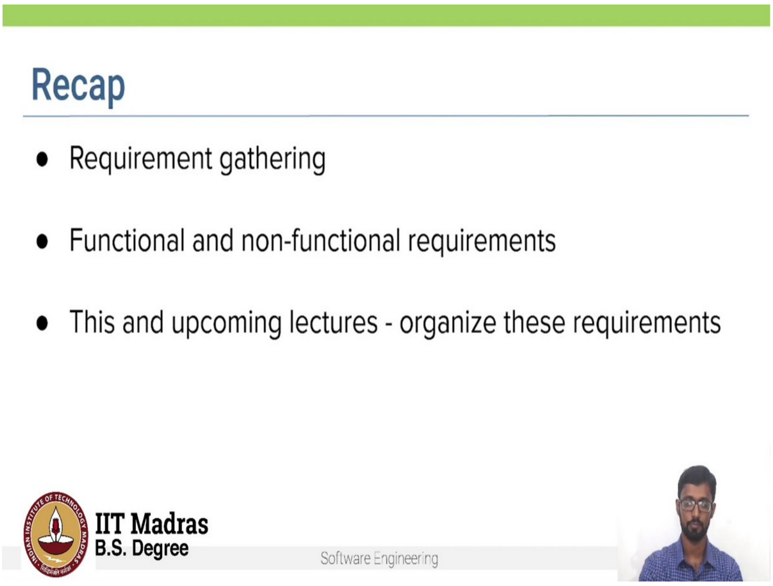
Software Engineering
Professor Sridhar Iyer
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Professor Prajish Prasad
Department of Computer Science
FLAME University
Software Requirement Specification

(Refer Slide Time: 0:28)

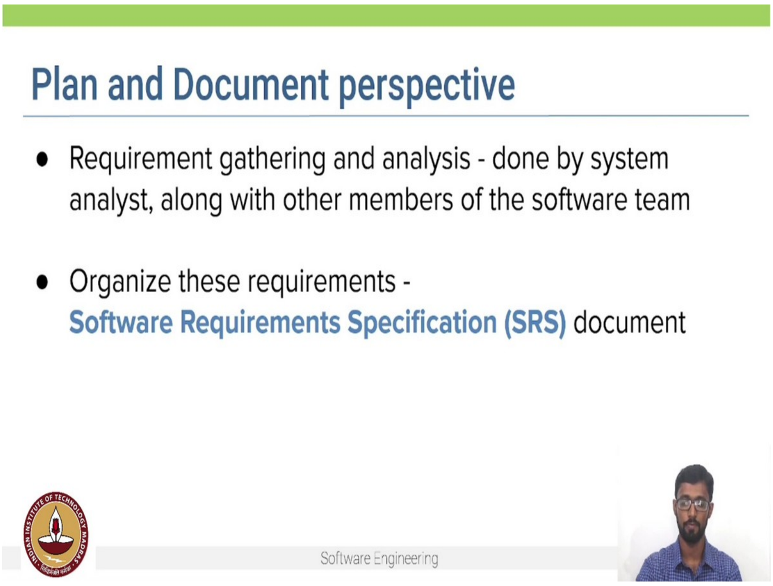


Recap

- Requirement gathering
- Functional and non-functional requirements
- This and upcoming lectures - organize these requirements

So, in the previous lecture, we looked at how we can gather and analyze requirements. And we categorize these requirements into two broad categories of functional and non-functional requirements. Now, another important point to consider is how we organize all these requirements that we have gathered and analyzed? So, in this and the upcoming lectures, we will look at means, by which we can effectively organize these requirements.

