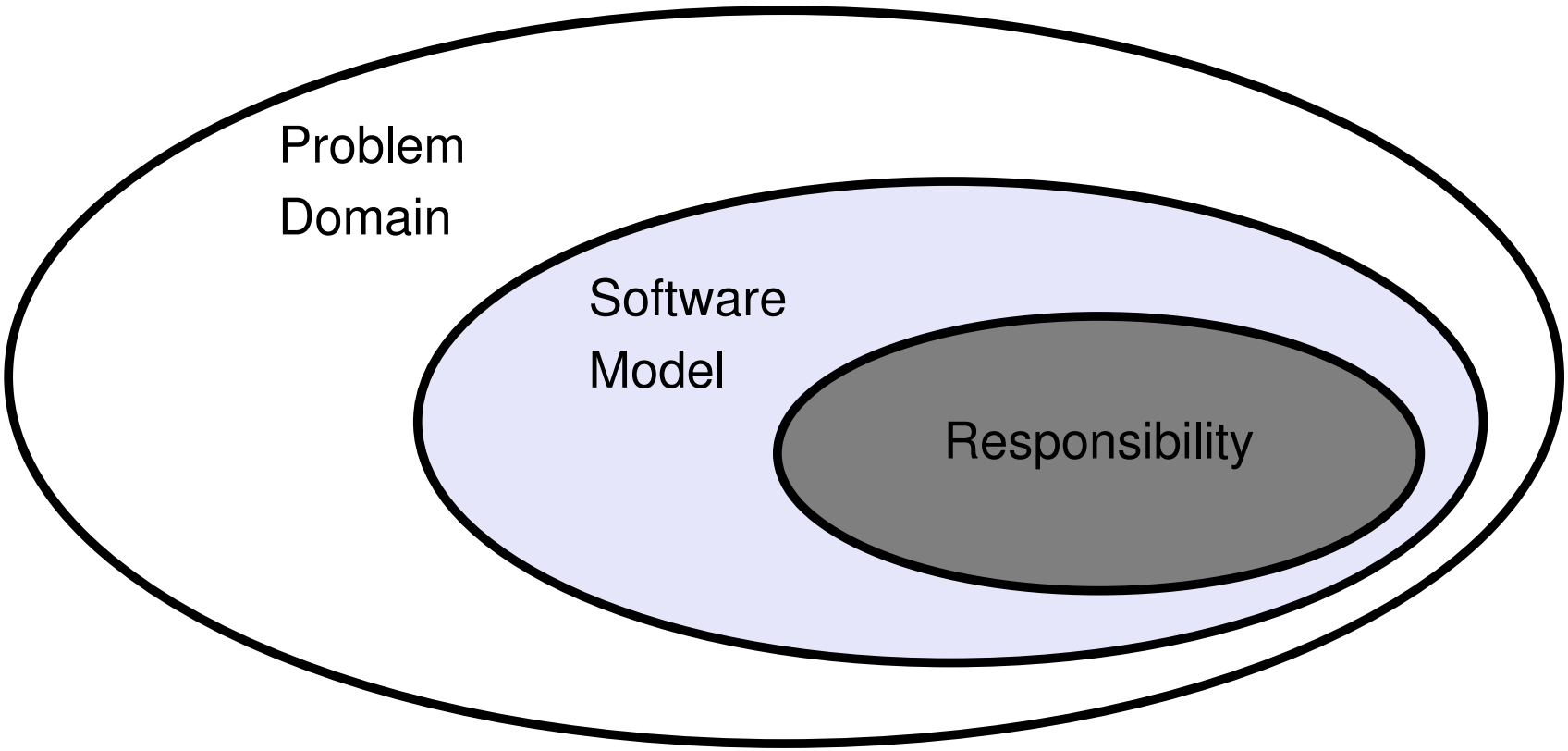
Software Analysis and Design

Software analysis goal

- The goal of requirement analysis phase is answer to question: what software must do (and with what constraints)?
- The goal of software analysis phase is answer to question: how system should work?
- The goal of software development phase is answer to question: how system should be implemented?

The problem domain, software model and responsibility



Object oriented analysis

Goal of object oriented analysis

The goal of object oriented analysis is to prepare model, that will represent software according to the user needs.

