



APLICAÇÃO DA UML AO LONGO DO PROCESSO DE DESENVOLVIMENTO

MODELAÇÃO E ANÁLISE DE SISTEMAS | TP

ILÍDIO OLIVEIRA ico@ua.pt
v2022-12-16

Aplicações principais da UML

estrutura e comportamento de sistemas de software

análise, desenho e implementação de sistemas baseados em software

elementos do modelo representam entidades do mundo do software

especialmente adequada para o desenvolvimento orientado por objetos

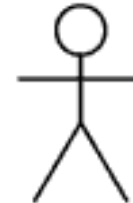
processos organizacionais novos ou já existentes

especificar ou documentar processos de negócio

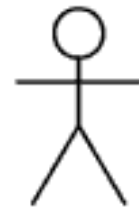
não implica ou assume uma implementação em software



Analista

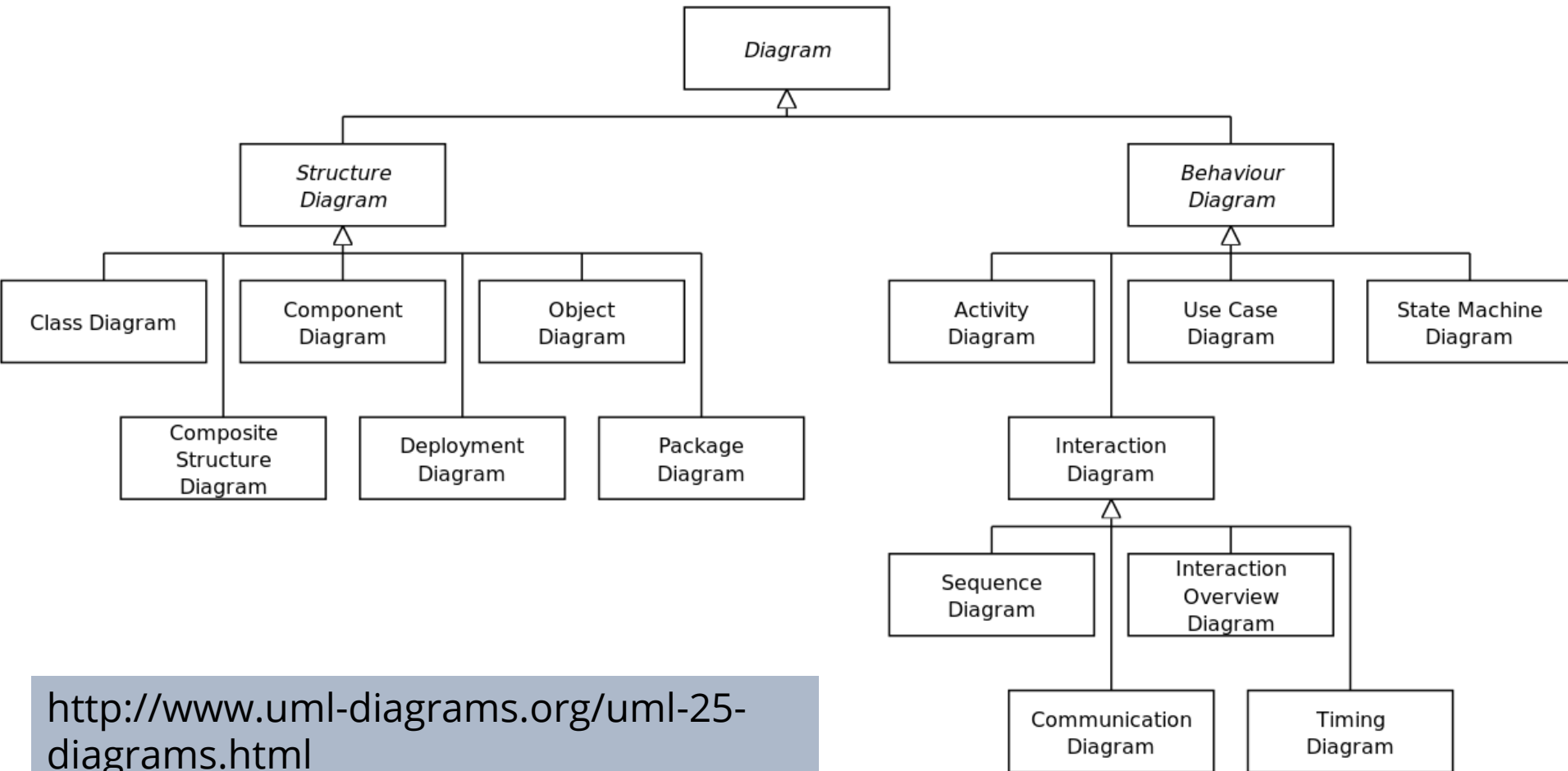


Programador



Arquiteto

UML ver. 2.0

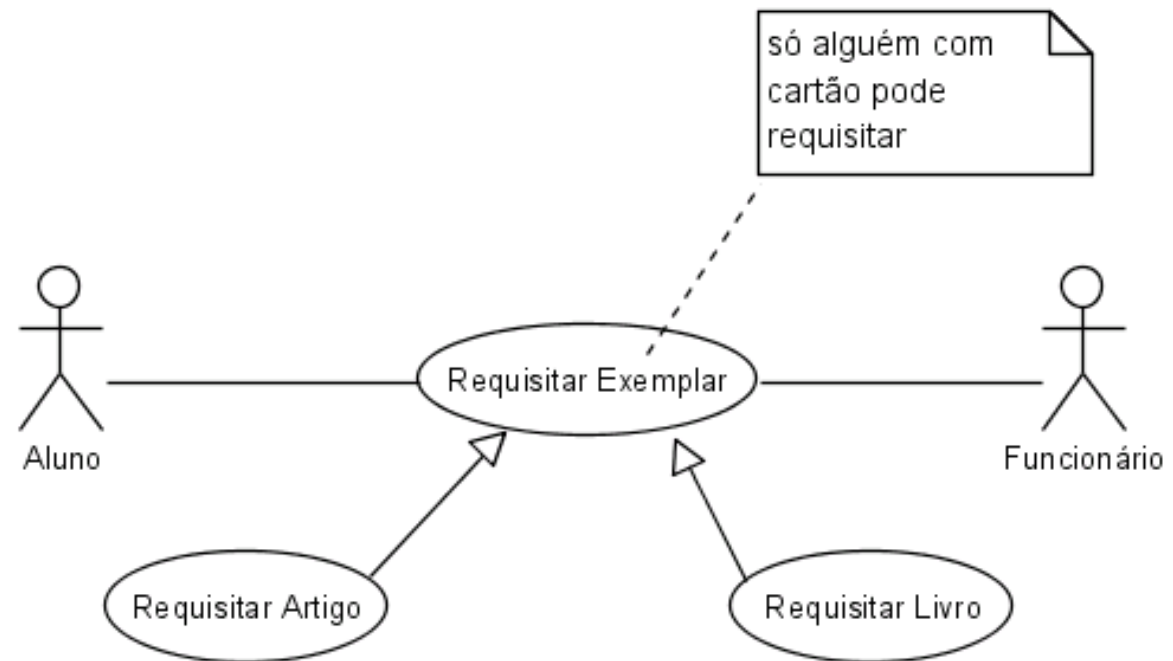


<http://www.uml-diagrams.org/uml-25-diagrams.html>

ELEMENTOS COMUNS

Anotações

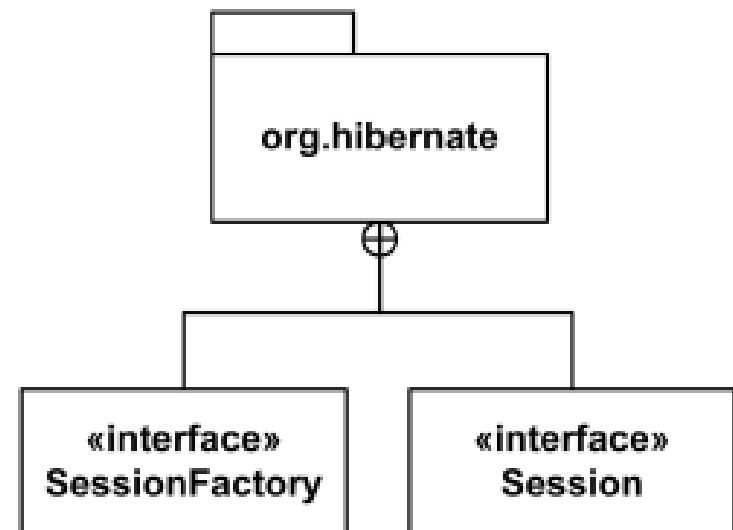
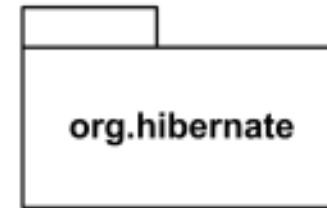
Um comentário que pode ser usado para anotar qualquer elemento



Pacotes

um mecanismo para dividir um modelo em partes

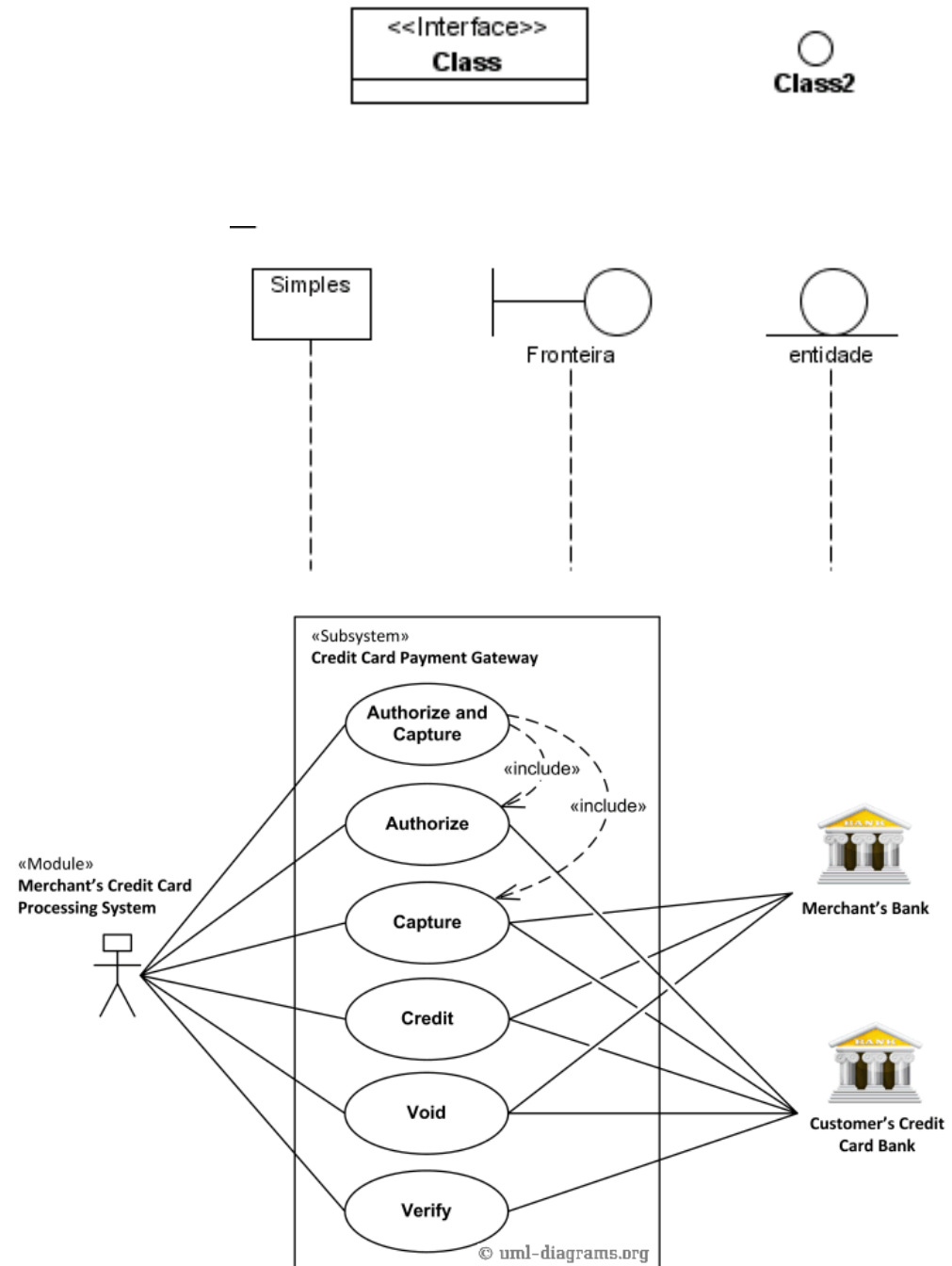
serve como mecanismo genérico para fazer agrupamentos



Estereotipo (stereotype)