(Refer Slide Time: 1:05)



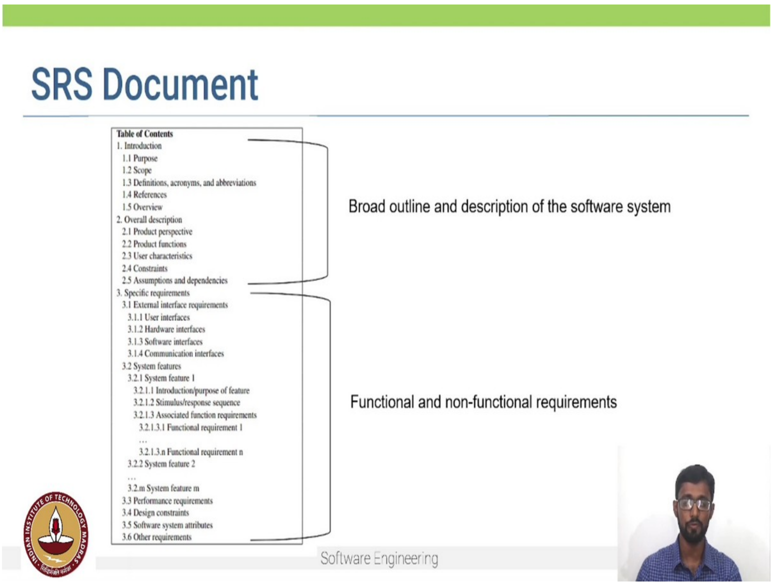
Plan and Document perspective

- Requirement gathering and analysis - done by system analyst, along with other members of the software team
- Organize these requirements - **Software Requirements Specification (SRS)** document

Software Engineering

In the first week, we looked at two prominent software process models, the plan and the document model and the agile model and recall that in the planning document model, significant time is spent in planning and documenting the entire software process. And in the Plan and Document perspective, requirements gathering and analysis is usually done by a person known as a system analyst in consultation with other members of the software team. And after the analyst has gathered all the required information regarding the software to be developed, he or she starts to organize these requirements in what is known as the software requirements specification or an SRS document. And this document contains all the user requirements in a structured form. So, now, let us look at the standard format of an SRS document.

(Refer Slide Time: 2:17)



SRS Document

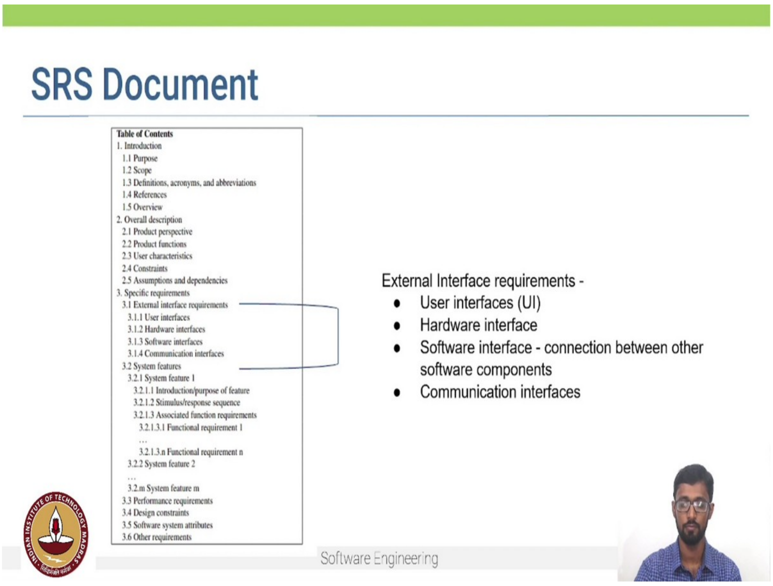
Table of Contents

1. Introduction
  1.1 Purpose
  1.2 Scope
  1.3 Definitions, acronyms, and abbreviations
  1.4 References
  1.5 Overview
2. Overall description
  2.1 Product perspective
  2.2 Product functions
  2.3 User characteristics
  2.4 Constraints
  2.5 Assumptions and dependencies
3. Specific requirements
  3.1 External interface requirements
    3.1.1 User interfaces
    3.1.2 Hardware interfaces
    3.1.3 Software interfaces
    3.1.4 Communication interfaces
  3.2 System features
    3.2.1 System feature 1
      3.2.1.1 Introduction/purpose of feature
      3.2.1.2 Stimulus/response sequence
      3.2.1.3 Associated function requirements
        3.2.1.3.1 Functional requirement 1
        …
        3.2.1.3.n Functional requirement n
    3.2.2 System feature 2
    …
    3.2.m System feature m
  3.3 Performance requirements
  3.4 Design constraints
  3.5 Software system attributes
  3.6 Other requirements

