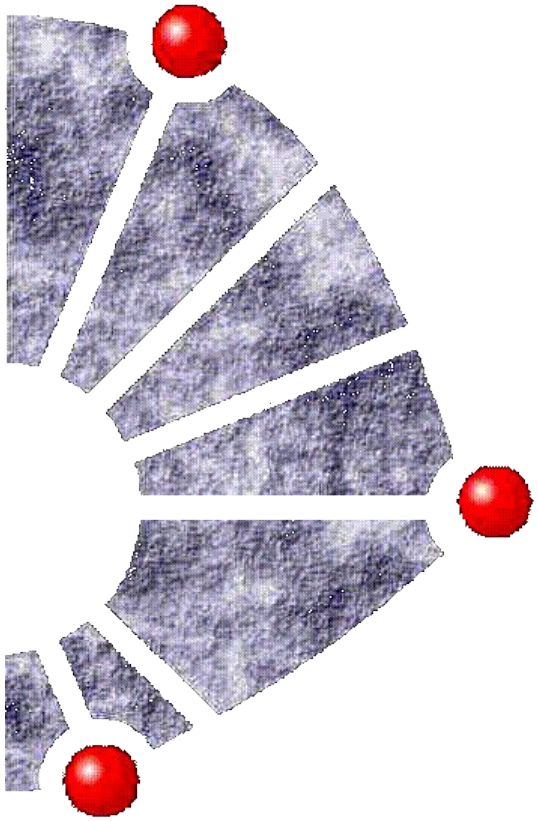
O Processo Unified e a Fase de Análise



FEEC-UNICAMP

Ricardo Gudwin

Análise

- **Análise dos Requisitos:** aprofundamento das investigações, detalhamento e refinamento das especificações dos requisitos
- **Objetivos**
  - obter uma melhor compreensão dos requisitos, mantendo uma descrição dos requisitos que seja compreensível e auxilie no posterior *design* do sistema
  - transladar as especificações dos requisitos que se encontram na "linguagem do cliente" para uma representação que use uma "linguagem do desenvolvedor"
  - passar de um modelo de casos de usos para um modelo de análise
- **Principal Desafio:** manter um nível abstrato de investigação, sem entrar em questões de design

Análise

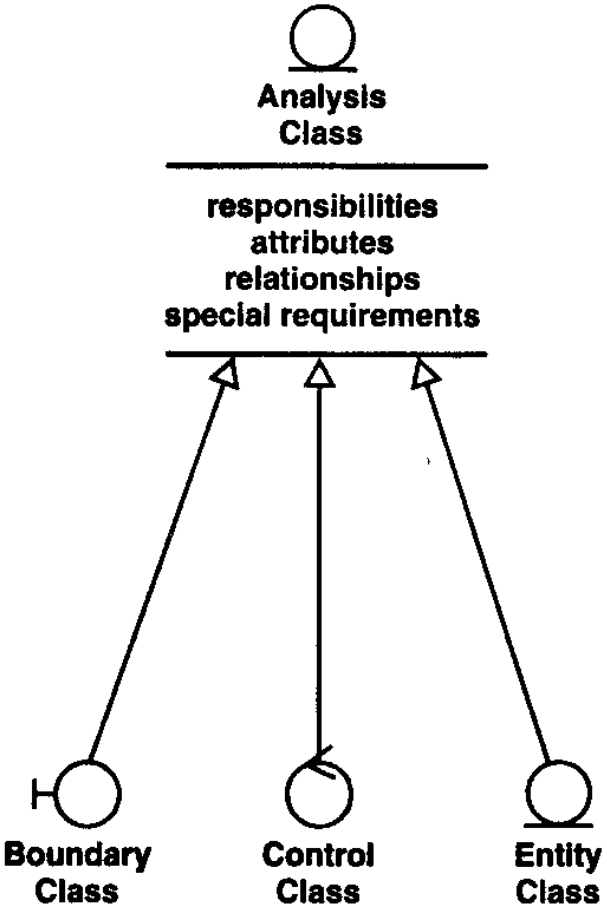
- Modelo de Análise
  - permite o refinamento do modelo de casos de uso por meio da especificação dos detalhes internos do sistema a ser construído
  - provê um poder de expressão e formalismo mais refinados, principalmente para a descrição da dinâmica do sistema
  - entretanto, utiliza **abstrações** e evita a solução definitiva de alguns problemas que são postergados para a fase de desig
- Estrutura desenvolvida na fase de Análise
  - nem sempre pode ser preservada durante o design, mas deve ser negociada e compromissada quanto à flexibilidade que permite
- Análise x Design
  - Análise = o quê ?
  - Design = como ?

