Requirements Engineering: Domain Models

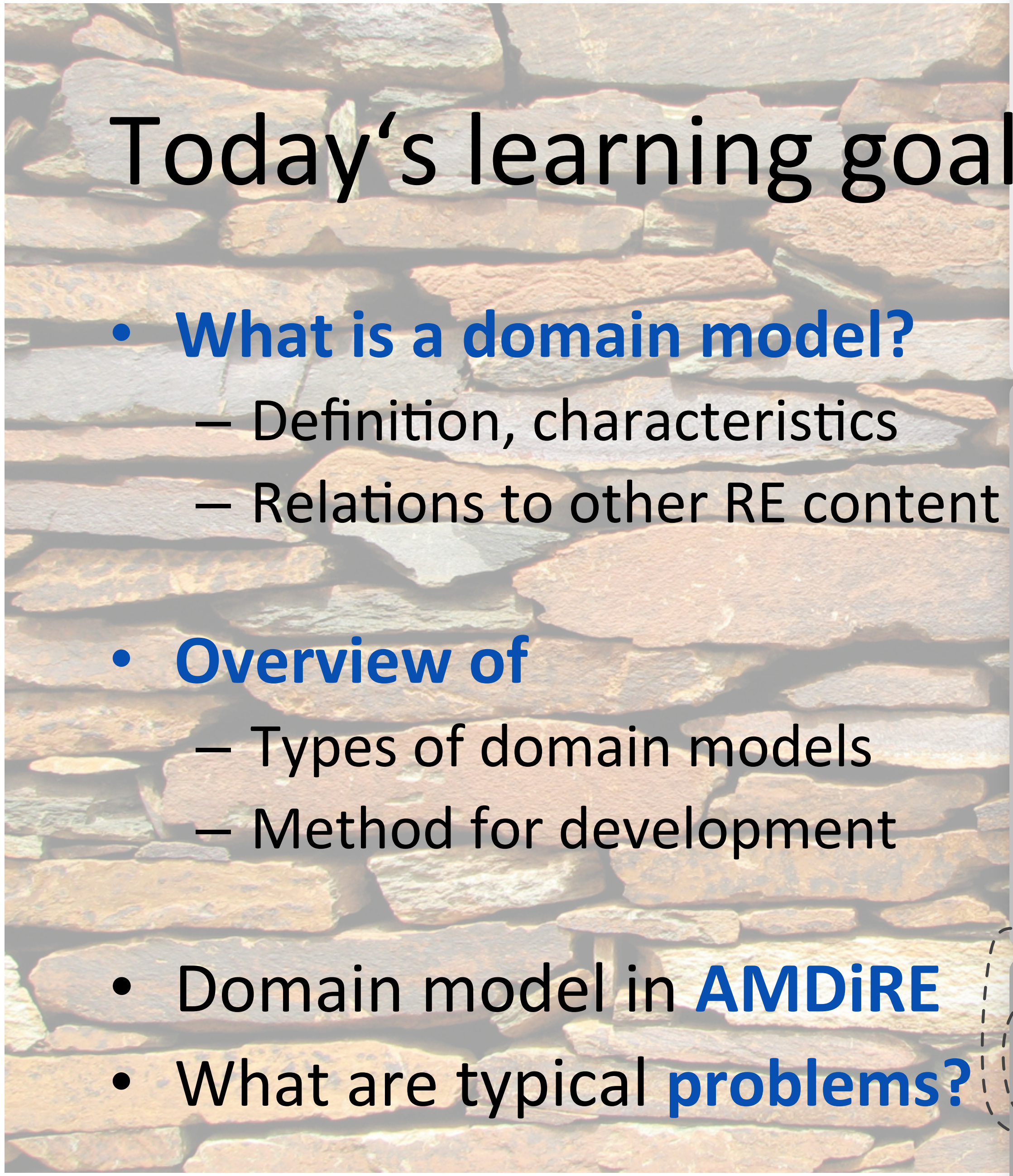
CECS 590

Recap time!

- System Vision
  - What is it?
  - Why do we need one?
  - Which elements should it contain?
  - Which methods do we use?