Broad outline and description of the software system

Functional and non-functional requirements

Software Engineering



SRS Document

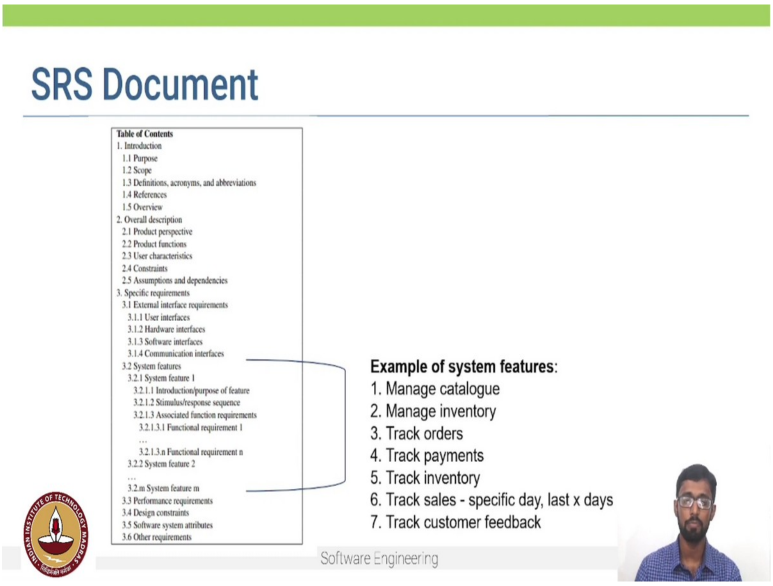
Table of Contents

1. Introduction
  1.1 Purpose
  1.2 Scope
  1.3 Definitions, acronyms, and abbreviations
  1.4 References
  1.5 Overview
2. Overall description
  2.1 Product perspective
  2.2 Product functions
  2.3 User characteristics
  2.4 Constraints
  2.5 Assumptions and dependencies
3. Specific requirements
  3.1 External interface requirements
    3.1.1 User interfaces
    3.1.2 Hardware interfaces
    3.1.3 Software interfaces
    3.1.4 Communication interfaces
  3.2 System features
    3.2.1 System feature 1
      3.2.1.1 Introduction/purpose of feature
      3.2.1.2 Stimulus/response sequence
      3.2.1.3 Associated function requirements
        3.2.1.3.1 Functional requirement 1
        …
        3.2.1.3.n Functional requirement n
    3.2.2 System feature 2
    …
    3.2.m System feature m
  3.3 Performance requirements
  3.4 Design constraints
  3.5 Software system attributes
  3.6 Other requirements

External Interface requirements -

- User interfaces (UI)
- Hardware interface
- Software interface - connection between other software components
- Communication interfaces

