Modelagem Estática

Metodologias de Desenvolvimento (I)

- **Método** é definido como sendo um conjunto de atividades sistemáticas para realizar uma tarefa.
- **Técnica** é um modo de executar as atividades recomendadas pelos métodos.
- **Metodologia** é um conjunto de métodos e técnicas com os quais um objetivo pode ser realizado.

Metodologias de Desenvolvimento (II)

- Uma boa metodologia de desenvolvimento deve proporcionar mais do que uma simples notação: ela deve fornecer orientações sobre os passos a serem tomados nos diversos estágios de desenvolvimento de software, e deve cobrir o ciclo de desenvolvimento de software completo.

Metodologias de Desenvolvimento (III)

- Uma metodologia de desenvolvimento de software provê orientações para a construção de um modelo do domínio de um problema e subsequente adição de detalhes de implementação.
- A abordagem orientada a objetos para construção de sistemas permite que um mesmo conjunto de conceitos e notação seja usado através de todo o ciclo de vida do software: análise, projeto e implementação.

Análise vs. Projeto Orientado a Objetos

- Análise orientada a objetos modela o mundo real de tal modo que ele possa ser compreendido. Durante a análise, a ênfase está em encontrar e descrever objetos que estejam no domínio do problema e que sejam relevantes para o sistema que se pretende construir.
- Projeto orientado a objetos define objetos de software que fazem parte do domínio da solução e que serão implementados em uma linguagem de programação orientada a objetos.

Por que Desenvolvimento OO?

- Modelagem direta do mundo real.
- Reutilização.
- Manutenção.
- Unificação de conceitos.
- Quando você produzirá componentes reutilizáveis? Quando você fará uso deles? Em geral, isso é vago.
- é mais fácil dar orientações precisas de como construir um sistema OO de fácil manutenção.
- Foco na manutenção: adicionar melhorias e modificações de forma efetiva.

Desenvolvimento OO

- Como podemos distinguir um projeto bom de um projeto ruim?
- Objetivos: melhorar produtividade e encorajar reutilização.
- Orientações: devem ser claras (o menos subjetivas possível) e a nível de código.

Análise Orientada a Objetos (I)

- A análise orientada a objetos cria uma especificação do domínio do problema e dos requisitos do ponto de vista da classificação baseada em objetos e do entendimento dos termos usados no domínio do problema.
- Durante a análise, são modelados aspectos estáticos e dinâmicos do domínio do problema.

Análise Orientada a Objetos (II)

- A **modelagem estática** visa identificar os conceitos do mundo real relevantes para o sistema.
- Esses conceitos são incluídos em um **diagrama de classes de análise**, ou modelo conceitual.
- A **modelagem dinâmica** descreve os aspectos do sistema de software que podem mudar com o tempo devido à ocorrência de eventos e que dizem respeito ao seu fluxo de controle.
- A modelagem dinâmica usa diagramas dinâmicos de UML, como **diagramas de sequência**, **atividades** e **colaboração**, para modelar as interações entre objetos do sistema.

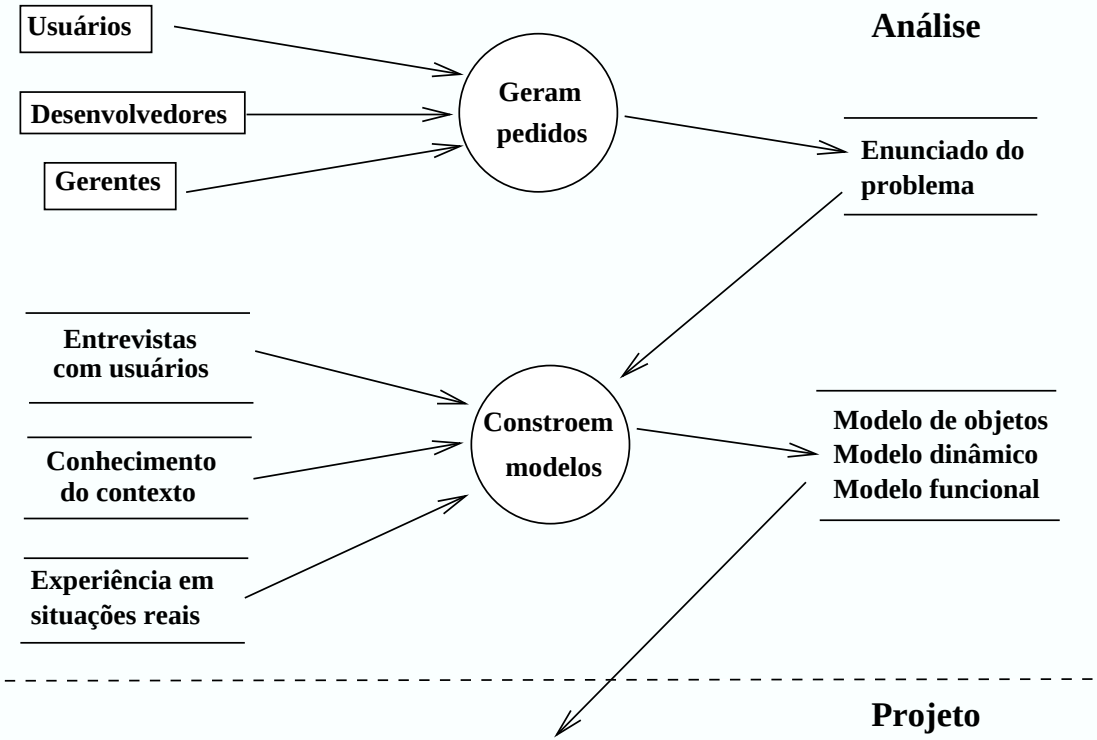
Metodologias de Análise OO

- Várias metodologias orientadas a objetos diferentes podem ser encontradas na literatura.
- Cada uma delas introduz um processo para análise de sistemas, um conjunto de diagramas e uma notação que permitem que um modelo de análise seja criado de uma maneira consistente.
- Podemos afirmar, sem medo de generalizar demais, que as metodologias existentes para desenvolvimento orientado a objetos possuem mais ou menos as mesmas etapas.

Metodologia OMT(I)

- Esta metodologia surgiu em 1991.
- **Estágio da Análise:** preocupa-se em entender e modelar a aplicação e o domínio do problema com o qual ela interage. é composto pelos seguintes modelos:
  - modelo de objetos (diagrama de classes de análise),
  - modelo dinâmico (modelo de seqüência e diagrama de colaboração) e
  - modelo funcional (DFD).

Metodologia OMT (II)

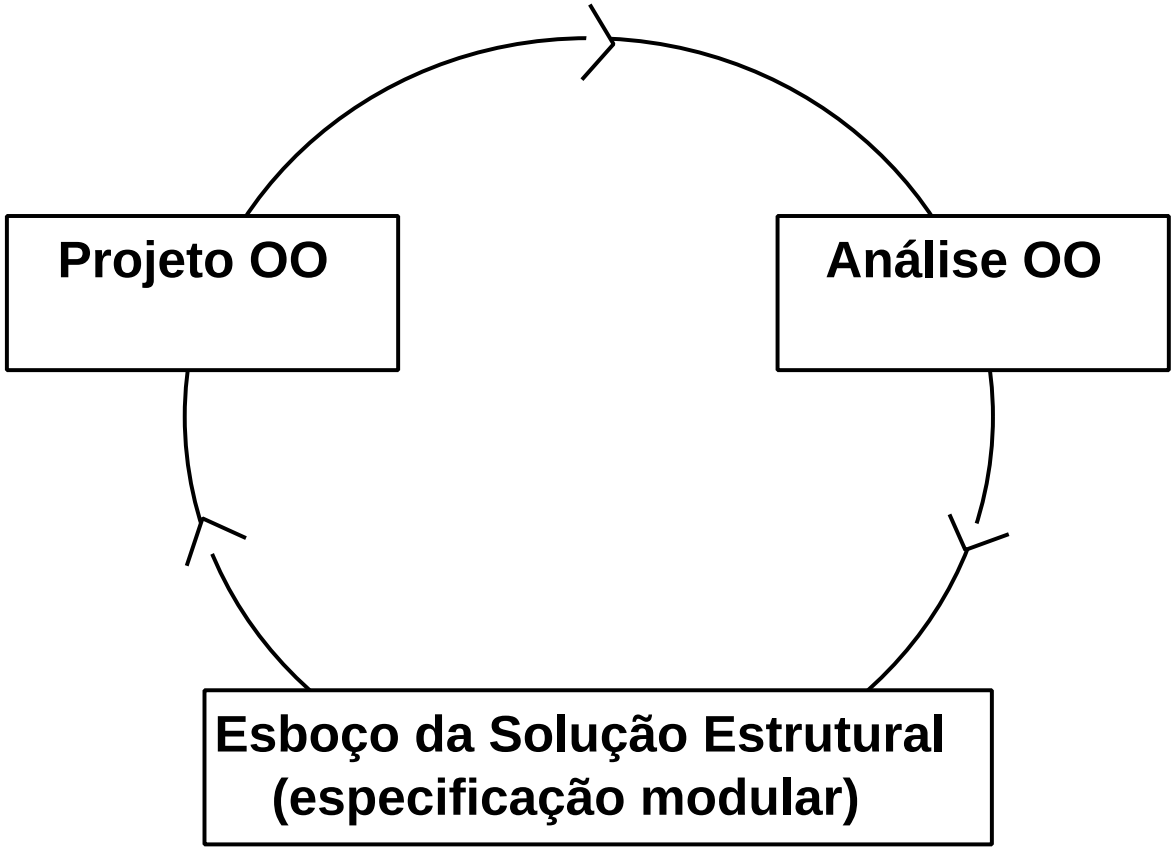


Metodologia OMT(III)

- **Estágio de Projeto do Sistema:** faz decisões a respeito da arquitetura geral do sistema. O projetista divide o sistema em subsistemas, preocupa-se com desempenho, etc.
- **Estágio de Projeto de Objetos:** preocupa-se com estruturas de dados e algoritmos necessários para implementar cada classe do modelo de objetos.

Ao final é gerado um documento contendo um modelo de objetos detalhado, um modelo dinâmico detalhado, e um modelo funcional detalhado.