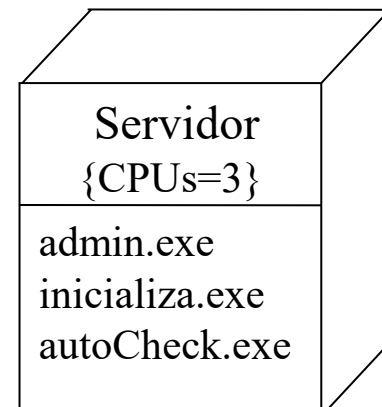
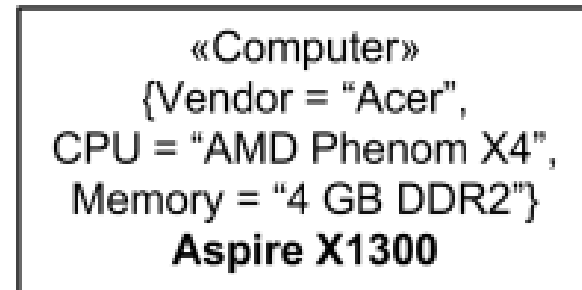
uma especialização da
semântica de um elemento do
modelação

marcada com «...» ou com a
alteração da decoração



Valores etiquetados (*tagged values*)

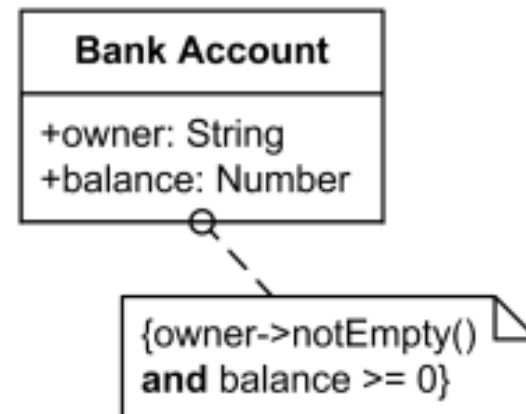
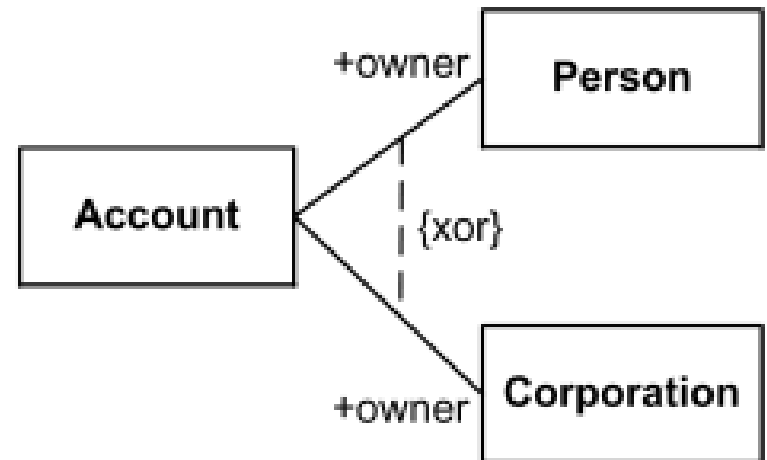
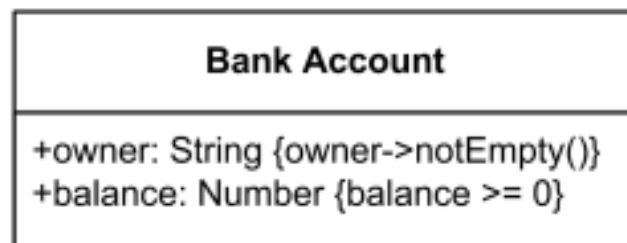
Estender elementos do modelo com uma linguagem “computável” (pares atributo/valor)



Restrições

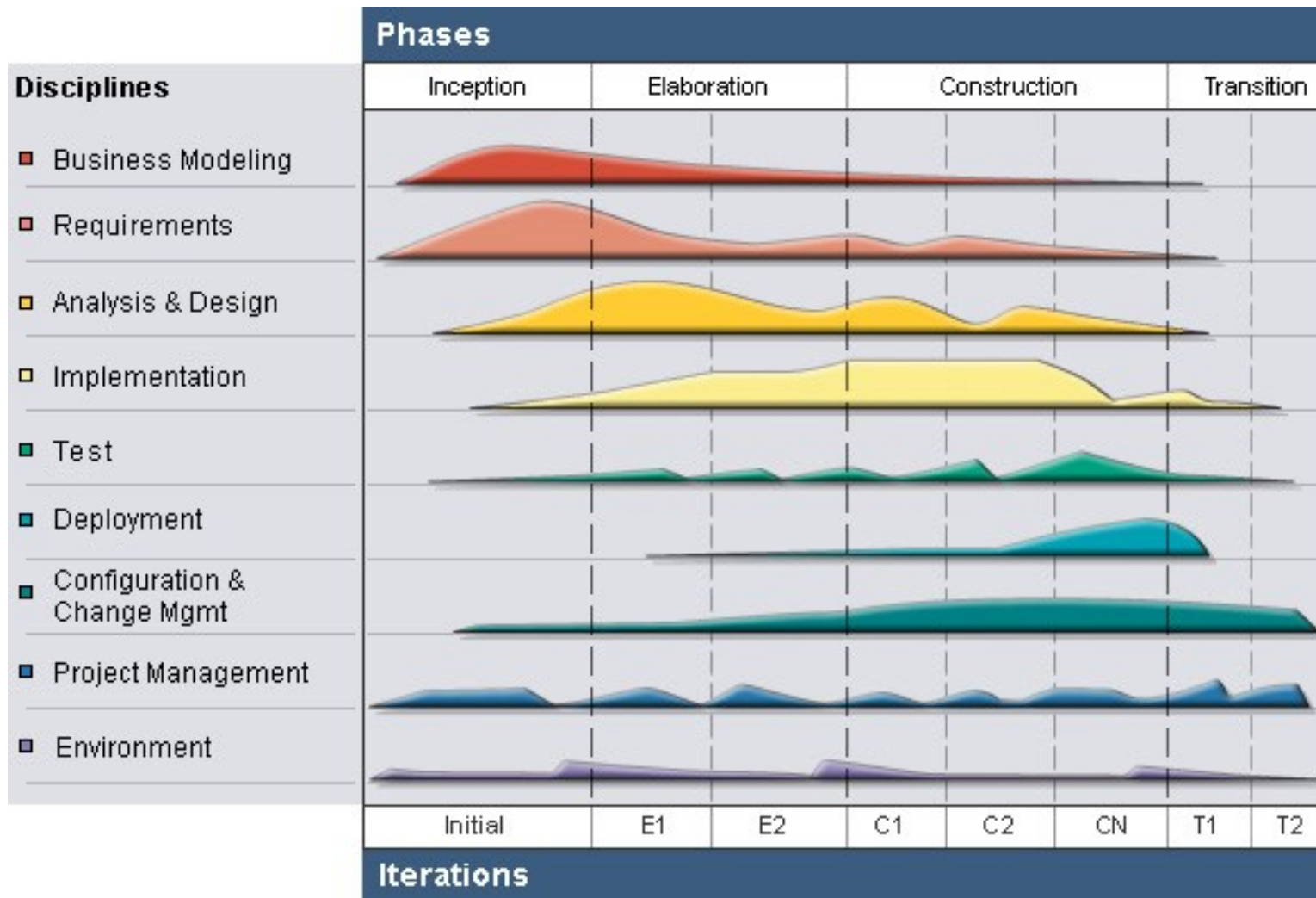
Linguagem para adicionar regras ao modelo ou condicionar a sua interpretação

condição ou restrição relacionada com um ou mais elementos

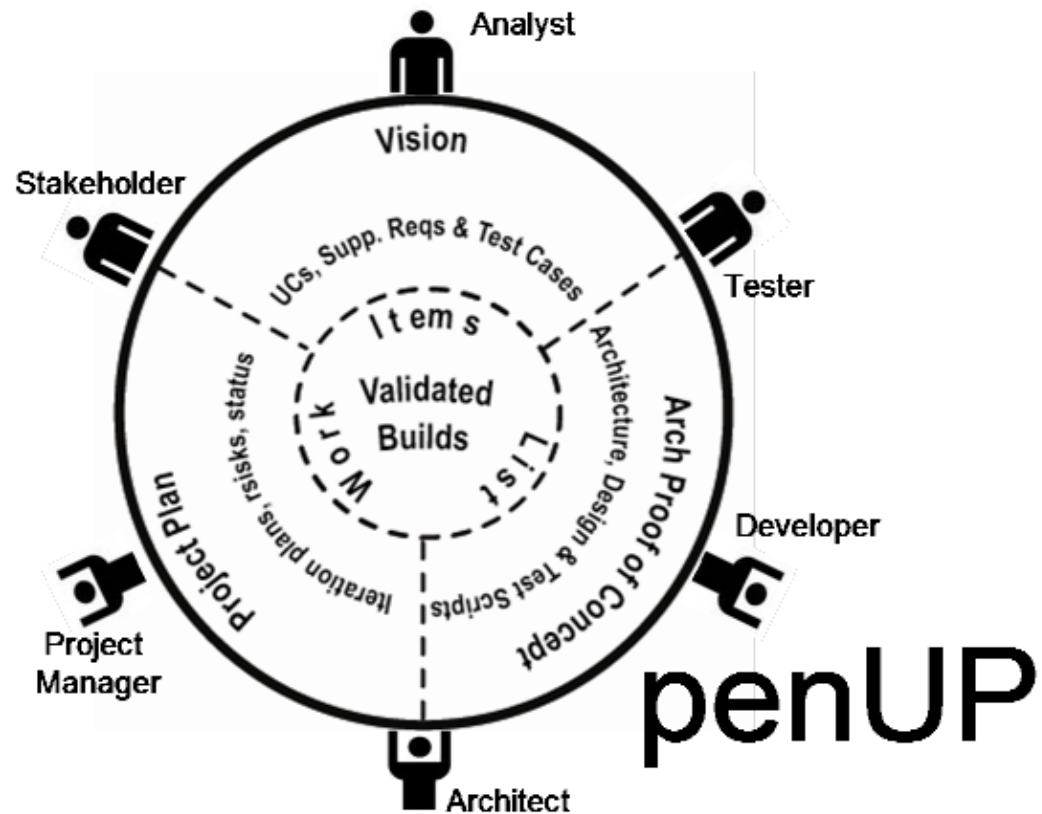


UML AO LONGO DO PROCESSO DE SOFTWARE

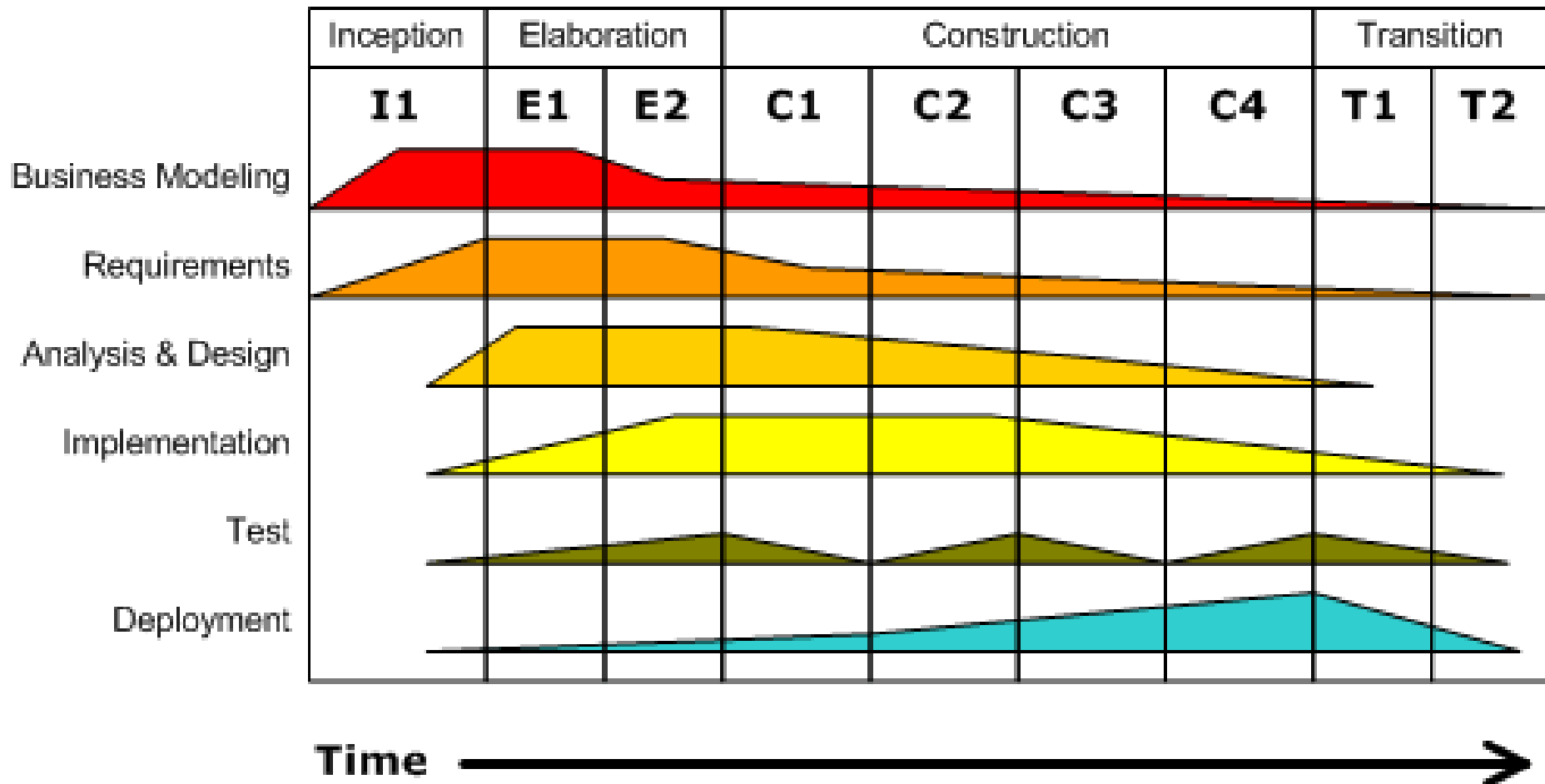
Modelos ao longo do SDLC



UML & OpenUP



São usadas várias disciplinas em cada iteração, com intensidade distinta



O nível de abstração da análise varia ao longo do projeto

	USE-CASE MODEL	USE-CASE NARRATIVE	USE-CASE REALIZATION	TEST CASE	SUPPORTING INFORMATION
SKETCH:		BRIEFLY DESCRIBED		TEST IDEAS FORMULATED	OUTLINED
BARE ESSENTIALS:	VALUE ESTABLISHED	BULLETED OUTLINE	IMPLEMENTATION ELEMENTS IDENTIFIED	SCENARIO CHOSEN	SIMPLY DEFINED
ENHANCED:	SYSTEM BOUNDARY ESTABLISHED	ESSENTIAL OUTLINE	RESPONSIBILITIES ALLOCATED	VARIABLES IDENTIFIED	MODELLED & ILLUSTRATED
EXPANDED:	STRUCTURED	FULLY DESCRIBED	INTERACTION DEFINED	VARIABLES SET	COMPREHENSIBLY DEFINED
FURTHER EXPANDED:				SCRIPTED /AUTOMATED	