Software Engineering



SRS Document

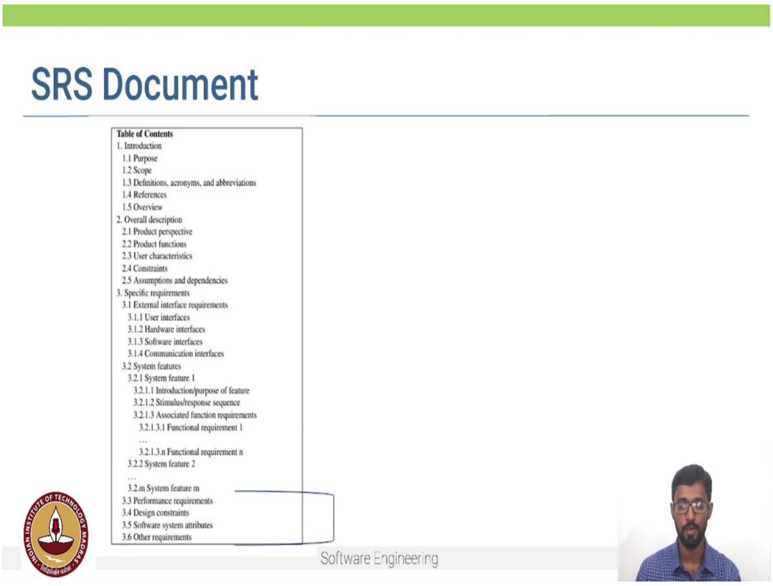
Table of Contents

1. Introduction
  1.1 Purpose
  1.2 Scope
  1.3 Definitions, acronyms, and abbreviations
  1.4 References
  1.5 Overview
2. Overall description
  2.1 Product perspective
  2.2 Product functions
  2.3 User characteristics
  2.4 Constraints
  2.5 Assumptions and dependencies
3. Specific requirements
  3.1 External interface requirements
    3.1.1 User interfaces
    3.1.2 Hardware interfaces
    3.1.3 Software interfaces
    3.1.4 Communication interfaces
  3.2 System features
    3.2.1 System feature 1
      3.2.1.1 Introduction/purpose of feature
      3.2.1.2 Stimulus/response sequence
      3.2.1.3 Associated function requirements
        3.2.1.3.1 Functional requirement 1
        …
        3.2.1.3.n Functional requirement n
    3.2.2 System feature 2
    …
    3.2.m System feature m
  3.3 Performance requirements
  3.4 Design constraints
  3.5 Software system attributes
  3.6 Other requirements

Example of system features:

1. Manage catalogue
2. Manage inventory
3. Track orders
4. Track payments
5. Track inventory
6. Track sales - specific day, last x days
7. Track customer feedback

Software Engineering



SRS Document

Table of Contents

1. Introduction
   - 1.1 Purpose
   - 1.2 Scope
   - 1.3 Definitions, acronyms, and abbreviations
   - 1.4 References
   - 1.5 Overview
2. Overall description
   - 2.1 Product perspective
   - 2.2 Product functions
   - 2.3 User characteristics
   - 2.4 Constraints
   - 2.5 Assumptions and dependencies
3. Specific requirements
   - 3.1 External interface requirements
     - 3.1.1 User interfaces
     - 3.1.2 Hardware interfaces
     - 3.1.3 Software interfaces
     - 3.1.4 Communication interfaces
   - 3.2 System features
     - 3.2.1 System feature 1
       - 3.2.1.1 Introduction/purpose of feature
       - 3.2.1.2 Stimulus/response sequence
       - 3.2.1.3 Associated function requirements
         - 3.2.1.3.1 Functional requirement 1
         - ...
         - 3.2.1.3.n Functional requirement n
     - 3.2.2 System feature 2
     - ...
     - 3.2.m System feature m
   - 3.3 Performance requirements
   - 3.4 Design constraints
   - 3.5 Software system attributes
   - 3.6 Other requirements

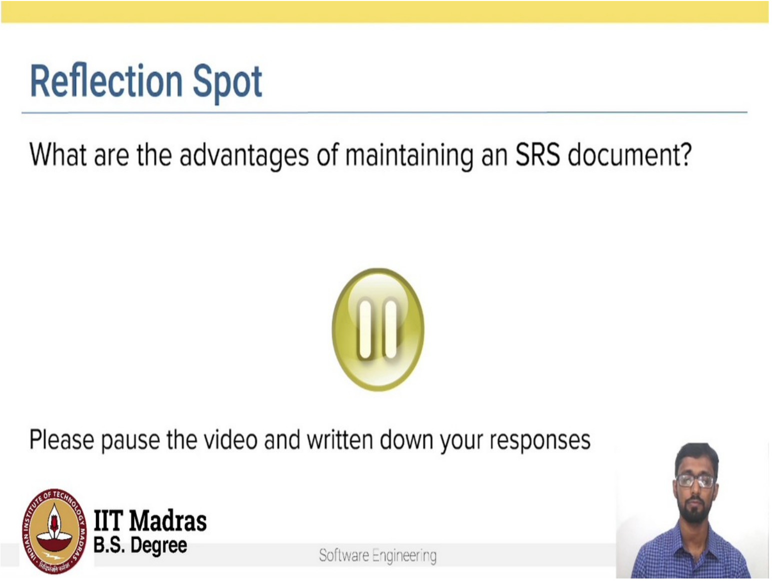
So, this is the standard structure of an SRS document. So, it has a table of contents which has several sections and subsections. Now, let us look at some of the important sections in the SRS document. Now, section 1 and 2, it describes the broad outline and description of the software system which we want to build. For example, it contains the purpose the scope, several definitions, acronyms and abbreviations which will be used in the software as well as the perspective, functions, constraints, assumptions, dependencies and so on. Now, section 3 of the SRS document contains the specific functional as well as non-functional requirements of the system.