Metodologia OMT(IV)



Passos da Análise na Metodologia OMT (I)

1. Identificar objetos e classes.
2. Preparar dicionário de dados.
3. Identificar associações (agregações).
4. Identificar atributos.
5. Refinar com herança.

Passos da Análise na Metodologia OMT (II)

6. Testar os caminhos de acesso aos atributos usando cenários.
7. Iterar e refinar.
8. Agrupar classes em módulos.

- Ao final, é gerado um documento contendo um modelo de objetos detalhado, um modelo dinâmico detalhado e um modelo funcional detalhado.

Processo RUP (I)

- Primeira versão em 1999.
- Surgiu da união das metodologias de Booch (Booch), Rumbaugh (OMT) e Jacobson (OOSE).
- Utiliza a notação UML.
- Imenso conjunto de métodos, técnicas, documentos e procedimentos que visa ser o mais genérico possível.
- Muito complexo para ser usado diretamente. é necessário eliminar partes que não sejam relevantes para as necessidades da organização.

Características do Desenvolvimento OO no RUP

- UML como linguagem de modelagem
- Direcionado por casos de uso
- Centrado na arquitetura
- Iterativo
- Incremental

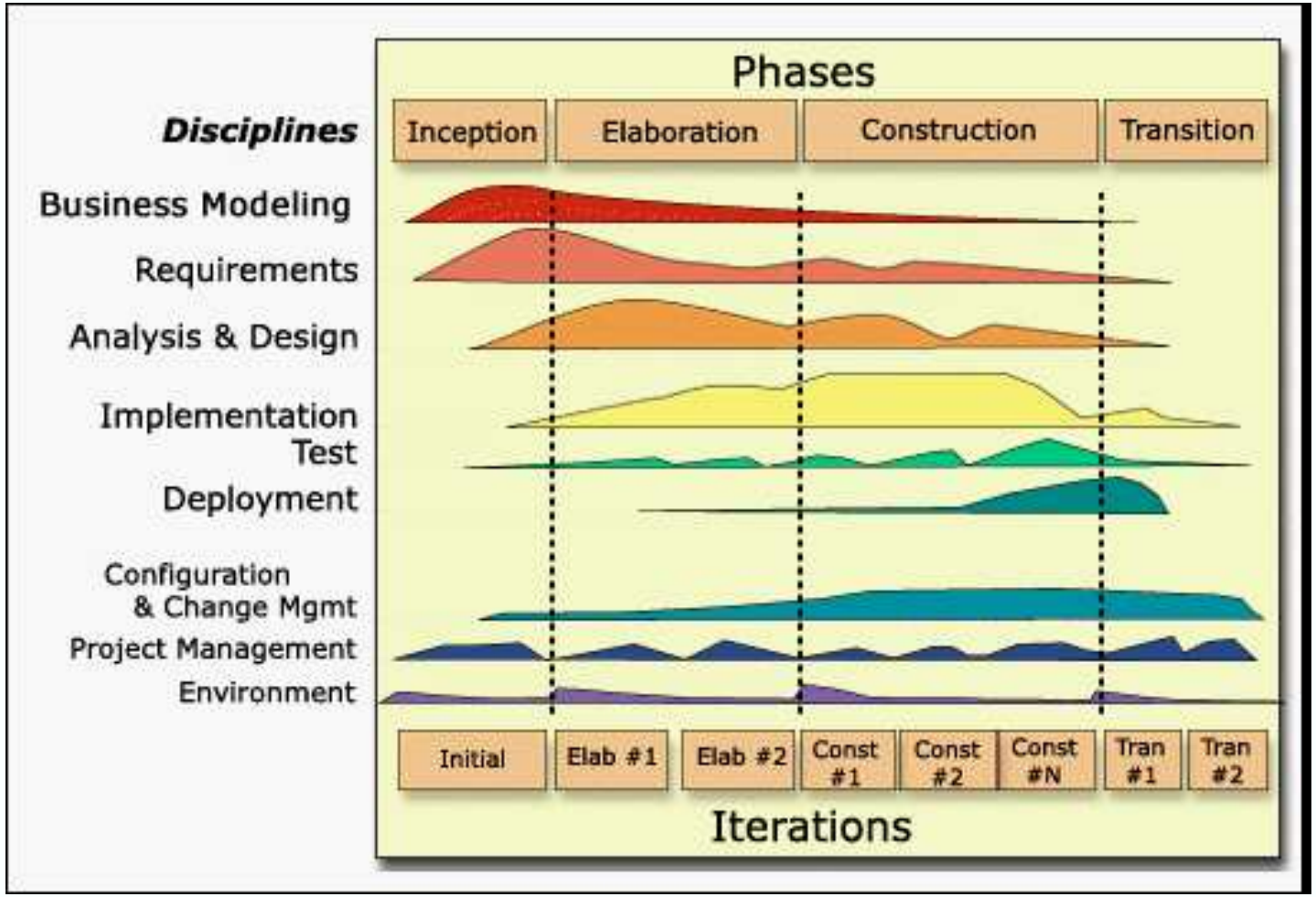
Etapas do RUP

- **Concepção:** estudo de viabilidade, análise de riscos e elicitação dos principais requisitos
- **Elaboração:** determina a arquitetura e os componentes para o projeto, junto com protótipos iniciais
- **Construção:** completa o desenvolimento com base na arquitetura inicial
- **Transição:** *release* completo do software e manutenção

Disciplinas do RUP

- Modelagem de Negócios
- Requisitos
- Análise e Projeto
- Implementação
- Testes
- Implantação
- Gerenciamento de Configuração e Mudanças
- Gerenciamento de Projeto
- Ambiente

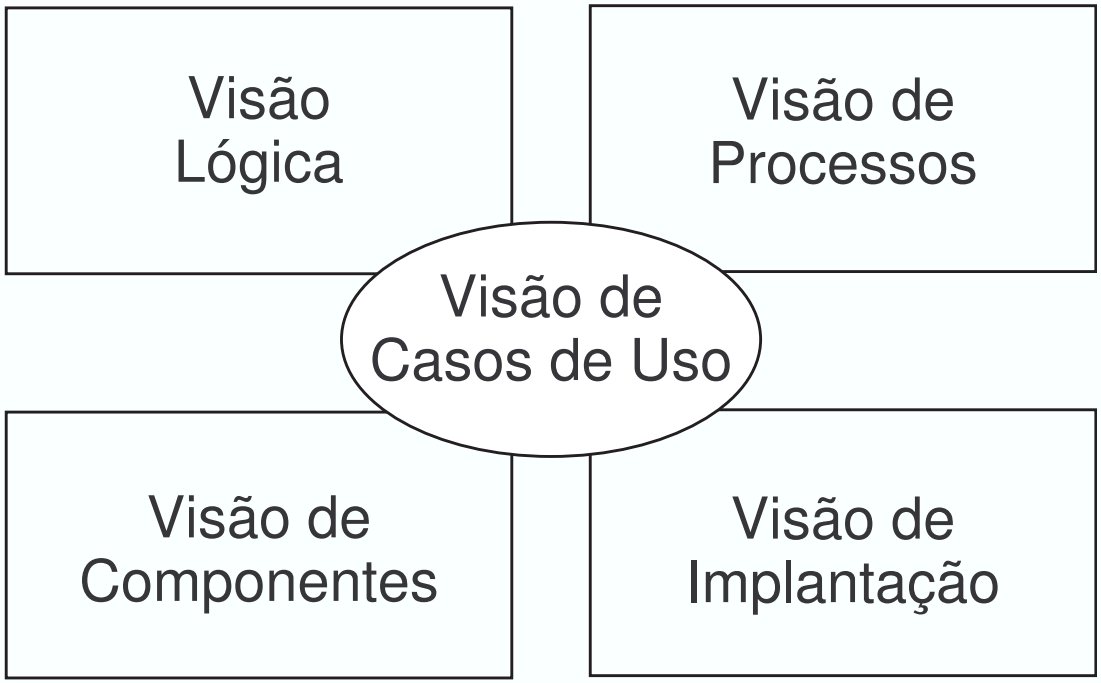
Fases e Fluxos do RUP



O Modelo 4+1 de Especificação (I)

- O RUP se baseia na premissa de que cinco **visões** complementares são necessárias para especificar um sistema de software:
  - Visão Lógica;
  - Visão de Componentes ou Implementação;
  - Visão de Processos ou Concorrência;
  - Visão de Implantação ou Física;
  - Visão de Casos de Uso ou Cenários.

O Modelo 4+1 de Especificação (II)



Análise OO no Processo RUP (I)

- Segundo o RUP, a análise OO visa:
  (i) identificar as classes que executam o fluxo de eventos de um caso de uso e os relacionamentos entre essas classes;
  (ii) distribuir o comportamento do caso de uso entre essas classes, através de realizações de casos de uso.

