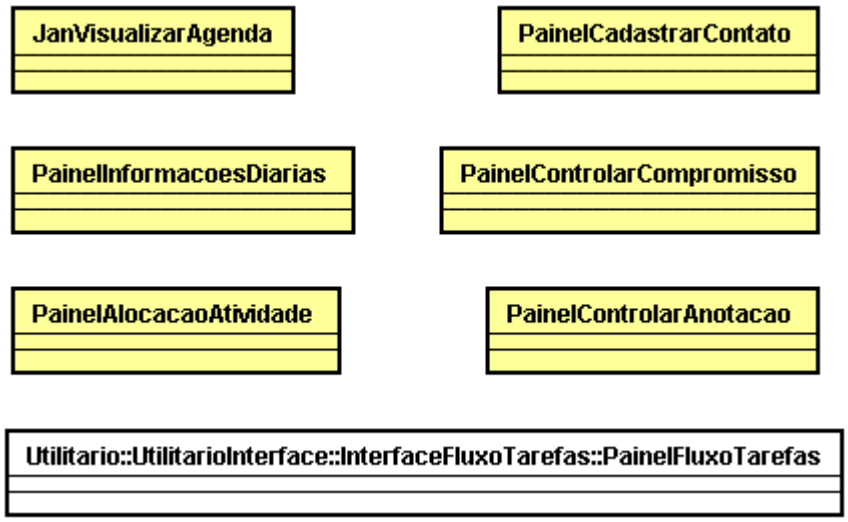

Figura B.12 - Componente de Interface Humana do pacote *Agenda*

A ferramenta é possui uma única janela (`JanVisualizarAgenda`) que comporta o `PainelFluxoTarefas`. Esse painel agrupa e organiza a ordem de exibição dos painéis em tela, conforme o novo padrão de interface desenvolvido para controlar o fluxo de tarefas exercidas pelos usuários em uma ferramenta do ambiente ODE, que se encontra mais detalhado no Documento do Utilitário de Interface Gráfica para Fluxos de Tarefas

O `PainelInformacoesDiarias` apresenta para o desenvolvedor as principais informações diárias sobre suas alocações no ambiente ODE, seus compromissos e suas anotações, tratando do caso de uso *Visualizar Informações Diárias*. Já o `PainelAlocacaoAtividade` apresenta as alocações a atividades que o desenvolvedor possui no ambiente ODE, sendo ainda possível selecionar ferramentas de software disponíveis, registrar esforços despendidos e solicitar encerramento da participação na

atividade, conforme a descrição dos casos de uso *Visualizar Alocações de Recurso Humano a Atividades*, *Selecionar Ferramenta de Trabalho* e *Registrar Esforço Despendido* (ferramenta *AlocaODE*) e o evento *Solicitar Encerramento da Participação na Atividade* do caso de uso *Controlar Andamento de Alocação de Recurso Humano a Atividade* (ferramenta *AlocaODE*) .

Os painéis `PainelControlarCompromisso`, `PainelControlarAnotacao` e `PainelCadastrarContato` mostram para o desenvolvedor funcionalidades relacionadas ao controle de compromissos, cadastro de contatos e cadastro de anotações, conforme a descrição dos casos de uso *Controlar Compromisso*, *Controlar Anotação* e *Cadastrar Contato.*

Conforme discutido na próxima seção, um objeto controlador é responsável pela exibição de todas as interfaces do sistema, o que explica a inexistência de relacionamentos entre os objetos do Cih.

B.3.3.5 – Componente de Controle de Interação (Cci)

O Componente de Controle de Interação tem como objetivo fazer a comunicação de dados entre os objetos da interface da ferramenta (Cih) e as classes de aplicação (Cgt), como mostra a Figura B.13.