Steps

1. Agree with the customer basic requirements of the system
2. Identify classes
3. Prepare class hierarchy
4. Describe relationships between objects
5. Prepare object behavior model
6. Repeat steps 1-5 until complete model is finished

Software analysis phase

Software engineering elements that are used during analysis phase:

- notations for model record,
- methods of model preparation,
- tools for easy use of notations and methods.

Roles and types of notation

Following notations can be used for model recording:

- natural language,
- graphical notations,
- specifications – partially structured text and numerical notation.

Notation can be used:

- as a primary tool used by an analyst,
- as a tool for communication with user,
- as a tool for communication with team members,
- as a basis for software implementation,
- as a technical documentation format.

Structural methods

Components of an analyzed system:

- passive components – reflect storing data in a system,
- active components – reflect executing operations in a system.

Object methods

The system is analyzed in object oriented way, when:

- it is divided into objects, that have:
  - identity,
  - state,
  - behavior,
- objects are grouped into classes composed of objects with similar state and behavior.

Object methods

Object oriented programming simplify:

- data encapsulation,
- reuse of components
- reuse of already prepared software,
- rapid prototyping,
- event-driven programming,
- incremental programming.

Class specification

Basic elements of class specification are:

- name,
- general description.

Additional elements of class specification are:

- attributes list,
- operations list,
- constraints, that must be fulfilled by attributes,
- estimated or accurate number of this class objects
- persistence.

Operations specification

Operations are specified by giving following information:

- input data,
- output data,
- algorithm,
- preconditions,
- postconditions,
- exceptions,
- time complexity,
- memory complexity.

Attributes specification

Basic attributes are specified by giving following data:

- type of stored value,
- unit of measure,
- range of acceptable values,
- list of possible values,
- required precision,
- default value,
- information, if attribute can be empty.

For all types of attributes following data can be specified:

- constraints,
- operations that can read or modify the attribute.

Object model preparation process

In the object model preparation process we can recognize four main tasks:

- identification of classes and objects,
- identification of classes and objects relationships,
- identification and specification of attributes,
- identification and specification of operations.

Classes and objects identification

Methods of classes and objects identification:

- Using tangible entities in the application domain
- The domain analysis process
- Using a grammatical analysis of a natural language description
- Using classes and objects relationships
- Using a scenario based analysis

Using typical classes in the application domain

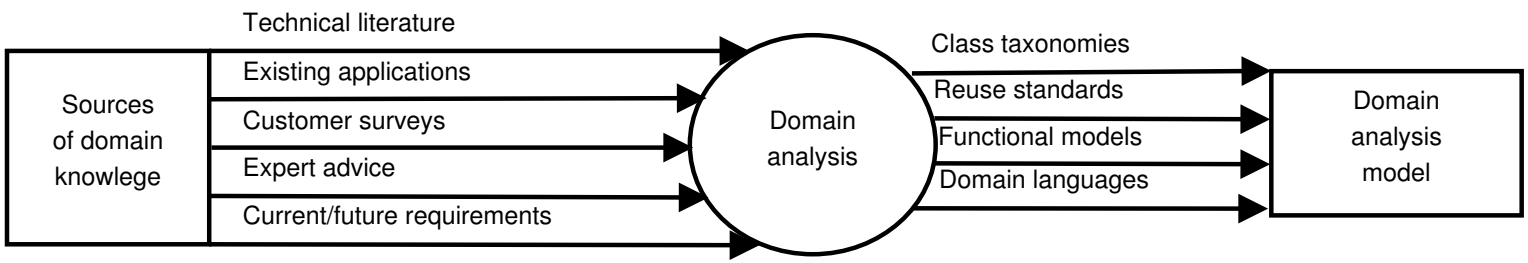
Typical classes:

- tangible items (car, sensor)
- roles played by persons (employee, lecturer, politician)
- events of which the system stores information (order, plane landing)
- interactions between individuals or systems for which the system stores information (loan, meeting)
- locations – places designed for people or objects
- groups of tangible items (cars, sensors)
- organizations (company, department, association)
- conceptions (quality standard, job)
- documents (driving license, invoice)
- classes which are interface to external systems
- classes which are interface to hardware

Domain Analysis

The Domain Analysis Process

Software domain analysis is the identification, analysis, and specification of common requirements from a specific application domain, typically for reuse on multiple projects within that application domain.



