Requirements Specification

(Week 4)

Software Requirements Specification (SRS)

- Main aim of requirements specification:
  - Systematically organize the requirements arrived during requirements analysis.
  - Document requirements properly.

Software Requirements Specification

- The SRS document is useful in various contexts:
  - Statement of user needs
  - Contract document
  - Reference document
  - Definition for implementation

Software Requirements Specification: A Contract Document

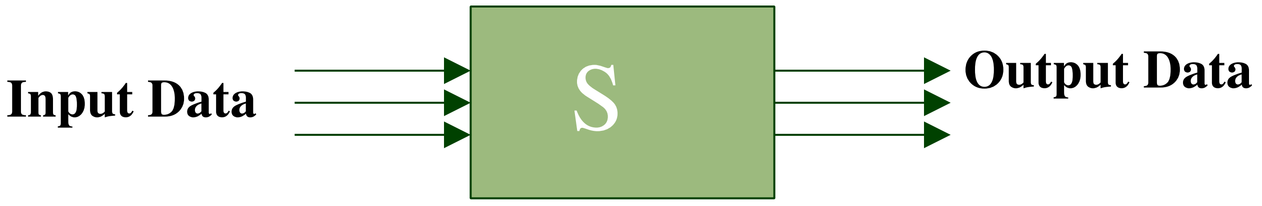
- Requirements document is a reference document.
- SRS document is a contract between the development team and the customer.
  - Once the SRS document is approved by the customer,
    - Any subsequent controversies are settled by referring the SRS document.

Software Requirements Specification: A Contract Document

- Once customer agrees to the SRS document:
  - Development team starts to develop the product according to the requirements recorded in the SRS document.
- The final product will be acceptable to the customer:
  - As long as it satisfies all the requirements recorded in the SRS document.

SRS Document (CONT.)

- The SRS document is known as black-box specification:
  - The system is considered as a black box whose internal details are not known.
  - Only its visible external (i.e. input/output) behavior is documented.



SRS Document (CONT.)

- SRS document concentrates on:
  - What needs to be done
  - Carefully avoids the solution ("how to do") aspects.
- The SRS document serves as a contract
  - Between development team and the customer.
  - Should be carefully written

SRS Document (CONT.)

- The requirements at this stage:
  - Written using end-user terminology.
- If necessary:
  - Later a formal requirement specification may be developed from it.

Properties of a Good SRS Document

- It should be concise
  - and at the same time should not be ambiguous.
- It should specify what the system must do
  - and not say how to do it.
- Easy to change.,
  - i.e. it should be well-structured.
- It should be consistent
- It should be complete

Properties of a Good SRS Document (cont...)

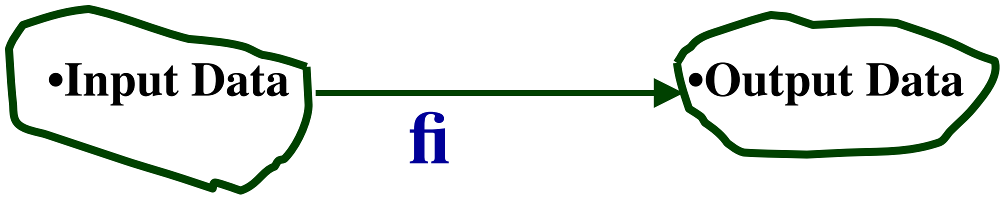
- It should be traceable
  - You should be able to trace which part of the specification corresponds to which part of the design, code, etc and vice versa.
- It should be verifiable
  - e.g. “system should be user friendly” is not verifiable

SRS Document (CONT.)

- SRS document, normally contains three important parts:
  - Functional requirements,
  - Non-functional requirements,
  - Goals of Implementation.

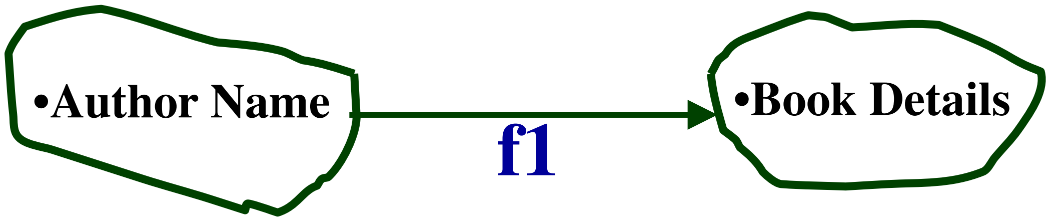
SRS Document (CONT.)

- It is desirable to consider every system:
  - Performing a set of functions {fi}.
  - Each function fi considered as:
  - Transforming a set of input data to corresponding output data.



Example: Functional Requirement

- F1: Search Book
  - Input:
    - an author's name:
  - Output:
    - details of the author's books and the locations of these books in the library.



