2.4 PROCESS ORGANIZATION AND INTERACTIONS

- A set of interrletared actions (tasks) and activities to achieve a predefined result is called as process. Process has its own set of inputs and produces the output(s).
- Process is a sequence of serial and/or concurrent operations or tasks that transform and/or add value to a set of inputs to produce an output. Software process is a set of required activities and the outcome of the activities with a target to produce a software product.
- Software is developed/engineered effectively and efficiently with the help of processes (activities). In general a process is defined as, "a series of steps involving activities and resources which produces the desired/expected output." **OR**
- Processes are defined as, "discrete events that have inputs, outputs and energy for alteration of input data to output data."
- The various processes can be specified under a process system. A complex process is made up of many simple processes which may include one or more processes to complete the action/event. It forms a structural hierarchy of abstraction.
- The process system can be easily managed and understood if analysed in a hierarchical order. Process system is a dual phenomenon of change/no change or form/transform.
- The analysis of a process system hierarchy is performed on the basis of process architecture. Process architecture is basically the structural design of process systems of varying degrees of complexity.

2.4.1 SRS (Software Requirements Specification)

- The software requirements document (sometimes called the Software Requirements Specification or SRS) is an official statement of what the system developers should implement. It should include both the user requirements for a system and a detailed specification of the system requirements.
- It is a requirements specification for a software system, and show a complete description of the behavior of a system to be developed and may include a set of use cases that describe interactions the users will have with the software.
- Software requirement specification is a document that completely describes what the proposed software should do without describing how software will do it.
- The basic goal of the requirement phase is to produce the SRS, which describes the complete behavior of the proposed software. SRS is also helping the clients to understand their own needs.
- IEEE defines software requirements specification as, "a document that clearly and precisely describes each of the essential requirements (functions, performance, design constraints and quality attributes) of the software and the external interfaces. Each requirement is defined in such a way that its achievement can be objectively verified by a prescribed method, for example, inspection, demonstration, analysis or test."

eatures of SRS:

1. It forms the basis for software development.
2. SRS provides a reference for validation of the final software product.
3. It is a medium or media through the client and used needs are accurately specified determined.
4. SRS helps to clients to understand their own needs and requirements.
5. It establishes the basis for agreement between the client and the supplier.

ırposes served by SRS:

1. **Feedback:** Provides a feedback, which ensures to the user that the organization (which develops the software) understands the issues or problems to be solved and the software behavior necessary to address those problems.
2. **Decompose Problem into Components:** Organizes the information and divides the problem into its component parts in an orderly manner.

3. **Validation:** Uses validation strategies applied to the requirements to acknowledge that requirements are stated properly.
4. **Input to Design:** Contains sufficient detail in the functional system requirements to devise a design solution.
5. **Basis for Agreement Between the User and the Organization:** Provides a complete description of the functions to be performed by the system. In addition, it helps the users to determine whether the specified requirements are accomplished.
6. **Reduce the Development Effort:** Enables developers to consider user requirements before the designing of the system commences. As a result, 'rework' and inconsistencies in the later stages can be reduced.
7. **Estimating Costs and Schedules:** Determines the requirements of the system and thus enables the developer to have a 'rough' estimate of the total cost and schedule of the project.

2.4.1.1 Concept of SRS

- Requirements documents are essential when an outside contractor is developing the software system. The Software Requirements Specification (SRS) is a technical specification of requirements for the software product.
- The SRS records the outcome of the software requirements definition activity. SRS also known as requirements document.
- The main goal of software requirements definition is to completely and consistently specify the technical requirements for the software product in a concise and unambiguous manner, using formal notations as appropriate.
- Depending on the size and complexity of the software product, the SRS may consists of a few pages and it is based on the system definition.
- The requirements specification will state the "what" of the software product without implying "how". Software design is concerned with specifying how the product will provide the required features and goals.

I. Introduction	A. System reference		
	B. Overall description		
	C. Software project constraints		
II. Information Description	A. Information content representation	1. Data flow	
	B. Information flow representation	2. Control flow	
III. Functional Description	A. Functional partitioning	1. Processing narrative	
	B. Functional description	2. Restrictions / limitations	
	C. Control description	3. Performance requirements	
		4. Design constraints	
		5. Supporting diagrams	
IV. Behavioral Description	1. Control specification	2. Design constraints	
	A. System states		
	B. Events and actions		
V. Validation and Criteria	A. Performance bounds		
	B. Classes of tests		
	C. Expected software response		
	D. Special consideration		
VI. Bibliography			
VII. Appendix			

Fig. 2.41: Software Requirements Specification (SRS) Outline

- However, the simplified outline presented in Fig. 2.41 may be used as a framework for the specificatio

I. The "**introduction**" states the goals and objectives of the software, describing it in the content of t computer-based system. Actually, it is nothing but the software scope.

