3

O Framework IssueNet

“Depois que conhece uma nova idéia, a mente do homem nunca pode voltar a suas dimensões originais.”
(Oliver Wendell Holmes Jr.)

PUC-Rio - Certificação Digital Nº 0521485/CA

Neste capítulo, serão apresentados inicialmente conceitos gerais sobre Frameworks e, em seguida, o Framework IssueNet. Primeiramente, seu escopo será definido e suas características, apresentadas. Em seguida, sua arquitetura será detalhada, com auxílio dos diagramas elaborados durante a fase de modelagem do sistema. Finalmente, será feita uma análise dos pontos flexíveis do Framework, detalhando as instâncias utilizadas nos estudos de caso desta pesquisa, que serão apresentados com mais detalhes no capítulo 4.

3.1.

Conceitos Gerais

A tecnologia de Frameworks possibilita que uma família de produtos seja gerada a partir de uma única estrutura que captura os conceitos mais gerais da família de aplicações (Pinto, 2000). A literatura apresenta várias definições de Frameworks. Nesta dissertação, será adotada a definição de (Buschmann *et al*., 1996, Pree, 1995) e Pinto, 2000): um Framework pode ser definido como um software parcialmente completo, projetado para ser instanciado. Um Framework define uma arquitetura para uma família de subsistemas, oferecendo os construtores básicos para a sua criação. Devem ser explicitados os pontos de extensão (*hot-spots*), nos quais devem ser realizadas adaptações do código de acordo com o funcionamento específico de cada módulo instanciado.

O uso de Frameworks no desenvolvimento traz uma série de benefícios. Primeiramente, diminuem o esforço de entendimento e manutenção da aplicação. Além disso, incentivam o reuso, já que capturam o conhecimento de desenvolvedores em um determinado domínio. Frameworks também são expansíveis por construção e, finalmente, o uso de inversão de controle, presente em alguns Frameworks, simplifica o desenvolvimento de aplicações. Entretanto, na sua utilização também são encontrados desafios. Dentre eles, destacam-se o esforço de desenvolvimento, a curva de aprendizagem, a eficiência e a manutenção, além do desafio da falta de padrões (Barreto, 2006).

Este capítulo apresentará o Framework proposto nesta dissertação. Na próxima seção, os módulos do Framework IssueNet serão detalhados.

3.2.

Módulos do IssueNet

O IssueNet apresenta dois módulos principais: o módulo Gerenciador de Tarefas e o módulo Gerenciador de Usuários. Enquanto o primeiro engloba todas as funcionalidades de gerência de tarefas e avaliações do sistema, o último é responsável pela gerência de usuários, grupos e permissões.

PUC-Rio - Certificação Digital Nº 0521485/CA

3.2.1.

Módulo Gerenciador de Tarefas

O módulo Gerenciador de Tarefas é o principal módulo do Framework. Ele é composto por uma implementação de tarefa genérica e operações de criação, alocação, resolução e avaliação de tarefas. Cada tarefa pode pertencer a um grupo de tarefas e tem associada uma data de início e uma data prazo. As instâncias, se necessário, podem extender os grupos de tarefa e a tarefa genérica, acrescentando ou sobrescrevendo propriedades e operações necessárias para a sua gerência.

Uma das principais propriedades da tarefa genérica é o status, que informa em qual etapa do ciclo de vida de tarefa ela se encontra. O status de uma tarefa genérica pode assumir um dos seguintes valores:

- ***Aberta:*** Este é o status da tarefa genérica assim que ela é criada por algum usuário. O seu criador deve obrigatoriamente selecionar um responsável e um prazo para a sua resolução;
- ***Resolvida:*** A tarefa genérica passa a ter este status quando o responsável pela sua resolução a resolve, informando o tempo de resolução e um comentário;
- ***Fechada:*** Este status pode ser atribuído a uma tarefa genérica que se encontre no status “Aberta” ou “Resolvida”. Esta alteração de status só poderá ser realizada pelo criador da tarefa genérica, quando este decidir que a tarefa e sua resolução deverão ser fechadas para alterações;
- ***Em avaliação:*** Este status pode ser atribuído a uma tarefa genérica que se encontre no status “Resolvida”. Esta alteração de status só poderá ser realizada pelo criador da tarefa genérica, e permite que qualquer usuário avalie a resolução desta tarefa, através de uma nota e um comentário;
- ***Expirada:*** Este status de tarefa é atribuído automaticamente pelo sistema quando uma tarefa no status “Aberta” ultrapassa a sua data limite de resolução.

A figura 3.1 ilustra o ciclo de vida de uma tarefa genérica no Framework IssueNet:

PUC-Rio - Certificação Digital Nº 0521485/CA

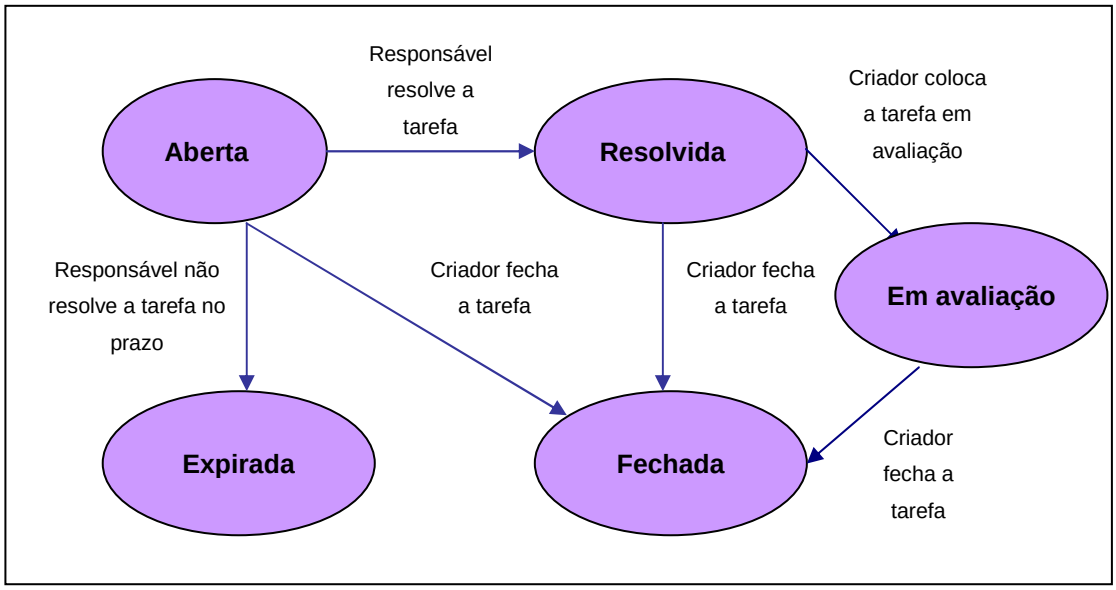

Figura 3.1 - Ciclo de vida de uma tarefa genérica

As operações de gerência de tarefas são realizadas pelo sistema ou por usuários cadastrados no mesmo. Na próxima subseção, o módulo Gerenciador de Usuários será detalhado e relacionado com o módulo Gerenciador de Tarefas, apresentado nesta seção.