Functional Requirements

- Functional requirements describe:
  - A set of high-level requirements
  - Each high-level requirement:
    - takes in some data from the user
    - outputs some data to the user
  - Each high-level requirement:
    - might consist of a set of identifiable functions

Functional Requirements

- For each high-level requirement:
  - Every function is described in terms of:
    - Input data set
    - Output data set
    - Processing required to obtain the output data set from the input data set.

Nonfunctional Requirements

- Characteristics of the system which can not be expressed as functions:
  - Maintainability,
  - Portability,
  - Usability, etc.

Nonfunctional Requirements

- Nonfunctional requirements include:
  - Reliability issues,
  - Performance issues:
    - Example: How fast the system can produce results
      - so that it does not overload another system to which it supplies data, etc.
  - Human-computer interface issues,
  - Interface with other external systems,
  - Security, maintainability, etc.

Non-Functional Requirements

- Hardware to be used,
- Operating system
  - or DBMS to be used
- Capabilities of I/O devices
- Standards compliance
- Data representations
  - by the interfaced system

Goals of Implementation

- Goals describe things that are desirable of the system:
  - But, would not be checked for compliance.
  - For example,
    - Reusability issues
    - Functionalities to be developed in future

Organization of the SRS Document

- Introduction.
- Functional Requirements
- Nonfunctional Requirements
  - External interface requirements
  - Performance requirements
- Goals of implementation

Functional Requirements

- A high-level function is one:
  - Using which the user can get some useful piece of work done.
- Can the receipt printing work during withdrawal of money from an ATM:
  - Be called a useful piece of work?
- A high-level requirement typically involves:
  - Accepting some data from the user,
  - Transforming it to the required response, and then
  - Outputting the system response to the user.

High-Level Function

- A high-level function:
  - Usually involves a series of interactions between the system and one or more users.
- Even for the same high-level function,
  - There can be different interaction sequences (or scenarios)
  - Due to users selecting different options or entering different data items.

Example Functional Requirements

- List all functional requirements
  - with proper numbering.
- Req. 1:
  - Once the user selects the “search” option,
    - he is asked to enter the key words.
  - The system should output details of all books
    - whose title or author name matches any of the key words entered.
    - Details include: Title, Author Name, Publisher name, Year of Publication, ISBN Number, Catalog Number, Location in the Library.

Example Functional Requirements

- Req. 2:
  - When the “renew” option is selected,
    - The user is asked to enter his membership number and password.
  - After password validation,
    - The list of the books borrowed by him are displayed.
  - The user can renew any of the books:
    - By clicking in the corresponding renew box.

Req. 1:

- R.1.1:
  - Input: “search” option,
  - Output: user prompted to enter the key words.
- R1.2:
  - Input: key words
  - Output: Details of all books whose title or author name matches any of the key words.
    - Details include: Title, Author Name, Publisher name, Year of Publication, ISBN Number, Catalog Number, Location in the Library.
  - Processing: Search the book list for the keywords

Alternatively/Additionally…

- **Use cases** can be used for representing functional requirements

Req. 2:

- R2.1:
  - Input: “renew” option selected,
  - Output: user prompted to enter his membership number and password.
- R2.2:
  - Input: membership number and password
  - Output:
    - list of the books borrowed by user are displayed. User prompted to enter books to be renewed or
    - user informed about bad password
  - Processing: Password validation, search books issued to the user from borrower list and display.

Req. 2:

- R2.3:
  - Input: user choice for renewal of the books issued to him through mouse clicks in the corresponding renew box.
  - Output: Confirmation of the books renewed
  - Processing: Renew the books selected by the in the borrower list.

Examples of Bad SRS Documents

- <u>Unstructured Specifications:</u>
  - Narrative essay --- one of the worst types of specification document:
    - Difficult to change,
    - Difficult to be precise,
    - Difficult to be unambiguous,
    - Scope for contradictions, etc.

Examples of Bad SRS Documents

- <u>Noise:</u>
  - Presence of text containing information irrelevant to the problem.
- <u>Silence:</u>
  - aspects important to proper solution of the problem are omitted.

Examples of Bad SRS Documents

- <u>Overspecification:</u>
  - Addressing "how to" aspects
  - For example, "Library member names should be stored in a sorted descending order"
  - Overspecification restricts the solution space for the designer.
- <u>Contradictions:</u>
  - Contradictions might arise
    - if the same thing described at several places in different ways.

Examples of Bad SRS Documents

- <u>Ambiguity:</u>
  - Literary expressions
  - Unquantifiable aspects, e.g. “good user interface”
- <u>Forward References:</u>
  - References to aspects of problem
    - defined only later on in the text.
- <u>Wishful Thinking:</u>
  - Descriptions of aspects
    - for which realistic solutions will be hard to find.

User Stories – example structure

As a [Type of USER],

[Function to Perform (some goal)]

so that [Business Value (some reason)]

Example: As a user, I can indicate folders not to backup so that my backup isn't filled up with things I don't need saved