II. The "**information description**" provides a detailed description of the problems that the softwa must solve. Information content and relationships, flow and structure are documented. Hardwa software, and human interfaces are described for external system elements and internal softwa functions.

III. A description of each function required to solve the problem is presented in the "**functio description**." A processing narrative is provided for each function, design constraints are stated an justified; performance characteristics are stated, and one or more diagrams are included t graphically represent the overall structure of the software and interplay among software functio and other system elements.

IV. The "**behavioral description**" section of the specification examines the operation of the software a a consequence of external events and internally generated control characteristics. Probably the mos important, and ironically, the most often neglected section of a software requirements specificatio is "validation criteria".

- How do we recognize a successful implementation?
- What classes of tests must be conducted to validate function, performance and constituents?

V. Specification of **validation criteria** acts as an implicit review of all other requirements. It is essentia that time and attention be given to this section.

VI. The **bibliography** contains references to all documents that relate to the software. These include:

- other software engineering documentation,
- technical references,
- vendor literature, and
- standards

VII. The **appendix** contains information that supplements the specification. Tabular data, detailed description of algorithms, charts, graphs and other material are presented as appendices.

2.4.1.2 Need/Importance of SRS

- Following points describes need for SRS:

1. The basic purpose of SRS is to bridge the communication gap between the client, the users, and the developer. SRS is the medium through which the client and user needs are accurately specified.
2. Another important purpose of developing an SRS is helping the clients understand their own needs or requirements.
3. An SRS is important because it establishes the basis for agreement between the client and the supplier on what the software product will do.
4. An SRS provides a reference for validation of the final product.
5. A high-quality SRS is a prerequisite to high-quality software.
6. SRS enables developer to consider user requirements before the designing of the system. This reduces rework and inconsistencies, which reduce the development efforts.
7. SRS needed to provide a feedback, which ensures to the user that the organisation understands the issues or problems to be solved.
8. SRS determines the requirements of the system and thus it enables the developer to have a rough estimate of the total cost and schedule the software project.
9. SRS uses validation strategies applied to the requirements to acknowledge that requirements are stated properly.

2.4.1.3 Characteristics of SRS

- Software requirements specification should be accurate, complete, efficient, and of high quality, so that it does not affect the entire project plan. An SRS is said to be of high quality when the developer and user easily understand the prepared document.
- The characteristics of SRS are discussed below:

1. **Correct:** SRS is correct when all user requirements are stated in the requirements document. The stated requirements should be according to the desired system. This implies that each requirement is examined to ensure that it (SRS) represents user requirements. Note that there is no specified tool or procedure to assure the correctness of SRS. Correctness ensures that all specified requirements are performed correctly.
2. **Unambiguous:** SRS is unambiguous when every stated requirement has only one interpretation. This implies that each requirement is uniquely interpreted. In case there is a term used with multiple meanings, the requirements document should specify the meanings in the SRS so that it is clear and easy to understand.
3. **Complete:** SRS is complete when the requirements clearly define what the software is required to do. This includes all the requirements related to performance, design and functionality.
4. **Ranked for Importance/Stability:** All requirements are not equally important, hence each requirement is identified to make differences among other requirements. For this, it is essential to clearly identify each requirement. Stability implies the probability of changes in the requirement in future.
5. **Modifiable:** The requirements of the user can change, hence requirements document should be created in such a manner that those changes can be modified easily, consistently maintaining the structure and style of the SRS.
6. **Traceable:** SRS is traceable when the source of each requirement is clear and facilitates the reference of each requirement in future. For this, forward tracing and backward tracing are used. Forward tracing implies that each requirement should be traceable to design and code elements. Backward tracing implies defining each requirement explicitly referencing its source.
7. **Verifiable:** SRS is verifiable when the specified requirements can be verified with a cost-effective process to check whether the final software meets those requirements. The requirements are verified with the help of reviews. Note that unambiguity is essential for verifiability.
8. **Consistent:** SRS is consistent when the subsets of individual requirements defined do not conflict with each other. For example, there can be a case when different requirements can use different terms to refer to the same object. There can be logical or temporal conflicts between the specified requirements and some requirements whose logical or temporal characteristics are not satisfied.

2.4.1.4 SRS Format

- SRS is the standard statement of what the system developers should implement. SRS includes the user's requirements for a system and a detailed specification of the system requirement.
- SRS is the final work product produced by the requirements engineer. It serves as the foundation for subsequent software engineering activities.
- SRS describes the function and performance requirements of software. It also lists the constraints that will affect its development.
- A typical structure (template) of an SRS document is given in Fig. 2.42. The requirements of software may be described against each heading and sub-heading of this template.