FIGURE 14: WORK PRODUCT LEVELS OF DETAIL

I Oliveira (2017)

USE-CASE 2.0 - The Guide to Succeeding with Use Cases by Ivar Jacobson, Ian Spence & Kurt Bittner

http://www.outsideininc.com/wp-content/uploads/2012/02/Use-Case-2_0_Feb14_2012.pdf










or inclass readings

Resultados do OpenUP

Architecture

-  Architecture Notebook

Deployment






-  Product Documentation
-  Support Documentation
-  User Documentation
-  Training Materials
-  Backout Plan
-  Deployment Plan
-  Infrastructure
-  Release Communications
-  Release Controls

-  Release




Development

-  Implementation
-  Build
-  Developer Test
-  Design

Requirements

-  Glossary
-  Vision
-  System-Wide Requirements
-  Use-Case Model
-  Use Case

Test

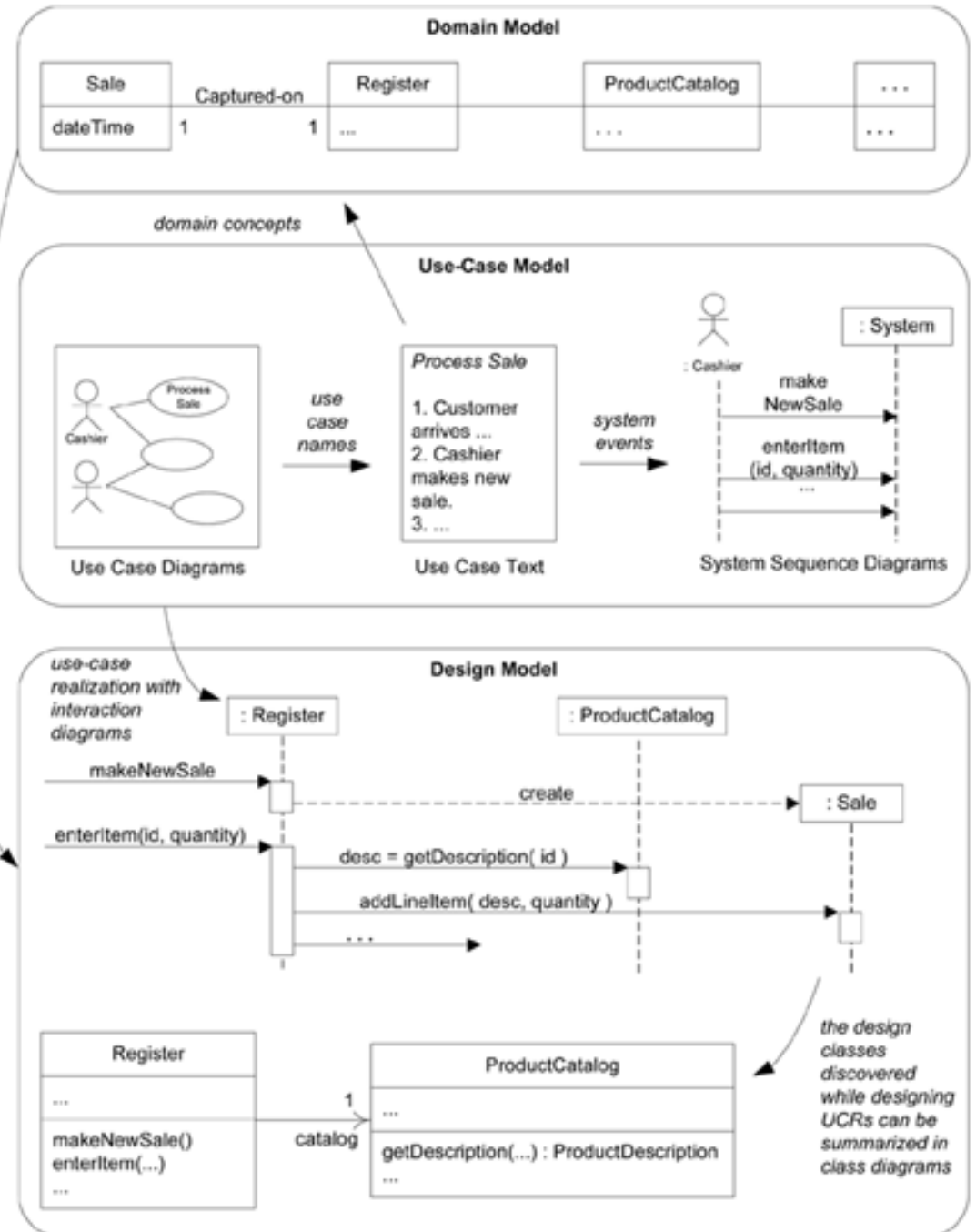
-  Test Case
-  Test Script
-  Test Log

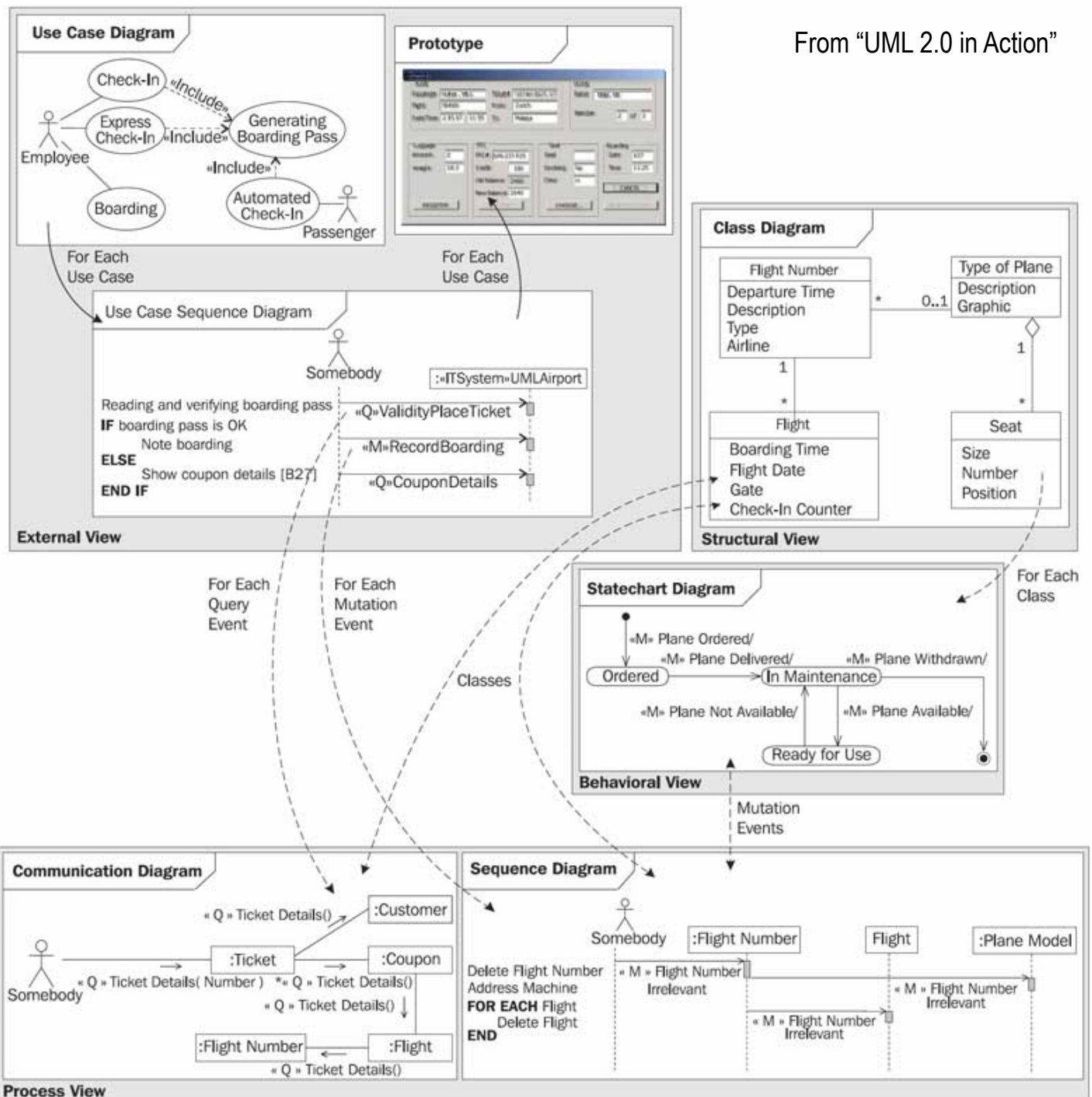
Sample Unified Process Artifact Relationships

Visão geral dos resultados da aplicação do Unified Process

In: Larmam

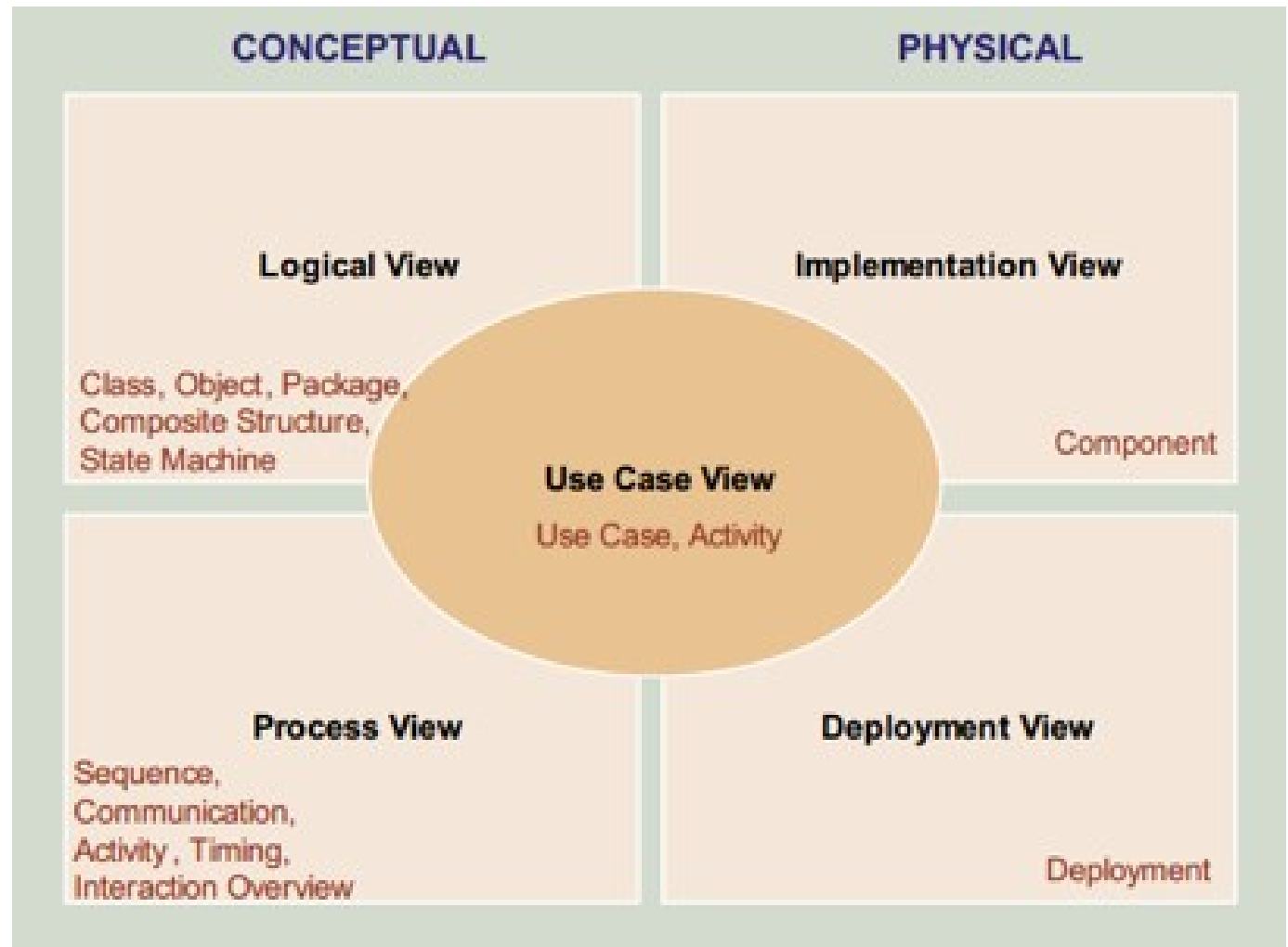
conceptual classes in the domain inspire the names of some software classes in the design