3.2.2.

Módulo Gerenciador de Usuários

O módulo de Gerência de Usuários trata os usuários, grupos e permissões do sistema. Apesar de algumas permissões serem sugeridas no Framework, as instâncias podem implementá-las de acordo com as suas regras de negócio. As permissões definidas no Framework são as seguintes:

- ***Criação de Tarefa:*** pode ser realizada por qualquer usuário cadastrado no sistema;
- ***Resolução de Tarefa:*** só pode ser realizada pelo usuário alocado como responsável por ela;
- ***Avaliação de Tarefa:*** pode ser realizada por qualquer usuário cadastrado no sistema, entretanto, para que uma tarefa seja avaliada, ela precisa estar no status “Em avaliação”;

PUC-Rio - Certificação Digital Nº 0521485/CA

- ***Alteração do status da Tarefa para "Fechada":*** só pode ser realizada pelo usuário criador da tarefa, desde que ela esteja no status "Aberta", "Resolvida" ou "Em avaliação";
- ***Alteração do status da Tarefa para "Fechada" e "Em avaliação":*** só pode ser realizada pelo usuário criador da tarefa;
- ***Gerência de usuários:*** pode ser realizada por qualquer usuário cadastrado no sistema;
- ***Gerência de perfis:*** pode ser realizada por qualquer usuário cadastrado no sistema.

Na próxima subseção, serão apresentados componentes complementares aos módulos de Gerência de Tarefas e de Usuários. As instâncias podem implementar estes componentes de acordo com os detalhes do negócio.

PUC-Rio - Certificação Digital Nº 0521485/CA

3.2.3.

Componentes Auxiliares

O Framework também possibilita a integração com componentes, como por exemplo:

- ***Componente de Avaliação:*** Responsável por calcular as notas de cada aluno com base nas avaliações das resoluções de suas tarefas;
- ***Componente de Relatórios:*** Responsável por exibir relatórios sobre as tarefas e usuários do sistema;
- ***Componente de Alertas:*** Responsável por notificar os usuários sobre eventos relacionados às tarefas do sistema.

A figura 3.2 ilustra esquematicamente os módulos e componentes do Framework IssueNet.

Figura 3.2 - Arquitetura do Framework

Na próxima seção, os módulos e componentes do Framework serão detalhados. O Framework terá a sua arquitetura técnica detalhada, e também serão apresentados outros Frameworks que foram usados no desenvolvimento do IssueNet.

3.3.

Arquitetura do Framework

O sistema foi desenvolvido usando o banco de dados *MySQL* (MySQL, 2007) e os Frameworks de infra-estrutura *Hibernate* (Hibernate, 2007) e *Spring*, (Sring, 2007). Para tornar o desenvolvimento mais simples, optou-se pela utilização de *Taglibs* (Taglibs, 2007), na camada de visão, e *Hibernate Annotations*, para relacionar a camada de negócio com a camada de persistência.

O Framework foi implementado usando o modelo MVC (*Model-View-Controller*). O MVC é um padrão arquitetural usado em engenharia de software, que tem o objetivo de separar os dados (*model*) da interface com o usuário (*view*), a fim de que as alterações em uma camada não afetem a outra. Este padrão desacopla o acesso a dados e a lógica de negócio da apresentação de dados e interação com o usuário, através da introdução do componente intermediário *controller* (Gamma *et al*., 2005). O modelo do Framework baseado em MVC é ilustrado pela figura 3.3.

PUC-Rio - Certificação Digital Nº 0521485/CA

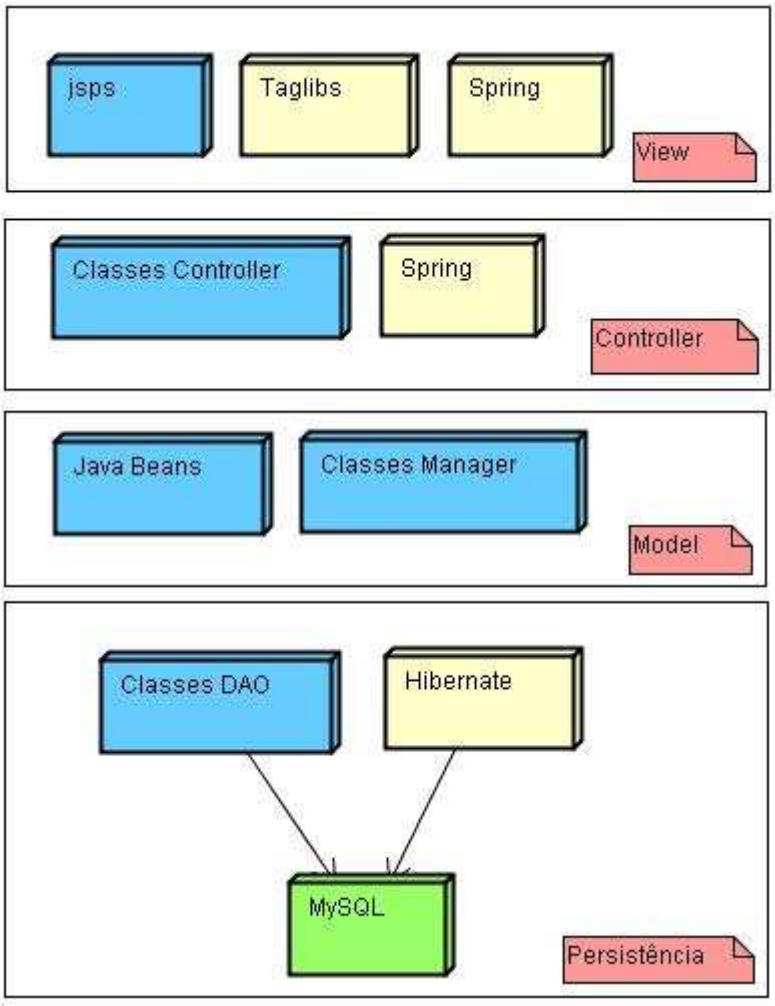

Figura 3.3 - Tecnologias do IssueNet agrupadas por camadas

A camada de apresentação (*view*) concentra os arquivos *jsp* do Framework, e nela são utilizadas *Taglibs* e o Framework *Spring*. As *Taglibs* representam uma biblioteca de *tags* customizadas usadas na composição de páginas JSP e a sua utilização ajuda a eliminar a redundância de código nas páginas. Já o Framework *Spring* oferece os serviços de controle de transações, segurança e exposição de arquivos remotos e tem o objetivo de simplificar o desenvolvimento de aplicações JEE.