Análise

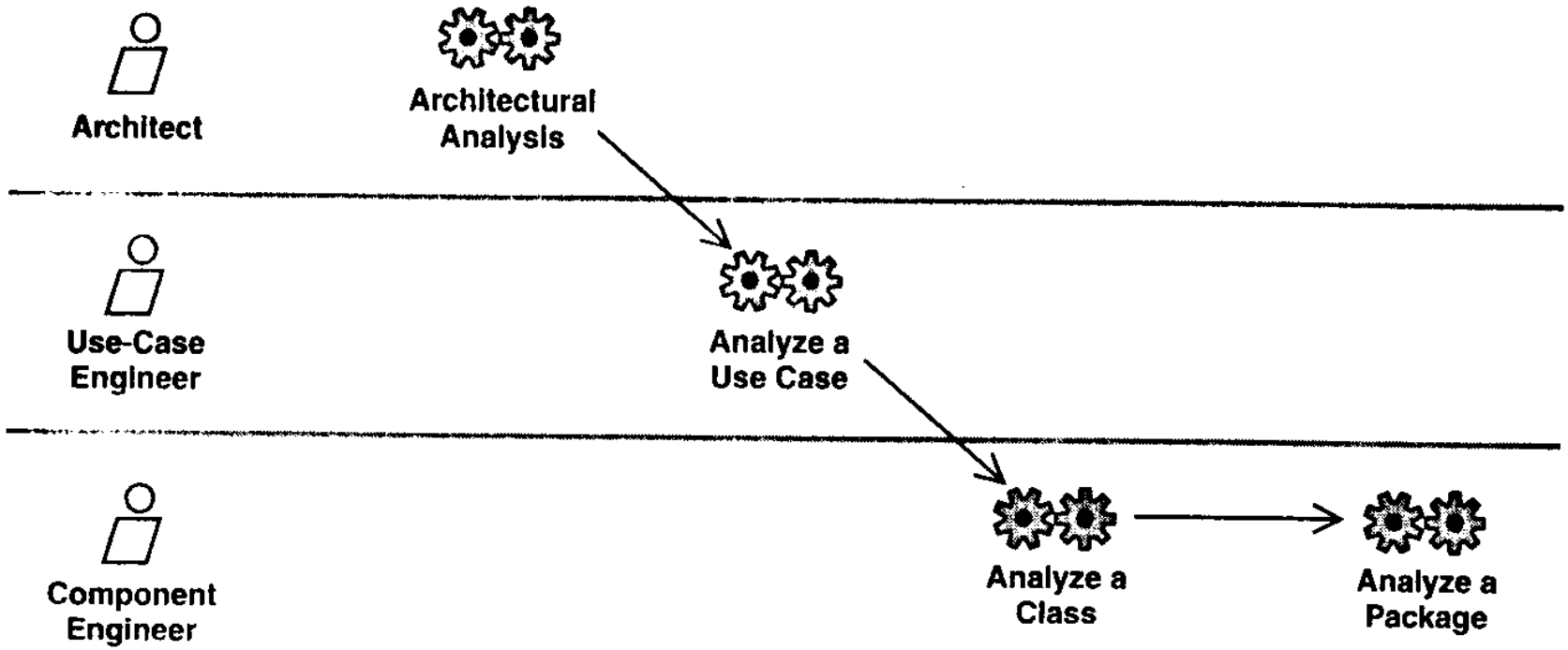
- Artefatos de Análise
  - Classes de Análise e Pacotes de Análise
  - Realização de Casos de Uso - Análise
  - Descrição da Arquitetura (Visão do Modelo de Análise)
- Classes de Análise
  - representam abstrações de uma ou mais classes ou subsistemas que irão aparecer no design do sistema
- Características
  - focalização nos requisitos funcionais, postergando os requisitos não-funcionais (tais como os detalhes da interface, etc...)
  - definição das interfaces em termos de responsabilidades
  - atributos de alto nível e relacionamentos conceituais
  - estereótipos “boundary”, “control” e “entity”

Classes Estereotipadas

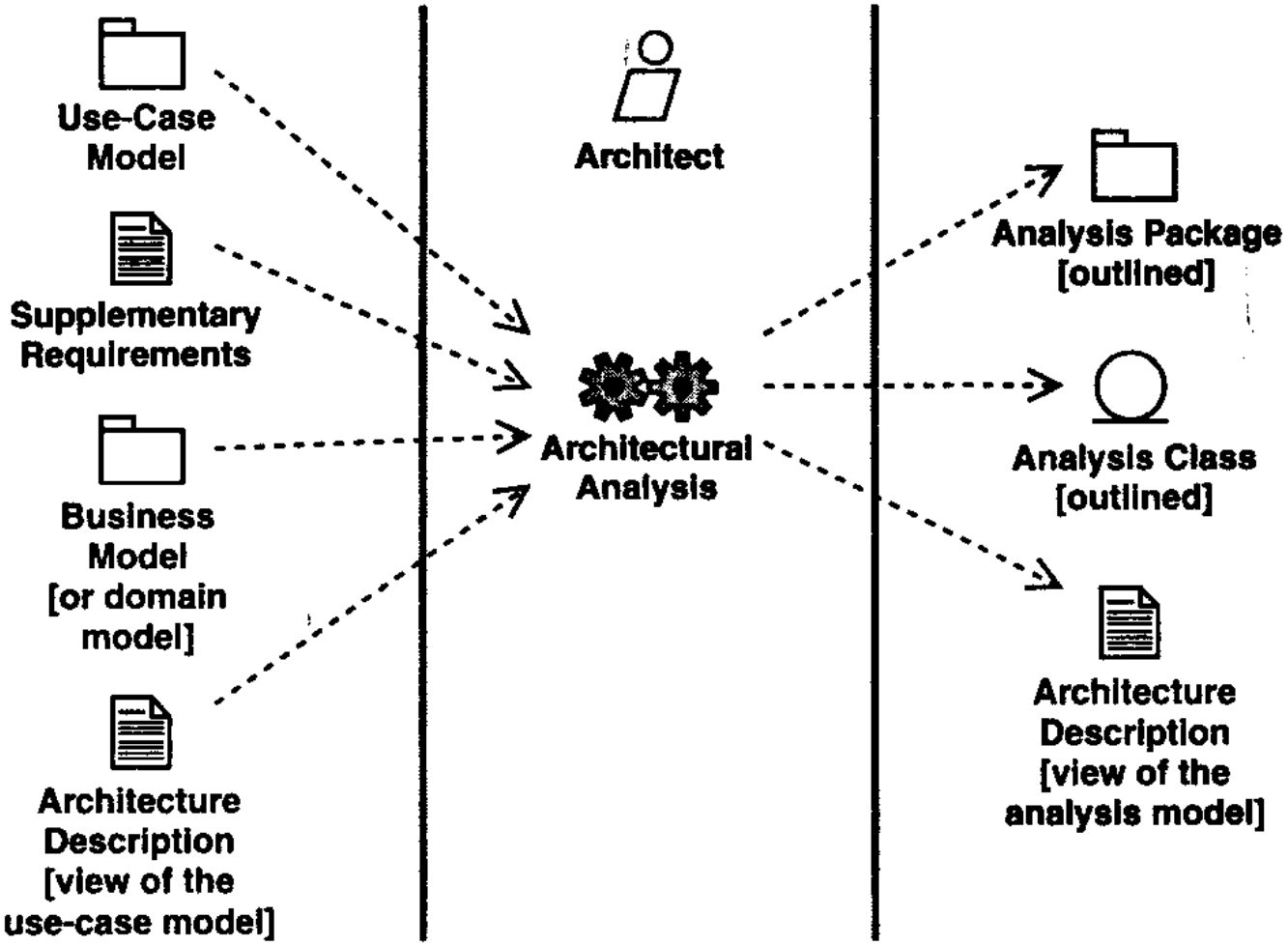
- Classes Boundary
  - utilizada para modelar a interação entre o sistema e os atores
  - interação envolve o recebimento e apresentação de informações
- Classes Control
  - representam a coordenação, sequenciamento, transações e controle de outros objetos
  - derivações e cálculos
- Classes Entity
  - modela informações (dados) de longa duração, frequentemente persistentes