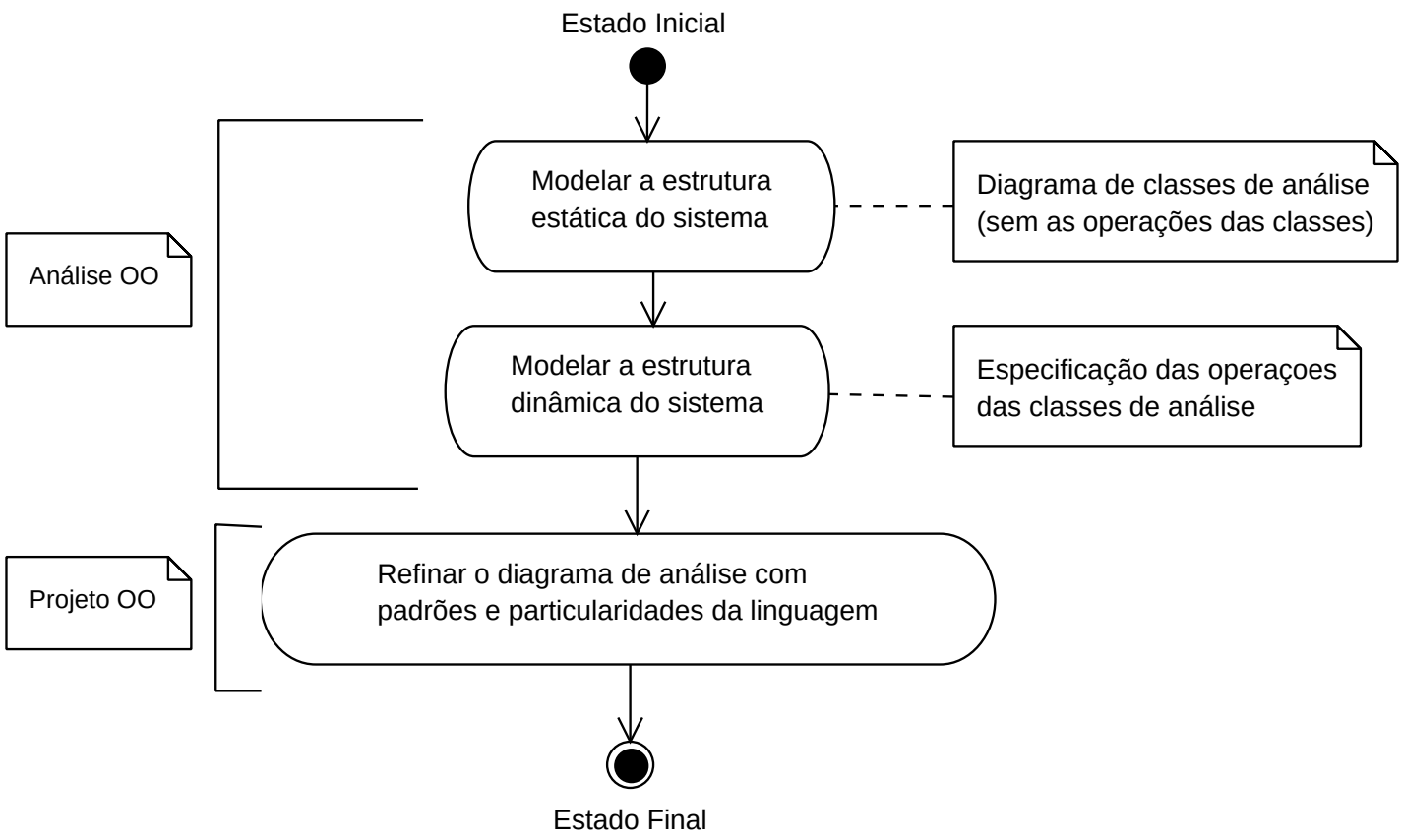
Análise OO no Processo RUP (II)

- Passos:
  1. Complementar as descrições dos casos de uso.
  2. Encontrar classes de análise a partir das descrições dos casos de uso.
  3. Distribuir comportamento entre as classes de análise.
  4. Descrever responsabilidades.
  5. Descrever atributos.
  6. Estabelecer associações entre classes de análise.

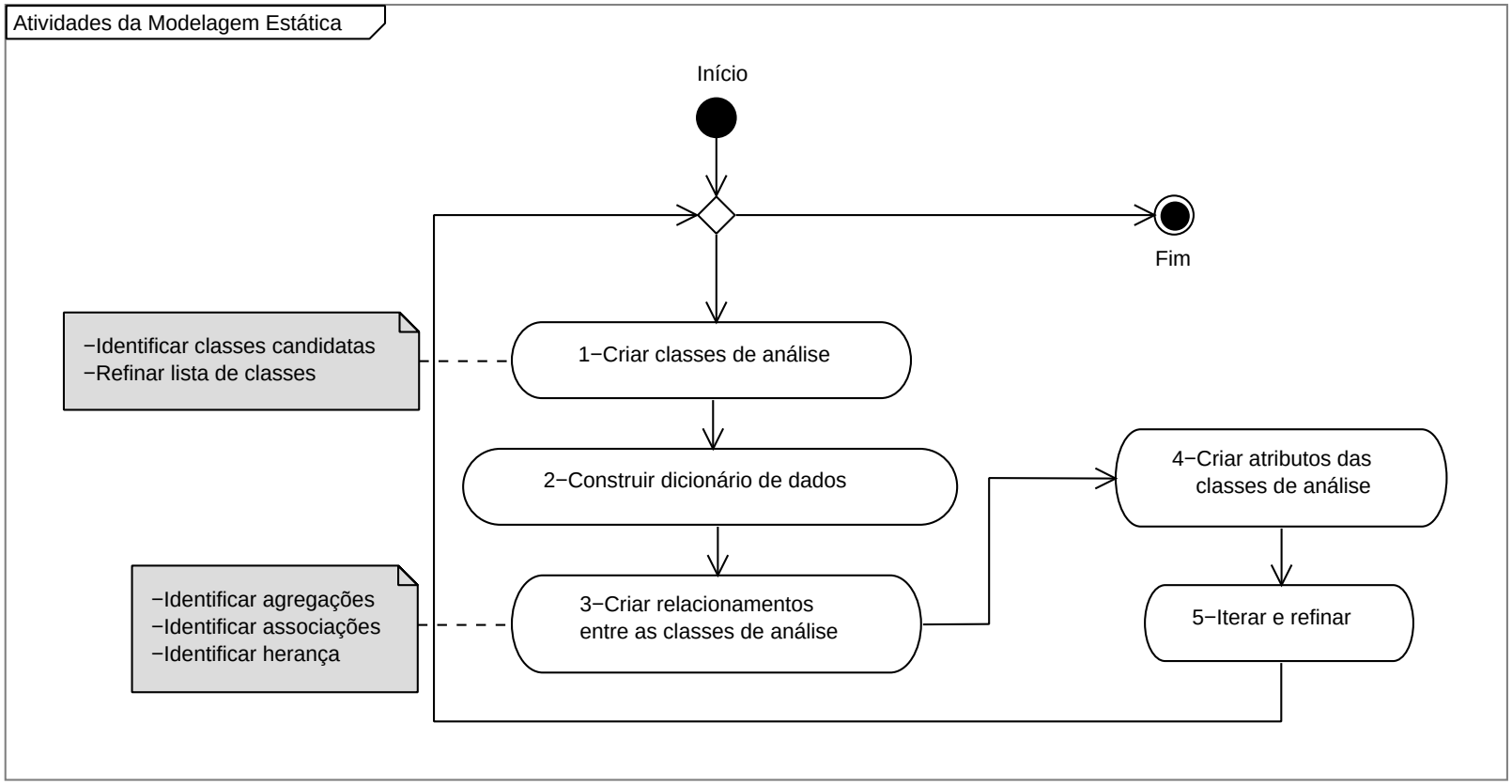
Um Método para Análise OO Baseada na UML (I)

- OMT, apesar das boas idéias, tem algumas deficiências (por exemplo, ela não explora o conceito de casos de uso).
- O RUP é moderno e bastante completo, mas muito complicado para ser usado sem modificações.
- Independentemente das diferenças entre as diversas metodologias existentes, os passos necessários para realizar a análise OO são semelhantes em todas elas.

Um Método para Análise OO Baseada na UML (II)



Um Método para Análise OO Baseada na UML (III)



Uma Metodologia para Análise OO Baseada na UML (IV)

- UML como linguagem de modelagem.
- Características que uma metodologia de desenvolvimento baseada na UML deve apresentar:
  - **Direcionado por casos de uso;**
  - **Centrado na arquitetura;**
  - **Iterativo;**
  - **Incremental.**

Estudo de Caso: Sistema para Controle de Bibliotecas (I)

Queremos construir um sistema de software para controlar o empréstimo e a devolução de exemplares de uma biblioteca. O usuário pode fazer um empréstimo de um exemplar durante um certo período e, ao final desse tempo, o exemplar deve ser devolvido. Renovações não são aceitas.

A atendente é uma funcionária que interage com os usuários e com o sistema de controle da biblioteca através de um terminal. As principais características do sistema são listadas a seguir:

1. Um usuário do sistema, que pode ser um aluno, um professor ou um outro funcionário da universidade, pode reservar publicações e também cancelar reservas previamente agendadas.

Estudo de Caso: Sistema para Controle de Bibliotecas (II)

2. Um usuário deve estar devidamente cadastrado no sistema para usar os seus serviços. O sistema é operado pela atendente da biblioteca, que também é uma funcionária da universidade.
3. Um usuário pode emprestar exemplares previamente reservados ou não. Se foi feita uma reserva, ela deve ser cancelada no momento do seu empréstimo.

Estudo de Caso: Sistema para Controle de Bibliotecas (III)

4. No caso da devolução de um exemplar em atraso, existe uma multa que deve ser paga. Essa multa é calculada com base no número de dias em atraso. Além disso, se o exemplar estiver atrasado por mais de 30 dias e se o usuário não for um professor, além de pagar a multa, o usuário é suspenso por um período de 2 meses.
5. Um exemplar da biblioteca pode ser bloqueado/desbloqueado por um professor por um período de tempo. Nesse caso, o exemplar fica disponível numa estante, podendo ser consultado por usuários da biblioteca, mas não pode ser emprestado.

Estudo de Caso: Sistema para Controle de Bibliotecas (IV)

6. O período de empréstimo é variável, dependendo do tipo de usuário (7 dias para alunos e funcionário, e 15 dias para professores).
7. A manutenção dos dados do acervo da biblioteca é feita pela bibliotecária, que também é funcionária da universidade. Ela é responsável pelo inclusão de novos exemplares, exclusão de exemplares antigos e pela atualização dos dados dos exemplares cadastrados.

