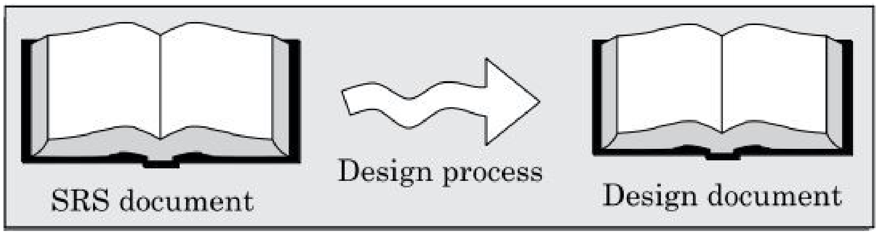
Design Concepts and Principles

The design creates a representation or model of the software, but unlike the requirements model (that focuses on describing required data, function, and behavior), the design model provides detail about software architecture, data structures, interfaces, and components that are necessary to implement the system. The design focuses on four major areas:

- **Data**
- **Architecture**
- **Interfaces**
- **components**.

the design process starts using the **SRS document** and completes with the production of the **design document**. The design document produced at the end of the design phase should be implementable using a programming language in the coding phase.



- **Who does it?** Software engineers conduct each of the design tasks.

- **Why is it important?** one wouldn't attempt to build a house without a blueprint, would you? Likewise, for computer software_ and which is considerably more complex than a house_ a blueprint is needed, and that is the design.

The importance of software design can be stated with a single word—***quality***.

- **What are the steps?** The design depicts the software in a number of different ways:
  - First, the architecture of the system or product must be represented.
  - Then, the interfaces that connect the software to end users or to other systems and devices are modeled.
  - Finally, the software components that are used to construct the system are designed.

  Each of these views represents a different design action, but all must conform to a set of basic design concepts that guide software design work.

- **Design levels**

The design process consists of two distinct levels:

<u>high-level design, and low-level design</u>.

Differences between High-Level Design and Low-Level Design:

High-Level Design	Low-Level Design
High-Level Design in short called as HLD.	Low-Level Design in short called as LLD.

High-Level Design is the general system design means it refers to the overall system design.	Low-Level Design is like detailing HI means it refers to the component-leve design process.
In HLD the input criteria are Software Requirement Specification (SRS).	In LLD the input criteria are reviewed High-Level Design (HLD).
In HLD the output criteria are database design, functional design, and review record.	In LLD the output criteria are program specification and unit test plan.
It is also known as macro-level/system design.	It is also known as micro-level/detailed design.

Software Design and Software Engineering

Software design sits at the technical kernel of software engineering. Each of the elements of the requirements model provides information that is necessary to create the four design models required for a complete specification of design.

The flow of information during software design is illustrated in Figure 1.1. The requirements model, clarified by scenario-based, class-based, flow-oriented, and behavioral elements, feed the design task. Using design notation and design methods will produce a data/class design, an architectural design, an interface design, and a component design.