O Framework *Spring* também é usado na camada de controle (*controller*) do IssueNet, que concentra as classes *Controller*, que funcionam como mecanismo de ligação entre a camada de negócios e a camada de apresentação. A camada de negócios (*model*) engloba os *beans* da aplicação, que representam entidades como tarefa, perfil e usuário; e as classes *Manager*, responsáveis por executar as operações de negócios do Framework como, por exemplo, criar um usuário ou calcular a nota final de uma tarefa.

PUC-Rio - Certificação Digital Nº 0521485/CA

A camada de persistência, por sua vez, concentra as classes DAO (*Data Access Object*), responsáveis pelo acesso do banco de dados do Framework. Estas classes utilizam o Framework *Hibernate*, que realiza o mapeamento objeto-relacional do Framework, ou seja, traduz as tabelas do banco de dados em objetos Java e vice-versa. Optou-se por utilizar o banco de dados *MySQL*, no Framework e nas instâncias do IssueNet.

Esta seção apresentou as tecnologias utilizadas no desenvolvimento do Framework proposto nesta pesquisa, agrupados por camadas baseadas no modelo MVC. Na próxima seção, serão detalhados os pontos flexíveis do Framework, que possibilitarão a sua instanciação.

3.4. Pontos Flexíveis

O módulo Gerenciador de Usuários do Framework concentra as operações referentes aos usuários do sistema. No Framework, são oferecidas implementações genéricas de usuário e perfil, que podem ser extendidas de acordo com a necessidade das instâncias. Por exemplo, podem ser criador perfis de acesso diferentes para delimitar as responsabilidades de cada papel de usuários, como Professores e Alunos.

Já o módulo Gerenciador de Tarefas concentra as operações referentes as tarefas do sistema. Da mesma forma que no módulo Gerenciador de Usuários, são oferecidas implementações genéricas de tarefa e grupo de tarefa, que podem ser extendidas. Tipos diferentes de tarefas podem ser criados, por exemplo, para possibilitar tarefas de submissão de arquivos e tarefas de preenchimento de questionários.

O Framework já define um ciclo de vida de tarefas, representado pela figura 3.1, que compreende os status ‘Aberta’, ‘Resolvida’, ‘Em avaliação’, ‘Fechada’ e ‘Expirada’. Entretanto, as instâncias podem modificá-lo conforme necessário, adicionando e excluindo status, ou até mesmo alterando a ordem no ciclo de vida dos status já implementados.

PUC-Rio - Certificação Digital Nº 0521485/CA

Outro ponto de extensão do Framework IssueNet é a forma com que as notas recebidas nas avaliações são tratadas. Caso em uma instância seja necessário calcular notas parciais ou médias, ela precisará decidir qual será a sua estratégia, e implementá-la da maneira que melhor lhe convier, através do componente de avaliação.

As instâncias também precisam decidir se irão implementar mecanismos de notificação aos usuários, como notificação por e-mail ou SMS. Caso optem por utilizar notificações, deverão estabelecer os casos e para quais usuários elas deverão ser enviadas. O Framework IssueNet traz uma implementação simples de notificação por e-mail, na qual notificações são enviadas ao criador e responsável por uma tarefa sempre que o seu status é alterado.

O Framework IssueNet sugere alguns relatórios, como relatório de usuários e perfis cadastrados, relatório de avaliações recebidas em uma tarefa, e média dos alunos por tarefa. As instâncias podem implementar outros relatórios de acordo com a sua necessidade.

Esta seção apresentou os pontos flexíveis do Framework, ou seja, aqueles que podem ser extendidos pelas instâncias conforme a sua necessidade. Na próxima seção, os detalhes técnicos do Framework IssueNet serão ilustrados através dos diagramas elaborados durante a fase de modelagem.

PUC-Rio - Certificação Digital Nº 0521485/CA

3.5.

Padrões de Projeto utilizados

Os padrões de projeto, ou *design patterns*, descrevem soluções para problemas recorrentes no desenvolvimento de sistemas software orientados a objeto. Um padrão de projeto define o problema, a solução, quando aplicar esta solução e suas conseqüências. Os padrões de projeto visam facilitar a reutilização soluções na fase de projeto do software. Também acarretam um vocabulário comum de desenho, facilitando comunicação, documentação e aprendizado dos sistemas de software (Gamma *et al.*, 1995).

O Framework IssueNet foi elaborado com base no padrão *Model-View-Controller*, já detalhado na seção 3.3. As próximas subseções mostrarão outros

padrões de projeto utilizados na elaboração do Framework: *Facade*, *Singleton* e *Flyweight*.

3.5.1. Facade

O *design pattern Facade* fornece uma interface unificada para um conjunto de interfaces em um subsistema, ou seja, define uma interface de nível mais alto que facilita a utilização desse subsistema. Sua aplicação é vantajosa para evitar a vinculação rígida entre clientes e subsistemas.