Análise


Architect
Architectural Analysis
Use-Case Engineer
Analyze a Use Case
Component Engineer
Analyze a Class
Analyze a Package

Análise Arquitetural



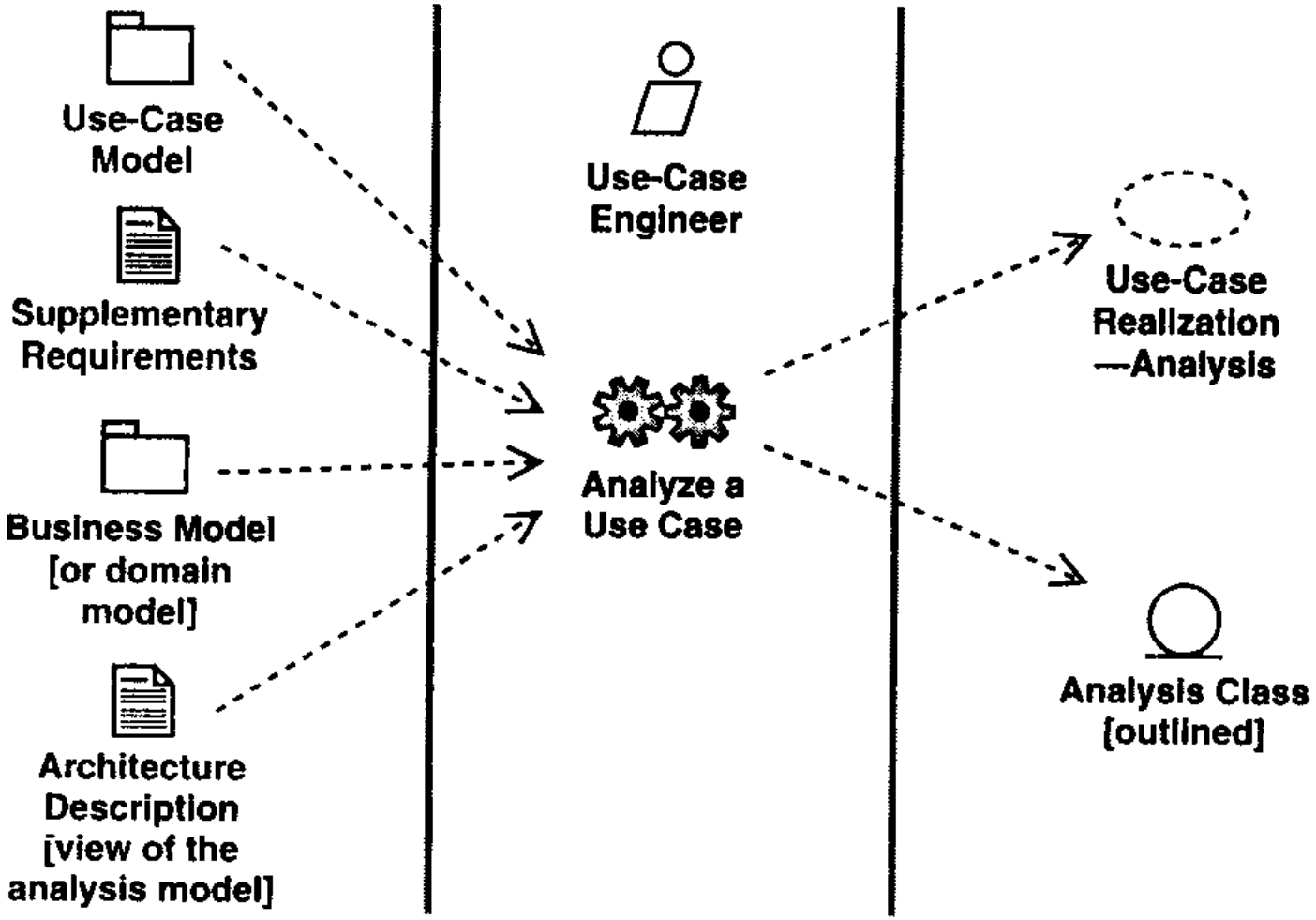
Análise Arquitetural

- Objetivo
  - gerar um **outline** do modelo de análise e da arquitetura por meio da identificação dos pacotes de análise, das classes de análise e de requisitos especiais necessários
- Identificação dos Pacotes de Análise
  - pacotes de análise permitem uma melhor organização do modelo de análise, dividindo-o em partes menores mais fáceis de serem gerenciadas
  - podem ser uma maneira de se dividir o trabalho entre diferentes integrantes da equipe
  - alocação de casos de usos com funcionalidades semelhantes em pacotes comuns
    - casos de uso que suportam processos organizacionais comuns
    - casos de uso que suportam um ator específico
    - casos de uso relacionados por generalização, extensão ou inclusão

Análise Arquitetural

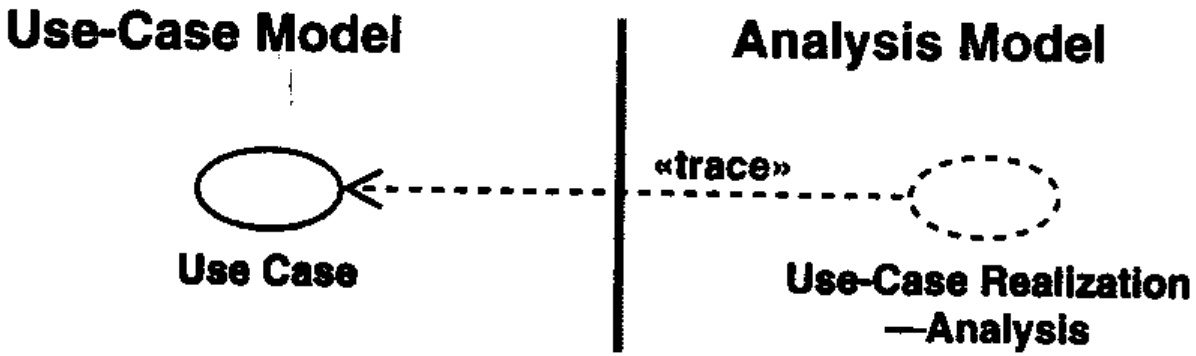
- Identificação de Classes Óbvias do tipo "Entity"
  - proposta preliminar das classes do tipo "entity" mais importantes (10 a 20 classes) baseando-se em entidades encontradas no domínio ou nas entidades organizacionais
  - as classes que não forem utilizadas posteriormente na realização de nenhum caso de uso devem ser eliminadas
- Identificação de Requisitos Especiais
  - requisitos especiais são aqueles que são importantes na determinação posterior no design e que não foram necessariamente levantados na especificação dos requisitos
  - persistência, distribuição ou concorrência, restrições de segurança, tolerância a falhas e gerenciamento de transações
  - em alguns casos, os requisitos especiais só poderão ser encontrados durante a realização dos casos de uso

Análise dos Casos de Uso



Análise dos Casos de Uso

- Objetivos
  - identificação das classes de análise cujos objetos são necessários para implementar o fluxo de eventos de um caso de uso em particular
  - distribuição do comportamento em um caso de uso por um conjunto de objetos interagindo entre si
  - captura dos requisitos especiais relativos à realização de um caso de uso em particular



