Course: Software Development Methods.
Prof. Luca Dan Serbanati
Conf. Andrei Vasilateanu

GUIDELINES FOR LABORATORY AND PROJECT ACTIVITY

HW3. Domain Analysis

Deliveries: Problem Domain Model

THIRD MACRO-ACTIVITY: IDENTIFY THE PROBLEM RESOURCES

A **problem domain** is the area of expertise or application that needs to be examined to solve a problem. A problem domain is simply looking at only the topics of an individual's interest, and excluding everything else: the domain refers to relevant topics solely within the delimited area of interest of the problem.

The scope of the problem domain needs to be identified upfront by the business analyst. The size and scope of the problem domain can vary greatly depending on the goals of the project being undertaken. The scope may align with the boundaries of an entire organization or it may be much more granular, aligning with a single organizational unit, a specific business process, or a particular system.

Even when the scope of the problem domain aligns with the boundaries of a particular group or system it may also include stakeholders outside of the process or organizational group such as customers, suppliers, or any other stakeholder which provides an input or accepts an output of a process, organization, or system, In short, the Problem Domain is anything and everything that is needed to define the area under analysis, fully understand the inputs and outputs of its processes, and achieve the goals of the area under analysis, but nothing more.

A problem domain model generally uses the vocabulary of the business domain so that a representation of the model can be used to communicate with non-technical stakeholders.

Problem domain modeling is a way to model and describe real world entities and the relationships between them, as well as the responsibilities that cover the problem domain, which collectively describe the problem domain space. Derived from an understanding of system-level requirements , identifying domain entities and their relationships provides an effective basis for understanding and helps practitioners design systems for maintainability, testability, and incremental development. Because there is often a gap between understanding the problem domain and the interpretation of requirements, domain modeling should be a primary modeling area concurrently carried out with requirements analysis. In an OO approach domain modeling envisions the problem solution as a set of domain models that collaborate to fulfill system-level scenarios.

Even the agile software methodologies - so parsimonious with documents writing - suggest creating a domain model as indispensable development activity.

In software engineering, a domain model is part of the conceptual model of the domain that incorporates both behavior and information. It is a system of abstractions that describes selected aspects of a sphere of knowledge, influence or activity, we called *domain* this sphere. The model can then be used to solve problems related to that domain. So, the domain model is a representation of meaningful real-world concepts pertinent to the domain that need to be modeled in software. The concepts include the information (data) involved in the business and rules the business uses in relation to that data.

Here are the steps for your homework.

A. **Building the problem domain model**. To analyze and model a problem domain, its main components should be first identified and described. Based on gathered requirements and other information sources:

   1. Find the concepts that can be applied to entities in the domain. Identify these concepts from the problem statement and use case descriptions.
   2. Promote the found concepts to object oriented classes. Draw them in a UML class diagram.
   3. Add to each pair of classes their semantic associations, mainly associations, that is identify conceptual connections between concepts from the problem statement and use case descriptions.
   4. Promote conceptual connections as associations between corresponding classes in the domain model. Refine them by identifying the compositions (eventually aggregations).
   5. Add to each class its attributes of interest for our problem. For the objects of each class identified at the previous step, identify and analyze the application-meaningful properties.

Have in mind the following procedure to be applied for concepts and relationships identification:

   1. Reuse or modify existing, available models of the same domain, if any.
   2. Use the category lists for concepts and respectively for semantic associations.
   3. Identify nouns and verbs in the documentation on the domain and select the most appropriate to be concepts, respectively semantic connections in your model.
   4. Use analysis models like Party, Organization hierarchy, Accountability, Quantity, Measurement, Observation, etc., to model combinations of concepts and relationships in your domain.
   5. Identify the concepts' properties of interest for your problem. Promote them as attributes in your model.

B. Construct the domain class diagram with the identified concepts, semantic connections and attributes.

C. Color the problem classes in your model according to the four class archetypes: Act, Role, Entity, and Description (eventually Participation, if needed). Verify that your model is complained with the basic model archetype.