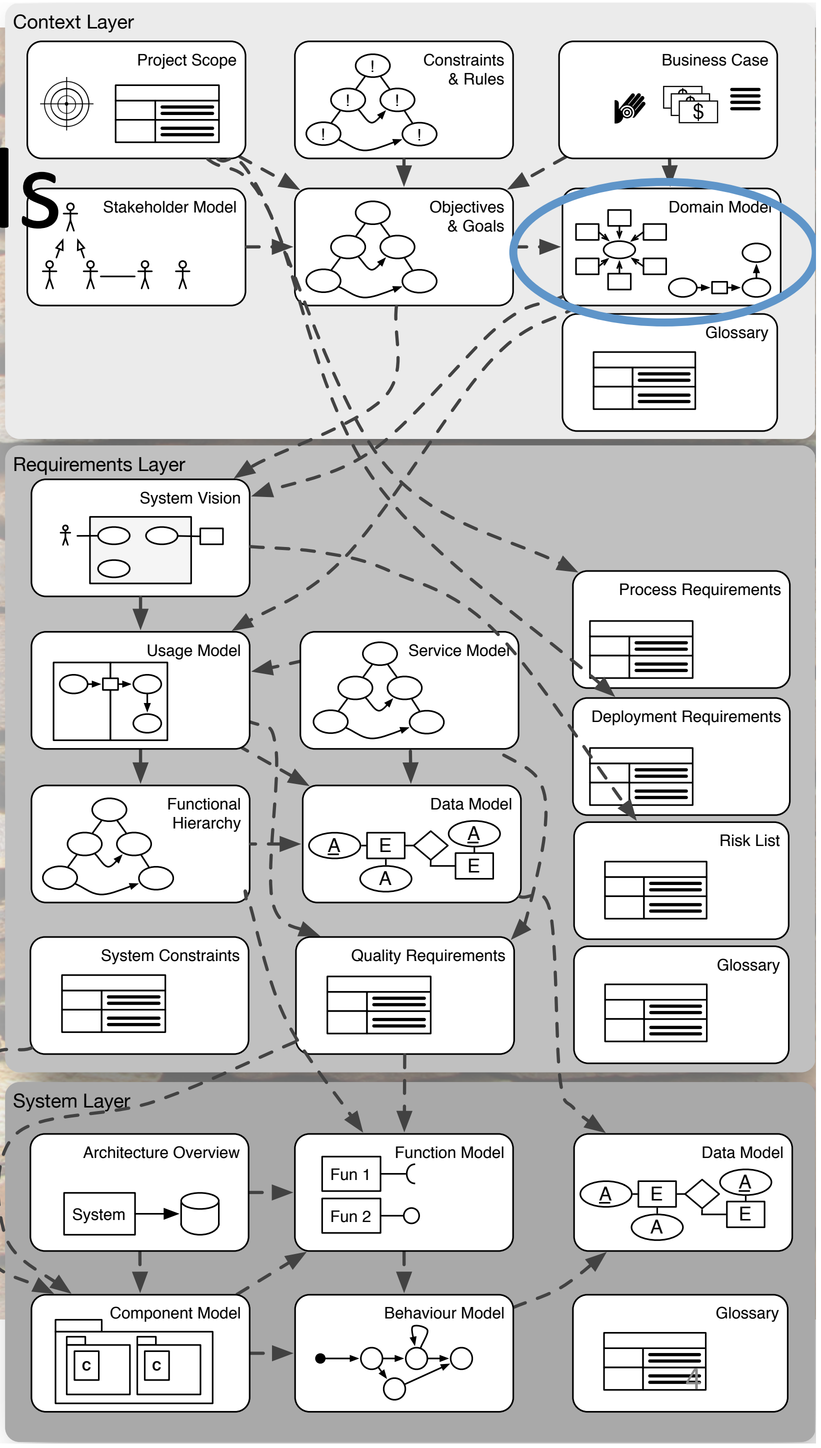
Requirements Engineering – Outline

- WHY do we need Requirements Engineering and what is it?
- Principles: Definitions, process, roles, problem/solution view, artifact orientation
- System Models: Decomposition and abstraction, system views
- Frameworks: What reference structures can I use for requirements?
- Business Case Analysis: Why are we building this system?
- Stakeholders: Who are the people to talk to about requirements?
- Goals and Constraints: What are the major objectives for the system?
- System Vision: What exactly do we want to achieve?
- **Domain Models: What are the surrounding systems ours interacts with?**
- Usage Models: How will the system interact with the user?
- Software quality models: How to determine the quality characteristics?
- Quality requirements: How to specify which qualities need to be met?
- Process requirements: How to specify constraints for development?
- Towards a system specification: How to hand over to design?
- Quality assurance: How to ensure that RE is done in a good way?
- Change management: How to evolve requirements?



Today‘s learning goals

- **What is a domain model?**
  - Definition, characteristics
  - Relations to other RE content
- **Overview of**
  - Types of domain models
  - Method for development
- Domain model in **AMDiRE**
- What are typical **problems?**



Definition

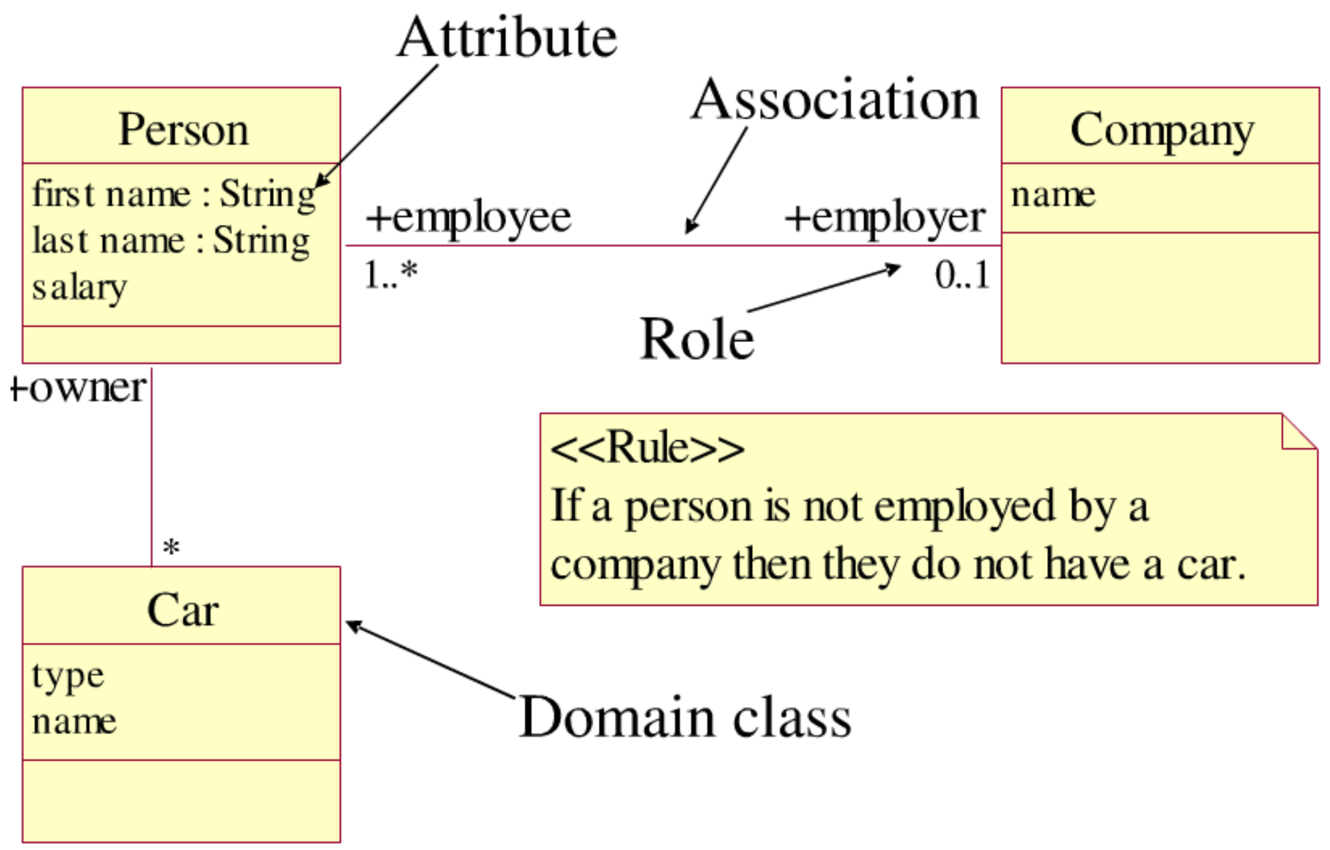
“A domain model captures the most important types of objects in the context of the business. The domain model represents the ‘things’ that exist or events that transpire in the business environment.” – I. Jacobsen

A **domain model** in problem solving and software engineering is a conceptual model of all the topics related to a specific problem.
It describes the various entities, their attributes, roles, and relationships, plus the constraints that govern the problem domain.