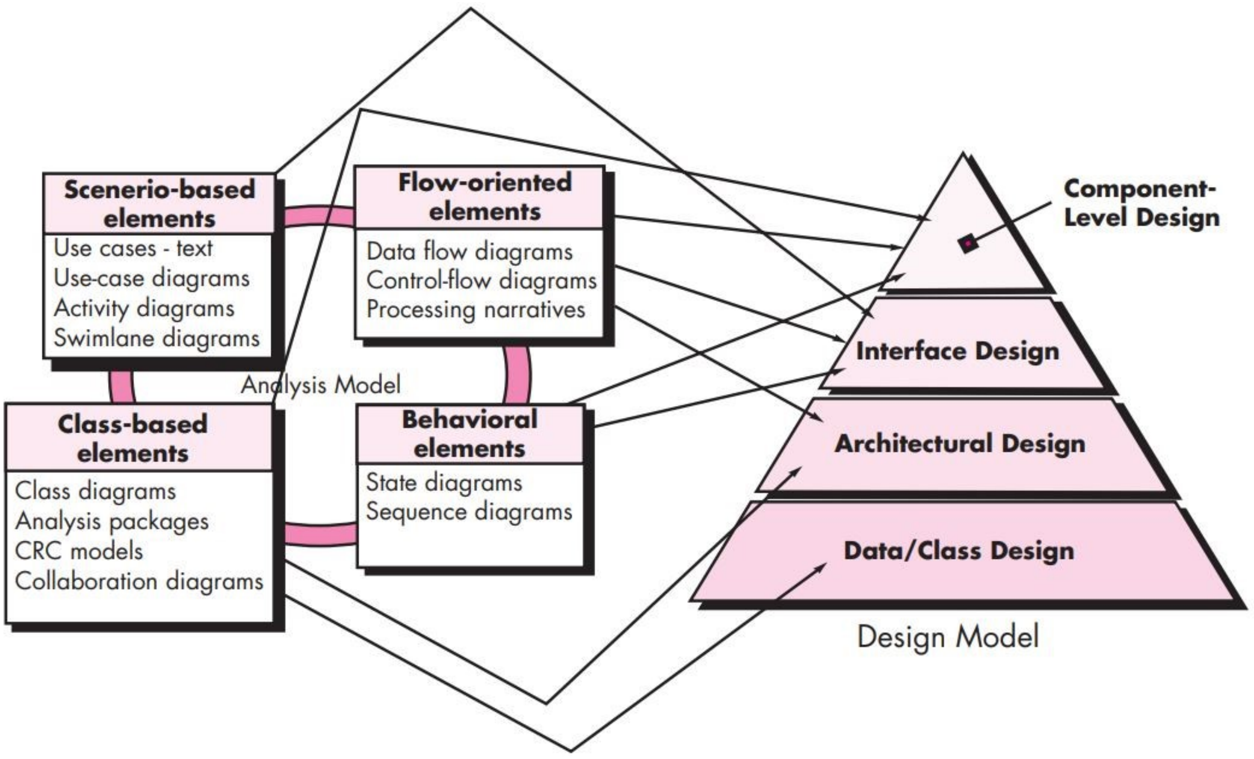


Figure 1.1: Translating the requirements model into the design model

1- **<u>The data/ class design</u>**

The data design transforms the information domain model *<u>(class models)</u>* created during analysis into the data structures that will be required to implement the software. Like other software engineering activities, data design (sometimes referred to as data architecting) creates a model of data and information that is represented at a high level of abstraction. This data model is then refined into gradually more implementation-specific representations that can be processed by the computer-based system. In many software applications, the architecture of the data will have a deep influence on the architecture of the software that must process it. The structure of data has always been an important part of software design. At the program component level, the design of data structures and the associated algorithms required to manipulate them is essential to the creation of high-quality applications.

2- **The architectural design**

Defines the relationship between:

- major structural elements of the software.
- the "design patterns" that can be used to achieve the requirements that have been defined for the system.
- and the constraints that affect the way in which architectural design patterns can be applied.

The architectural design representation can be derived from the analysis model (class-based element and flow-oriented element).

3- **The interface design**

Describes how the software communicates within:

- Itself
- with systems that interoperate with it
- and with humans who use it.

The interface design representation can be derived from the analysis model (scenario-based element, behavior element, and flow-oriented element).

4- **The component-level design**

Transforms structural elements of the software architecture into a procedural description of software components.

<u>There are three characteristics that use as a guide for the evaluation of a good design:</u>

1-The design must implement all of the explicit requirements contained in the analysis model, and it must accommodate all of the implicit requirements desired by the customer.

2- The design must be a readable, understandable guide for those who generate code and for those who test and subsequently support the software.

3- The design should provide a complete picture of the software, addressing the data, functional, and behavioral domains from an implementation.