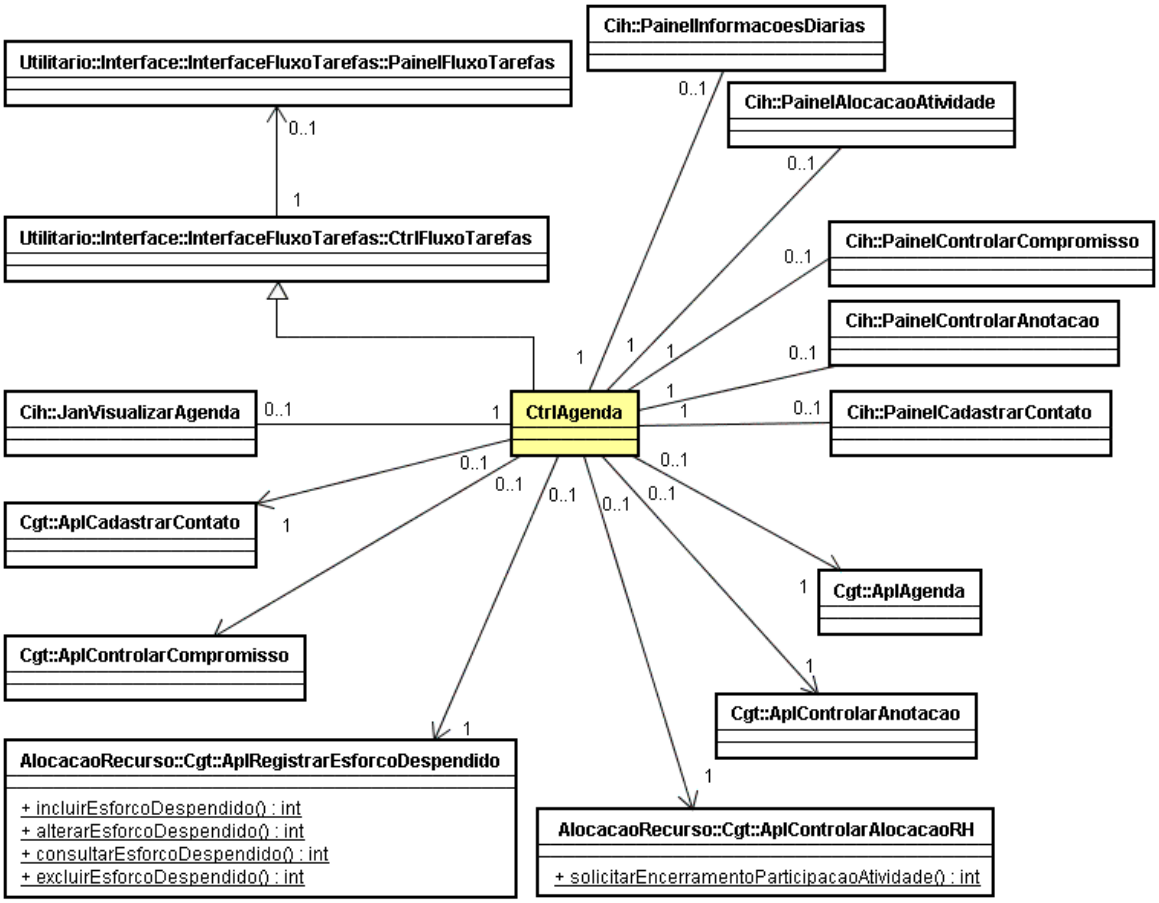

Figura B.13 - Componente de Controle de Interação do Pacote *Agenda*

A navegabilidade no sentido de `CtrlAgenda` para as classes do Cgt, tanto da ferramenta Agenda quanto da ferramenta de Alocação de Recursos, é justamente feita para permitir que as funcionalidades da ferramenta possam ser utilizadas por outras ferramentas do ambiente ODE e, caso exista necessidade de modificar a interface, apenas o controlador deve ser reformulado.

Quando uma interface solicita a realização de uma funcionalidade de uma classe de aplicação (classe do Cgt), essa é feita por meio do controlador (`CtrlAgenda`), sendo que este conhece todas as operações necessárias para atender às requisições das interfaces. Após a aplicação ser requisitada, a aplicação retorna um resultado para o controlador que o interpreta e atualiza as interfaces de acordo com as necessidades.