Rationale: Why a domain model?

- Conceptual framework of things in problem space
- Helps you think – and focus on semantics
- Provides a glossary of terms
- Static view – structure of time-invariant parts
- Based on that structure, we can describe the state of the problem domain at any time (in behavioral, dynamic views)

Example: Simple domain model



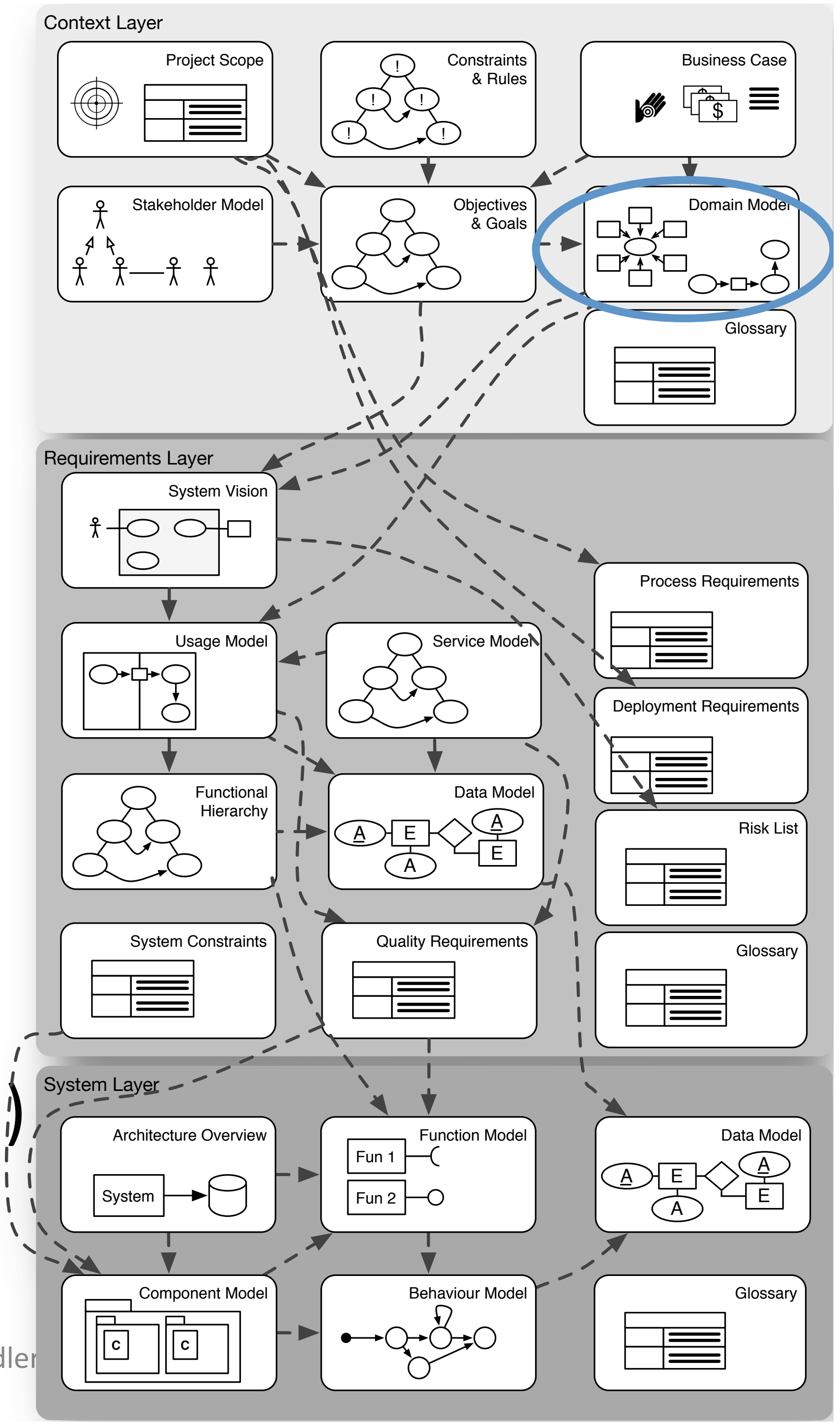
Domain objects are *not* software objects, but a representation of real-life conceptual objects.