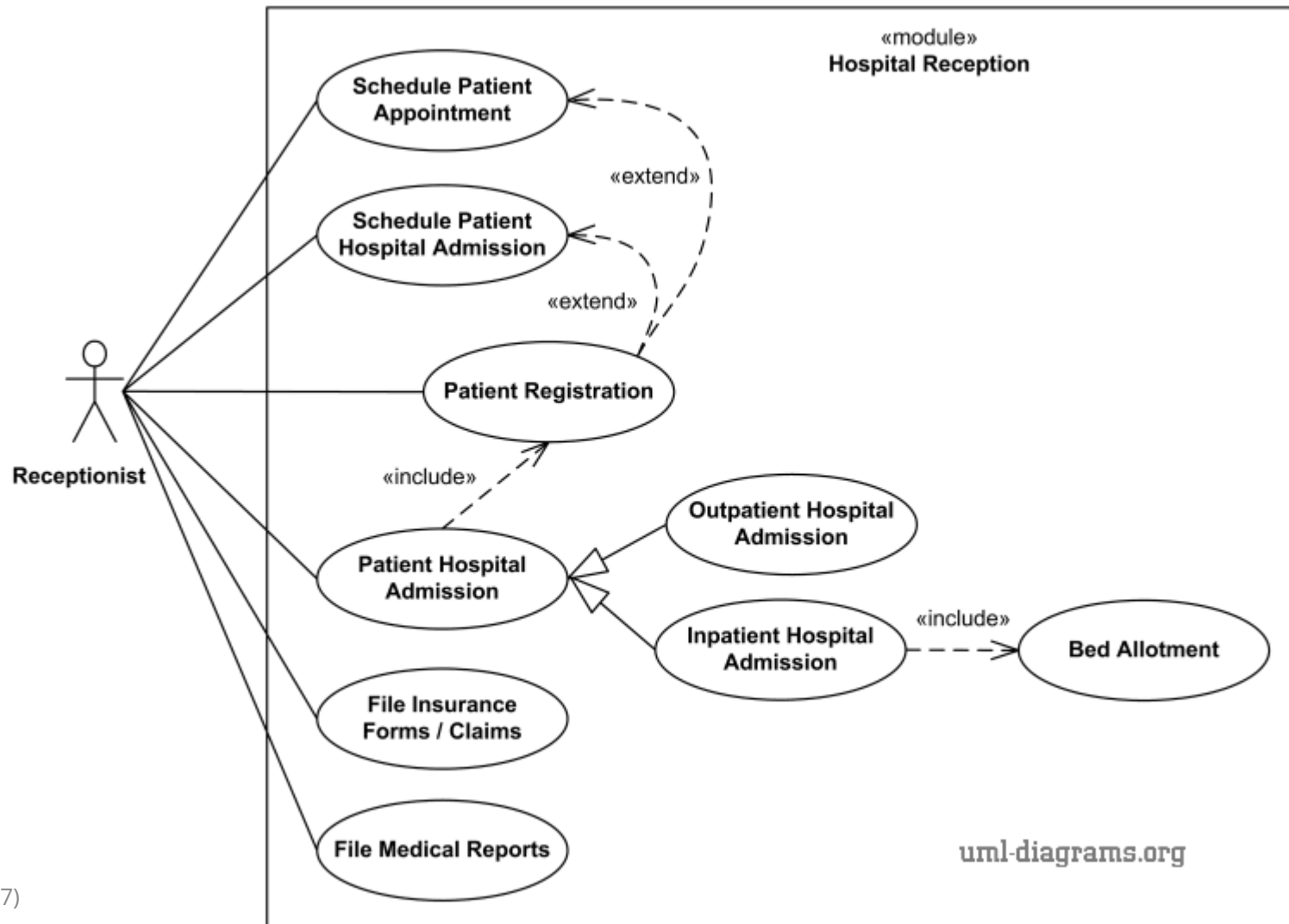


MODELO 4+1 (PHILIPPE KRUCHTEN)

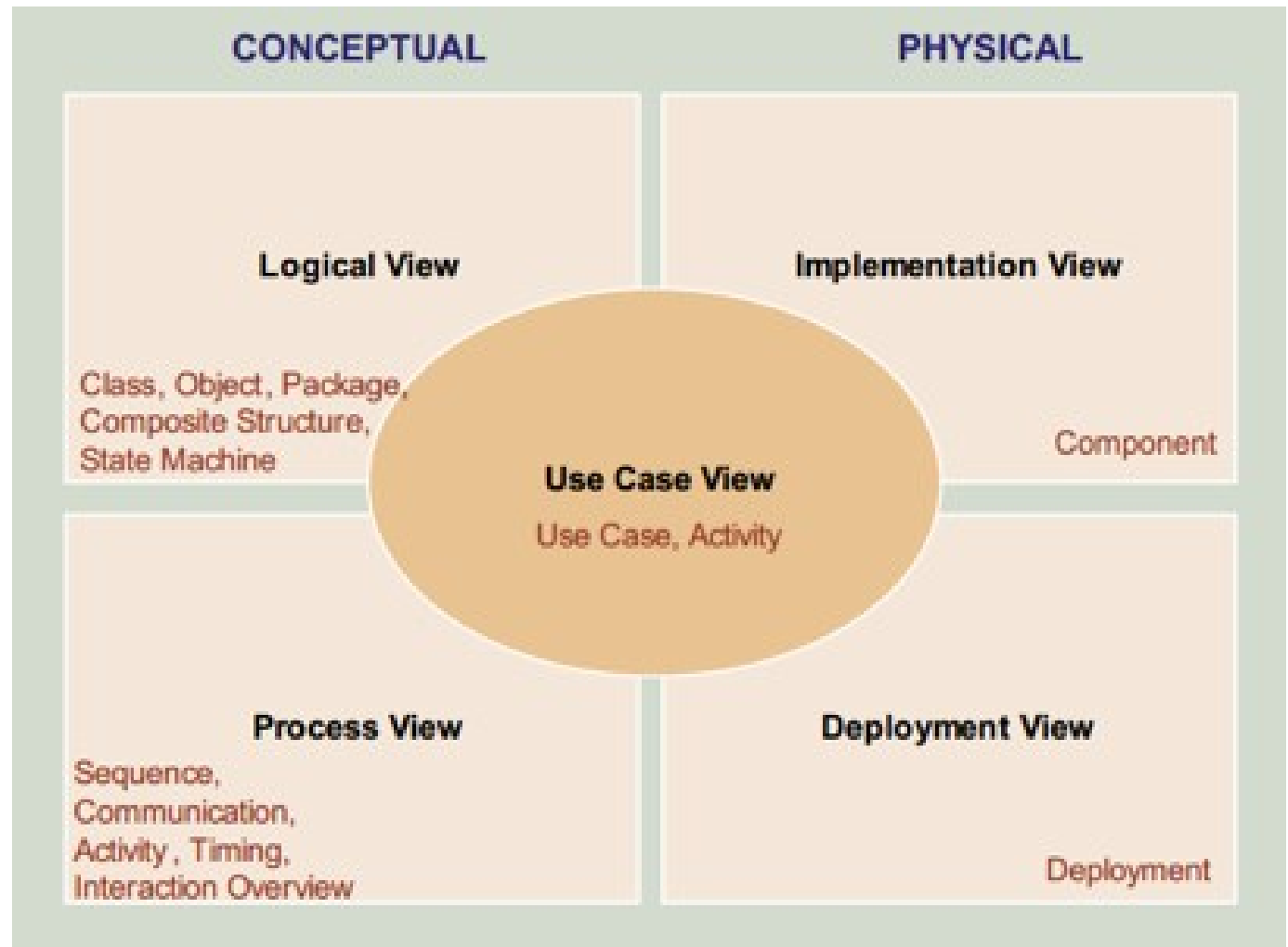
Diversos diagramas para abranger diferentes perspectivas de análise - Modelo 4+1 (Philippe Kruchten)



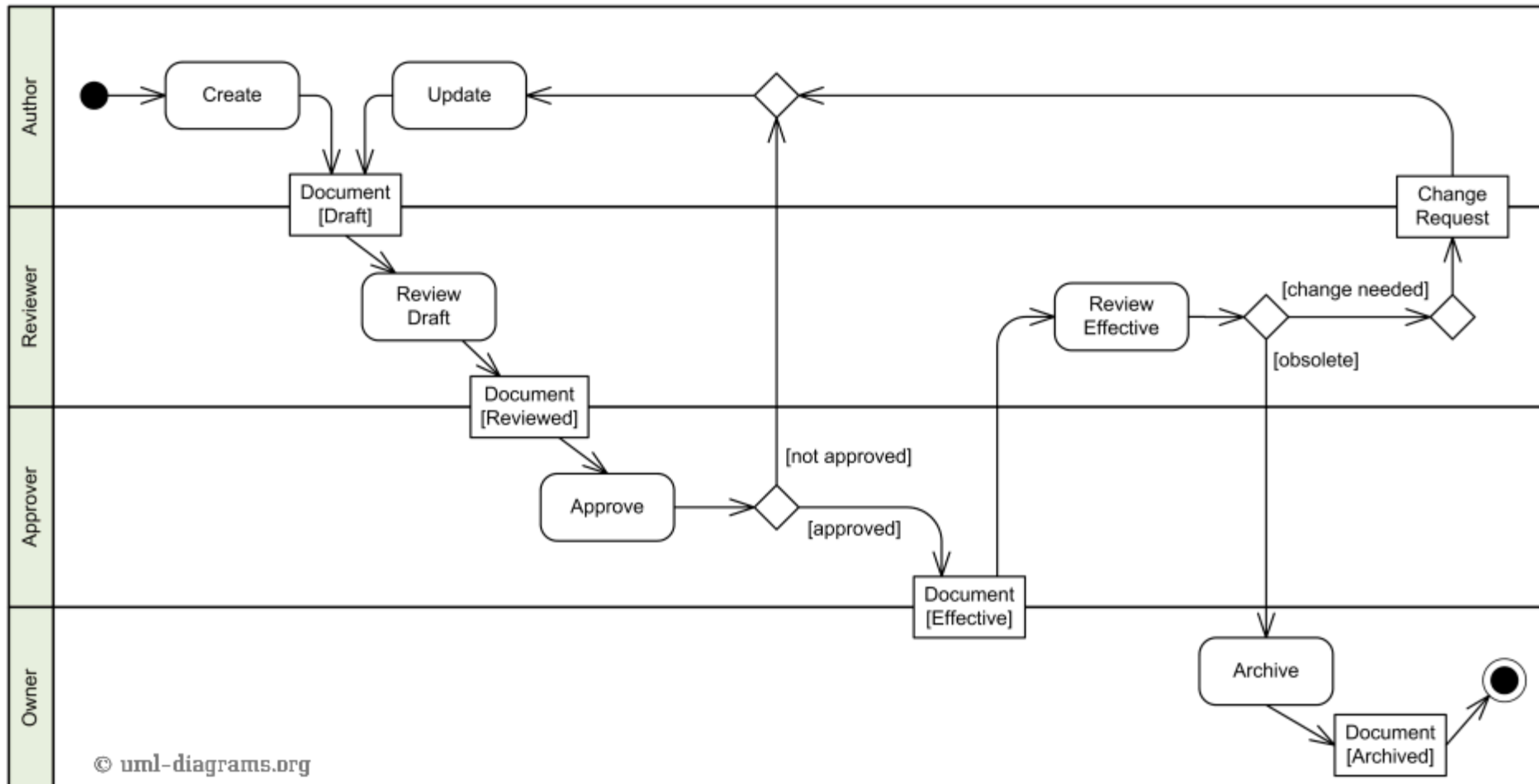
CaU do Sistema: organizar a funcionalidade do sistema em episódios de utilização