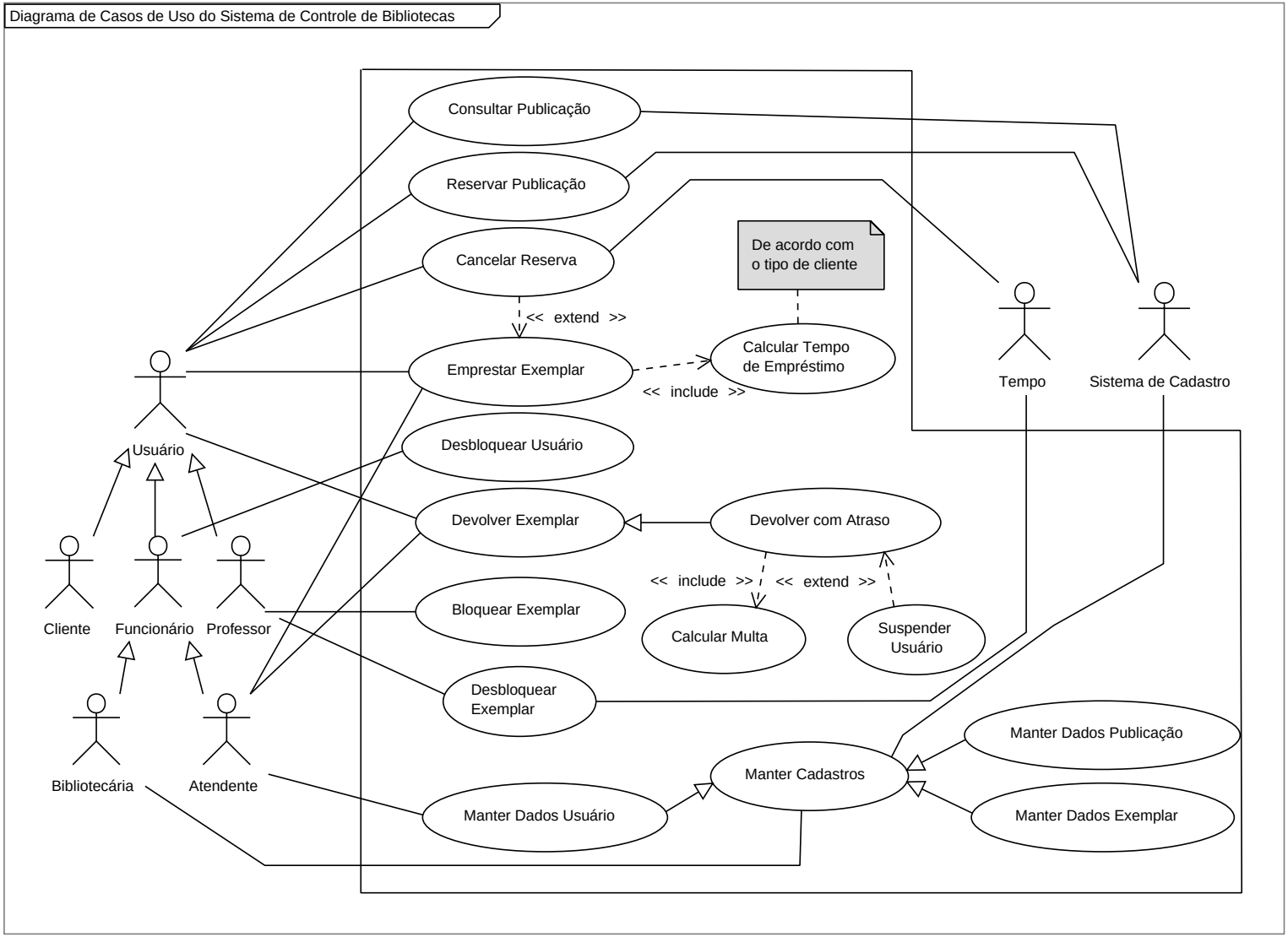
Os exemplares podem ser livros, periódicos, manuais e teses. As publicações são identificadas pelo seu número do tombo, além de outras características como o título, nome do autor, editora e número da edição correspondente.

Estudo de Caso: Sistema para Controle de Bibliotecas (V)

A biblioteca só empresta suas obras para usuários cadastrados. Um usuário é identificado através de seu número de registro. Outras informações relevantes são seu nome, instituto/faculdade a que pertence e seu tipo (aluno/funcionário/professor).

Estas informações adicionais e dados a respeito dos usuários e publicações devem poder ser acessados através de um sistema de cadastro de publicações e usuários.

Diagrama de Casos de Uso: Biblioteca



Caso de Uso Emprestar Exemplar (I)

Breve Descrição : Este caso de uso representa o processo de empréstimo de um ou vários exemplares da biblioteca. O empréstimo se inicia com a solicitação feita pelo cliente à atendente. Em seguida, através de um terminal, a atendente solicita ao sistema o empréstimo de um ou mais exemplares.

Atores : Cliente, Atendente, Sistema de Cadastro.

Pré-condição : O exemplar da publicação está disponível, o cliente está cadastrado no sistema de cadastro, o cliente não está suspenso.

Pos-condição : O exemplar está emprestado.

Requisitos Especiais : nenhum.

Caso de Uso Emprestar Exemplar (II)

Fluxo Básico :

1. O cliente solicita empréstimo de um exemplar de alguma publicação (livro, periódico, tese ou manual), fornecendo o seu número de registro e o número de tombo da publicação desejada.
2. A atendente solicita o empréstimo ao sistema, fornecendo o código do cliente e o tombo da publicação
3. O sistema valida o cliente e verifica o seu status no sistema de cadastro ("Normal" ou "Suspenso") através de seu número de registro.
   (<< *include* >> Validar Usuário)

Caso de Uso Emprestar Exemplar (III)

Fluxo Básico :

4. O sistema verifica se existe algum exemplar disponível da publicação desejada.
5. Se o status do cliente for "Normal" e algum exemplar da publicação estiver disponível

5.1. O sistema registra um novo empréstimo;

5.2. O sistema verifica o período do empréstimo, que depende do tipo de usuário - 7 dias para alunos ou funcionários e 15 para professores

5.3. O sistema atualiza seu banco de dados com a informação de que o exemplar não irá se encontrar na biblioteca até completar o período.

Caso de Uso Emprestar Exemplar (IV)

Fluxo Alternativo 1 :

No passo 5, se o usuário estiver suspenso, este é informado de sua proibição de retirar exemplares e o empréstimo não é realizado.

Fluxo Alternativo 2 :

No passo 5, se todas as cópias da publicação estiverem emprestadas ou reservadas, o sistema informa à atendente que não será possível realizar o empréstimo.

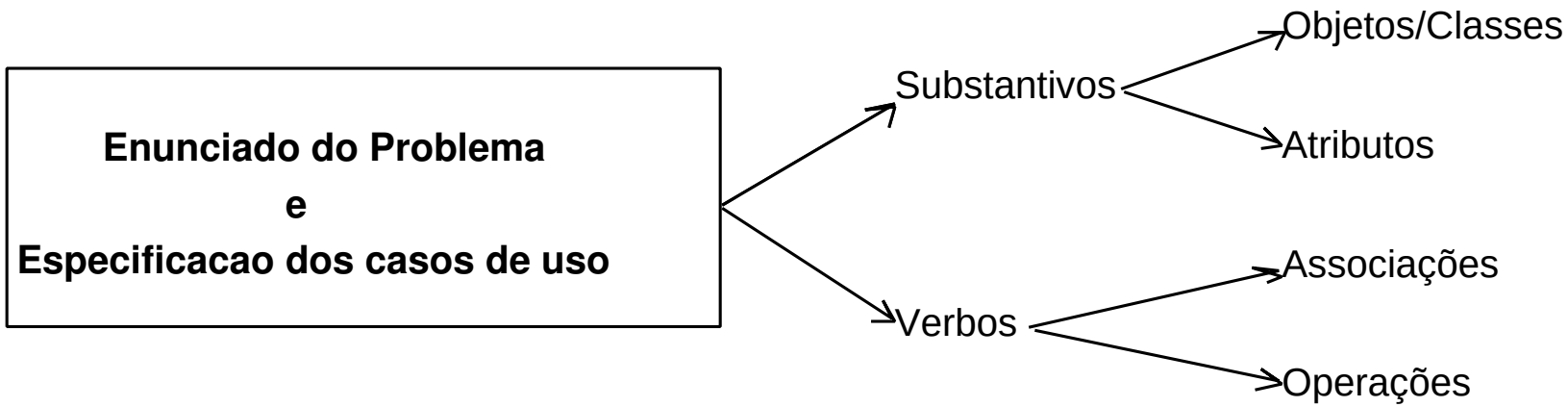
Atividade 1: Identificar Classes de Análise

- Usa-se especificações dos casos de uso.
- **Alternativa 1:** Identificar os conceitos do domínio do problema, que são relevantes para o sistema que se pretende construir. Esses conceitos se tranformam em classes de análise.
- **Alternativa 2:** Fazer uma análise textual da descrição do problema e das especificações dos casos de uso para identificar classes relevantes.
- **Obs.(1)**: As alternativas 1 e 2 podem ser usadas isoladamente ou de forma complementar.
- **Obs.(2)**: o diagrama de classes de análise é uma descrição de abstrações no domínio do problema do mundo real, não no do projeto de software!