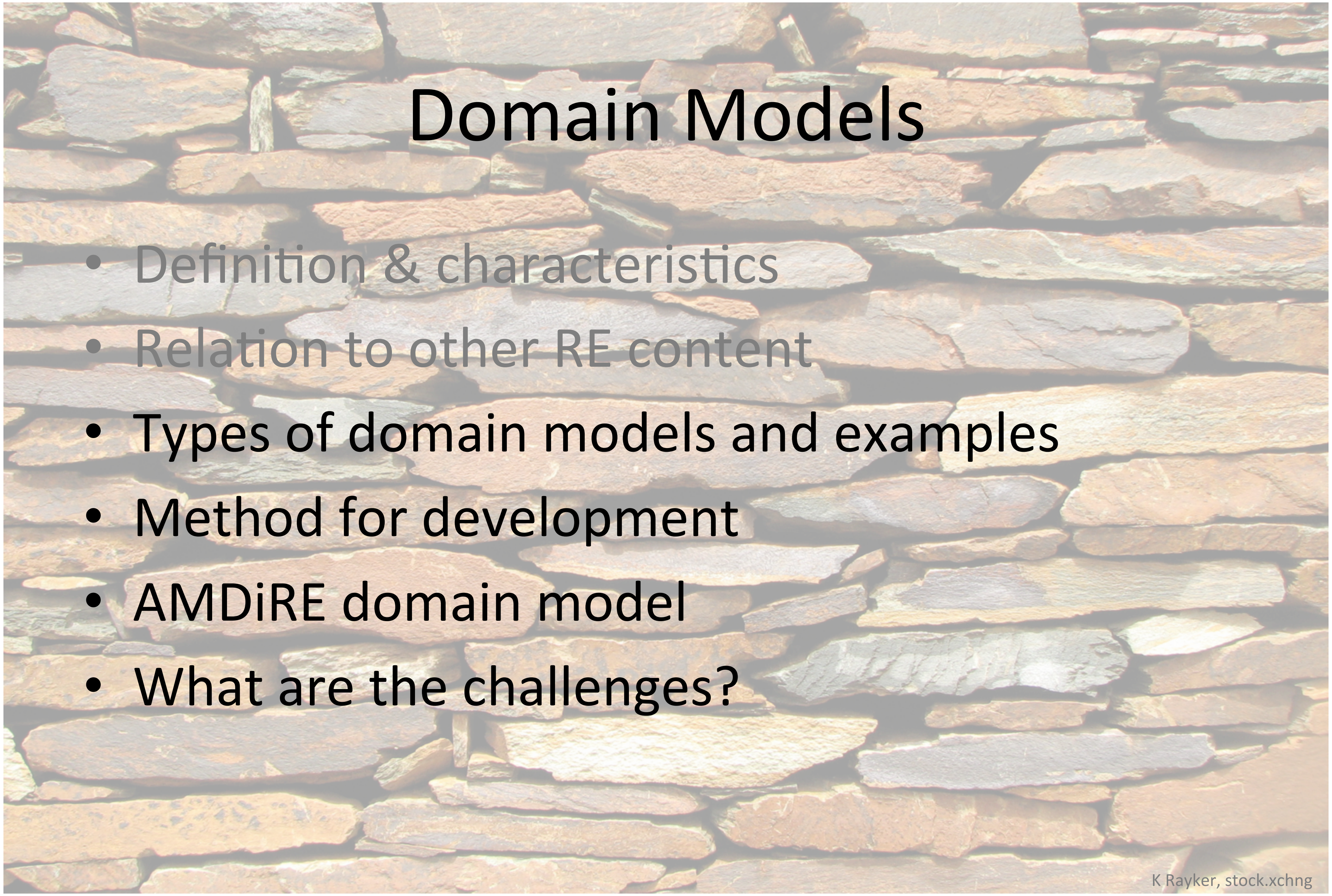
Features

- **Domain classes** – each domain class denotes a type of object
  - Business objects: things manipulated in the business, e.g., *Order*
  - Real world objects: things the business keeps track of, e.g., *Contact*, *Site*
  - Events that transpire, e.g. *Sale* and *Payment*
- **Attributes** – an attribute is the description of a named slot of a specified type in a domain class; each instance of the class separately holds a value, e.g. *name* or *ID*
- **Associations** – an association is a relationship between two (or more) domain classes that describes links between their object instances, e.g., *owns*. Associations can have roles, describing the multiplicity and participation of a class in the relationship, e.g. *employee*.
- **Additional rules** – complex rules that cannot be shown with symbology can be shown with attached notes.

Relation to other RE content

- Input
  - Business Case
  - Goal Model
- Output
  - Glossary
  - System Vision (usually developed in iteration with Dom. Model)
  - Usage Model



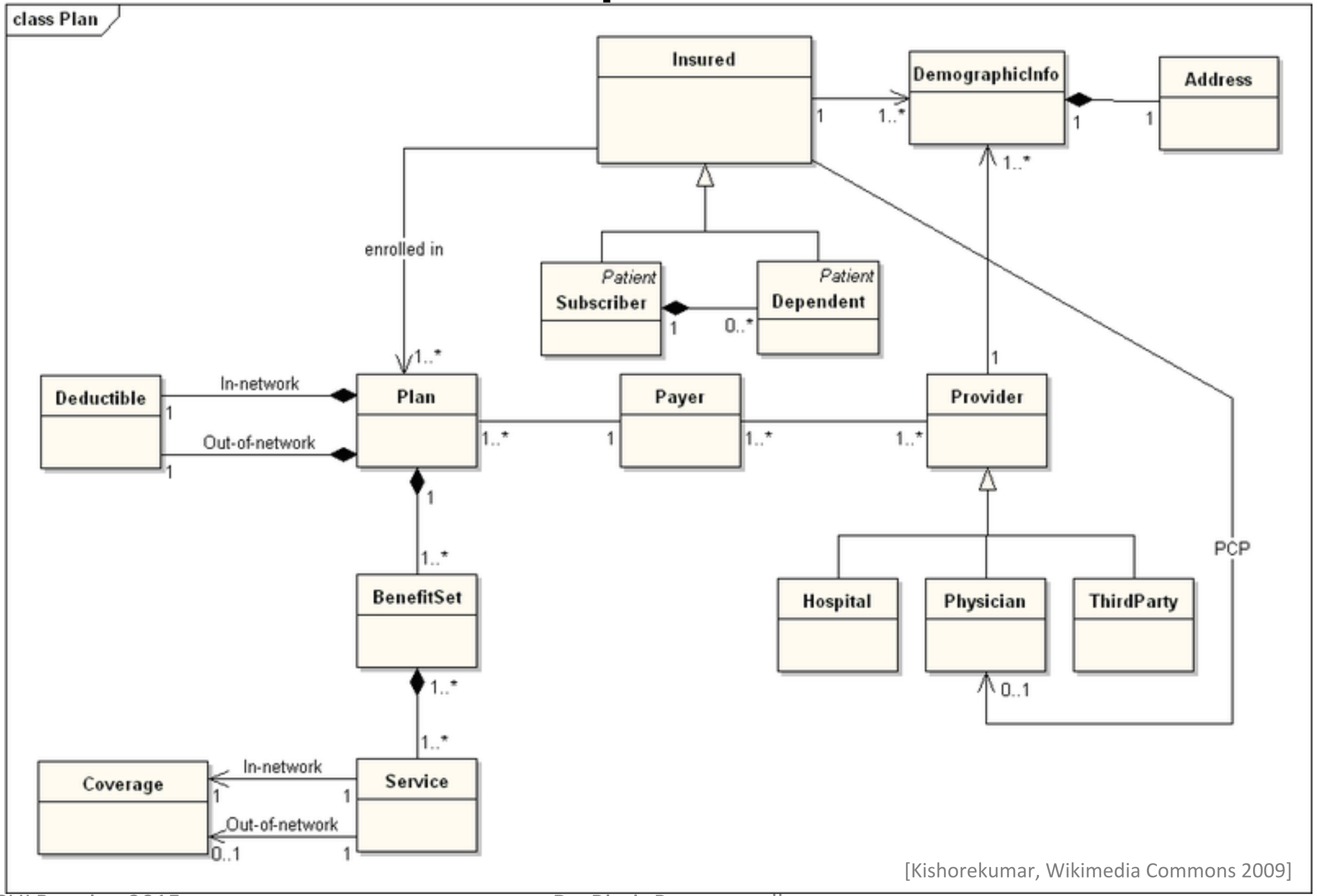


Domain Models

- Definition & characteristics
- Relation to other RE content
- Types of domain models and examples
- Method for development
- AMDiRE domain model
- What are the challenges?