Análise dos Casos de Uso

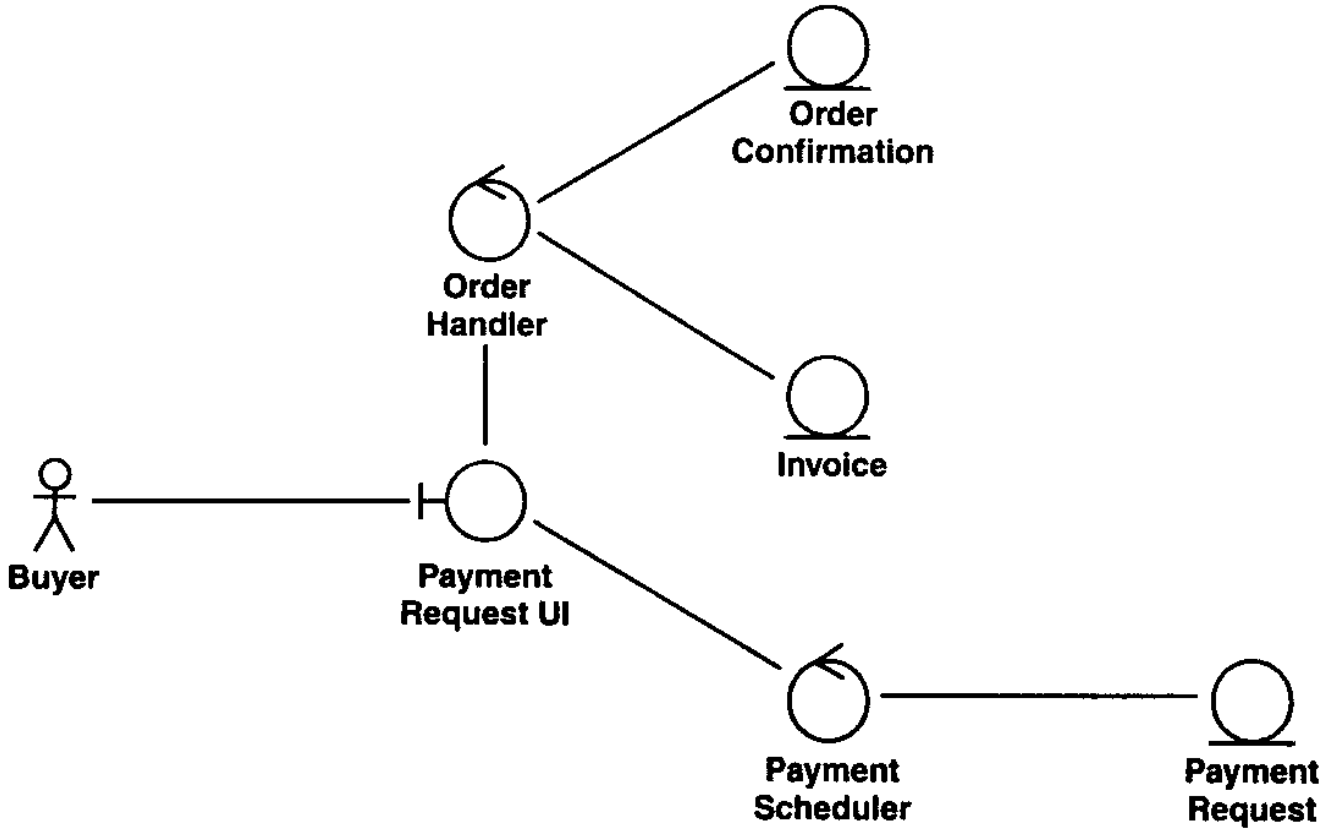
- Identificação de Classes de Análise
  - identificação das classes do tipo control, entity e boundary necessárias para a realização de um caso de uso, determinando-se seus nomes, responsabilidades, atributos e relacionamentos
- Guidelines para a identificação de classes
  - classes do tipo "entity" podem ser obtidas por meio do estudo das descrições dos **casos de uso** e do **modelo de domínio**
  - deve haver uma classe do tipo "boundary" para cada ator participante de um caso de uso, sendo que esta classe representa a janela da interface com o usuário com a qual o ator está interagindo
  - deve haver uma ou mais classes do tipo "control" que coordenam a realização do caso de uso

Análise dos Casos de Uso

- Descrevendo a Interação entre Objetos de Análise
  - após a determinação das classes de análise, é necessária a descrição das interações entre objetos dessas classes
  - essa descrição pode ser feita por meio de diagramas de colaboração contendo os atores, os objetos de análise e as conexões entre eles
  - caso hajam fluxos distintos, deve haver um diagrama de colaboração para cada fluxo
- Captura de Requisitos Especiais
  - neste passo, capturam-se todos os requisitos, tais como os requisitos não funcionais relacionados com a realização de um caso de uso que forem descobertos durante a fase de análise e que forem pertinentes para os posteriores design e implementação