PUC-Rio - Certificação Digital Nº 0521485/CA

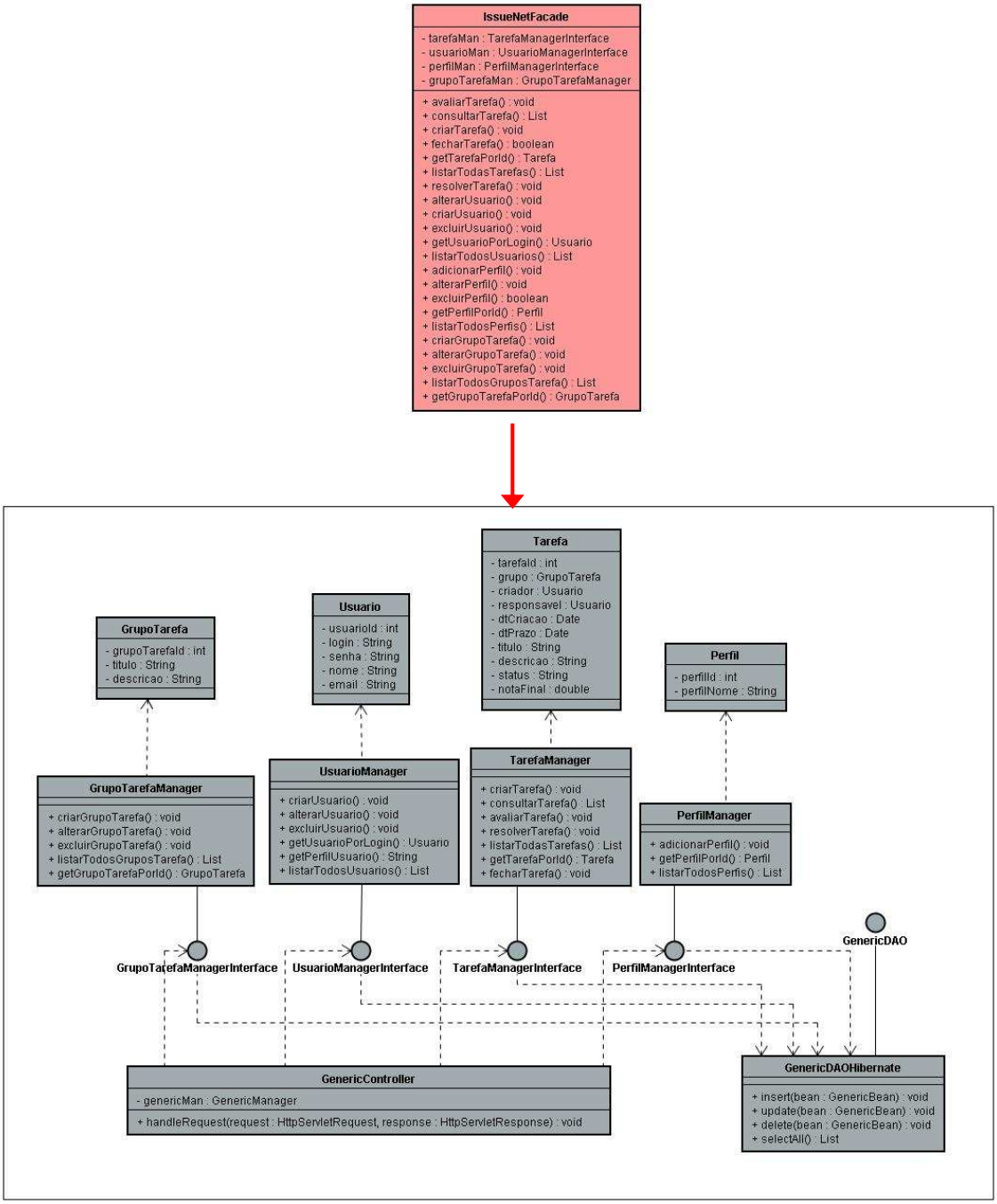

Figura 3.4 – Aplicação do *design pattern Facade*

No Framework IssueNet, este padrão foi utilizado para fornecer um único ponto de acesso comum às funcionalidades do sistema. Desta forma, todas as operações passam inicialmente pelo *Facade*, que é encarregado de encaminhar a solicitação para a classe correspondente. A aplicação do *design pattern Facade* no IssueNet é ilustrada pela figura 3.4. Por questões de legibilidade do diagrama, optou-se por não representar algumas dependências entre as entidades.

PUC-Rio - Certificação Digital Nº 0521485/CA

3.5.2. Singleton

O *design pattern Singleton* garante que uma classe tenha apenas uma instância, e fornece um ponto de acesso global para ela. O Framework Spring, por definição, garante que as implementações contidas no seu *ApplicationContext* sejam *beans singleton*, através dos princípios de Inversão de Controle e Injeção de Dependências.

A Inversão de Controle é uma característica comum aos Frameworks que implementam o fluxo principal de parte do sistema, fornecendo pontos de extensão para o desenvolvedor de aplicação adicionar a lógica específica. Assim, não é a aplicação que 'chama' o framework (como funcionam as bibliotecas), mas sim o framework que 'chama' os módulos da aplicação.

Já a Injeção de Dependências, descreve um modo de gerência de componentes e resolução de dependências onde as dependências dos componentes (recursos como *DataSources*, configurações e outros componentes) são 'injetadas' pelo container na inicialização do componente. Esta inversão faz com que toda a configuração seja feita no container, retirando a necessidade da criação de fábricas customizadas e promove o desacoplamento entre os componentes (Walls & Breidenbach, 2006).