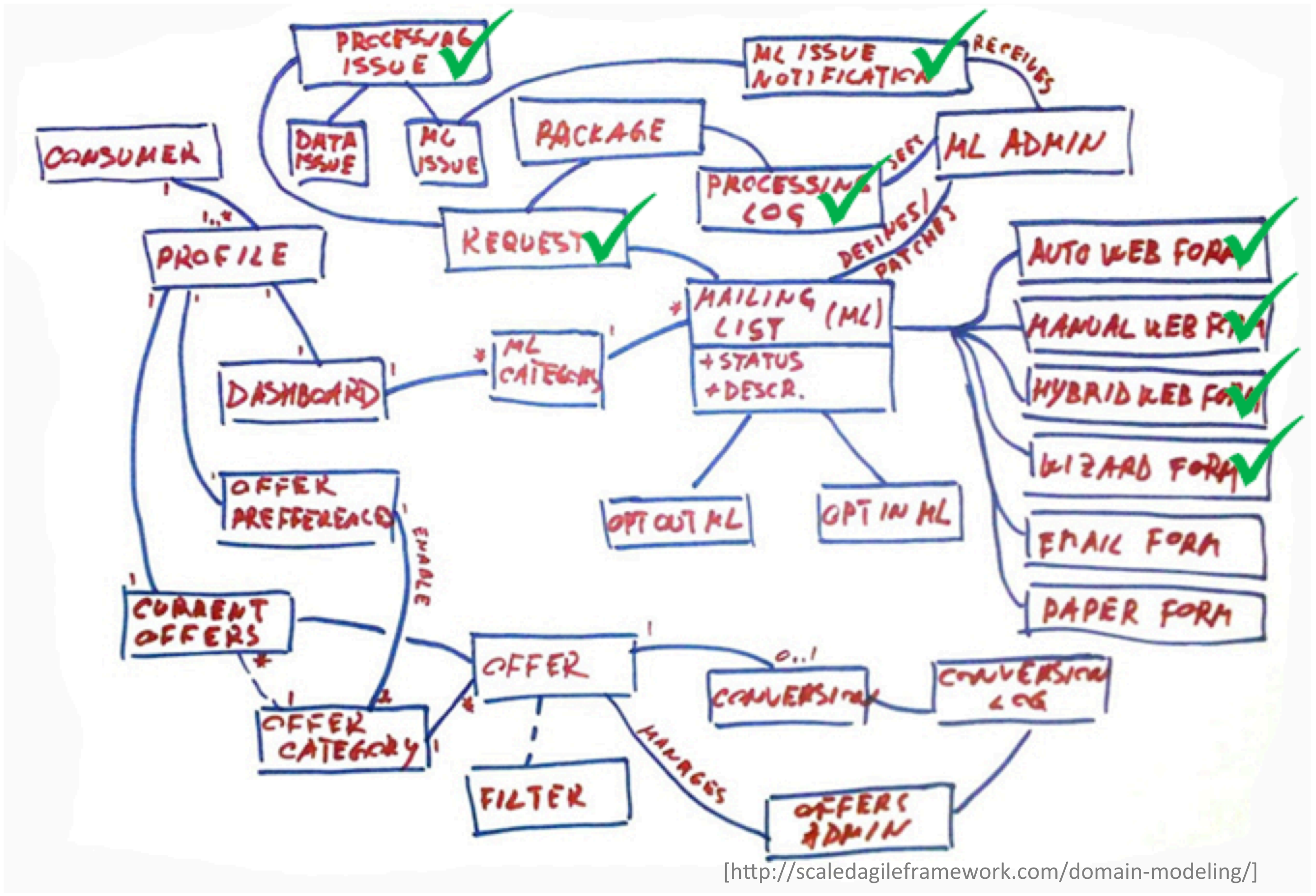
K Rayker, stock.xchng

Example: Sample domain model for a health insurance plan



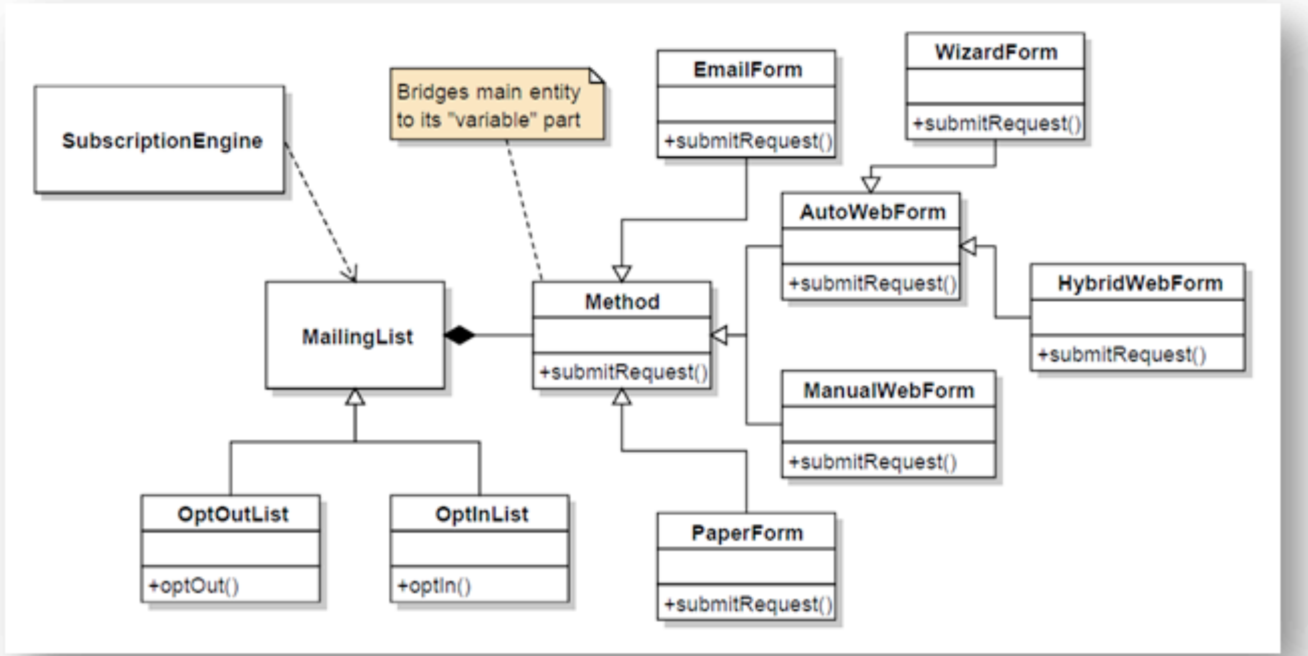
[Kishorekumar, Wikimedia Commons 2009]