Rysunek: Input and output for domain analysis

Domain Analysis

- Define the domain to be investigated
- Categorize the items extracted from the domain
- Collect a representative sample of applications in the domain
- Analyze each application in the sample
- Develop an analysis model for the objects

Grammatical analysis of a natural language description

Properties:

- natural language description
- distinguishing nouns (classes, objects, attributes)
- distinguishing verbs (operations, relationships)

Identifying elements of an object model

Objects can be:

- External entities
- Things
- Occurrences or events
- Roles
- Organizational Units
- Places
- Structures

Potential object inclusion in the model:

Selection characteristics:

1. Retained information
2. Needed services
3. Multiple attributes
4. Common attributes
5. Common operations
6. Essential requirements

Potential object inclusion in the model – example

Description:

SafeHome software enables the homeowner to configure the security system when it is installed, monitors all sensors connected to the security system, and interacts with the homeowner through a keypad and function keys contained in the *SafeHome* control panel.
During installation, the *SafeHome* control panel is used to "program" and configure the system. Each sensor is assigned a number and type, a master password is programmed for arming and disarming the system, and telephone number(s) are input for dialing when a sensor event occurs. When a sensor event is sensed by the software, it rings an audible alarm attached to the system. After a delay time that is specified by the homeowner during system configuration activities, the software dials a telephone number of a monitoring service, provides information about the location, reporting and the nature of the event that has been detected. The number will be redialed every 20 seconds until telephone connection is obtained.

Potential object inclusion in the model – example

Potential Object/Class	General classification
homeowner	role or external entity
sensor	external entity
control panel	external entity
installation	occurrence
system (alias security system)	thing
number, type	not objects, attributes of sensor
master password	thing
telephone number	thing
sensor event	occurrence
audible alarm	external entity
monitoring service	organizational unit or external entity

Potential object inclusion in the model – example

Potential Object/Class	Characteristic Number That Applies
homeowner	rejected: 1 i 2 fail even though 6 applies
sensor	accepted: all apply
control panel	accepted: all apply
installation	rejected
system (alias security system)	accepted: all apply
number, type	rejected: 3 fails, attributes of sensor
master password	rejected: 3 fails
telephone number	rejected: 3 fails
sensor event	accepted: all apply
audible alarm	accepted: 2, 3, 4, 5, 6 apply
monitoring service	rejected: 1, 2 fail even though 6 applies

Identifying elements of an object model

- attributes specification
- operations specification
- supplementing object details

Using classes and objects relationships

Consider:

- Does class has the potential specializations and/or generalizations?
- Does class has components and/or is part of a larger whole?
- Whether class remains in relationships with other classes?

Using a scenario (use case) based analysis

Method of conducting

- selection of some system function
- creation of the functions implementation description in a natural language
- creation of the objects interaction description

Use Cases

Use cases are identified during requirement analysis to:

- describe the functional requirements in the use case scenarios
- prepare clear and unambiguous description of system and users interaction
- allow validation

Example

Alarm system – cases of user interaction with the system

- enter password in order to get an authorization for work with the system
- check status of security zones
- check status of sensors
- press button during emergency
- system start or shutdown

Class and object verification

Potential problems:

- Lack of attributes and operations
- Few attributes or methods
- Only one object per class
- Lack of relationships with other classes

Attributes identification and specification

Answer the questions:

- What is needed to describe the class as part of the problem domain?
- What data methods of the class will need to fulfill their tasks?
- What attributes should be introduced to describe the states in which objects of that class may be?

Operations identification and specification

Operations division:

- algorithmic simple,
  - constructors,
  - destructors,
  - methods for getting/setting values of attributes,
  - methods for setting relationships between objects,
  - methods for editing values of attributes,
- algorithmic complex,
  - methods for making calculations,
  - methods for tracking external systems or devices.

Bibliography

- Pressman R. S.: *Software Engineering: A Practitioner's Approach*, WNT, Warszawa 2004