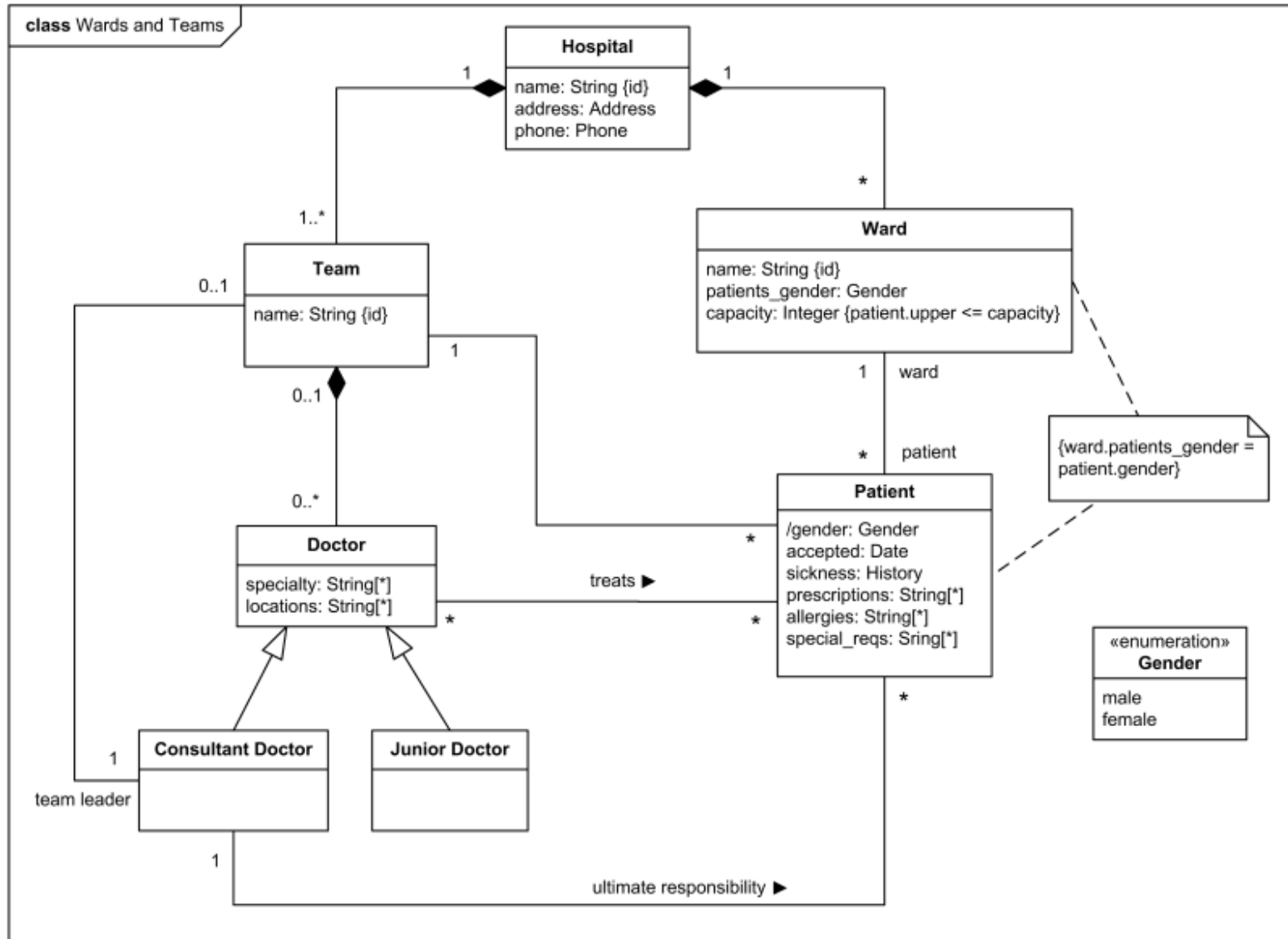
Diversos diagramas para abranger diferentes perspectivas de análise



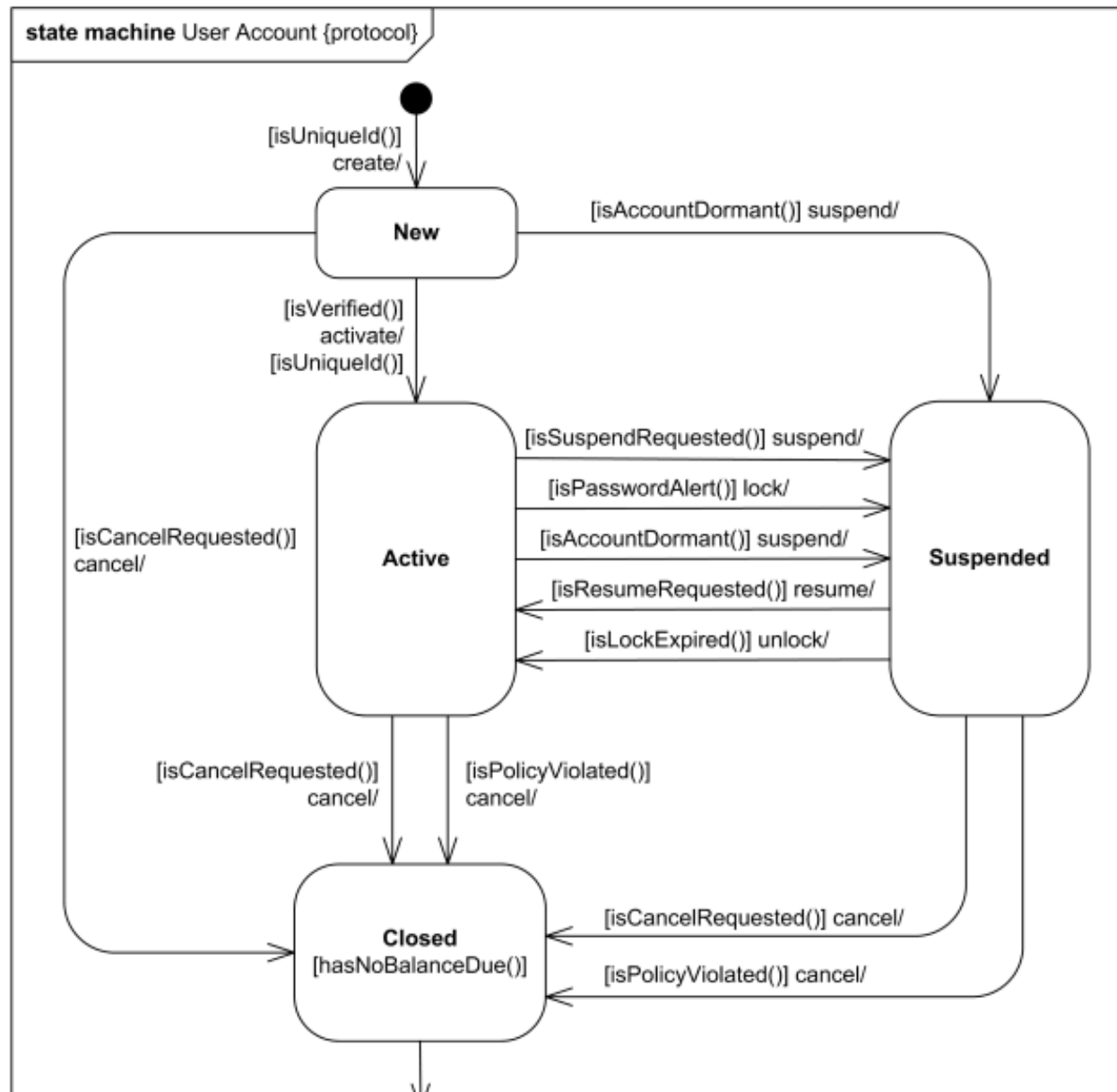
Diagramas de atividades para explicar procedimentos do domínio



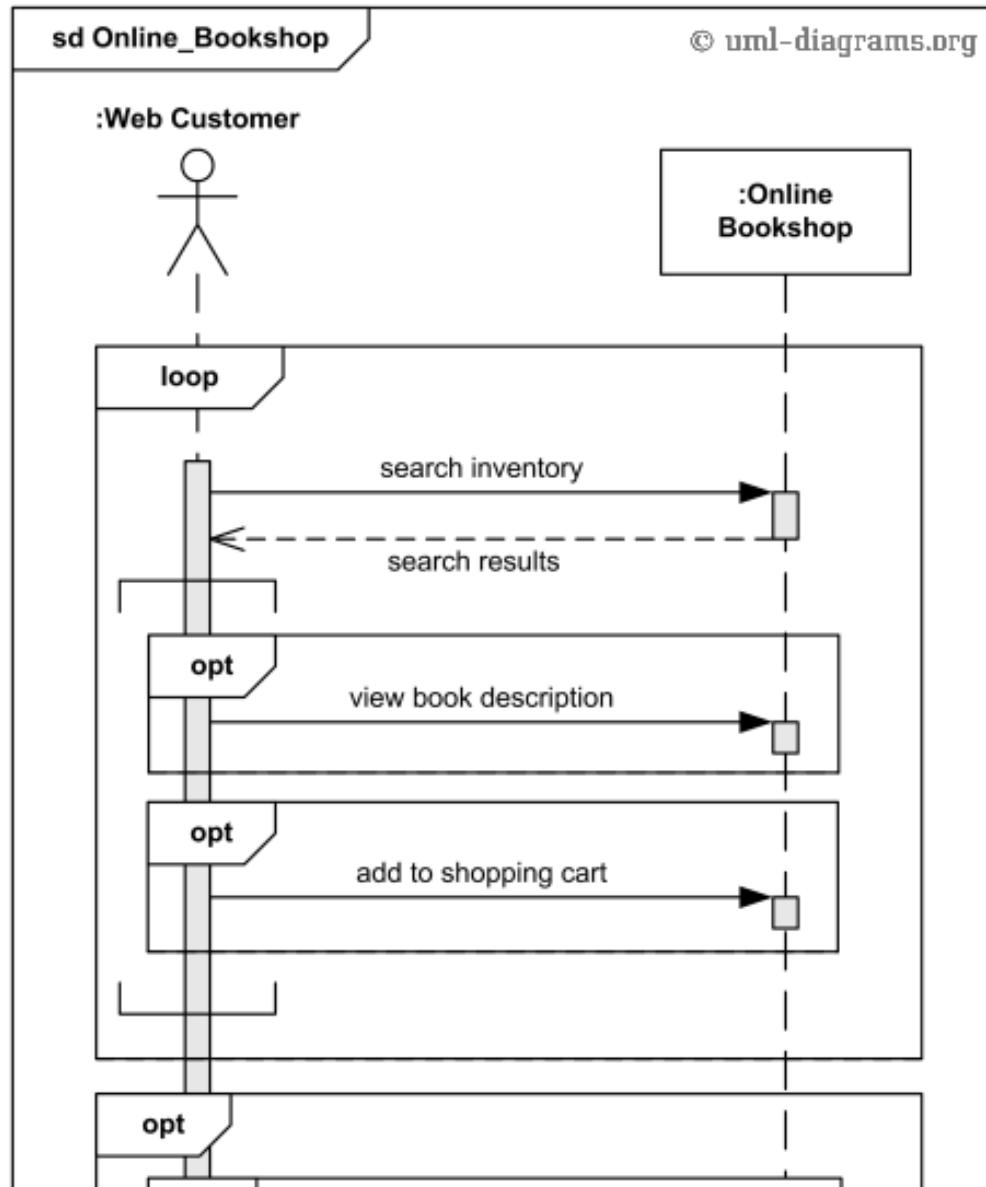
Classes para representar os conceitos da área do problema (modelo do domínio)