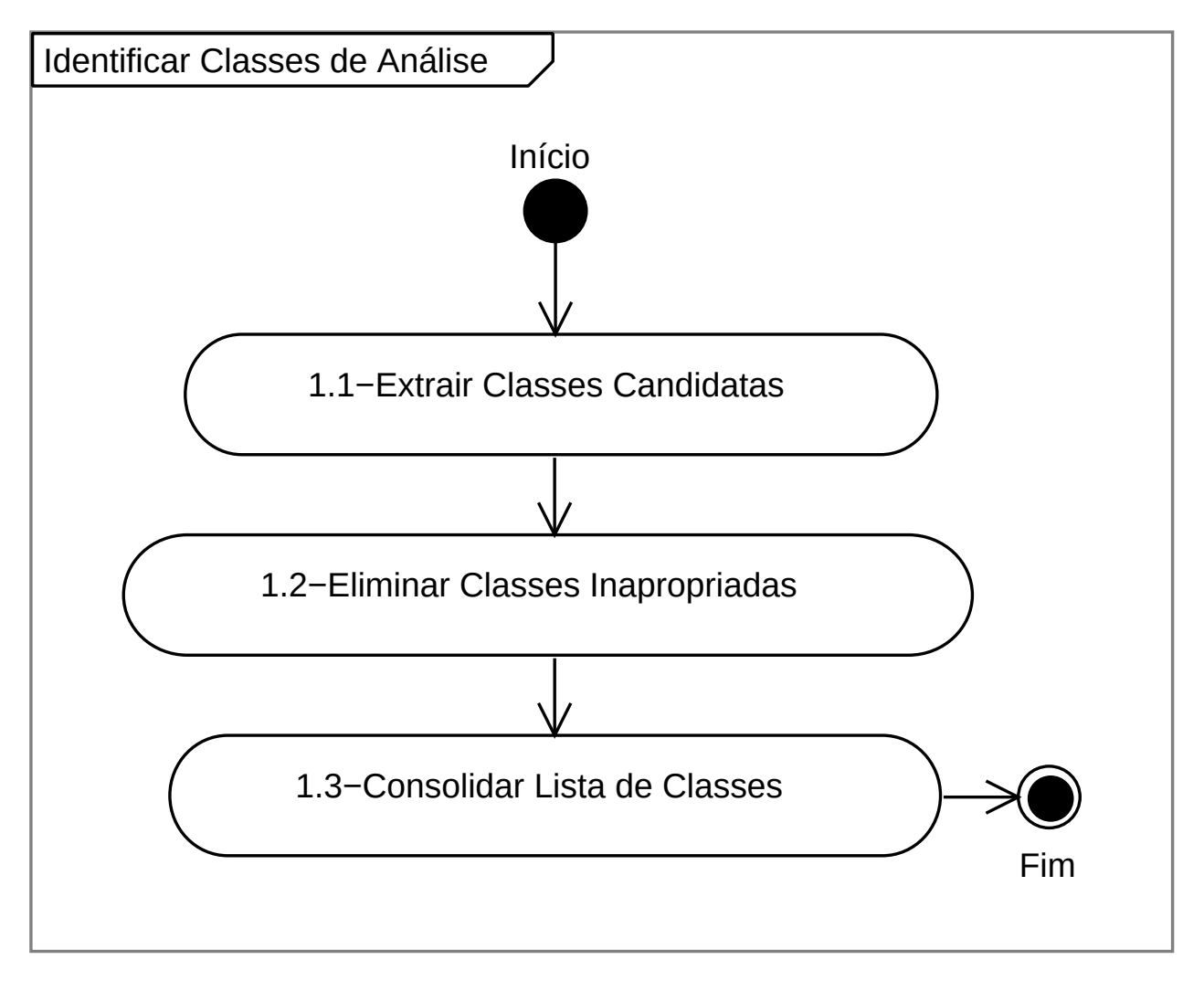
Domínio de Locação/Devolução de Bens

Categoria	**Exemplo**
Objetos físicos ou tangíveis	livro, periódico, tese, manual, cartão da biblioteca
Locais	biblioteca
Transações	empréstimo, reserva, cadastro de usuário, inclusão de nova obra no acervo
Items de transações	exemplar, publicação
Papéis de pessoas	atendente, usuário, bibliotecária
Outros sistemas ou dispositivos externos ao sistema	sistema de cadastro de publicações e usuários
Eventos	devolução com atraso, reserva, empréstimo, perda, bloqueio, desbloqueio

Análise Textual



Atividade 1: Identificar Classes de Análise



Atividade 1.1: Extrair Classes Candidatas

- Utilizamos análise textual para identificar as classes candidatas a partir da especificação do caso de uso *Emprestar Exemplar.*
- As responsabilidades das classes (operações) serão identificadas durante a modelagem dinâmica.

Caso de Uso Emprestar Exemplar (I)

Breve Descrição : Este caso de uso representa o processo de empréstimo de um ou vários exemplares da biblioteca. O empréstimo se inicia com a solicitação feita pelo cliente à atendente. Em seguida, através de um terminal, a atendente solicita ao sistema o empréstimo de um ou mais exemplares.

Atores : Cliente, Atendente, Sistema de Cadastro.

Pré-condição : O exemplar da publicação está disponível, o cliente está cadastrado no sistema de cadastro, o cliente não está suspenso.

Pos-condição : O exemplar está emprestado.

Requisitos Especiais : nenhum.

Caso de Uso Emprestar Exemplar (II)

Fluxo Básico :

1. O cliente solicita empréstimo de um exemplar de alguma publicação (livro, periódico, tese ou manual), fornecendo o seu número de registro e o número de tombo da publicação desejada.
2. A atendente solicita o empréstimo ao sistema, fornecendo o código do cliente e o tombo da publicação
3. O sistema valida o cliente e verifica o seu status no sistema de cadastro ("Normal" ou "Suspenso") através de seu número de registro.
   (<< *include* >> Validar Usuário)

Caso de Uso Emprestar Exemplar (III)

Fluxo Básico :

4. O sistema verifica se existe algum exemplar disponível da publicação desejada.
5. Se o status do cliente for "Normal" e algum exemplar da publicação estiver disponível

5.1. O sistema registra um novo empréstimo;

5.2. O sistema verifica o período do empréstimo, que depende do tipo de **usuário** - 7 dias para alunos ou funcionários e 15 para professores

5.3. O sistema atualiza seu banco de dados com a informação de que o exemplar não irá se encontrar na biblioteca até completar o período.

Caso de Uso Emprestar Exemplar (IV)

Fluxo Alternativo 1 :

No passo 5, se o usuário estiver suspenso, este é informado de sua proibição de retirar exemplares e o empréstimo não é realizado.

Fluxo Alternativo 2 :

No passo 5, se todas as retirar cópias da publicação estiverem emprestadas ou reservadas, o sistema informa à atendente que não será possível realizar o empréstimo.

Classes Candidatas

processo de empréstimo	exemplares	biblioteca
empréstimo	cliente	atendente
terminal	sistema	sistema de cadastro
publicação	disponível	suspenso
emprestado	livro	periódico
tese	manual	número de registro
número de tombo	código do cliente	status do cliente
período de empréstimo	tipo de usuário	usuário
dias	alunos	professores
banco de dados	informação	período
proibição	cópias	

Atividade 1.2: Eliminar Classes Inapropriadas

- Depois que a lista inicial de classes candidatas é obtida, precisamos refiná-la para eliminar classes redundantes ou irrelevantes.

Critérios para Eliminar Classes Inapropriadas

- **Classes Redundantes:** quando duas palavras significam a mesma coisa, escolha a palavra mais significativa.
- **Termos Irrelevantes ou Vagos:** aquelas classes que não estão diretamente relacionadas com o problema.
- **Atributos:** alguns atributos podem ser descritos por substantivos.
- **Operações:** alguns substantivos podem ser operações.
- **Papéis:** papéis são representados por atores e/ou se referem a processos dinâmicos, ao invés de classes propriamente ditas.
- **Construções de Implementação:** qualquer coisa que faça referência a estruturas de dados, etc.

Classes Candidatas Eliminadas (I)

1. **processo de empréstimo**: sinônimo de empréstimo.
2. **cliente**: sinônimo de usuário.
3. **atendente**: representa apenas o papel de um ator, já que o sistema não mantém uma lista de cadastro das atendentes.
4. **sistema de cadastro**: representa o papel de um ator.
5. **disponível**: atributo de publicação.
6. **suspenso**: atributo de usuário.
7. **emprestado**: atributo de publicação.

Classes Candidatas Eliminadas (II)

8. **número de registro**: atributo de usuário.
9. **número de tombo**: atributo de publicação.
10. **código do cliente**: atributo de usuário.
11. **status do cliente**: atributo de usuário.
12. **período de empréstimo**: sinônimo de período.
13. **tipo de usuário**: essa informação é capturada pela hierarquia formada entre usuario, aluno e professor.

Classes Candidatas Eliminadas (III)

14. **dias**: termo vago.
15. **banco de dados**: faz parte do domínio da solução.
16. **informação**: termo vago, mas no contexto do caso de uso, é algum atributo de empréstimo.
17. **período**: atributo de empréstimo.
18. **proibição**: representa um papel (processo dinâmico).
19. **cópias**: sinônimo de exemplar.

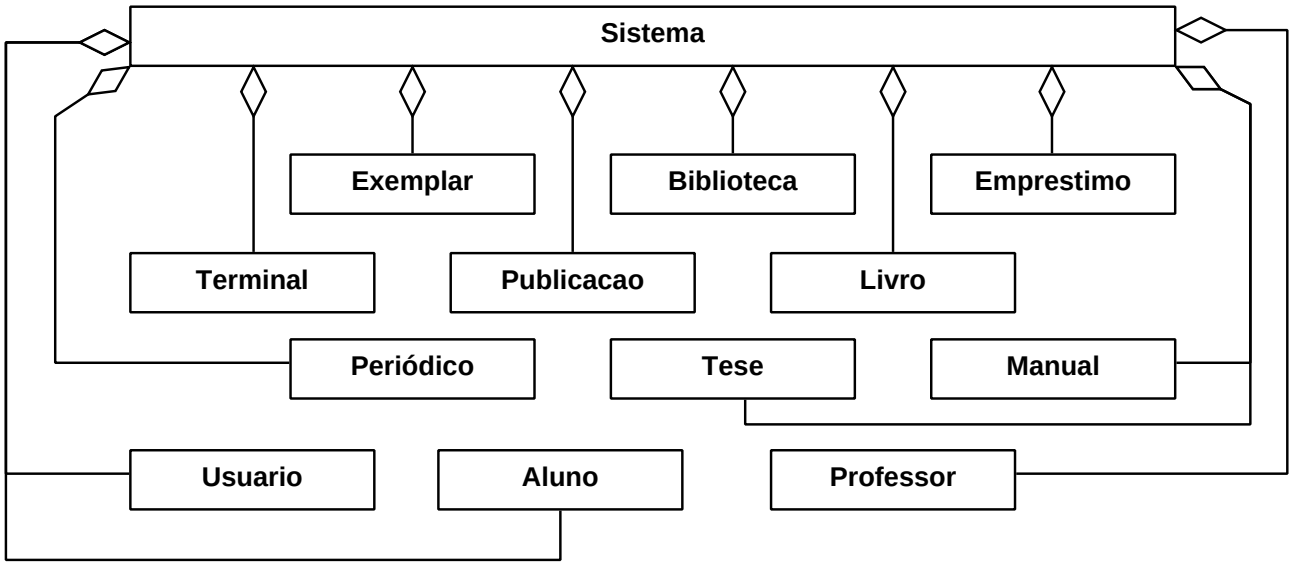
Atividade 1.3: Consolidar Lista de Classes

processo de empréstimo
empréstimo
terminal
publicação
emprestado
tese
número de tombo
período de empréstimo
dias
banco de dados
proibição

exemplares
cliente
sistema
disponível
livro
manual
código do cliente
tipo de usuário
alunos
informação
cópias

biblioteca
atendente
sistema de cadastro
suspenso
periódico
número de registro
status do cliente
usuário
professores
período