PUC-Rio - Certificação Digital Nº 0521485/CA

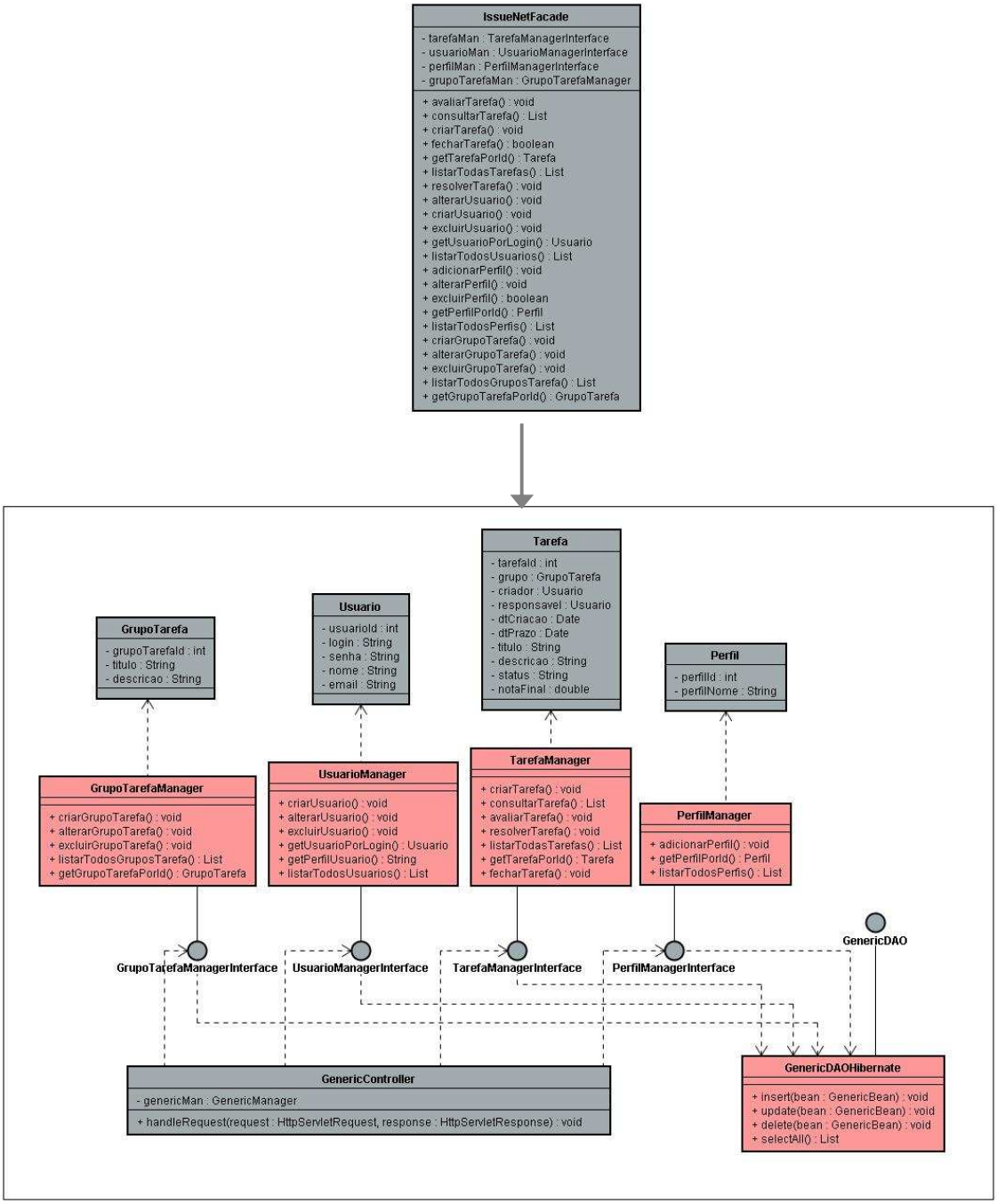

Figura 3.5 – Aplicação do *design pattern Singleton*

O Framework IssueNet implementa vários exemplos do padrão *Singleton*, como por exemplo, as classes *Manager* e *DAO*. A figura 3.5 ilustra estes exemplos. Por questões de legibilidade do diagrama, optou-se por não representar algumas dependências entre as entidades

PUC-Rio - Certificação Digital Nº 0521485/CA

3.5.3. Flyweight

O *design pattern Flyweight* é usado quando uma classe possui muitas instâncias e todas podem ser controladas de maneira idêntica. Este padrão é vantajoso para reduzir o número de instâncias de objetos durante a execução, economizando memória. Além disso, o *Flyweight* centraliza o estado de muitos objetos em um único local.