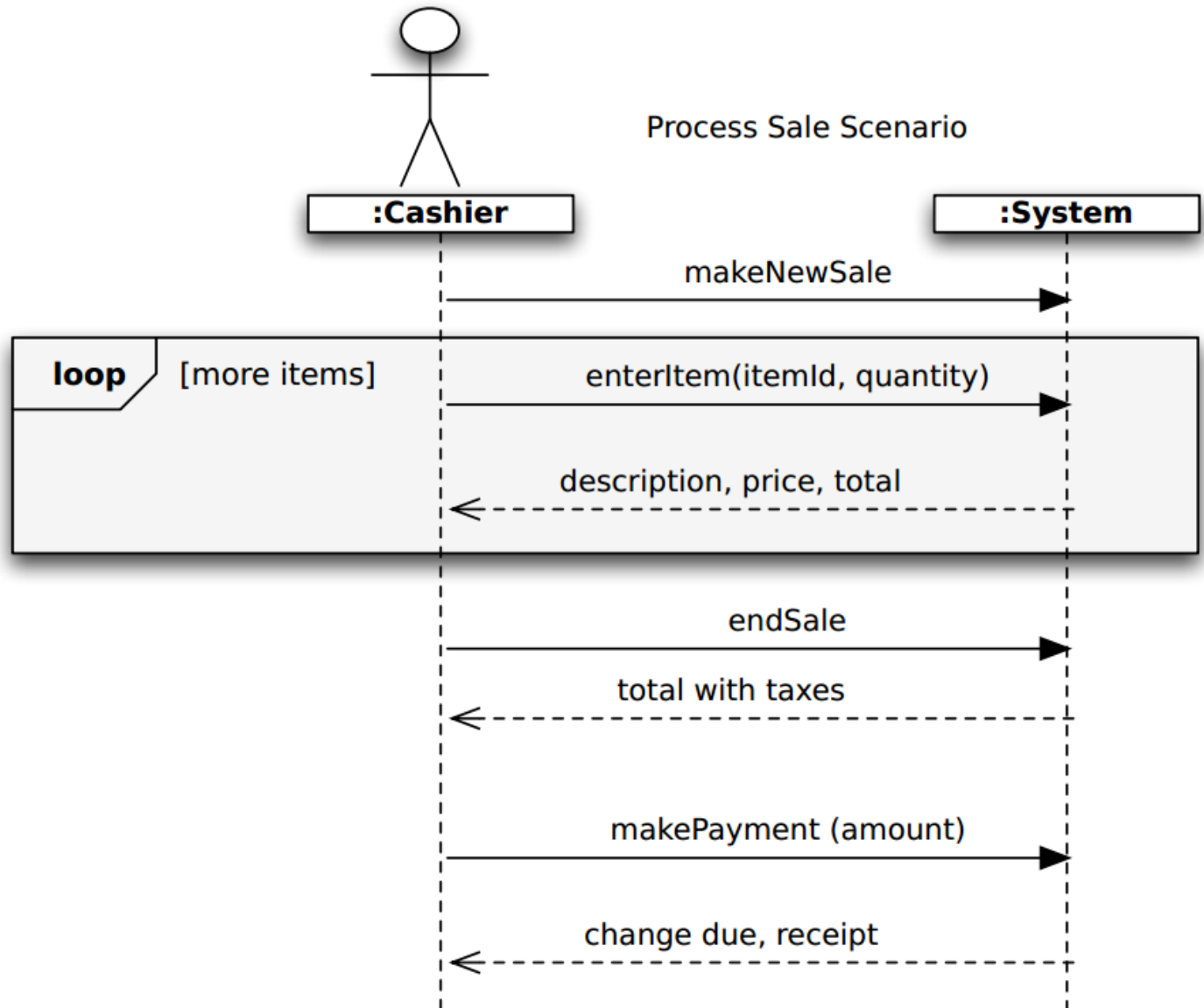
Máquina de estados de entidades/objetos



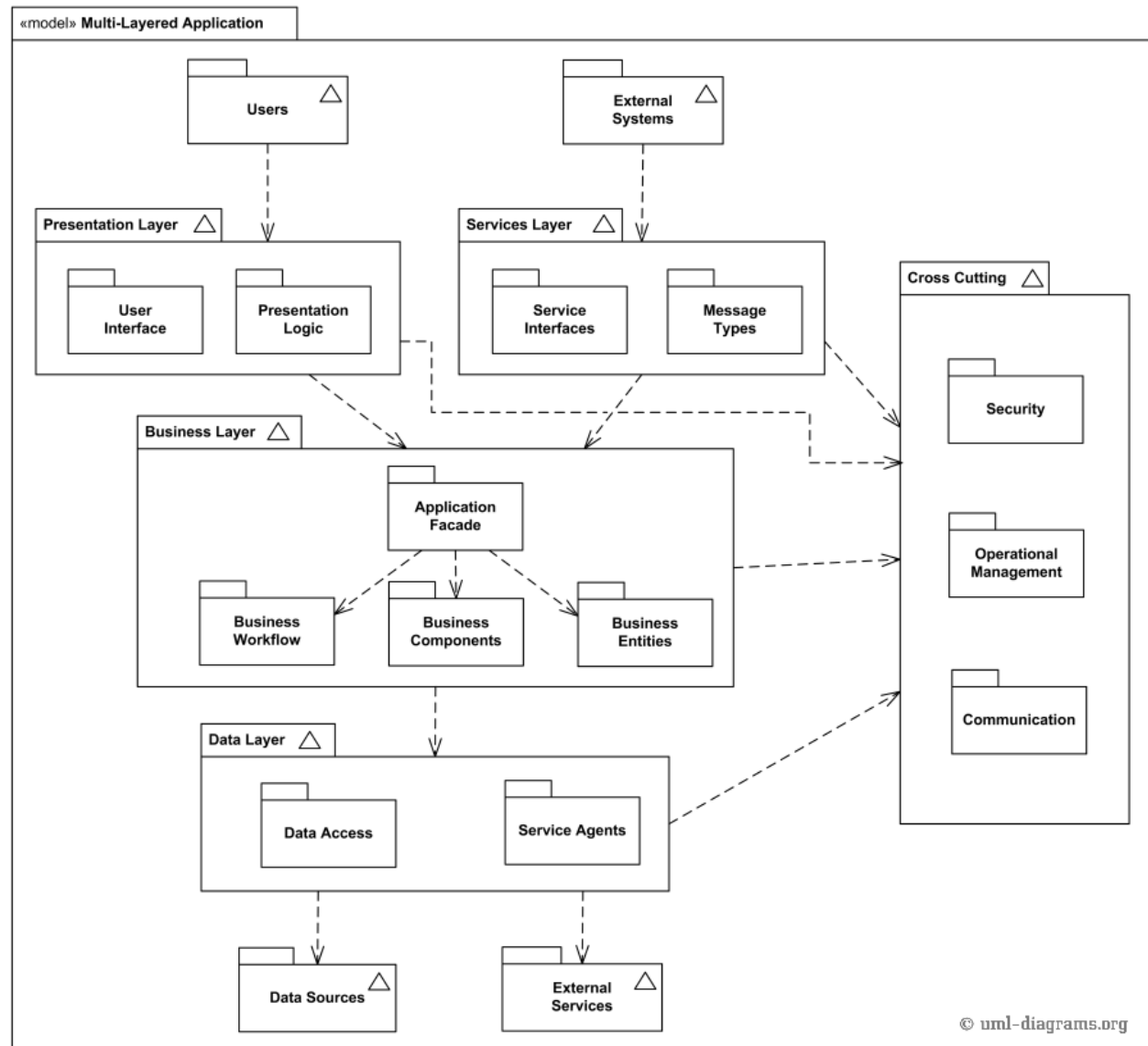
Interação entre atores e cenários dos CaU



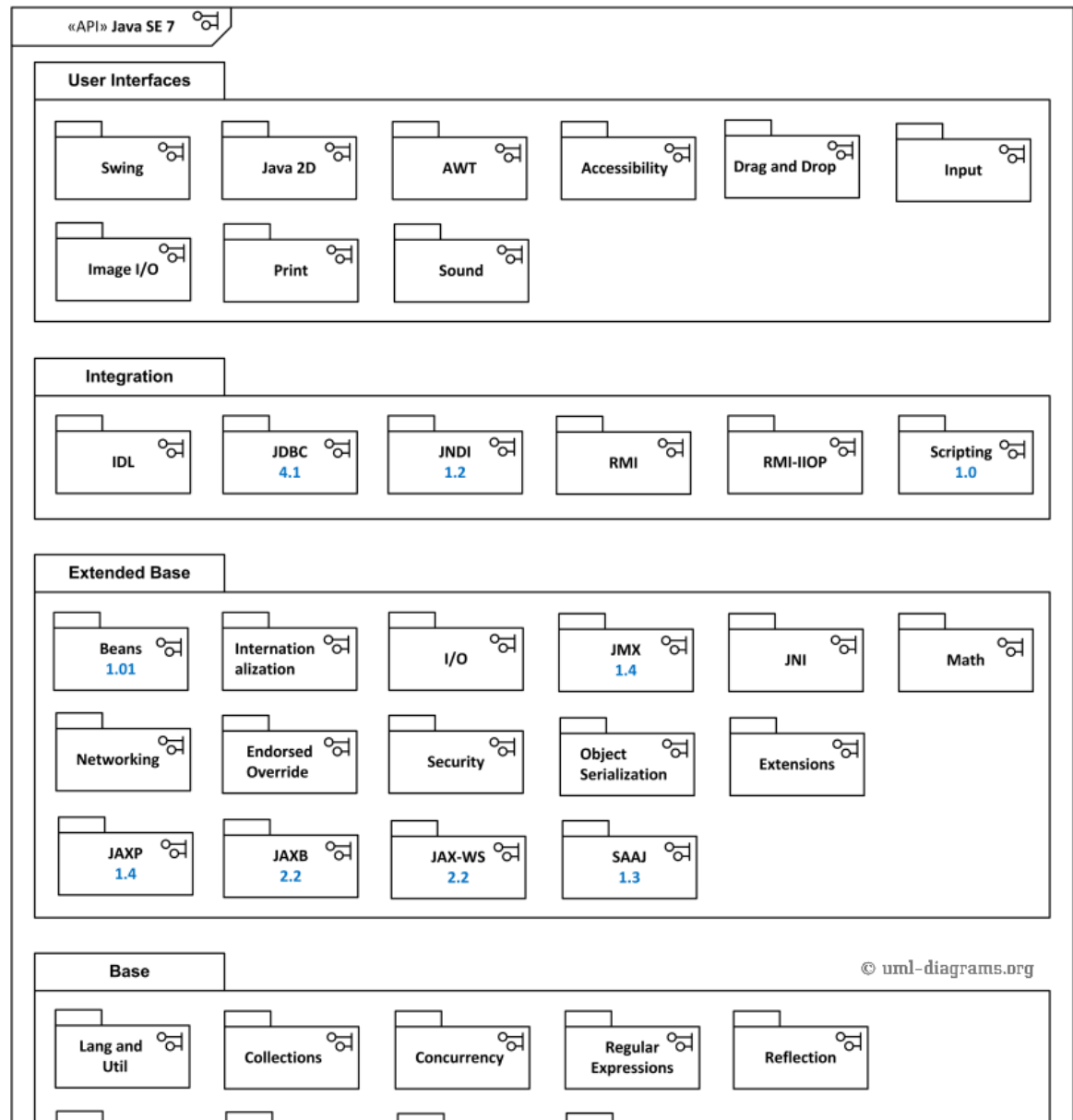
Caso particular: DS de sistema



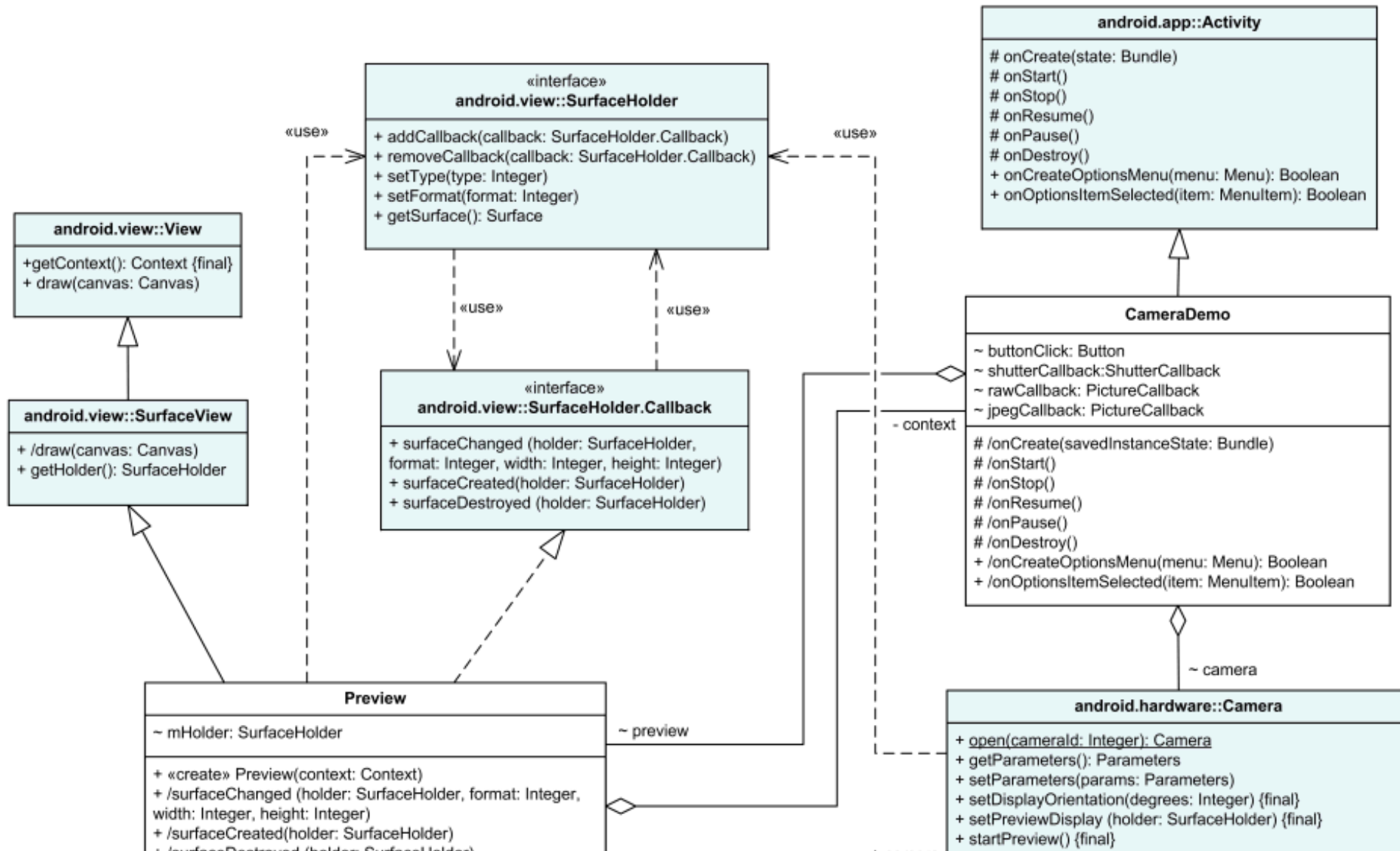
Pensar a arquitetura lógica com D. Pacotes