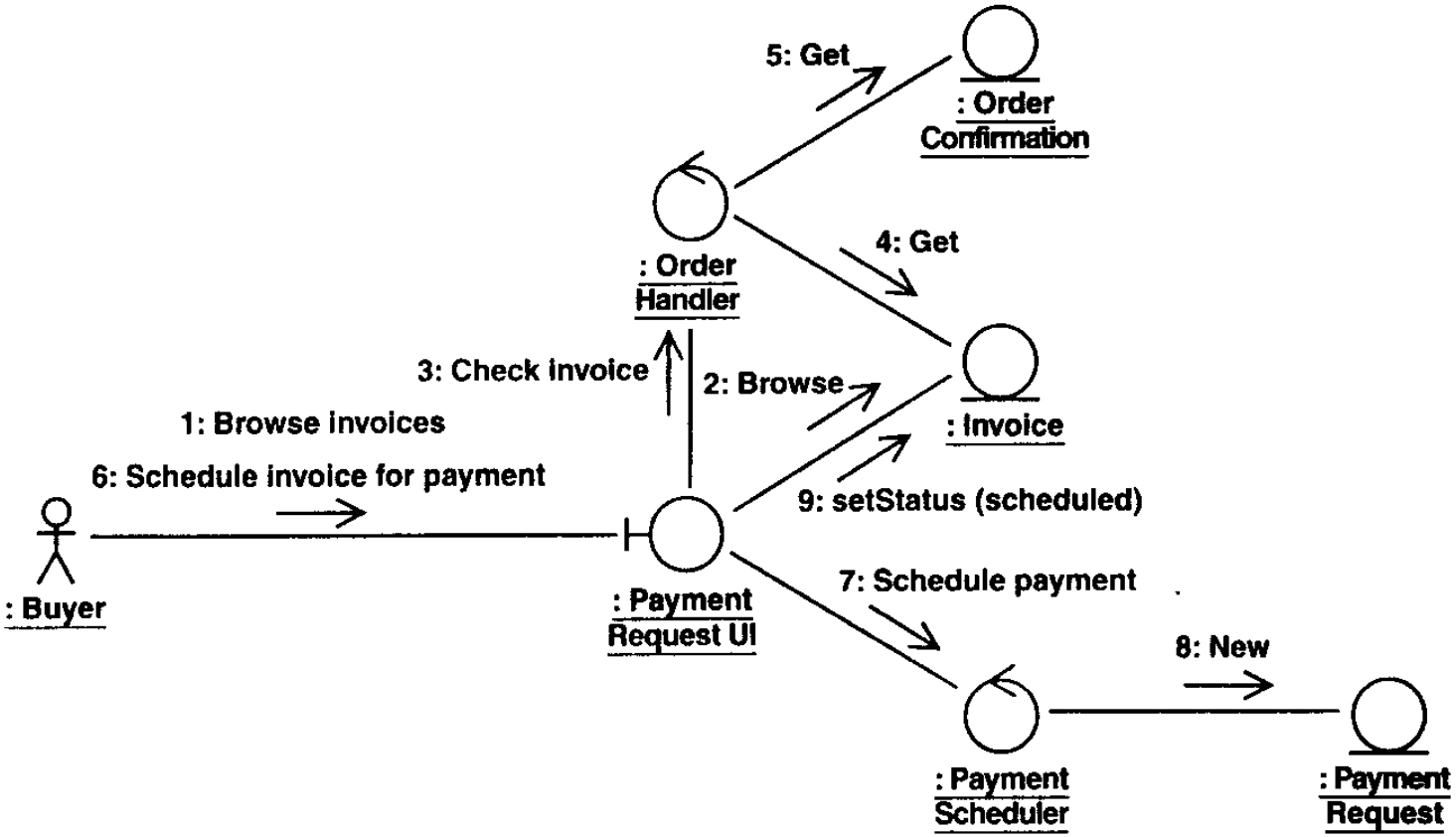
Análise de Casos de Uso

- Exemplo de Diagrama de Classes de Análise

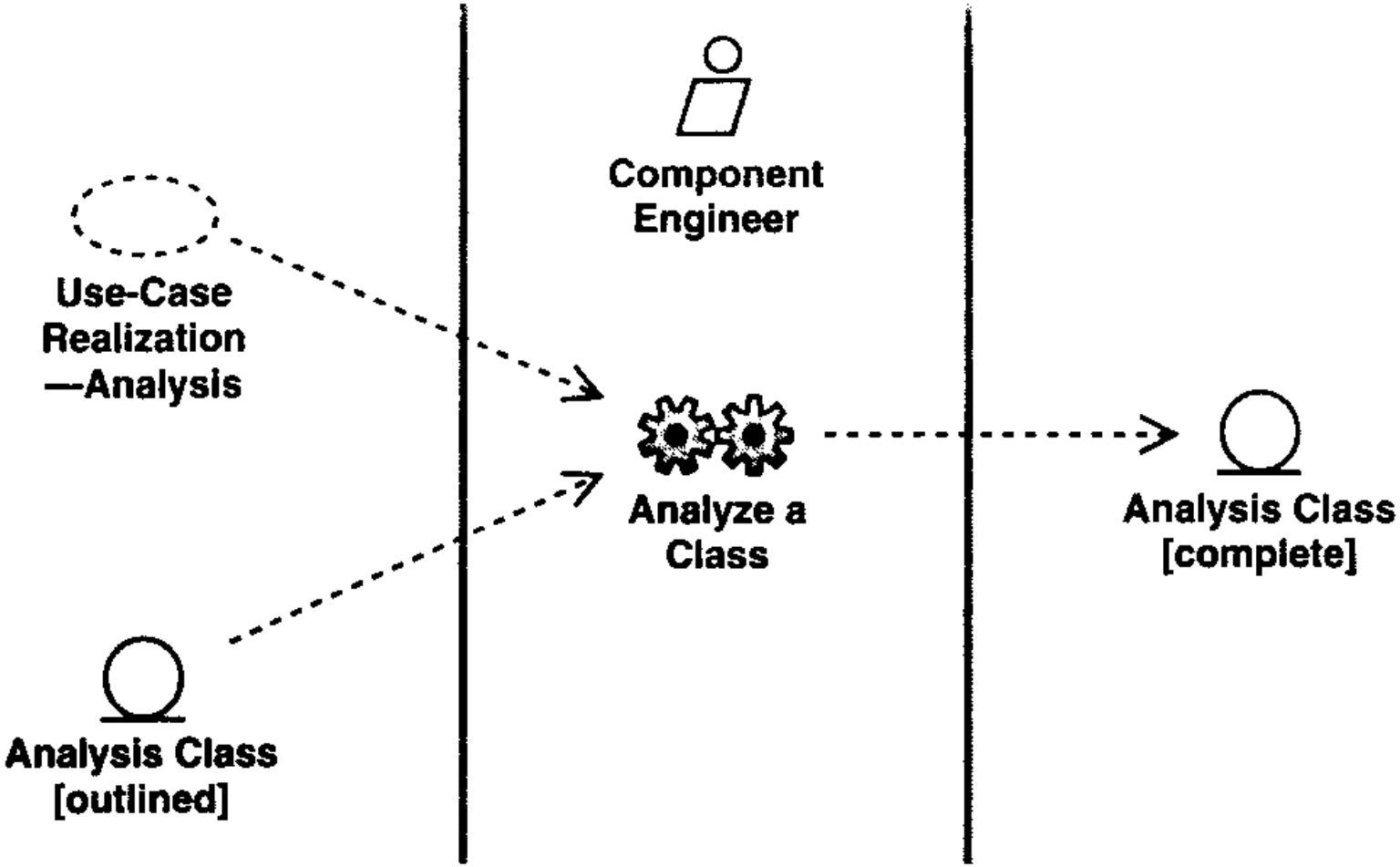


Análise de Casos de Uso

- Exemplo de Diagrama de Colaboração da Análise



Análise de uma Classe



Análise de uma Classe

- Objetivos
  - identificação e formalização das responsabilidades de uma classe de análise, baseadas em seu papel na realização de um caso de uso
  - identificação e formalização dos atributos e relacionamentos de uma classe de análise
  - captura de requisitos especiais na realização de uma classe de análise