Conforme mostra a Figura B.13, a classe `CtrlAgenda` herda de `CtrlFluxoTarefas`, que é um controlador especial para controle de fluxo de tarefas. Ele

possui um componente de interface, `PainelFluxoTarefas`, que implementa vários métodos para auxiliar o agrupamento e a ordem de exibição de outros painéis em tela. Mais detalhes sobre essa interface pode ser visto no Documento do Utilitário de Interface Gráfica para Fluxos de Tarefas.

Apesar dos métodos não estarem explícitos no diagrama da Figura B.13, o controlador possui vários métodos, cujas principais funcionalidades são de atualização das interfaces e de requisições às aplicações.

Vale a pena destacar que somente os métodos utilizados pela ferramenta Agenda das classes `AplAlocacaoRecurso` e `AplRegistrarEsforcoDependido` da ferramenta de Alocação de Recursos estão explícitos na Figura B.13. No Documento de Especificação de Projeto da ferramenta de Alocação de Recursos é possível ver todos os métodos dessas classes.

Anexo C

Documento do Utilitário de Interface Gráfica para Fluxos de Tarefas

C.1 Introdução

Apesar de um software ser reconhecido principalmente pelas suas funcionalidades, um outro fator de grande importância que também se destaca é a usabilidade do mesmo, que, se for alta, proporciona uma maior satisfação dos clientes.

Um aspecto altamente relacionado com a usabilidade de um sistema é a interface com o usuário. Essa deve ser construída de modo que toda interação com o usuário seja feita de uma maneira bem natural, proporcionando maior facilidade na utilização do software.

Este documento tem como objetivo apresentar um utilitário de interface gráfica orientada a tarefas, desenvolvido para ser utilizado por ferramentas do ambiente ODE, caso essas necessitem. A Seção C.2 apresenta o projeto desse utilitário, descrevendo sucintamente as principais classes envolvidas. A Seção C.3 apresenta o layout da interface. A Seção C.4 trata de como utilizar esse utilitário. Por fim, a Seção C.5 apresenta algumas considerações finais.

C.2 Projeto

Nesta seção é apresentado o diagrama de classes referente ao utilitário de interface gráfica para fluxo de tarefas, bem como a definição das principais classes envolvidas. A Figura C.1 apresenta o diagrama de classes da interface. Como se pode perceber, em relação à arquitetura padrão do ambiente ODE, dois pacotes são tratados neste utilitário: o Componente de Interação Humana (CIH) e o Componente de Controle de Interação (CCI).