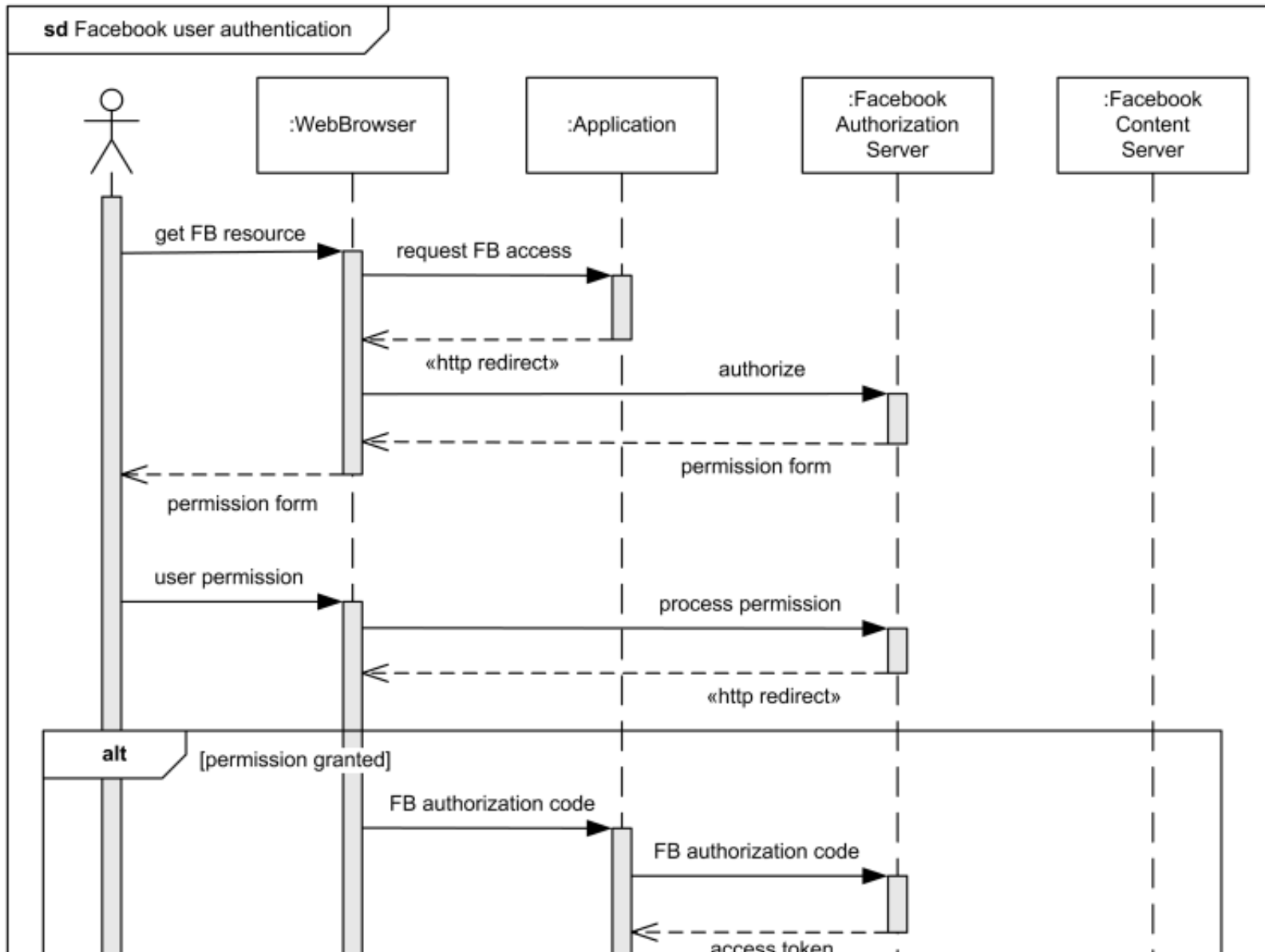
Pacotes do SDK do Java 1.7



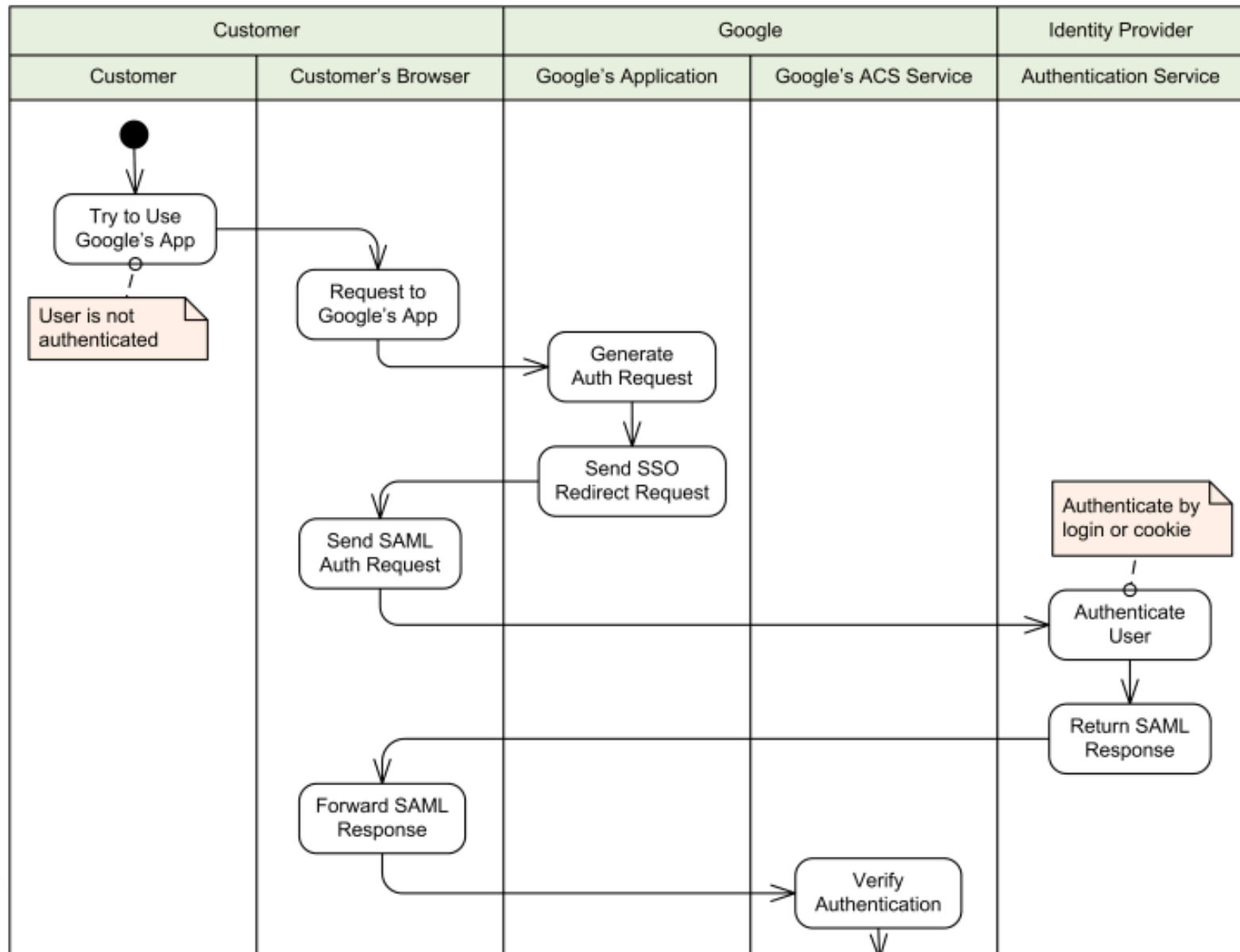
Classes para visualizar objetos de um linguagem de programação



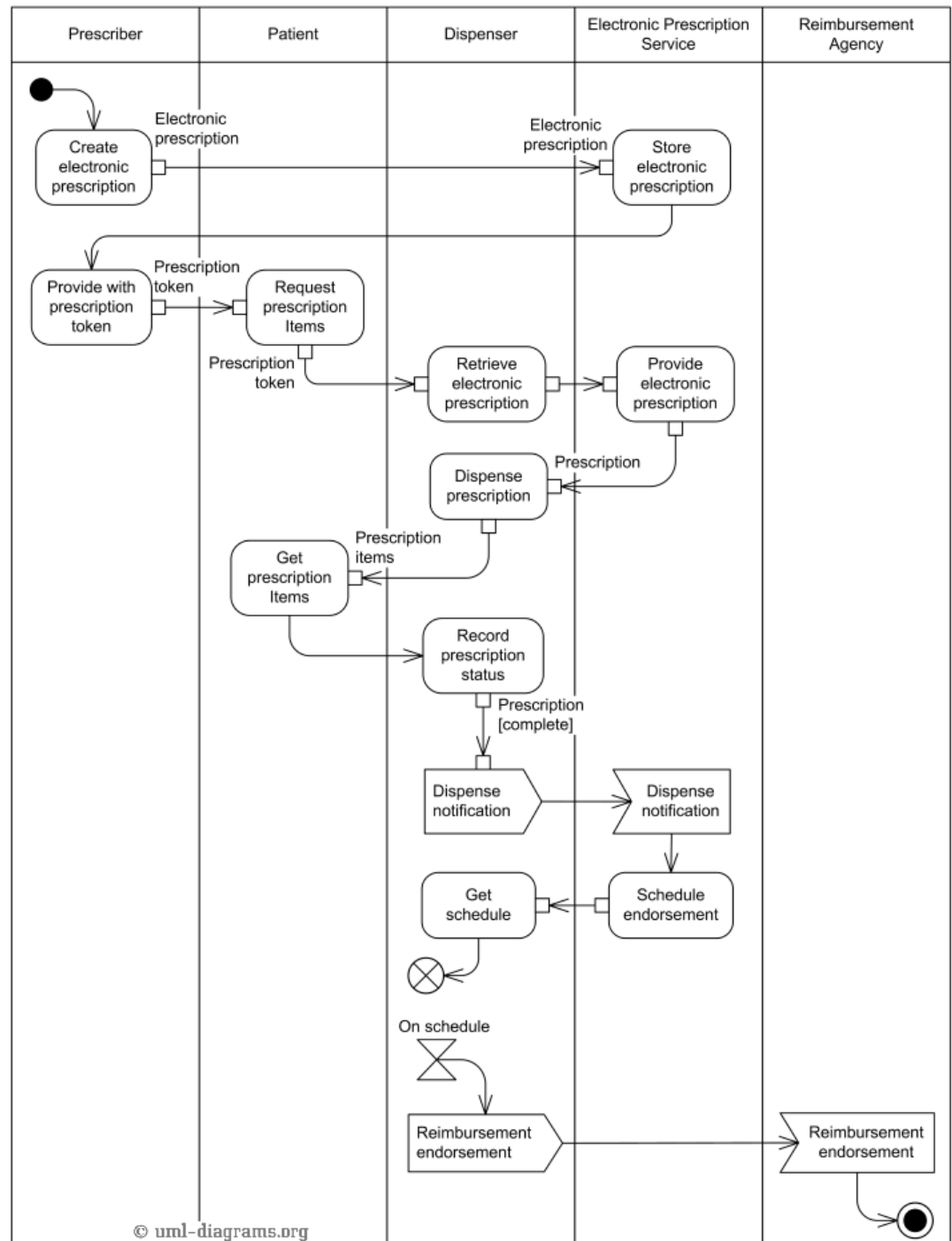
Interações entre componentes do software