- Identificação de Responsabilidades
  - responsabilidades podem ser identificadas por meio dos papéis que estas assumem em diferentes realizações de casos de uso
  - essas realizações encontram-se nos diagramas de classes e diagramas de interação que realizam cada caso de uso
  - responsabilidades estão associadas às "mensagens" que uma classe pode receber

Análise de uma Classe

- Descrição de Responsabilidades por meio de Contratos
  - Contratos: descrevem as responsabilidades de uma determinada classe, em termos de quais mudanças no estado do objeto são realizadas quando este recebe mensagens (ou invocações)
  - descrevem O QUE o objeto deve fazer, sem explicar COMO ele o faz
- Para cada mensagem aparecendo em um diagrama de colaboração, deve haver um contrato correspondente
- Seções de um Contrato
  - Um contrato está dividido em seções
  - Cada seção provê informações sobre uma parte específica do contrato
  - Nem todas as seções são obrigatórias, devendo aparecer quando necessário

Análise de uma Classe

- Seções de um Contrato
  - **Nome:** nome da operação e eventualmente seus parâmetros
  - **Responsabilidades:** uma descrição informal das responsabilidades que a operação deve implementar
  - **Tipo:**conceitual, classe de software ou interface
  - **Referências Cruzadas:** outras classes que utilizam o mesmo contrato, etc.
  - **Notas:** informações adicionais, algoritmos, etc.
  - **Exceções:** casos excepcionais
  - **Saídas Secundárias:** saídas não relacionadas à interface com o usuário, e.g. mensagens de erros, *logs*, etc.
  - **Pré-Condições:** condições referentes ao estado do sistema antes da execução da operação
  - **Pós-Condições:** estado do sistema após a conclusão da operação

Análise de uma Classe

- Como Fazer um Contrato
  - Identificar as operações a serem reguladas por contratos a partir dos diagramas de colaboração
  - Para cada operação, deve-se implementar um contrato
  - Começa-se pela seção "Responsabilidades", descrevendo informalmente o propósito da operação
  - Deve-se então passar à seção de "Pós-Condições", descrevendo declarativamente as mudanças no estado do objeto que devem ocorrer em virtude da operação
  - Para descrever as pós-condições, devemos atentar para os seguintes pontos:
    - Criação e Destruição de Instâncias de outros objetos
    - Modificação de Atributos
    - Formação ou Destruição de Associações

Análise de uma Classe

- Observações sobre Contratos
  - As pós-condições são a parte mais importante de um contrato
  - Não devemos confundir as Saídas Secundárias com Pós-Condições
  - Pós-condições não são ações a serem executadas durante a operação, mas declarações sobre o estado do objeto após a conclusão da operação contratada
  - Pós-condições dizem respeito ao Modelo de Análise
    - as classes dos objetos criados devem existir no diagrama de classes, do mesmo modo que as associações
  - Durante a criação de contratos, é comum se descobrir a necessidade de uma nova classe, atributo ou associação não anteriormente definida. Neste caso, elas devem ser incluídas de modo a atualizar o modelo de análise

Análise de uma Classe

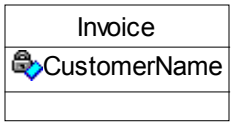
- Quão completas devem ser as Pós-Condições ?
  - A geração de um conjunto de pós-condições completo e acurado é muito improvável - e nem mesmo necessária - durante a fase de análise
  - detalhes maiores poderão ser descobertos durante a fase de design
- Pré-Condições
  - definem condições sobre o estado do sistema que são necessárias para que a operação possa ser realizada
  - Exemplos
    - Coisas que se deve testar a certo ponto da operação
    - Coisas que não serão testadas, mas que são determinantes no sucesso da operação - devem ser documentadas para auxiliar a futura implementação da operação em fases futuras

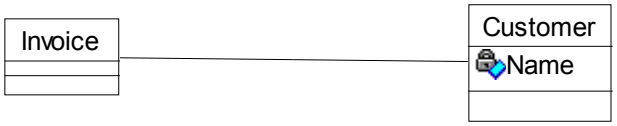
Análise de uma Classe

- Identificando Atributos
  - atributos especificam propriedades de uma classe de análise
  - relacionados com as responsabilidades da classe
- Guidelines para o Levantamento de Atributos
  - nome de um atributo deve ser um substantivo
  - tipos de atributos devem ser abstratos
    - por exemplo - "quantidade", ao invés de "inteiro"
  - deve-se tentar reutilizar ao máximo os tipos utilizados
  - atributos são atrelados a cada instância de objeto, e não à classe
  - se houver muitos atributos, alguns deles devem ser transformados em classes
  - nem sempre é necessário ter atributos