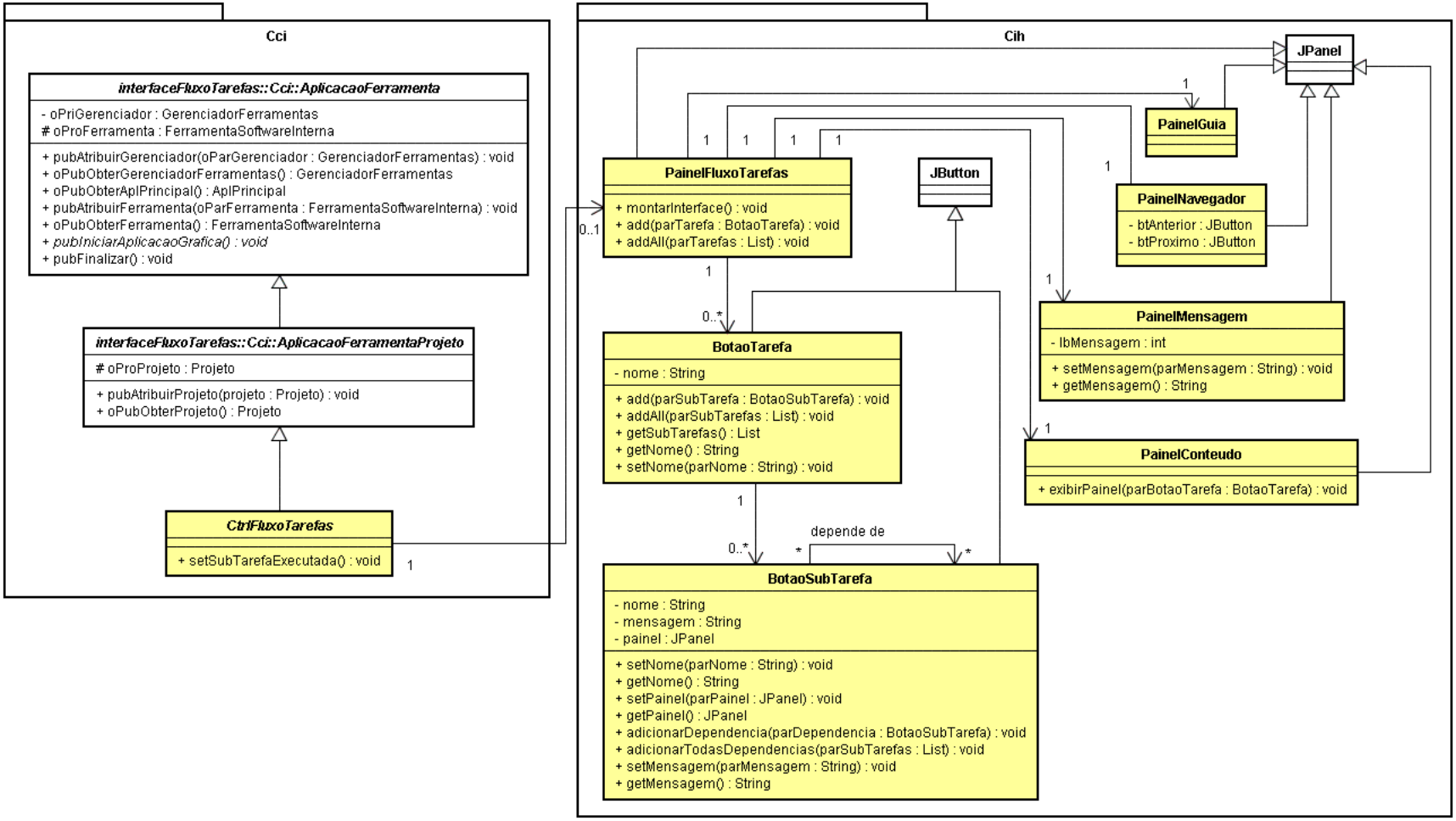

Figura C.1 – Diagrama de classes do Utilitário de Interface

As classes `CtrlFluxoTarefas` e `PainelFluxoTarefas` representam, respectivamente, o controlador de interface desse utilitário e o painel responsável por exibir e controlar o fluxo de tarefas da interface. É importante lembrar que a classe `CtrlFluxoTarefas` herda da classe `AplFerramentaProjeto`, que por sua vez herda de `AplFerramenta`. Por isso, qualquer classe que herdar da classe `CtrlFluxoTarefas` precisará implementar os métodos abstratos da classe `AplFerramenta`.

As classes `PainelMensagem` e `PainelConteudo` servem para exibir a mensagem e o painel associados ao `BotaoSubTarefa`, respectivamente. A classe `PainelGuia` é responsável por exibir os botões das tarefas e subtarefas. Já o `PainelNavegador` é responsável por exibir os botões que permitem navegar pelas subtarefas.

Como a idéia principal deste utilitário de interface gráfica é controlar o fluxo de tarefas, as classes `BotaoTarefa` e `BotaoSubTarefa` representam, respectivamente, uma tarefa e uma subtarefa na interface.

Fazendo uma analogia, o `PainelFluxoTarefas` pode ser comparado ao componente Java `JMenuBar`. `BotaoTarefa` atua como se fosse um `JMenu` e `BotaoSubTarefa` como um `JMenuItem`.