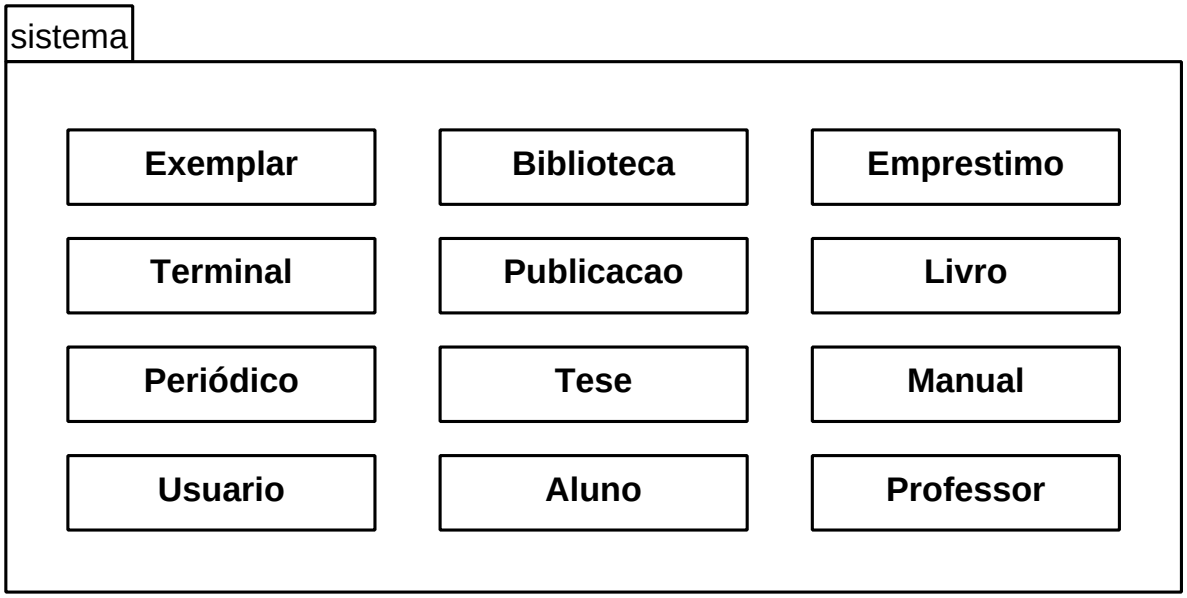
Classes de Análise Identificadas (I)



- A classe **Sistema** é uma representação semântica do sistema como um todo

Classes de Análise Identificadas (II)

- A classe **Sistema** pode ser representada por um módulo (pacote UML):



Ativ.2: Constr./Atualizar Dic. de Dados (I)

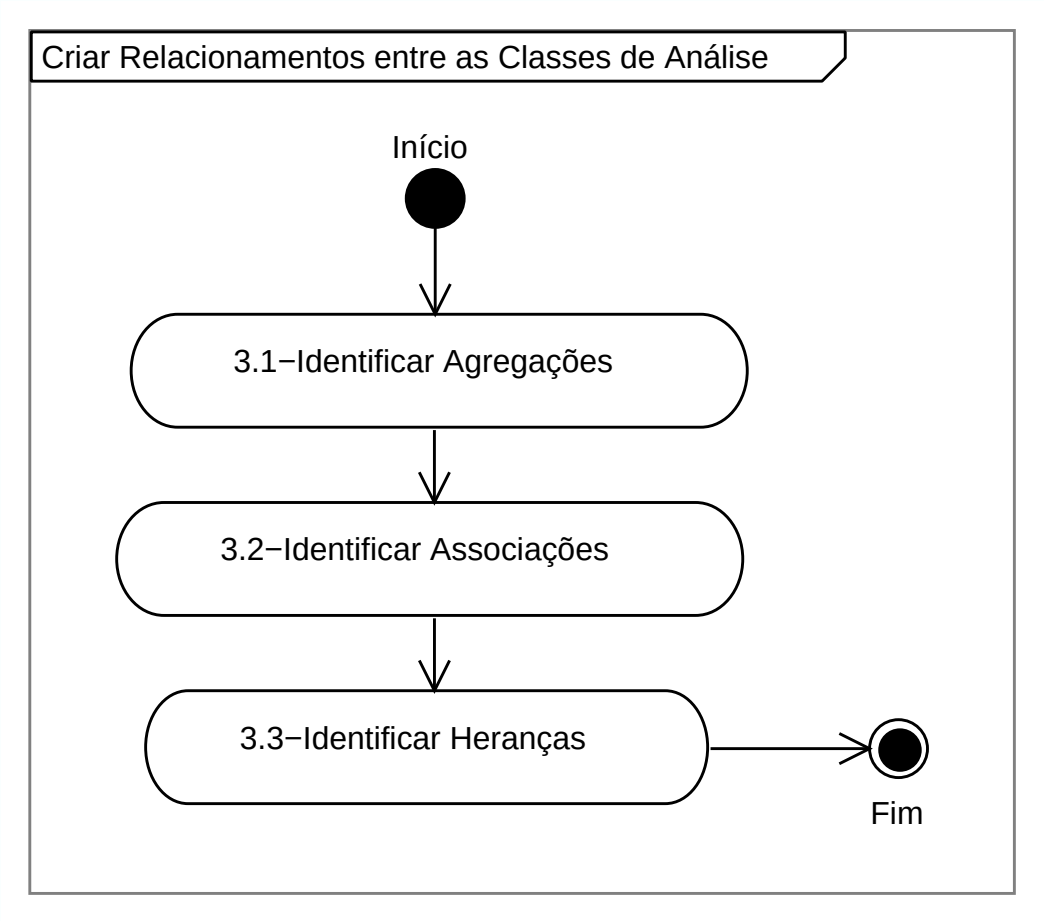
- O dicionário de dados descreve as classes de análise identificadas.

  Classe **Empréstimo**: materializa o empréstimo de exemplares relativos a um usuário, guardando informações dos exemplares emprestados, quem os pegou, quando o empréstimo foi realizado e qual o prazo para devolução.

Ativ.2: Constr./Atualizar Dic. de Dados (II)

- Classe **Publicação**: classe que materializa um título em si. Não representa o objeto físico propriamente dito.
- Classe **Exemplar**: classe que materializa as cópias de uma publicação. Representa o objeto físico que é emprestado.

Atividade 3: Criar Relacionamentos entre as Classes de Análise (I)



Atividade 3: Criar Relacionamentos entre as Classes de Análise (II)

- Esses relacionamentos podem ser identificadas através do estudo das especificações dos casos de uso e das descrições dos elementos no dicionário de dados.
  - Normalmente são identificados analisando-se os verbos
- Relações do tipo "conhece", "é composto" e "é um tipo de" entre classes freqüentemente indicam a existência respectivamente de associações, agregações e generalizações/especializações.
- Deve-se ter cuidado para não poluir o diagrama de classes de análise com um número excessivo de associações.

Atividade 3.1: Identificar Agregações

- Agregações indicam relações parte-todo entre duas ou mais classes distintas.
- Critérios para identificar agregações:
  - A é uma parte física ou lógica de B (Motor e Carro);
  - A é um membro de B (Funcionário e Departamento);
  - A está contida em B (Animal e Floresta);
  - A é uma sub-unidade organizacional de B (Departamento e Empresa);

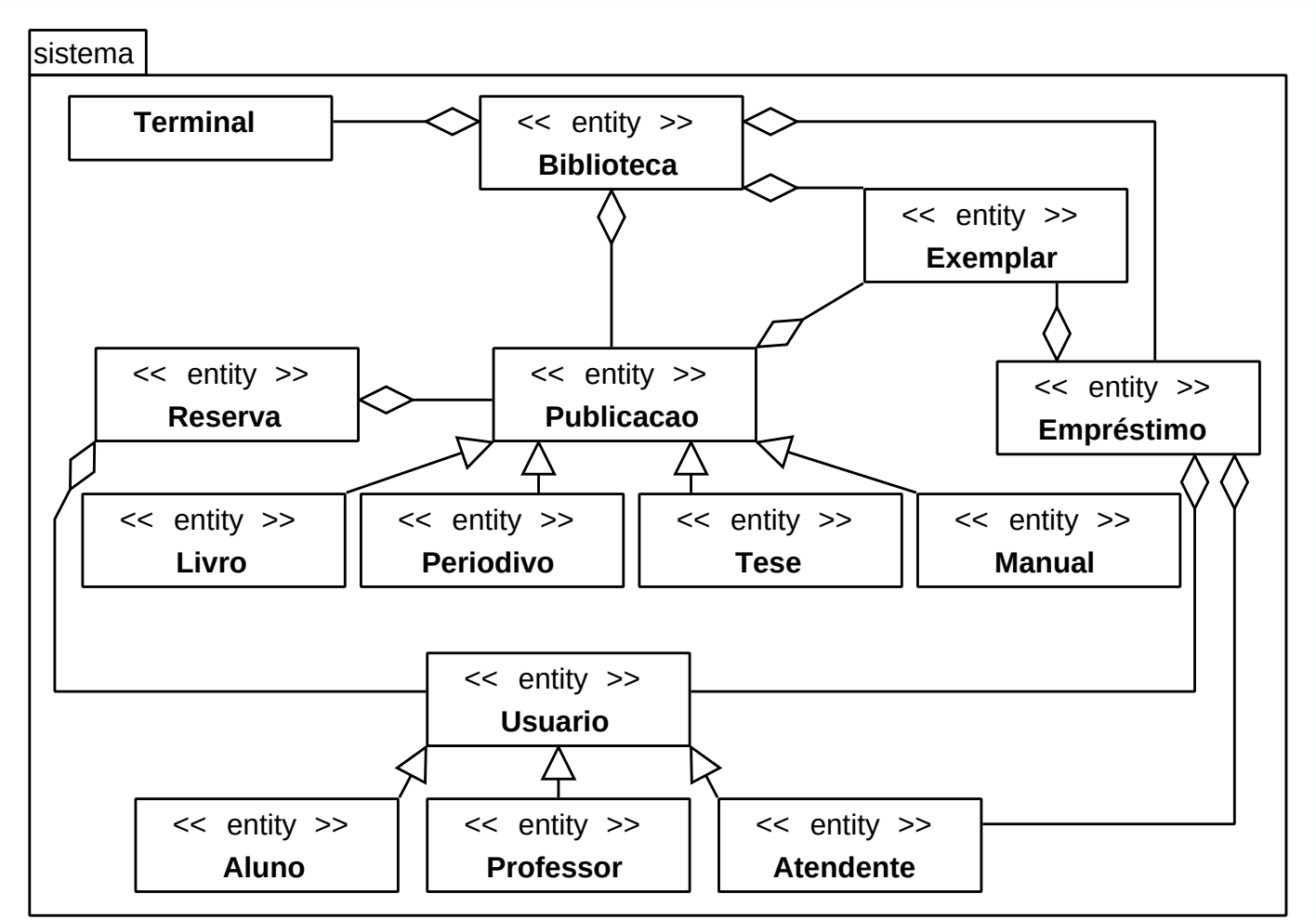
Atividade 3.2: Identificar Associações

- Critérios para a identificação de associações:
  - A usa ou gerencia B (Motorista e Carro);
  - A se comunica com B (Telefone e Central Telefônica);
  - A está relacionada com uma transação de B (Operadora de Cartão e Banco);
- **Obs.:** Todo relacionamento de agregação pode ser modelado como um relacionamento de associação.