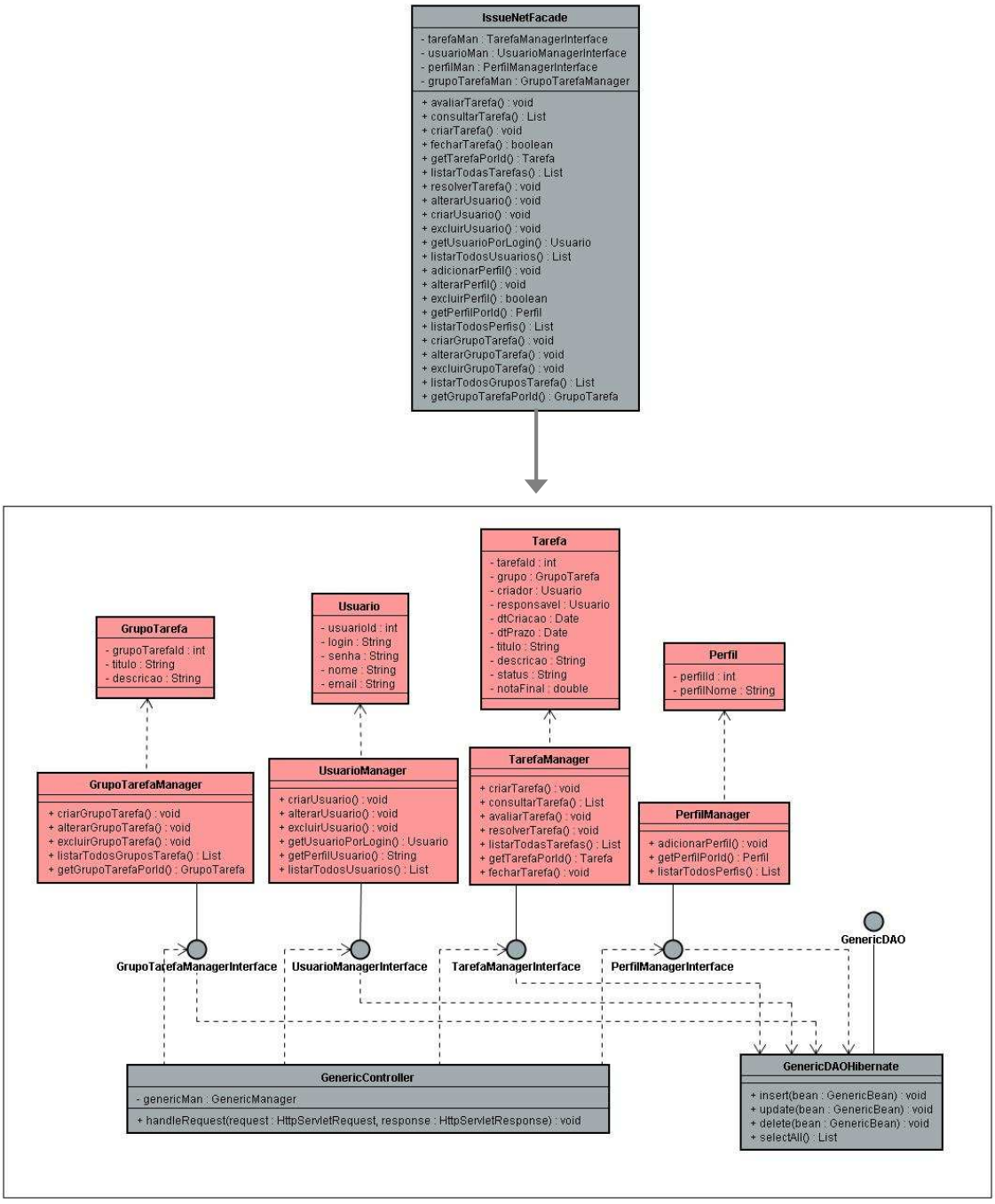

Figura 3.6 – Aplicação do *design pattern Flyweight*

PUC-Rio - Certificação Digital Nº 0521485/CA

No Framework IssueNet, este padrão foi utilizado através das classes *Manager*. Estas classes são responsáveis por centralizar todas as operações de negócio dos *beans* do Framework. A aplicação do *design pattern Flyweight* no IssueNet é ilustrada pela figura 3.6. Por questões de legibilidade do diagrama, optou-se por não representar algumas dependências entre as entidades.

3.6.

Diagramas de Modelagem

Esta seção apresentará os diagramas elaborados durante a fase de modelagem do Framework IssueNet. Inicialmente, será apresentado o digrama de Casos de Uso e, em seguida, o Diagrama de Classes. Finalmente, serão apresentados os diagramas de atividades de um cenário exemplo antes e depois da utilização do Framework.

Em todos os diagramas foi utilizada a linguagem UML (*Unified Modeling Language*), uma especificação da OMG (*Object Management Group*) que busca modelar não só a estrutura, comportamento e arquitetura de uma aplicação, mas também os seus processos de negócio e estrutura de dados (UML, 2007).

PUC-Rio - Certificação Digital Nº 0521485/CA

3.6.1.

Diagrama de Casos de Uso

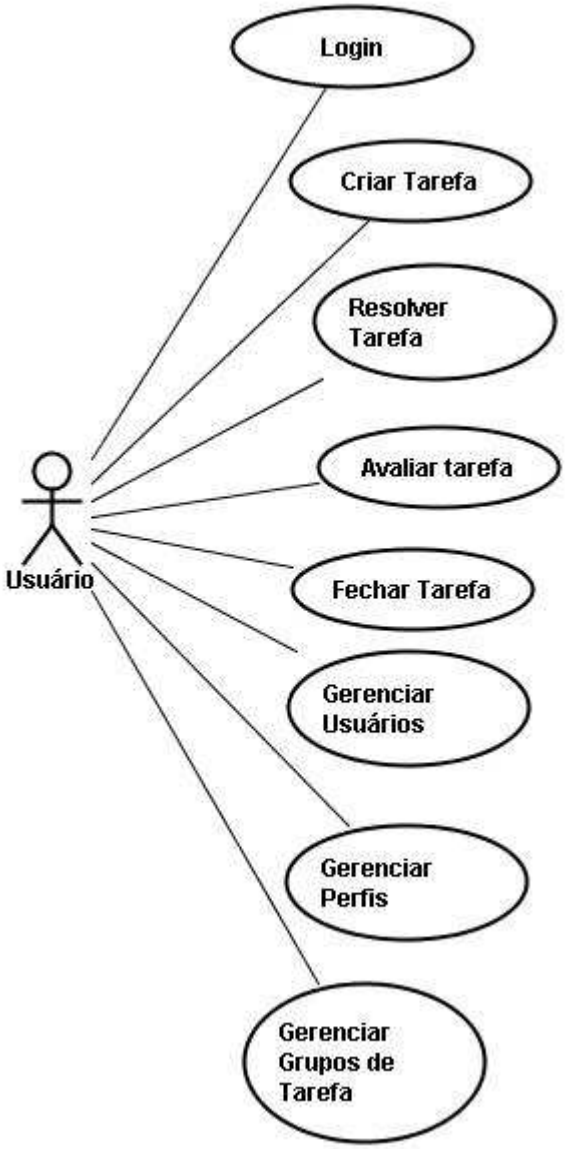

Figura 3.7 - Diagrama de Casos de Uso do Framework

O diagrama de casos de uso, mostrado na figura 3.7, representa as atividades que podem ser realizadas pelos usuários do sistema. No Framework IssueNet, os usuários podem Gerenciar Usuários, Perfis e Grupos de Tarefas; e criar, resolver, avaliar e fechar tarefas.

As ações representadas no diagrama de casos de uso do Framework são implementadas através de entidades – classes e interfaces - do sistema. Na próxima

PUC-Rio - Certificação Digital Nº 0521485/CA

subseção, o relacionamento entre entidades do IssueNet será ilustrado através do diagrama de classes.

3.6.2.

Diagrama de Classes

A figura 3.5 representa um fragmento do diagrama de classes do Framework IssueNet, englobando os módulos Gerenciador de tarefas e Gerenciador de usuários. As entidades em azul representam os pontos flexíveis, e as entidades em vermelho representam um exemplo de instanciação genérica do Framework.