Seja o seguinte exemplo: para se criar uma barra de menu com um menu *Arquivo* e um item *Novo* nesse menu, teria de ser implementado o seguinte código:

```
JMenuBar barraMenu = new JMenuBar();
JMenu menuArquivo = new JMenu ("Arquivo");
JMenuItem menuItemNovo = new JMenuItem("Novo");
barraMenu.add(menuArquivo);
menuArquivo.add(menuItemNovo);
```

O mesmo exemplo poderia ser implementado utilizando o `PainelFluxoTarefas`, conforme abaixo.

```
PainelFluxoTarefas painel = new PainelFluxoTarefas();
BotaoTarefa tarefaArquivo = new BotaoTarefa("Arquivo");
BotaoSubTarefa subTarNovo = new BotaoSubTarefa("Novo", pnelSubTar);
painel.add(tarefaArquivo);
tarefaArquivo.add(subTarefaNovo);
```

Analisando os dois trechos de código, pode-se perceber que a única diferença está no construtor de `BotaoSubTarefa`, que possui um parâmetro `pnelSubTar` representando o painel associado à subtarefa. Esse painel deve ser criado previamente e passado como parâmetro no construtor. Além disso, qualquer controlador desenvolvido herdando de `CtrlFluxoTarefas` deve conhecer os painéis específicos das subtarefas e vice-versa. Ou seja, deve haver associações com navegabilidade dupla entre esse controlador e os painéis das subtarefas.

Por fim, é válido ressaltar que as classes `PainelFluxoTarefas`, `PainelNavegador`, `PainelMensagem`, `PainelConteudo` e `PainelGuia` e as classes `BotaoTarefa` e `BotaoSubTarefa` herdam, respectivamente, das classes `JPanel` e `JButton`, sendo essas componentes gráficos da linguagem Java. Além disso, no diagrama apresentado, apenas os principais atributos e métodos foram mostrados. Algumas classes de painéis que foram utilizadas apenas para organizar a estrutura da interface não foram mostradas. São elas: `PainelInferior`, `PainelEsquerda`, `PainelDireita`, `PainelSubTarefas`, `PainelTarefasExecutadas` e `PainelTarefasNaoExecutadas`.

C.3 Implementação

Esta seção apresenta a interface do utilitário de controle de fluxo de tarefa, bem como o funcionamento do mesmo. A Figura C.2 mostra esse utilitário aplicado à ferramenta de Gerência de Riscos do ambiente ODE.