Section 1:	Product Overview and Summary
Section 2:	Development, Operating, and Maintenance Environments
Section 3:	External Interfaces and Data Flow
Section 4:	Functional Requirements
Section 5:	Performance Requirements
Section 6:	Exception Handling
Section 7:	Early Subsets and Implementation Priorities
Section 8:	Foreseeable Modifications and Enhancement
Section 9:	Acceptance Criteria
Section 10:	Design Hints and Guidelines
Section 11:	Cross-reference Index
Section 12:	Glossary of Terms

Fig. 2.42: Structure/Format of SRS

- SRS is a formal document. It uses natural language, graphical representations, mathematical models usage scenarios, prototype model, or any combination of these to describe the software to be developed. There are standard templates for presenting requirement specifications in a consistent and more understandable manner.
- The SRS document should be well-structured. A well-structured document is easy to understand and modify.
- The requirements document is devised in a manner that is easier to write, review, and maintain. It is organized into independent sections and each section is organized into modules or units.
- Fig. 2.42 shows structure of SRS document. SRS format's sections are described below:
  - **Section 1 and 2** of SRS document present an overview of product features and summarize the processing environments for development, operation, and maintenance of the software product.
  - **Section 3** of SRS document specifies the externally observable characteristics of the software product and it includes user displays and report formats, a summary of user commands and report options, data flow diagrams, and a data dictionary.
  - **Section 4** of SRS document specifies the functional requirements for the software product. Typically, functional requirements are expressed in relational and state-oriented notations that specify relationships among inputs, actions and outputs etc.
  - **Section 5** of SRS document specifies the performance characteristics like response time for various activities, processing time for various processes, throughput, primary and secondary memory constraints, required telecommunication bandwidth, and unusual reliability requirements etc.
  - **Section 6** of SRS document specifies the exception handling, including the actions to be taken and the messages to be displayed in response to undesired situations or events or errors. Various categories of possible exceptions include temporary and permanent resource failure, incorrect, inconsistent or out of range input data, violation of capacity limits and violations of restrictions on operators etc.
  - **Section 7** of SRS document specifies early subsets and implementation priorities for the system under development and it is important to specify implementation priorities for various system capabilities.
  - **Section 8** of SRS document specifies foreseeable modifications and enhancements that may be incorporated into the product following initial product release.

- **Section 9** of SRS document specifies the software product acceptance criteria. Acceptance criteria specify functional and performance tests that must be performed, and the standards to be applied to source code, internal documentation and external documentations like the design specifications, the test plan, the user's manual, the principles of operations, and the installation and maintenance procedures and so on.
- **Section 10** of SRS document specifies design hints and guidelines. The "how to" of product implementation is the topic of software design, and should be deferred to the design phase of product.
- **Section 11** of SRS document specifies product requirements to the source of information used in deriving the requirements.
- **Section 12** of SRS document specifies definition of terms that may be unfamiliar to the customer and the product developers. Proper care should be taken to define standard terms that are used in non-standard ways.

- **Examples for SRS:** Table 2.1 shows what a basic SRS outline might look like. This example is an adaptation and extension of the IEEE Standard 830-1998.

Table 2.1: A Sample of a Basic SRS Outline

1. **Introduction:**
   - 1.1 Purpose
   - 1.2 Document conventions
   - 1.3 Intended audience
   - 1.4 Additional information
   - 1.5 Contact information/SRS team members
   - 1.6 References
2. **Overall Description:**
   - 2.1 Product perspective
   - 2.2 Product functions
   - 2.3 User classes and characteristics
   - 2.4 Operating environment
   - 2.5 User environment
   - 2.6 Design/implementation constraints
   - 2.7 Assumptions and dependencies
3. **External Interface Requirements:**
   - 3.1 User interfaces
   - 3.2 Hardware interfaces
   - 3.3 Software interfaces
   - 3.4 Communication protocols and interfaces
4. **System Features:**
   - 4.1 System feature A
     - 4.1.1 Description and priority
     - 4.1.2 Action/result
     - 4.1.3 Functional requirements
   - 4.2 System feature B
5. **Other Non-functional Requirements:**
   - 5.1 Performance requirements

Contd...

5.2 Safety requirements
5.3 Security requirements
5.4 Software quality attributes
5.5 Project documentation
5.6 User documentation

6. **Other Requirements:**

Appendix A: Terminology/Glossary/Definitions list
Appendix B: To be determined

Advantages of SRS:

1. Software SRS establishes the basic for agreement between the client and the supplier on what the software product will do.
2. A SRS provides a reference for validation of the final product.
3. A high-quality SRS is a prerequisite to high-quality software.
4. A high-quality SRS reduces the development cost.

2.5 SYSTEMS DESIGN