Example: Sample domain model for a health insurance plan



[http://scaledagileframework.com/domain-modeling/]

Example: Sample domain model for a health insurance plan

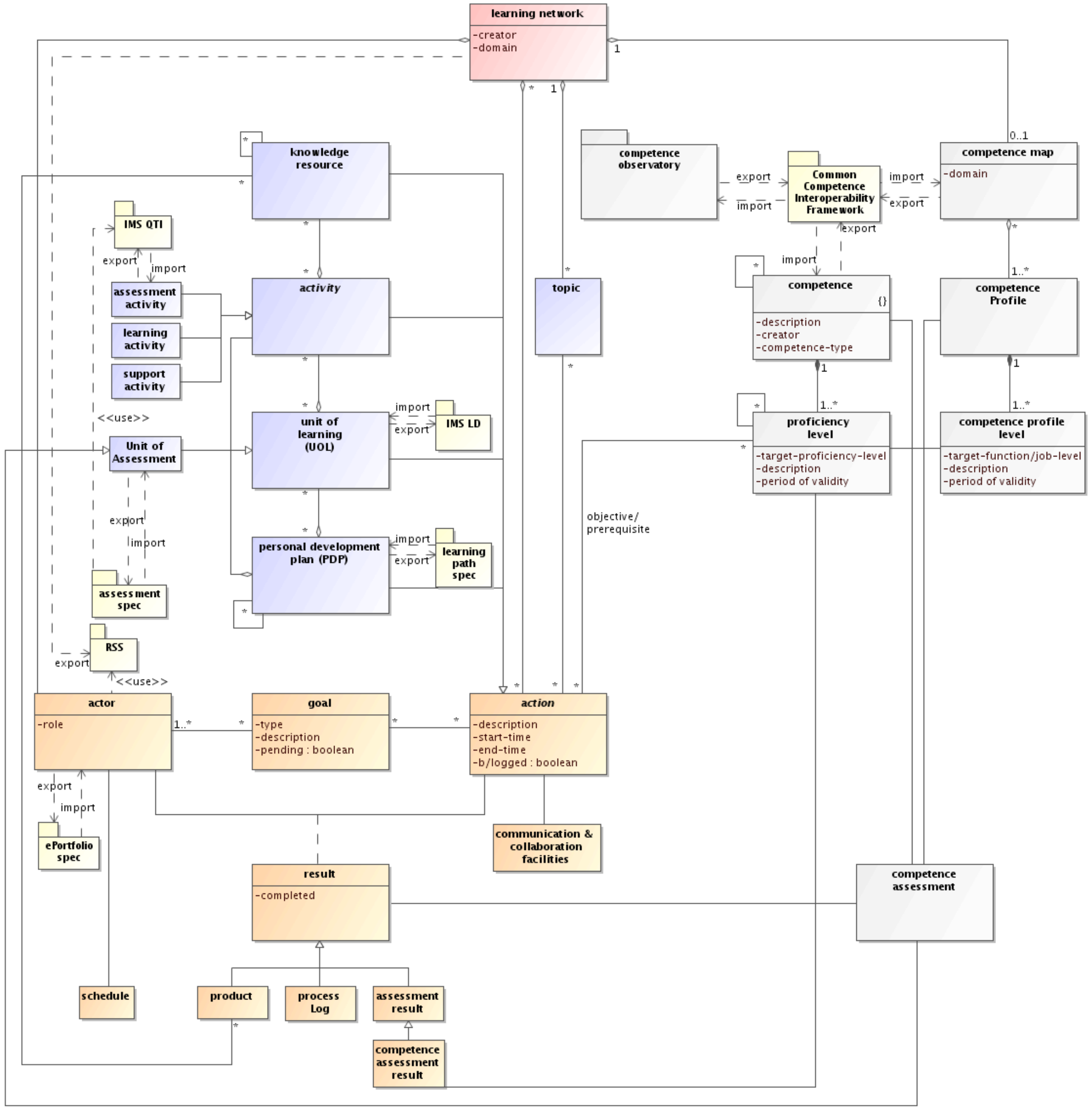


[http://scaledagileframework.com/domain-modeling/]