Atividade 3.3: Identificar Relacionamentos de Herança

- Para encontrar relacionamentos de herança entre as classes de análise identificadas, deve-se procurar por relações do tipo "é-um" entre elas.

Relacionamentos entre Classes no Estudo de Caso



Atividade 4: Identificar/Refinar Atributos

- Atributos são identificados através de estudo das especificações dos casos de uso do sistema, do enunciado do problema e do dicionário de dados.
- Normalmente representam conceitos simples que podem ser expressados usando-se tipos primitivos, como inteiros e caracteres.
- Também são transformados em atributos os **tipos de dados compostos**.
  Exemplos: *Endereço*, *Telefone*, *Cor*, *Ponto*, etc.
- Se um atributo é muito complexo, provavelmente ele deveria ser definido como uma entidade à parte.

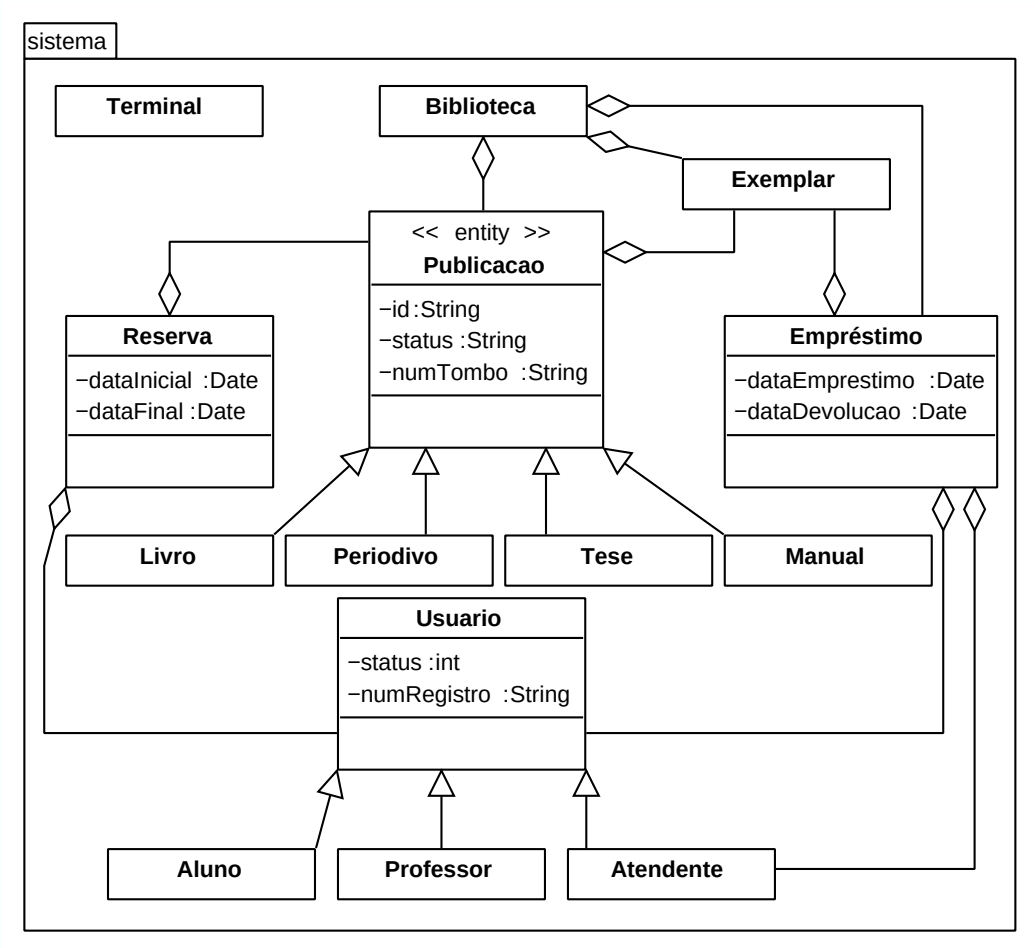
Atributos das Classes de Análise no Estudo de Caso (I)

Classes de Análise	**Atributos**
Empréstimo	Data de Empréstimo Data de Devolução
Usuário	Status Número de Registro
Publicação	Número de Tombo Status

Atributos das Classes de Análise no Estudo de Caso (II)

- Algumas informações foram modificadas para se tornarem mais simples.
  Exemplo: o período de um empréstimo foi transformado em duas datas, uma de empréstimo e outra de devolução.
- Atributos "óbvios" como o título da publicação foram adicionados.

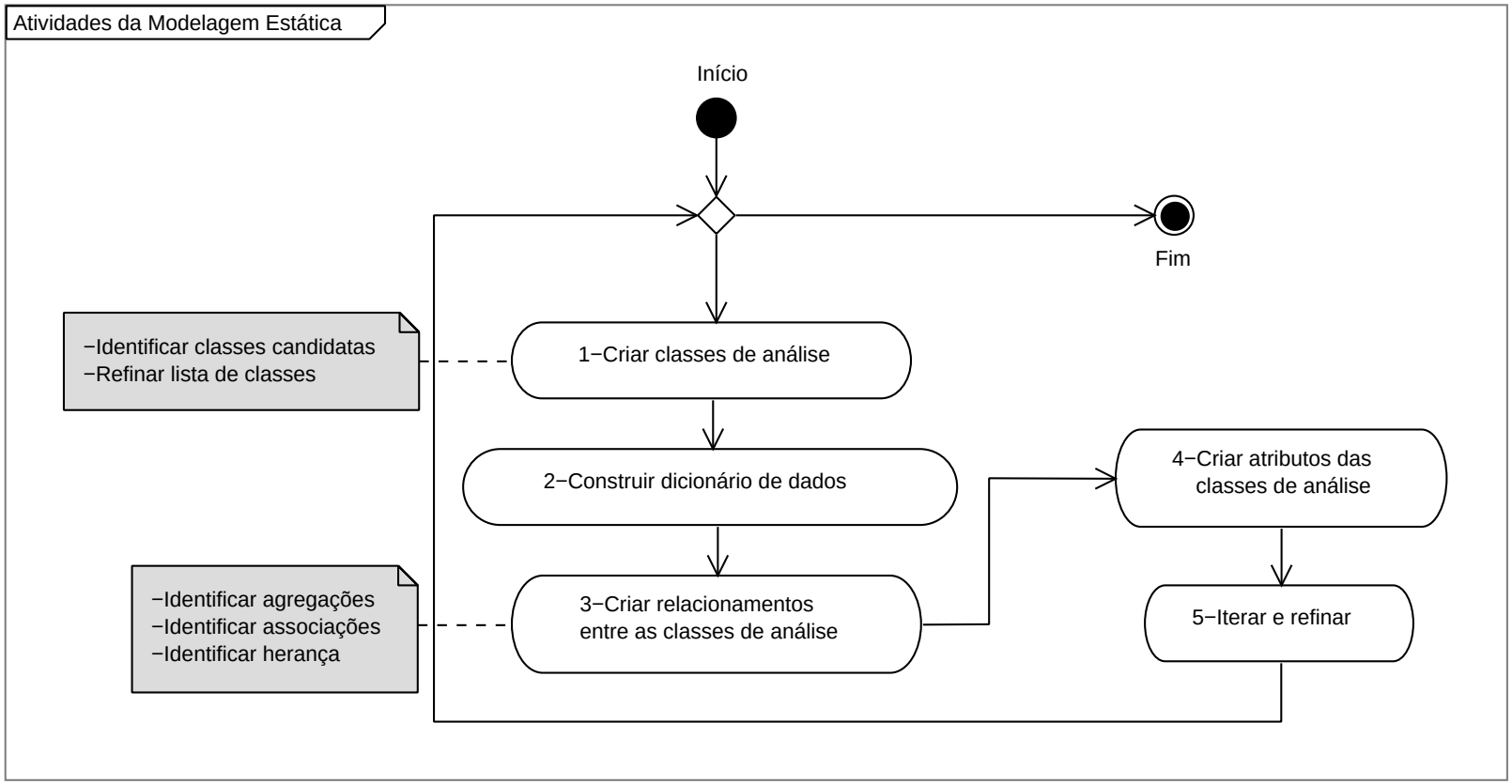
Atributos das Classes de Análise no Estudo de Caso (III)



Atividade 5: Iterar e Refinar

- Por se tratar de um processo iterativo, o sistema é construído gradativamente, a partir de refinamentos sucessivos dos modelos produzidos.
- As principais vantagens dessa construção gradual do sistema são:
  - Possibilidade de efetuar mudanças tardias dos requisitos
  - Distribuição da sua complexidade do desenvolvimento, através da evolução progressiva dos modelos
  - Melhoria da qualidade final do software produzido

Iteração 2

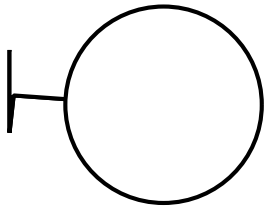

Atividades da Modelagem Estática
Início
Fim
−Identificar classes candidatas
−Refinar lista de classes
1−Criar classes de análise
2−Construir dicionário de dados
4−Criar atributos das classes de análise
−Identificar agregações
−Identificar associações
−Identificar herança
3−Criar relacionamentos entre as classes de análise
5−Iterar e refinar

Atividade 1 (iteração 2): Identificar/Refinar classes de análise (I)

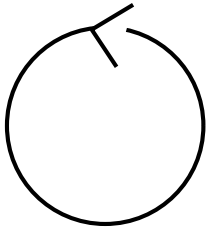
- O processo utilizado sugere que classes de análise sejam classificadas segundo o padrão de modelagem MVC
- Esse padrão divide as classes em três grupos: **entidade**, **fronteira** e **controle**.
- Essa divisão visa
  - Separar elementos não-relacionados
  - Tornar mais fácil a identificação dessas classes
  - Simplificar a transição da análise para o projeto

Atividade 1 (iteração 2): Identificar/Refinar classes de análise (II)

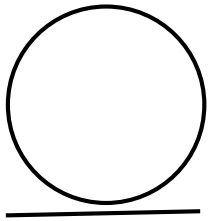
- Para cada tipo de classe de análise, o RUP define um estereótipo.