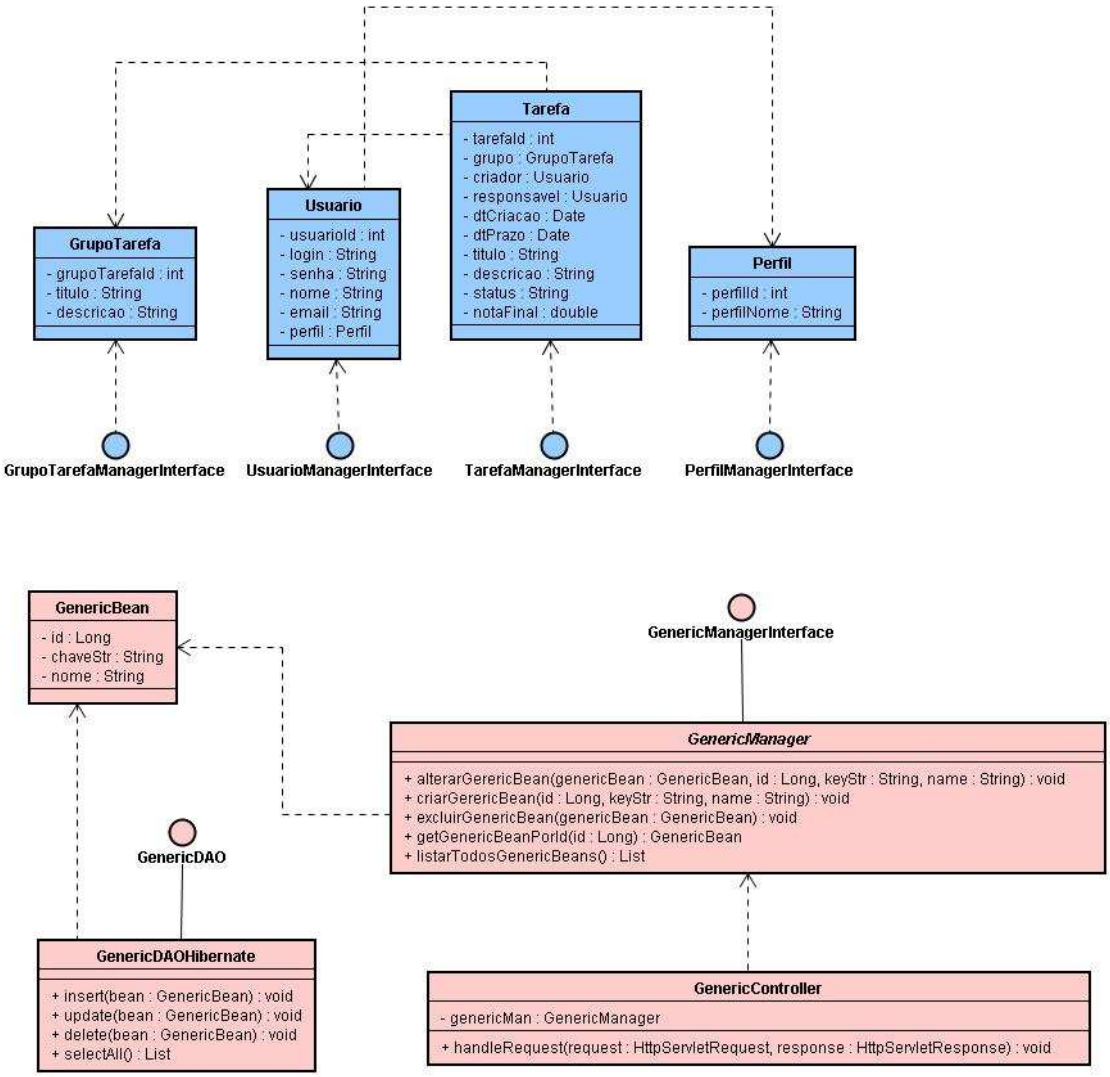

Figura 3.8 – Fragmento do diagrama de Classes do Framework

PUC-Rio - Certificação Digital Nº 0521485/CA

Como mostrado na figura 3.8, a camada *model* do Framework consiste nos *beans GrupoTarefa, Tarefa*, *Usuário* e *Perfil* e nas *Interfaces GrupoTarefaManagerInterface*, *TarefaManagerInterface*, *UsuarioManagerInterface* e *PerfilManagerInterface*. Os dois primeiros *beans* e as duas primeiras interfaces pertencem ao módulo de Gerenciamento de Tarefas, e as demais, ao módulo de Gerenciamento de Usuários, já detalhados na seção 3.2. O componente de avaliação é representado dentro da entidade *Tarefa*, através do atributo *notaFinal*.

Esta subseção mostrou o relacionamento entre as entidades do sistema através do diagrama de classes. Na próxima subseção, serão apresentados dois diagramas de atividades, que representam um exemplo de um possível cenário de uso do Framework IssueNet.

3.6.3.

Diagrama de Atividades

Para que os benefícios da aplicação do Framework IssueNet sejam melhor compreendidos, serão apresentados dois diagramas de atividades referentes a possíveis cenários de uso do sistema: o primeiro diagrama representa a dinâmica antes da utilização do IssueNet, e o segundo, a dinâmica após a aplicação do IssueNet.