How to develop a DM

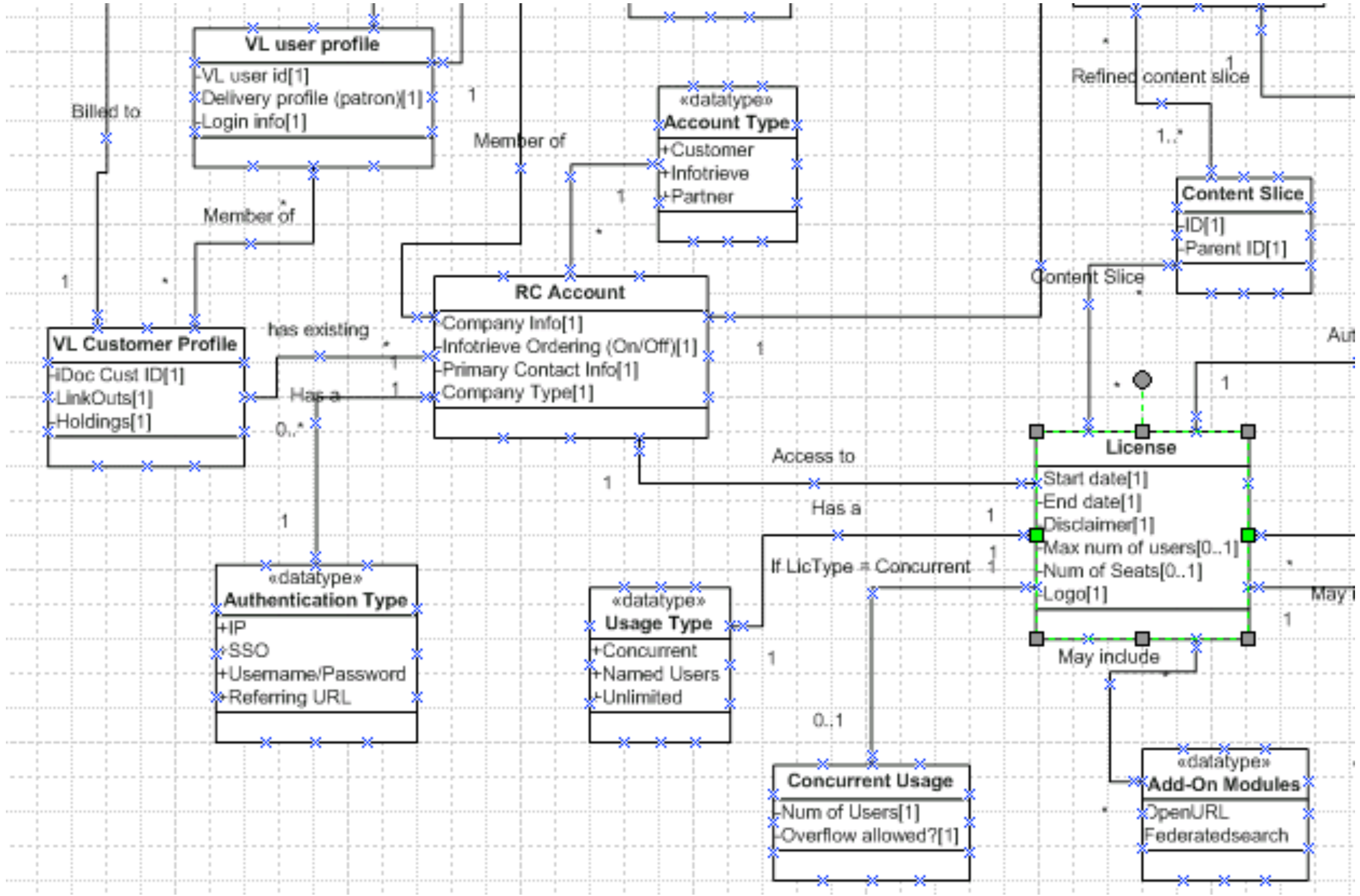
- Make a **list** of candidate domain classes – identify nouns.
- Draw these classes in a UML **class** diagram.
- If possible, add brief **descriptions** for the classes.
- Identify any **associations** that are necessary.
- Decide if some domain classes are really just **attributes**.
- Where helpful, identify **role** names and **multiplicity** for associations.
- Add any additional **static rules** as UML notes that cannot be conveyed with UML symbols.
- Group diagrams/domain classes by category into **packages**.

Example: TENCompetence learning network



[http://dspace.ou.nl/handle/1820/649]

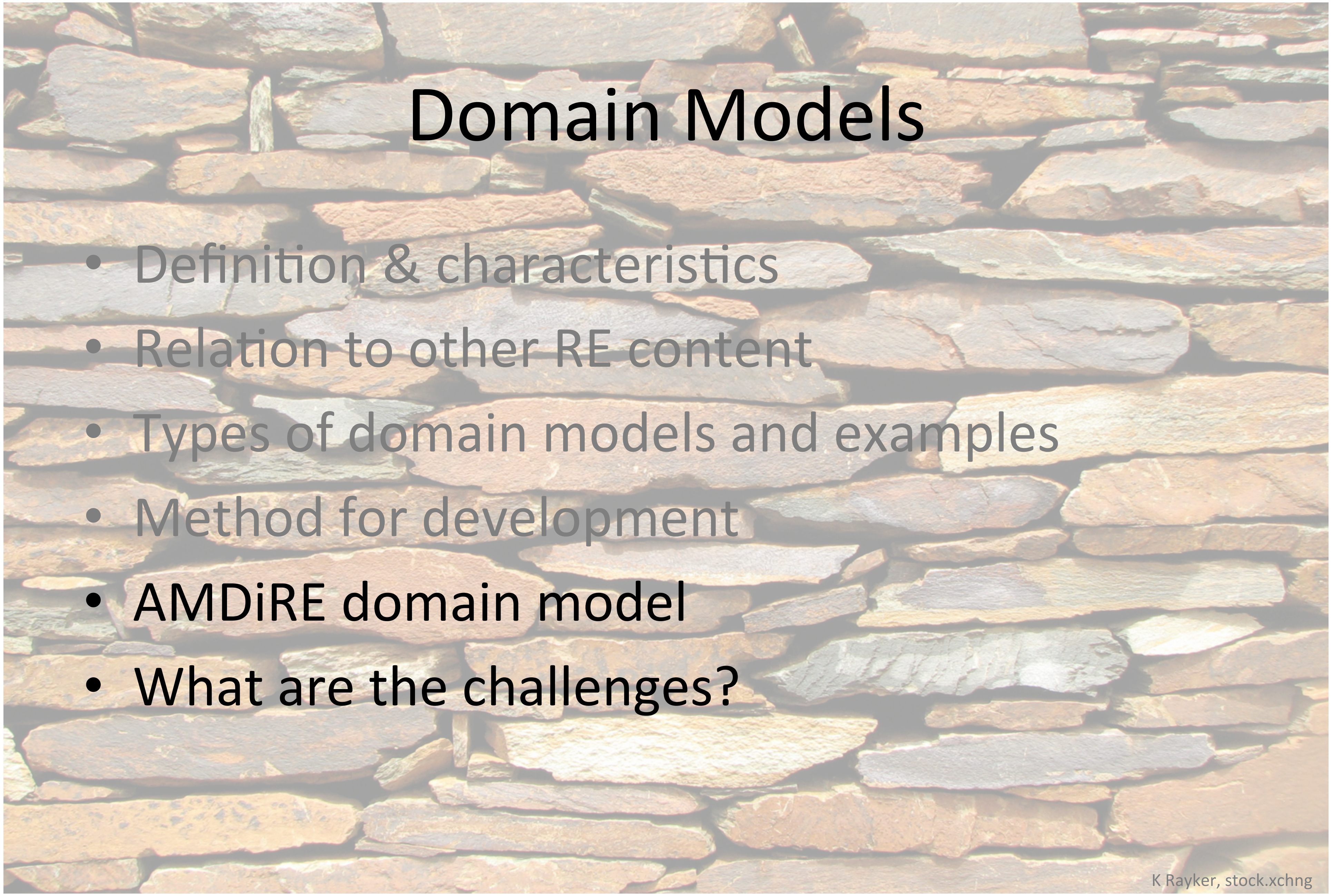
Example: Research Computing Domain Model



[http://www.bridging-the-gap.com/domain-models/]

Sources & QA

- Elaboration from
  - Business case analysis
  - Stakeholder input (goals, constraints)
  - Background research
- Quality: Completeness
  - All important concepts of problem domain
  - Adequate attributes
  - All important associations