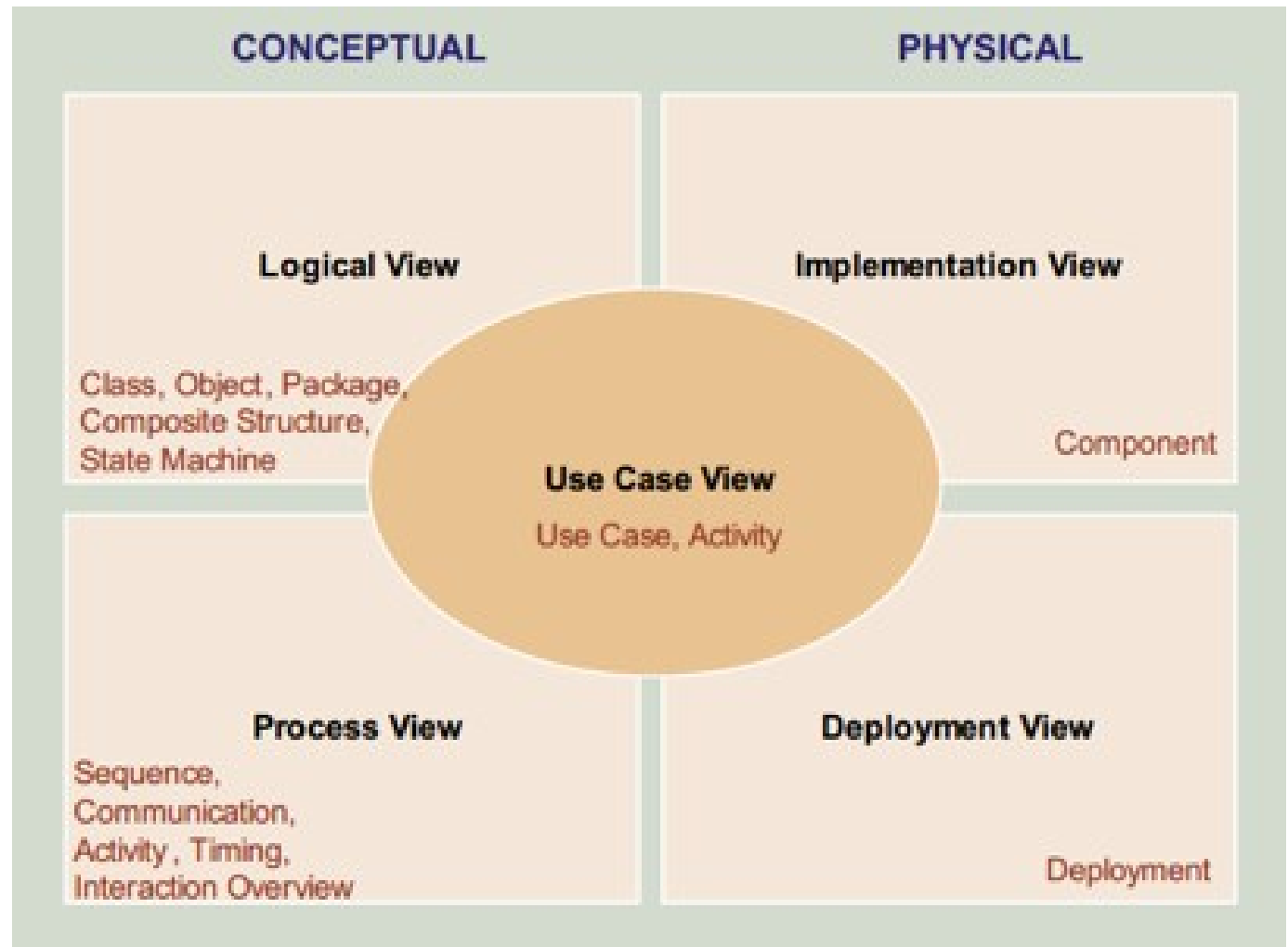
DA para explicar algoritmos



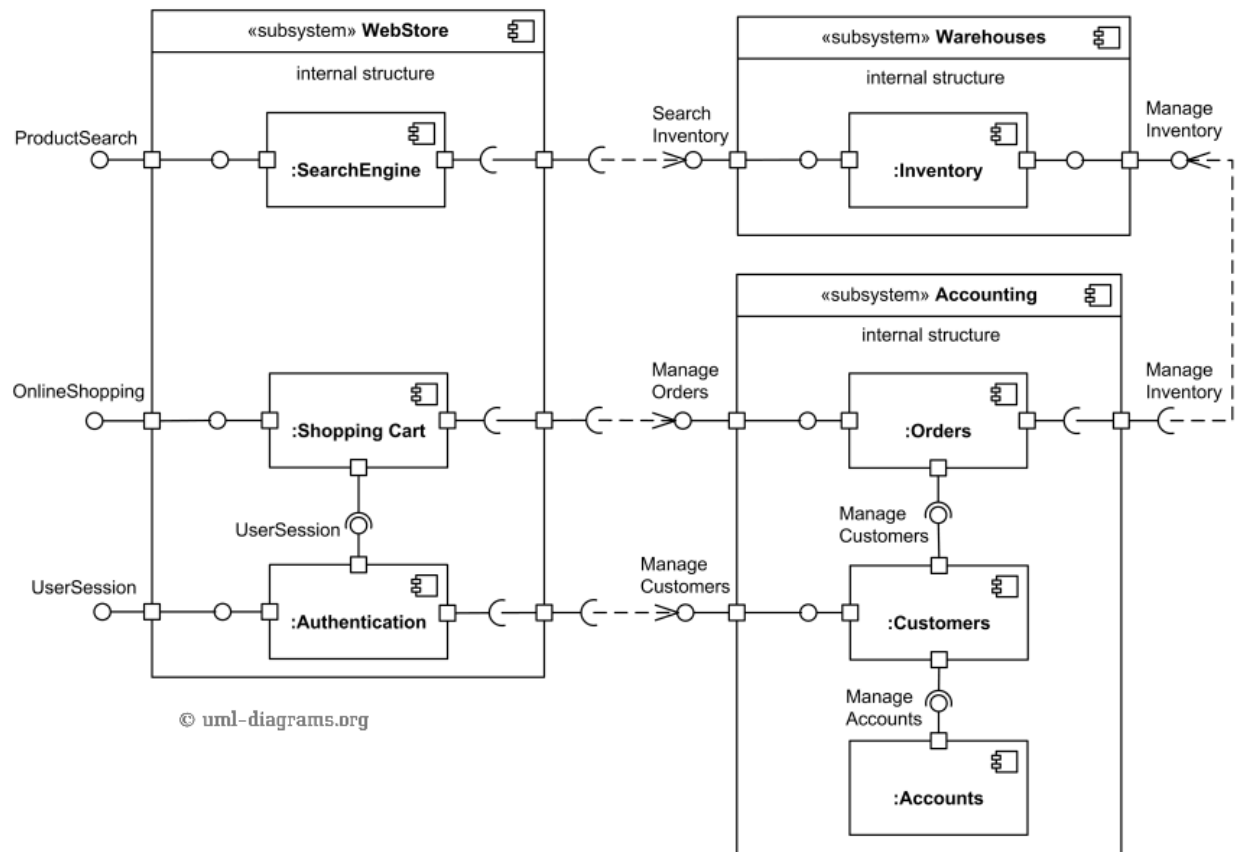
Os DA incluem semântica para mostrar eventos e passagem de informação



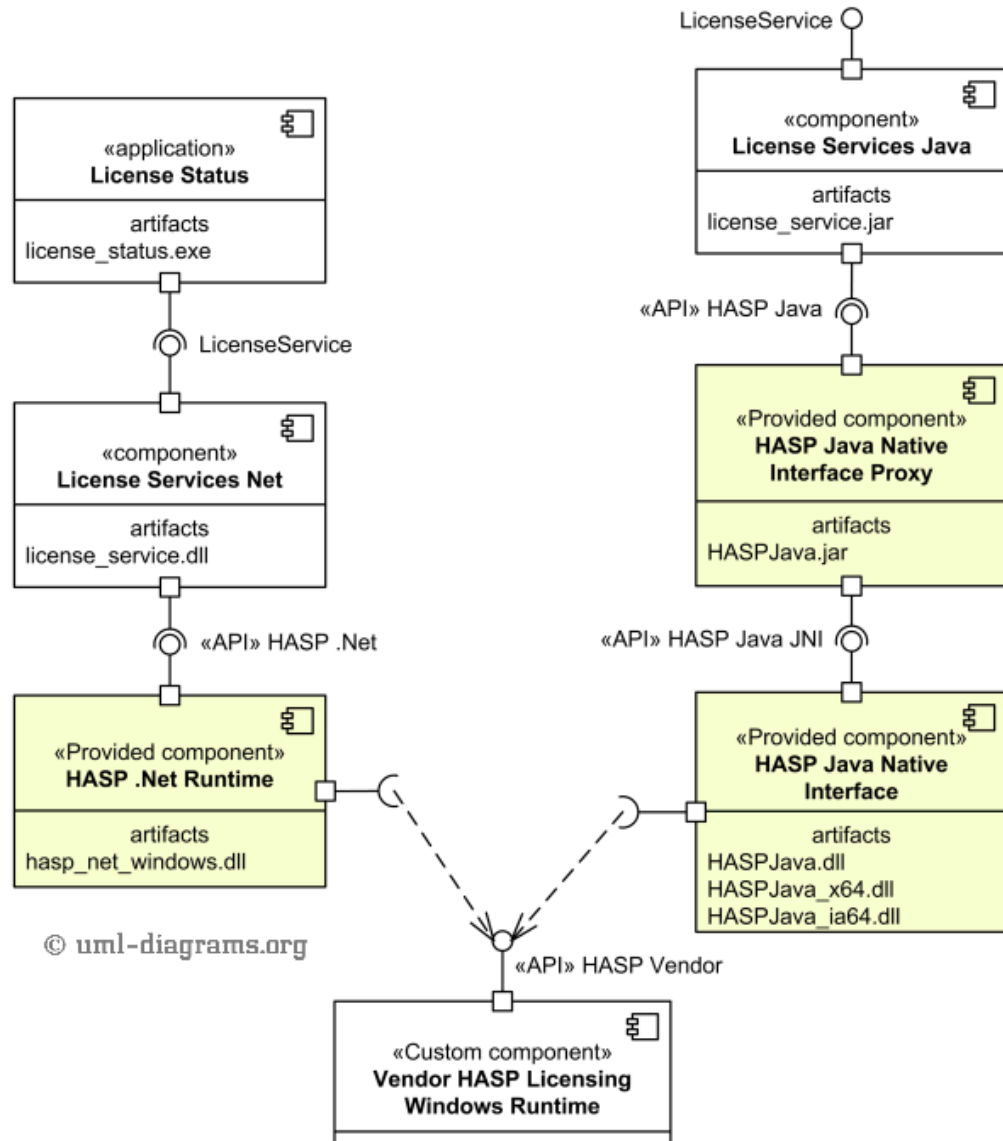
Diversos diagramas para abranger diferentes perspectivas de análise



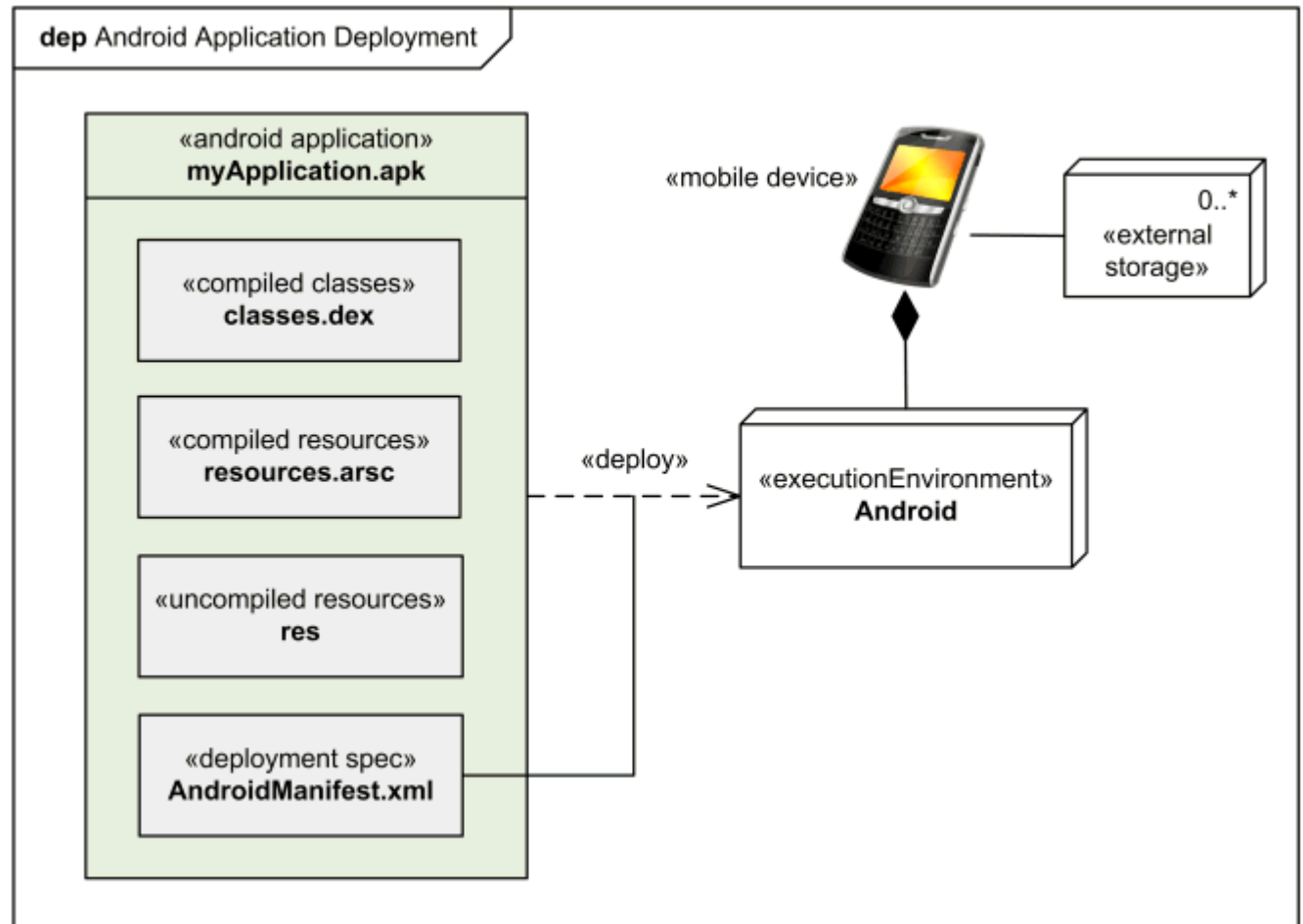
Módulos (executáveis) da solução captados em componentes



Os componentes têm correspondência em artefactos concretos



D. Instalação: mostrar o *setup* para produção



O nível de detalhe é variável, conforme aquilo que se quer comunicar