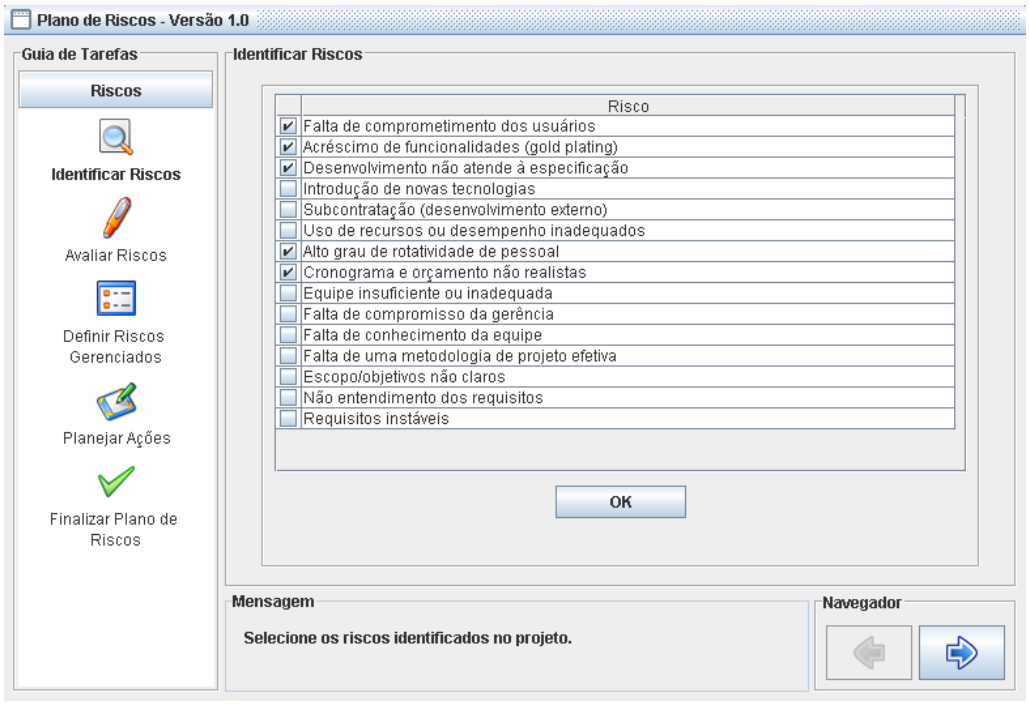

Figura C.2 - Utilitário de Interface Aplicado à Ferramenta de Gerência de Riscos

Na figura C.2, existe apenas uma tarefa (Riscos) com cinco subtarefas (Identificar Riscos, Avaliar Riscos, Definir Riscos Gerenciados, Planejar Ações, Finalizar Plano de Riscos). Cada `BotaoSubTarefa` é interligada a um painel, que representa a interface da subtarefa a ser executada, e a uma mensagem de informação, sendo esses sempre exibidos quando a subtarefa é selecionada.

No exemplo da Figura C.2, o `BotaoSubTarefa` "Identificar Riscos" está associado ao painel no qual os riscos são identificados e com a mensagem "Selecione os riscos identificados no projeto". Tanto o painel quanto a mensagem são criados por quem utiliza a interface e eles são apenas passados como parâmetros para serem utilizados na mesma.

A Figura C.3 explica cada detalhe da interface implementada.