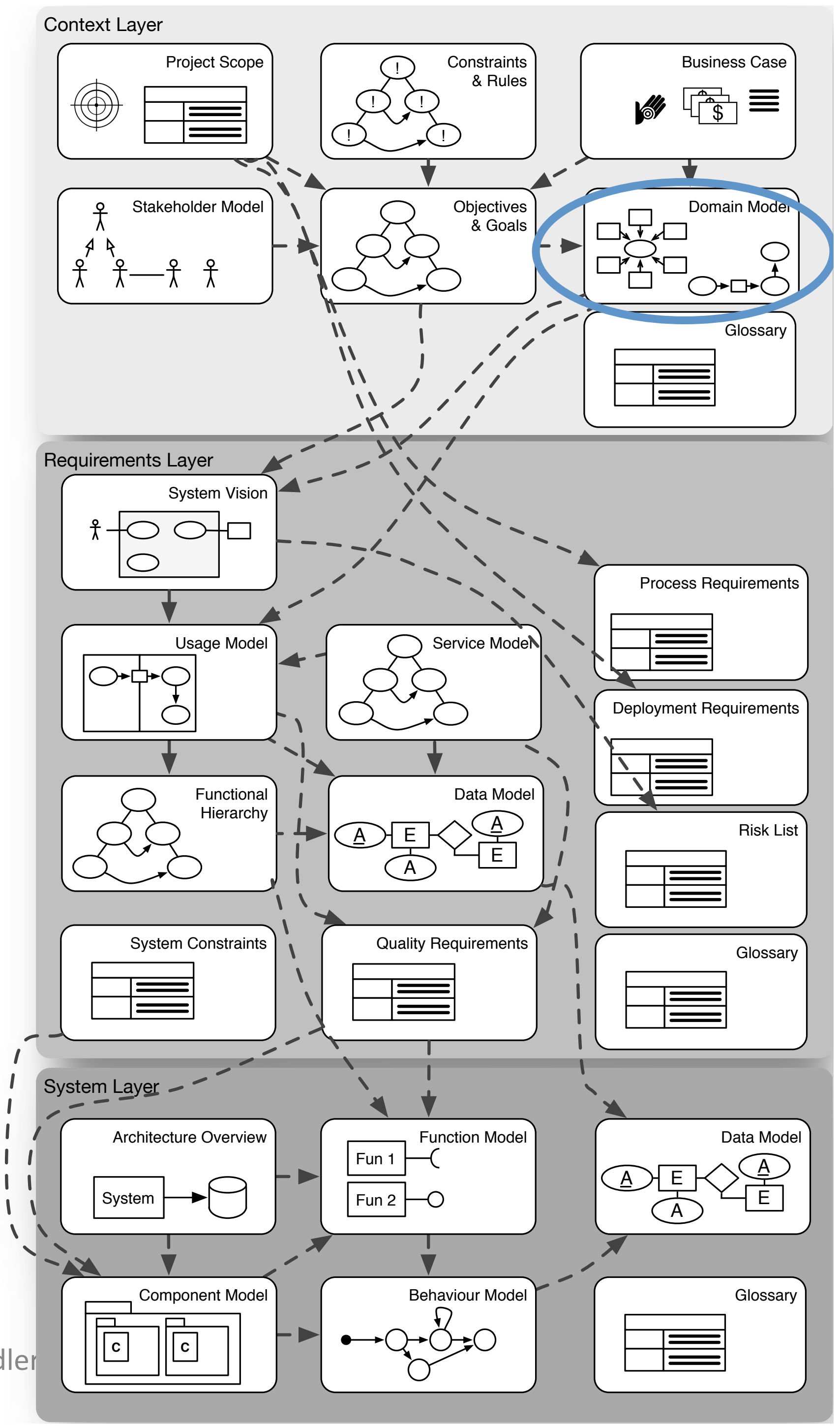
Domain Models

- Definition & characteristics
- Relation to other RE content
- Types of domain models and examples
- Method for development
- AMDiRE domain model
- What are the challenges?

K Rayker, stock.xchng

Elaboration in AMDiRE

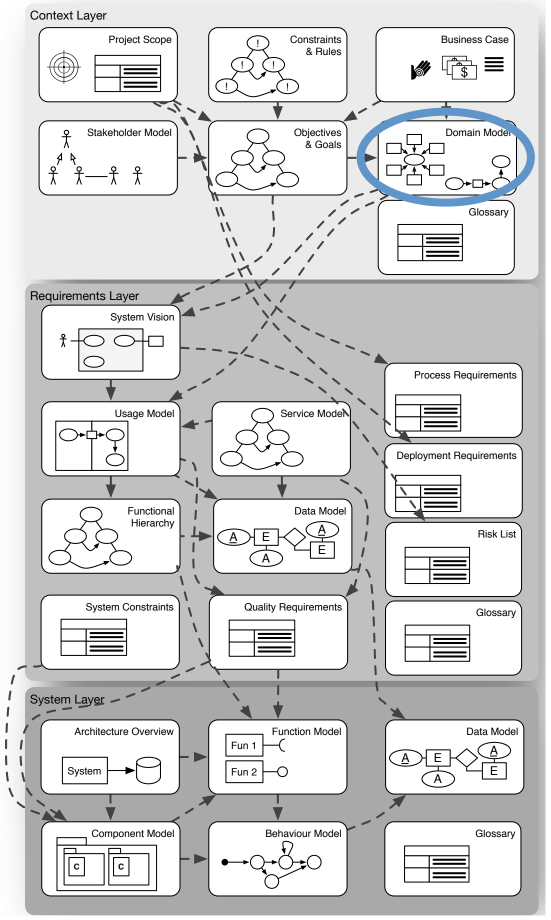
- Hand sketch (draft)
- Class diagram (iteration)



Challenges

- Problem domain (conceptual objects)
- Completeness (classes, associations, attributes)

→ The better your domain model, the easier it will be to develop the system vision and usage model.


Context Layer
Project Scope
Constraints & Rules
Business Case
Stakeholder Model
Objectives & Goals
Domain Model
Glossary
Requirements Layer
System Vision
Process Requirements
Usage Model
Service Model
Deployment Requirements
Functional Hierarchy
Data Model
Risk List
System Constraints
Quality Requirements
Glossary
System Layer
Architecture Overview
System
Function Model
Fun 1
Fun 2
Data Model
Component Model
Behaviour Model
Glossary