Requirement Analysis

Requirements analysis results in

- Specification of software's operational characteristics.
- Indicates software's interface with other system elements.
- Establishes constraints that software must meet.

The requirements modelling action results in one or more of the following types of models:

- **Scenario-based models** of requirements from the point of view of various system "actors".
- **Data models** that depict the information domain for the problem
- **Class-oriented models** that represent object-oriented classes (attributes and operations) and the manner in which classes collaborate to achieve system requirements
- **Flow-oriented models** that represent the functional elements of the system and how they transform data as it moves through the system
- **Behavioural models** that depict how the software behaves as a consequence of external "events"

Overall Objectives and Philosophy

Throughout requirements modelling, your primary focus is on what, not how.

- What user interaction occurs in a particular circumstance
- what objects does the system manipulate
- what functions must the system perform what behaviours does the system exhibit
- what interfaces are defined
- what constraints apply

Requirements analysis and modelling follows iterative approach as customers unsure of what is needed and developer unsure what approach will give good performance.
The requirements model must achieve three primary objectives:

1. To describe what the customer requires.
2. To establish a basis for the creation of a software design.
3. To define a set of requirements that can be validated once the software is built.

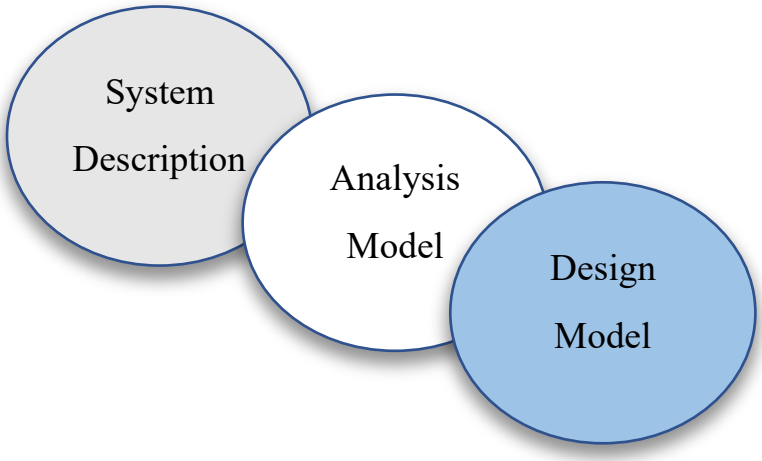

Figure: Requirement Model

The requirements model acts as a bridge between the system description and the design model as shown in the following figure.

Analysis Rules of Thumb

The rules of thumb that must be followed while creating the analysis model.

The rules are as follows:

- The model focuses on the requirements that are visible in the business domain. The level of abstraction must be high that is do not give internal details.
- Every element in the model should give overall understanding of the software requirement and focus on the information, function and behaviour of the system.
- Delay consideration of infrastructure and other non-functional models until design. For example**,** in case of database, the classes, functions and behaviour of that database are not initially required. Hence, they should be considered only after the completion of problem domain.
- Minimize coupling throughout the system. The interconnections between the modules is known as 'coupling'.
- The model should give value to all the stakeholders participated in that model.
- Keep the model as simple as it can be. Because simple model always helps in easy understanding of the requirement.

Domain Analysis

Analysis patterns often reoccur across many applications within a specific business domain. Domain analysis defines these analysis patterns, categorise them and make them ready to use. Software domain analysis is the identification, analysis, and specification of common requirements from a specific application domain.
The role of the domain analyst is to discover and define analysis patterns, analysis classes, and related information that may be used by many people working on similar but not necessarily the same applications.

The following figure illustrates key inputs and outputs for the domain analysis process. Sources of domain knowledge are surveyed in an attempt to identify objects that can be reused across the domain.

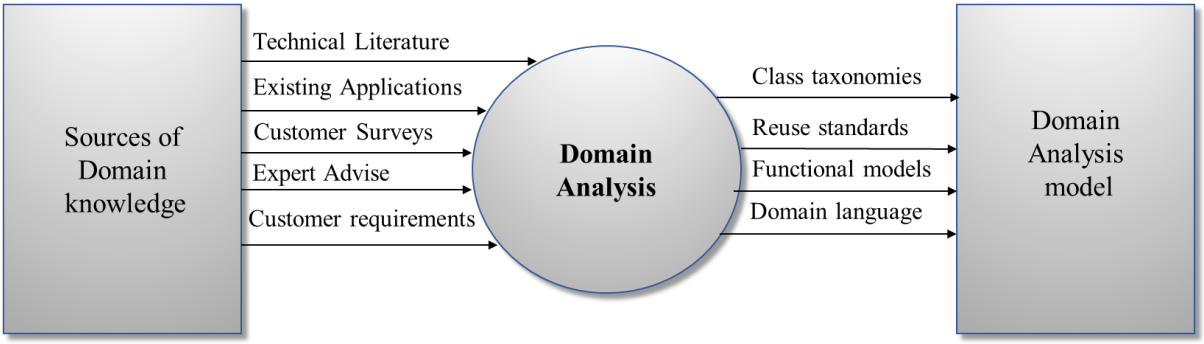

Figure: Inputs and outputs for domain analysis

Text Books

1. Roger Pressman S., “Software Engineering: A Practitioner's Approach”, 7th Edition, McGraw Hill, 2010.
2. Sommerville, “Software Engineering”, Eighth Edition, Pearson Education, 2007

Web Links

1. https://drive.google.com/file/d/1JUKl9zg1piYxbj0KPiwSybCJc7Rb3COb/view
2. https://drive.google.com/file/d/1wLbTovaAsAwZse4Pfrc3_7KOjX2qdwp6/view
3. https://cdn.shopify.com/s/files/1/0457/4009/7694/files/software_engineering_pdf_pressman_7th_edition.pdf