Classes de Fronteira
<<boundary>>

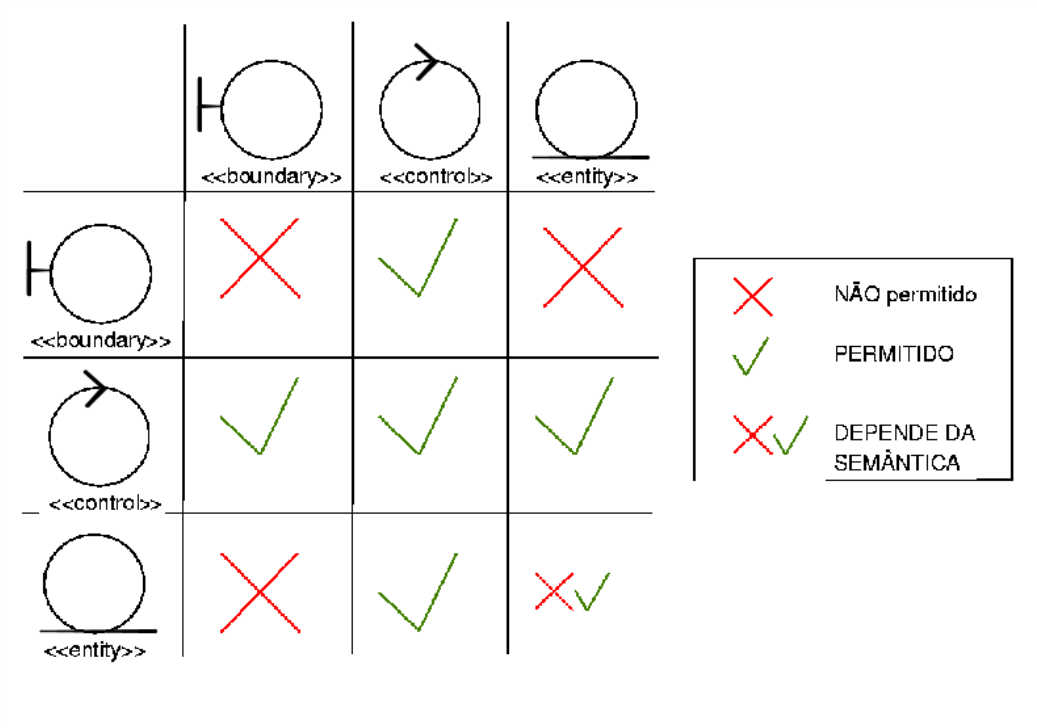


Classes de Controle
<<control>>



Classes de Entidade
<<entity>>

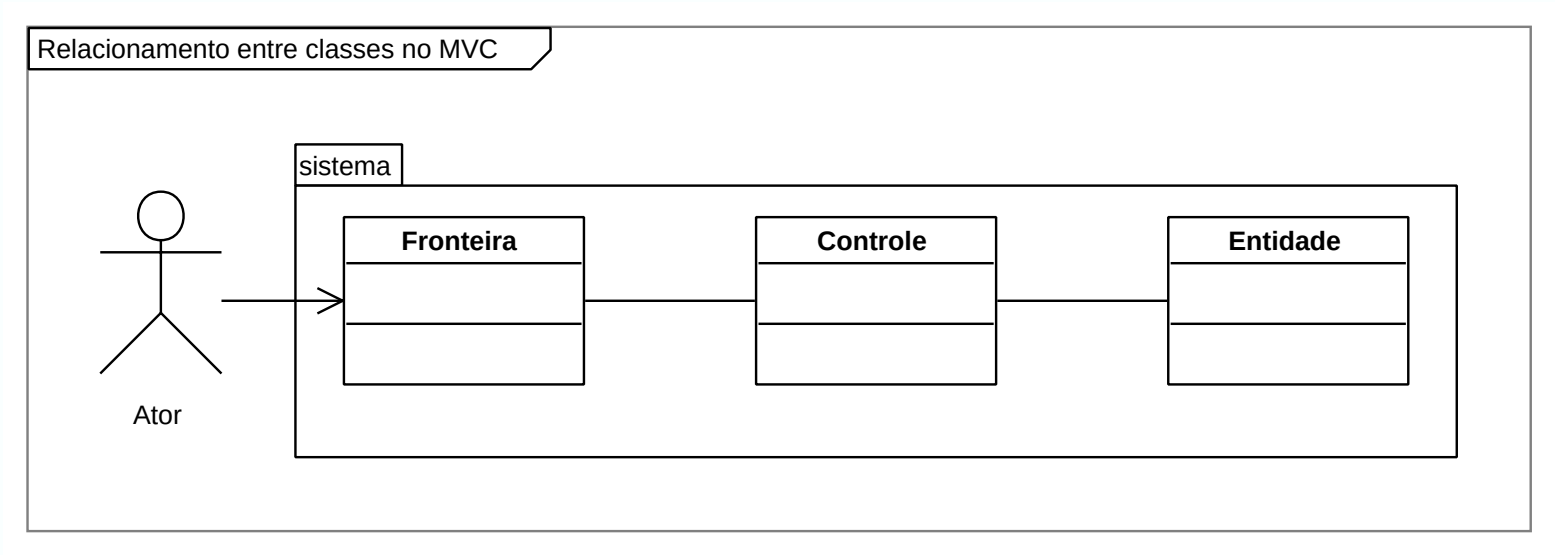
Associação entre Classes de Fronteira, Controle **e** Entidade



- As interações entre as classes devem ser estruturadas da seguinte maneira:

<<boundary>> ↔ <<control>> ↔ <<entity>>

O Nível de Abstração Mais Alto de um Sistema em MVC



Refinamento do Diag. de Classes com MVC

- Classe de fronteira (<< *boundary* >>):
  - `Terminal`
- Classe de controle (<< *control* >>):
  - **Decisão 1:** `Biblioteca`
  - **Decisão 2:** `Criar uma nova classe: Controlador`
    - Há necessidade de armazenar as informações da biblioteca
    - Ela deve ser considerada uma entidade (<< *entity* >>)
    - Exemplo: Sistema que contemple uma rede de bibliotecas
- Classe de entidade (<< *entity* >>):
  - `Exemplar, Biblioteca, Emprestimo, Publicacao, Livro, Periodico, Tese, Manual, Usuario, Aluno, Professor`

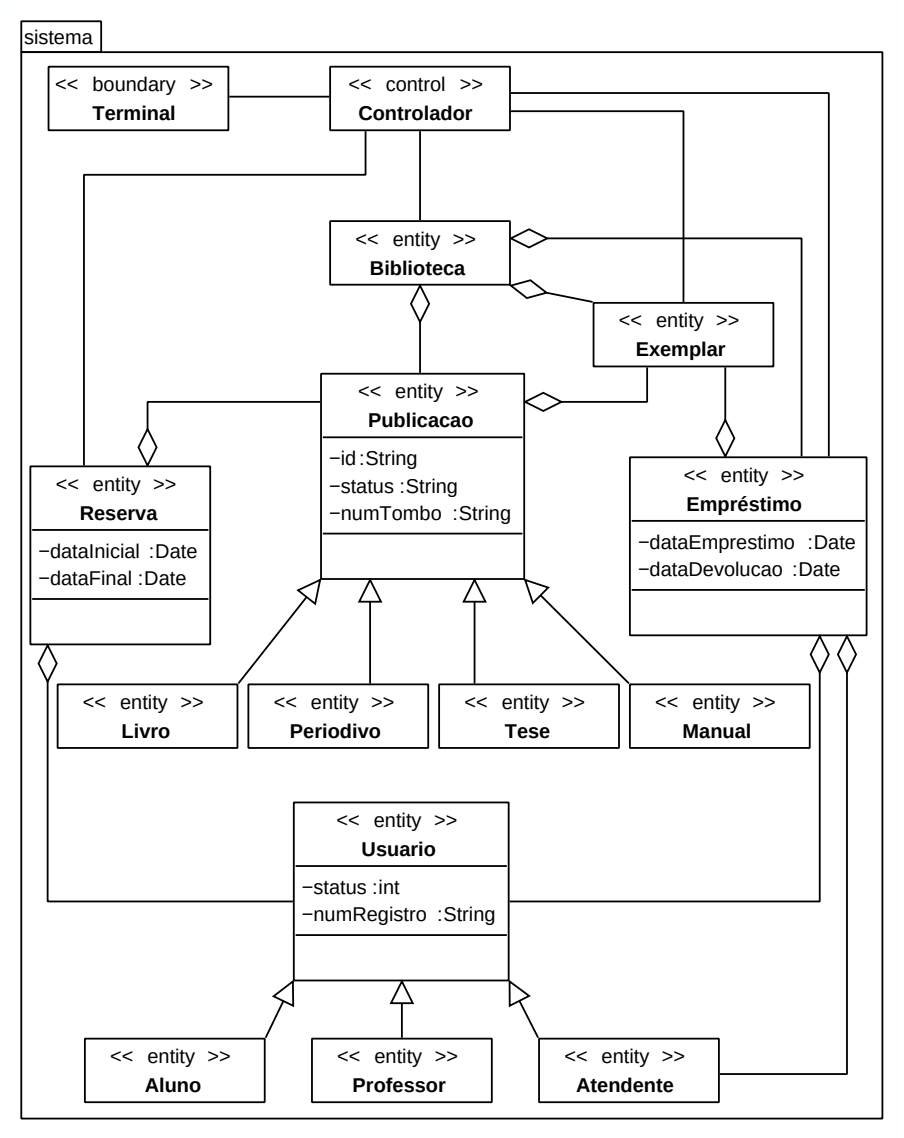



Diagrama de Classes de Análise Refinado