Análise de uma Classe

- Que atributos incluir ?
  - Somente aqueles que forem realmente necessários para satisfazer os requisitos de informações relevantes para os casos de uso sendo analisados
  - aqueles que devem ser "lembrados" para serem utilizados em algum ponto de um caso de uso
- Modelando Quantidades e Unidades de Atributos
  - devemos usar um conceito separado "unidade", que é associado a quantidades
- Evitar Referências a Atributos de Classes Externas como Atributos

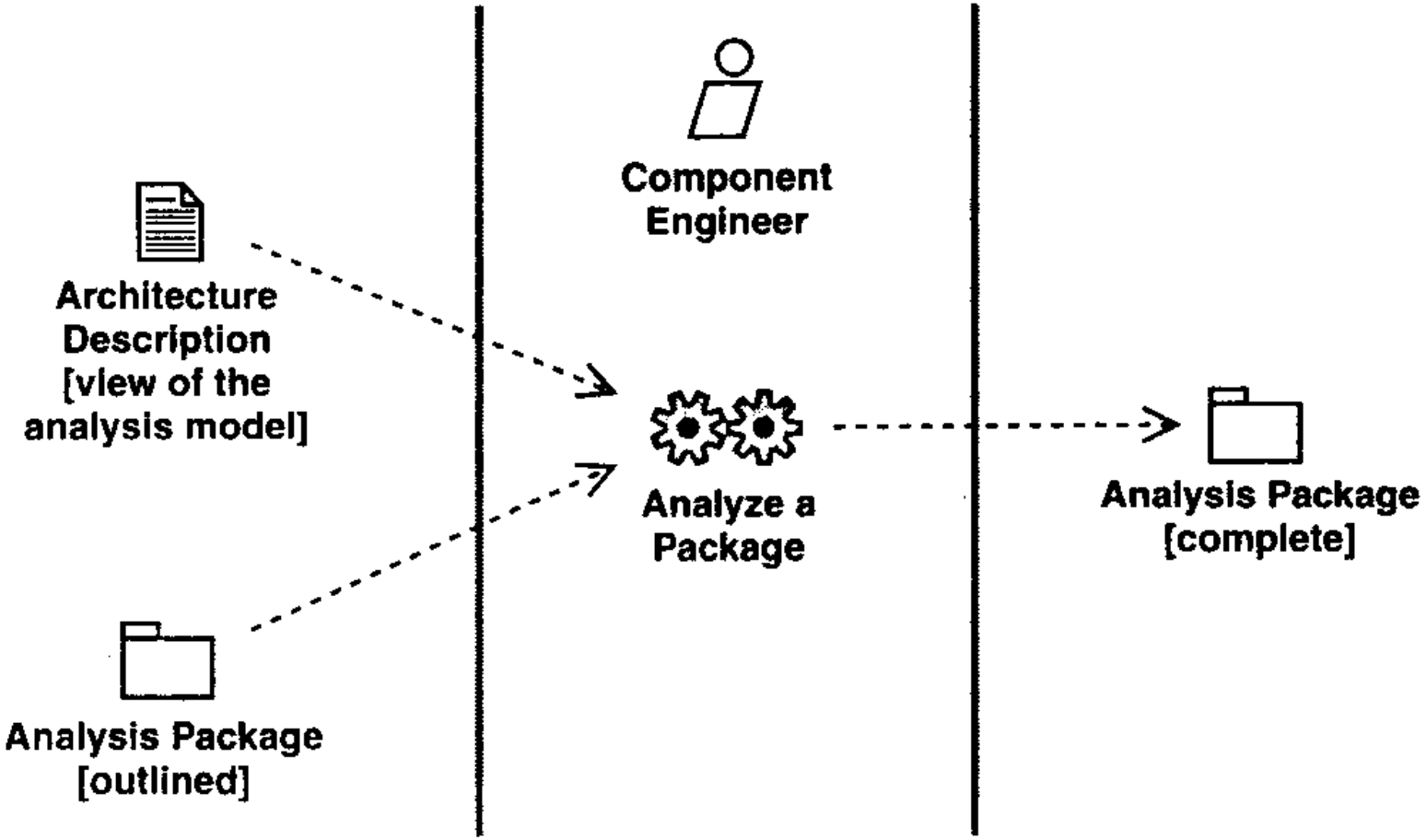




Análise de uma Classe

- Identificando Associações e Agregações
  - objetos de análise interagem entre si por meio de conexões em diagramas de colaboração
  - estas conexões são normalmente instâncias de associações entre as classes correspondentes
  - deve-se estudar as conexões necessárias em um diagrama de colaborações de modo a determinar as associações necessárias no diagrama de classes
  - o número de relacionamentos entre classes deve ser minimizado
  - deve-se modelar somente os relacionamentos realmente necessários para se modelar a realização dos casos de uso
- Identificando Generalizações
  - devem ser utilizadas para extrair comportamentos comuns ou compartilhados entre diferentes classes de análise

Analisando Pacotes



Analisando Pacotes

- Objetivos
  - garantir a independência entre diferentes pacotes
  - garantir que um pacote de análise implementa seu propósito de realizar classes de domínio ou casos de uso
  - descrever dependências entre pacotes, de tal forma que o efeito de mudanças futuras possa ser estimado
- Guidelines para a Análise de Pacotes
  - definir e formalizar as dependências entre pacotes, dependendo das classes que estes contiverem
  - garantir que os pacotes contenham as classes corretas. Deve-se tentar manter os pacotes coesos, por meio da inclusão somente de objetos que estejam funcionalmente correlacionados
  - tentar limitar as dependências entre pacotes - deve-se considerar a relocação de classes que criem muitas dependências entre pacotes