SWEN-262
Engineering of Software Subsystems
Domain Modeling
Application Domain

The domain for a software system defines the context in which the software operates.

- This is also referred to as the *application domain*, for example:
  - Retail sales
  - Banking
  - Customer contact management
  - Checkers playing

- The domain model describes the world in which the system’s experts and users exist and work on a daily basis.
  - Domain entities
  - Domain language
  - Associations/relationships between entities
Domain Modeling

Domain modeling provides an understanding of the application problem space.

**Application Domain**
What are the *entities* and *relationships* (associations) within this domain?

**Domain Model**
What *entities* and *relationships* (associations) are important for this application?

**Class Model**
What software structure will provide the implementation of this application?

Use the language of the application and user! The domain model is *not* meant to define the implementation.

The *domain model* influences the naming and definition of the software class model.
Domain Modeling

The domain model identifies important aspects of the application, not the implementation.

- Only use vocabulary from the *problem statement*.
  - For example, a unique identifier needed to store data with no meaning to the user would not be in the domain model.
  - This helps to establish a common understanding of the problem for the customer/user and the software team.

- Domain entities are *not* classes.
  - Do not add entities to the diagram to represent software classes that are not described in the problem statement.
  - The point of a domain model is to make sure that you understand the customer problem. Do not begin your design before you understand the problem!

Believe it or not, the customer knows their needs better than you do (or ever will).

It is far better for you to speak their language when learning about their problem.
Domain Modeling

The domain model is typically drawn using a simplified class diagram (UML) notation.

- Show the following information
  - Domain entities
  - Attributes in domain entities
  - Relationships (Associations) between domain entities

- Use *user* vocabulary
  - Attributes *do not* indicate data type.

- Relationships come from the problem statement
  - Place a label on the association line
  - Usually completes a phrase between two domain entities:
    - $DE_1 <\text{relationship}> DE_2$
  - Indicate multiplicity, if known
  - Use inheritance, if appropriate
Relationships

- All relationships (associations) should have an arrow to indicate the direction to read the association.
  - entity₁ - relationship - entity₂ read in the direction of the arrow should read like an English sentence.

Darth Vader is the father of Luke Skywalker

Luke Skywalker is a huge disappointment to Darth Vader

I always liked your sister more...

No! That's not true! It's impossible!
Relationships

- Labels for the relationship that use a generic term like “has,” “is,” or “contains;” are not very descriptive or useful.
  - Try reversing the direction of the relationship to force yourself to rephrase it.

Instead of...

try...

Raphael

has

Sai

Raphael

is the preferred weapon of

Sai
Domain Modeling

This partial domain model for a game of Monopoly demonstrates these ideas.

Die is played with 2 Count

Monopoly Game played on 1

Board is made up of 1

Player plays 1

Piece owns 1

Square occupies and is affected by 0..8

Die

Monopoly Game

Board

Player

Piece

Square

Character

Location
Domain Modeling

- As you learn more about the project, the domain model will continue to evolve.
  - Working on the project gives you a different understanding of the domain.
  - New features change your understanding of the domain.
  - When user stories are refined during backlog refinement, more details may come out about the domain.

- Keep your domain model up-to-date so that there is always a common understanding between the development team and the Product Owner.

The domain model helps to insure that you and I are on the same page at every stage of planning and development.