PUC-Rio - Certificação Digital Nº 0521485/CA

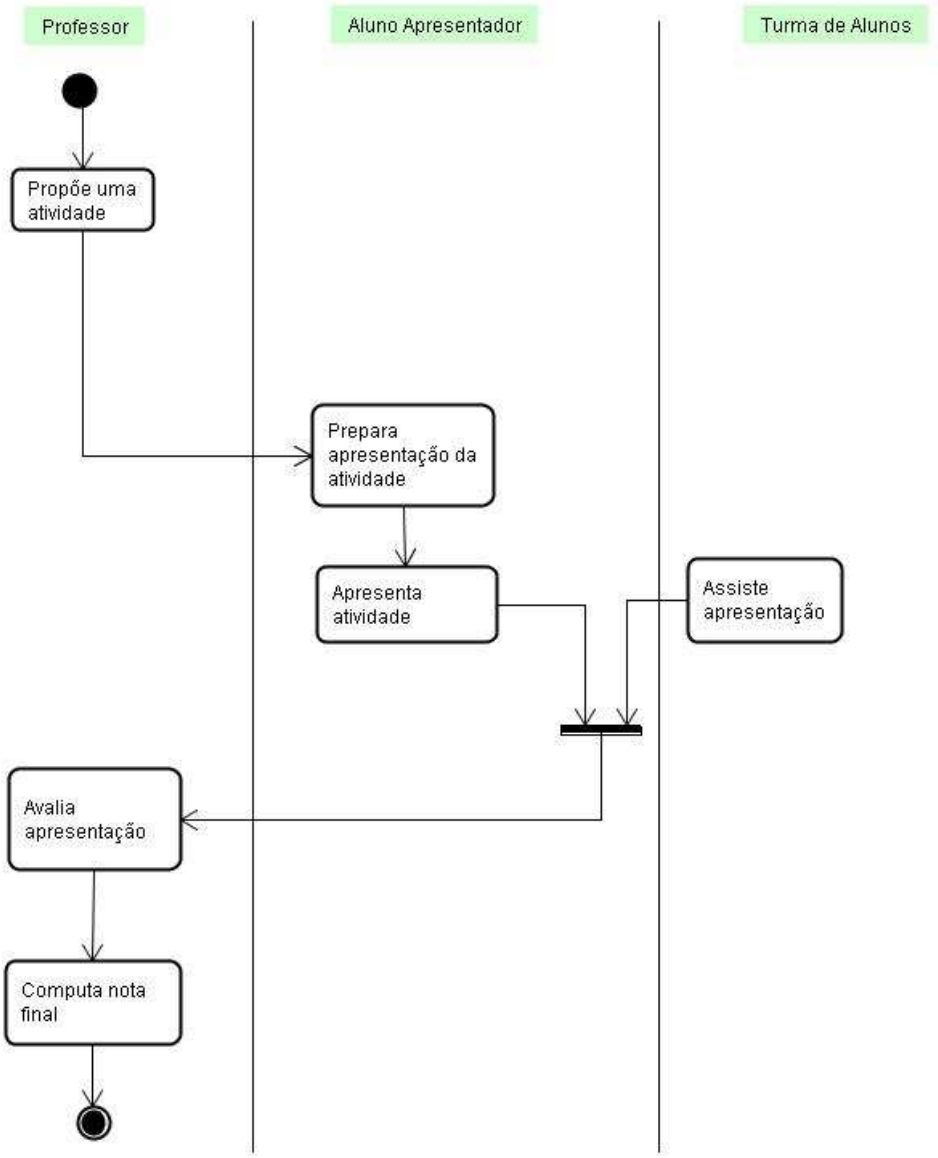

Figura 3.9 - Diagrama de Atividades de cenários em que seja aplicável a utilização do Framework, antes da sua implementação

Como se pode perceber pela figura 3.9, a dinâmica apresentada concentra todas as atividades de avaliação no professor, enquanto que o resto da turma apenas assiste passivamente a apresentação. A aplicação do IssueNet neste caso possibilitará a avaliação colaborativa, possibilitando aos outros alunos da turma contribuírem na avaliação dos colegas, como ilustra a figura 3.10. Os benefícios da avaliação colaborativa já foram discutidos no capítulo 2.

PUC-Rio - Certificação Digital Nº 0521485/CA

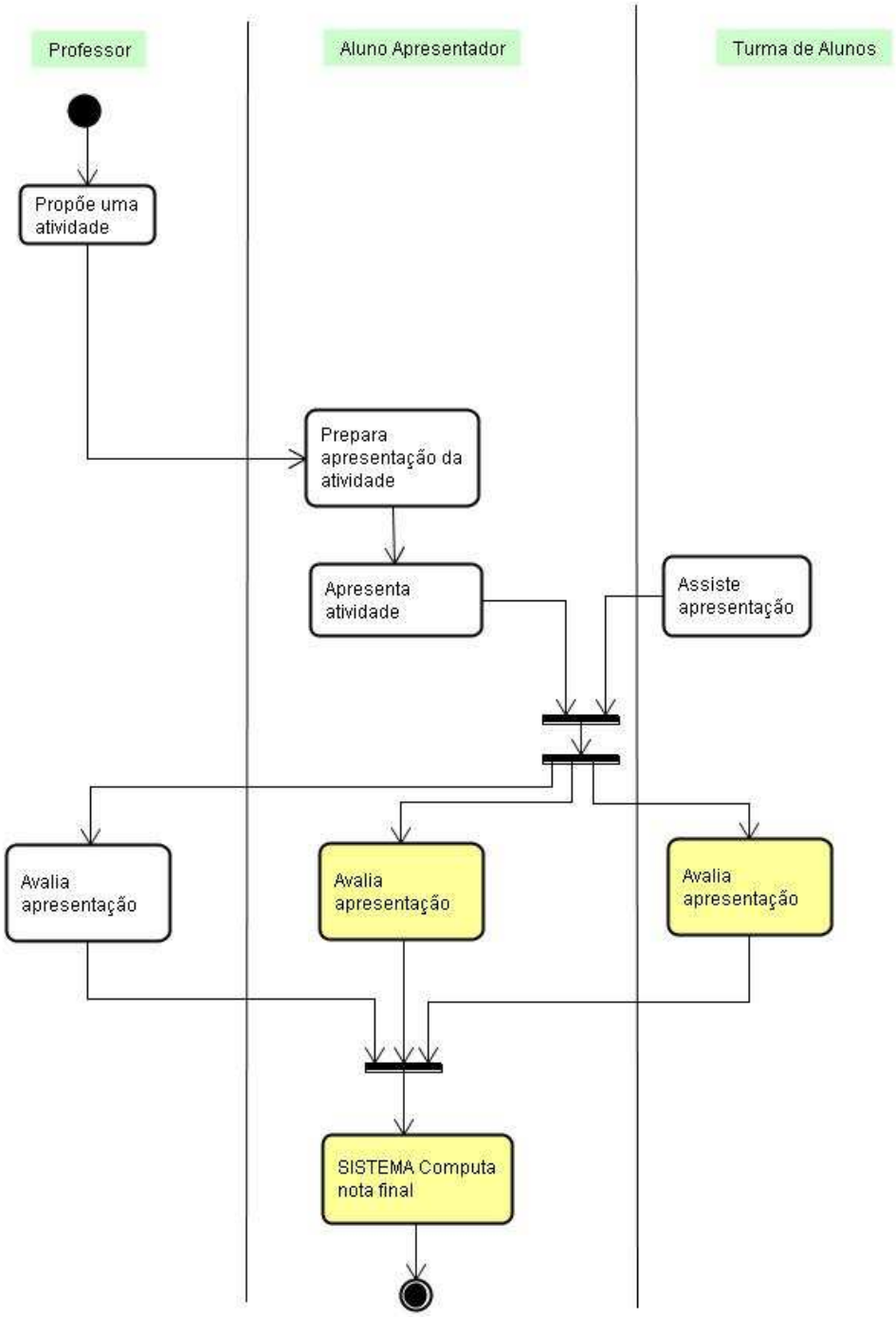

Figura 3.10 - Diagrama de Atividades de cenários em que seja aplicável a utilização do Framework, após a sua implementação

Este capítulo buscou apresentar o Framework IssueNet em sua totalidade: primeiramente, foram discutidos conceitos gerais sobre Frameworks e, em seguida, os módulos e a arquitetura do Framework foram detalhados. Por fim, apresentaram-se os pontos flexíveis que permitirão a extensão do Framework e os *design* patterns e diagramas UML utilizados na etapa de modelagem do sistema.

PUC-Rio - Certificação Digital Nº 0521485/CA

O próximo capítulo apresentará com detalhes os dois estudos de caso realizados com instâncias do IssueNet. Primeiramente, será apresentado um estudo de caso relacionado à aprendizagem colaborativa e, em seguida, um estudo de caso na área de trabalho colaborativo.

PUC-Rio - Certificação Digital Nº 0521485/CA