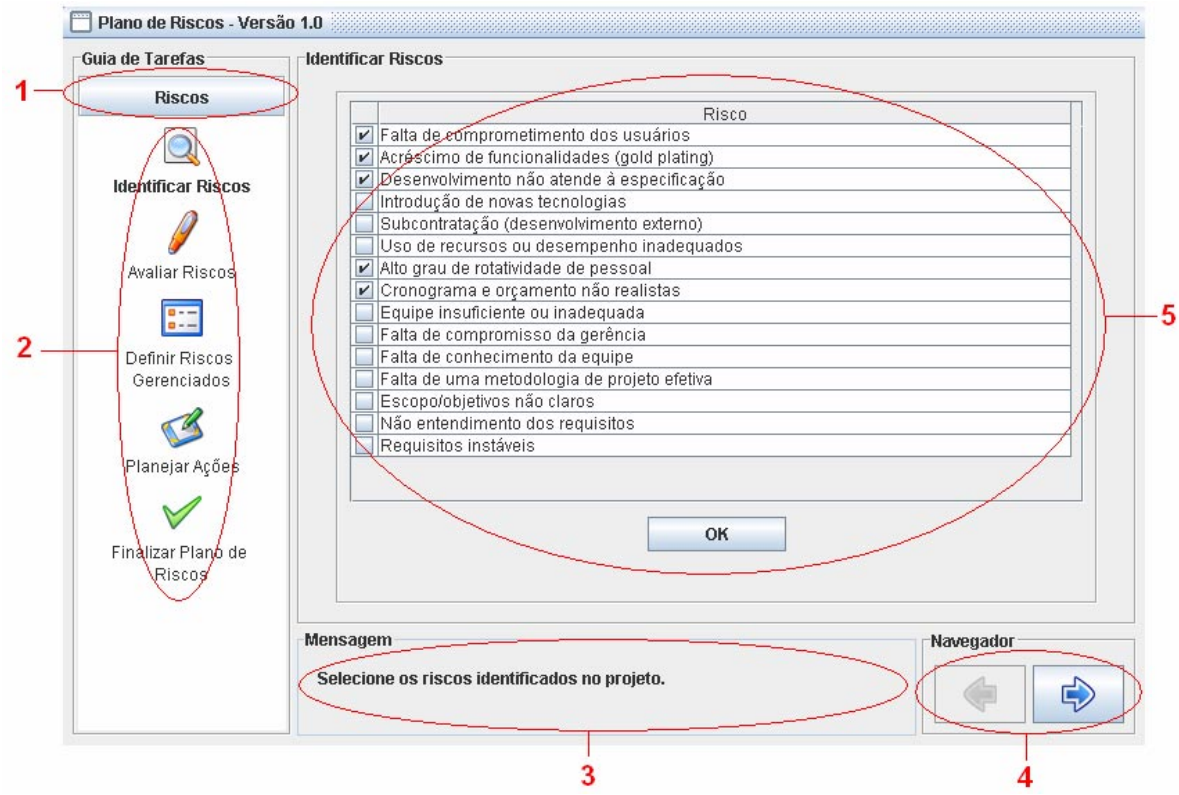

Figura C.3 – Detalhes da Interface Implementada

Os números na figura apontam para informações importantes da interface, conforme descrito abaixo:

1. Botões de Tarefas: representam as tarefas do processo sendo suportado pela interface, sendo que cada tarefa possui um conjunto de subtarefas.
2. Botões de Subtarefas: representam as subtarefas de uma tarefa, para as quais há painéis e mensagens específicos.
3. Painel de Mensagem: representa o local onde é exibida a mensagem associada à subtarefa selecionada.
4. Painel Navegador: tem como funcionalidade navegar entre as subtarefas da interface. A navegação é feita entre todas as subtarefas habilitadas para serem executadas, desde a primeira subtarefa da primeira tarefa até a última subtarefa última tarefa.
5. Painel Conteúdo: representa o local onde é exibido o painel associado à subtarefa selecionada.

C.4 Utilização

Esta seção mostra um passo-a-passo para se utilizar o utilitário de interface gráfica para fluxo de tarefas.

Para construir uma interface baseada nesse utilitário, é necessário efetuar as seguintes etapas, ilustradas com o exemplo da Figura C.3:

1. Defina quais serão as tarefas e subtarefas do sistema, projetando os respectivos painéis e mensagens de cada subtarefa.
2. Implemente os painéis de cada subtarefa.
3. Crie um controlador que herde da classe `CtrlFluxoTarefas`, apresentada no diagrama de classes da Figura 1. No exemplo da gerência de riscos, esse controlador é `CtrlGerenciaRiscos`.
4. Instancie o painel de fluxo de tarefas do controlador criado no passo anterior (`painelFluxoTarefas = new PainelFluxoTarefas();`).
5. Crie as tarefas, sendo que para cada tarefa é necessário passar o nome para o construtor da mesma (no exemplo, `BotaoTarefa riscos = new BotaoTarefa("Riscos");`).
6. Crie as subtarefas, sendo que, para cada subtarefa, é necessário passar como parâmetros o nome e o painel criado no passo 2 (no exemplo, `BotaoSubTarefa identificarRiscos = new BotaoSubTarefa ("Identificar Riscos", painelIdentificarRiscos);`)
7. Relacione uma mensagem a cada subtarefa, utilizando o método *setMensagem* e passando a mensagem como parâmetro (no exemplo, `identificarRiscos.setMensagem("Selecione os riscos identificados no projeto");`).
8. Inclua todas as subtarefas na tarefa, conforme projetado no passo 1. Para incluir uma subtarefa em uma tarefa, basta utilizar o método *add* da tarefa, passando a subtarefa como parâmetro (no exemplo: `riscos.add(identificarRiscos);`). Faça isso para todas as subtarefas.

9. Inclua todas as tarefas no painel de fluxo de tarefas instanciado no passo 4. Para tal, basta utilizar o método *add* do painel para cada tarefa, passando a tarefa como parâmetro (no exemplo, `painelFluxoTarefas.add(riscos);`).
10. Monte a interface, utilizando o método do painel de fluxo de tarefas *montarInterface* `(painelFluxoTarefas. montarInterface ());`

A Figura C.4 apresenta o modelo de classes do exemplo de utilização da interface na ferramenta de Gerência de Riscos.

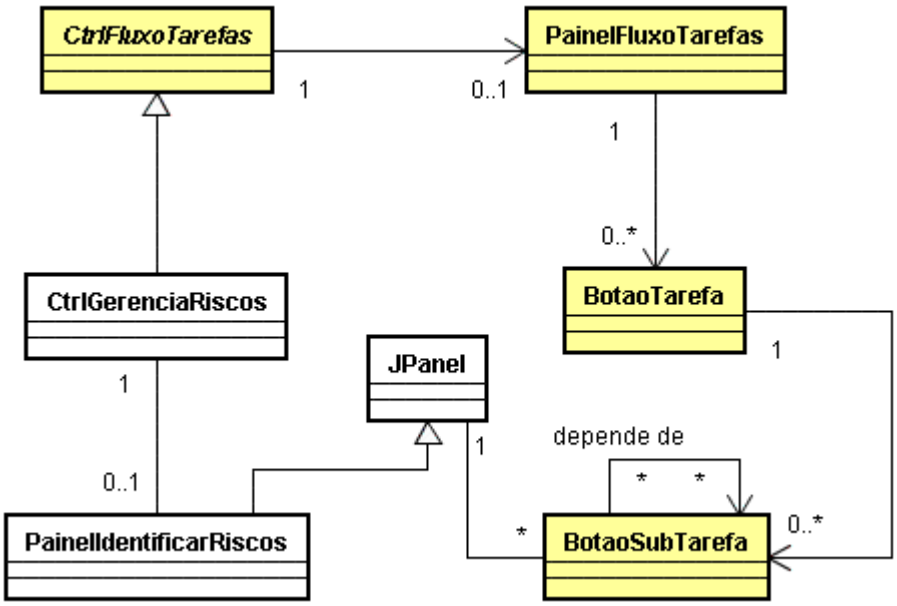

Figura C.4 – Modelo da Utilização da Interface

Pequenas considerações devem ser feitas à respeito da construção da interface. O passo 7 é opcional. Caso não seja efetuado, nada será exibido no painel de mensagem.

Outro passo opcional, não mostrado anteriormente, é a utilização de dependências entre as subtarefas, ou seja, nos casos em que uma subtarefa só pode ser executada se todas suas subtarefas tiverem sido executadas. Para isso, é necessário utilizar o método *setDependencias* para cada subtarefa, passando como parâmetro uma lista de subtarefas das quais a subtarefa depende e, sempre que uma subtarefa for executada, é necessário informar ao controlador que a

mesma foi executada, utilizando o método *setSubTarefaExecutada* da classe `CtrlFluxoTarefas`. Sendo assim, a subtarefa atual é atribuída como executada.

Além disso, como as classes `BotaoTarefa` e `BotaoSubTarefa` herdam da classe `JButton`, qualquer método desta última pode ser executado pelas mesmas, como por exemplo, alterar o ícone de uma subtarefa.

C.5 Considerações Finais

O utilitário de interface gráfica para fluxo de tarefas proporciona, de uma maneira geral, maior usabilidade para sistemas que possuem uma série de passos ou tarefas a serem executadas, tornando-as mais amigáveis.

Antes de utilizar a interface, o desenvolvedor deve fazer uma análise se esse tipo de interface realmente cabe à sua aplicação, pois, como já dito, esta se comporta especialmente bem em casos em que ocorrem fluxos de tarefas a serem executadas.

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

CENTRO TECNOLÓGICO
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ALEXANDRE GUILHERME NICCO COELHO

Apoio à Gerência de Recursos em ODE

Vitória
2007