Now, let us look at Section 3.1. Section 3.1 refers to the external interface requirements. And here we specify things like what are the user interfaces? For example, what are the sample screen images, GUI standard, screen layout etc. Then we have hardware interfaces that is the interface between the hardware and the software such as the description of supported device types, the nature of data and control interactions between the software and the hardware. For example, in an ATM system, we need to specify the interface between the ATM hardware, the ATM card and the ATM software. Next is descriptions about the software interface which is the connection between this software and other software components.

For example, the database what is the operating system, what are other tools and libraries which will be used and so on? So, for example in the Amazon seller portal example, which we described in the previous lectures. So, the seller portal has to interface with the Amazon buying portal database. So, this is an important software interface which we have to specify. Then we have communication interfaces, which are any communication which is required by the software example, sending an email or SMS notification about orders in the seller portal. So, that requires us to interface with an email server and such communication interfaces have to be specified. Now, when we look at Section 3.2, it outlines a broad high level function which is known as the system feature and the corresponding functional requirements for each of these system features.

So, for example, taking the example of the Amazon seller portal, we identified several key requirements such as managing the catalogue, inventory, tracking orders, payments, inventory and all of those things. So, all of this forms specific system features which can be listed in the system features section. And for each of these system features, we add additional functional requirements and finally, in Section 3.3 to 3.6 contains details of non-functional requirements like performance, security etc. So, one thing to remember is that this is a guideline of how an SRS should be organized and it is generally not very rigid based on the organization and the context you are in modifications to the structure and the content are possible.

(Refer Slide Time: 6:59)



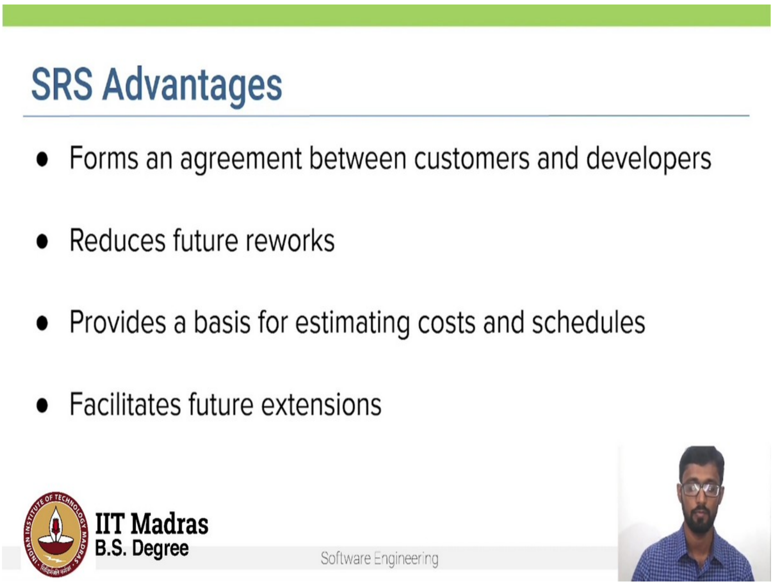
Reflection Spot

What are the advantages of maintaining an SRS document?

Please pause the video and written down your responses

Now, that we have seen what an SRS document contains, let us reflect on why maintaining such an SRS document is important. So, what do you think are some advantages of maintaining such an SRS document? Please pause this video and write down your responses before proceeding.

(Refer Slide Time: 7:28)



SRS Advantages

- Forms an agreement between customers and developers
- Reduces future reworks
- Provides a basis for estimating costs and schedules
- Facilitates future extensions

IIT Madras
B.S. Degree

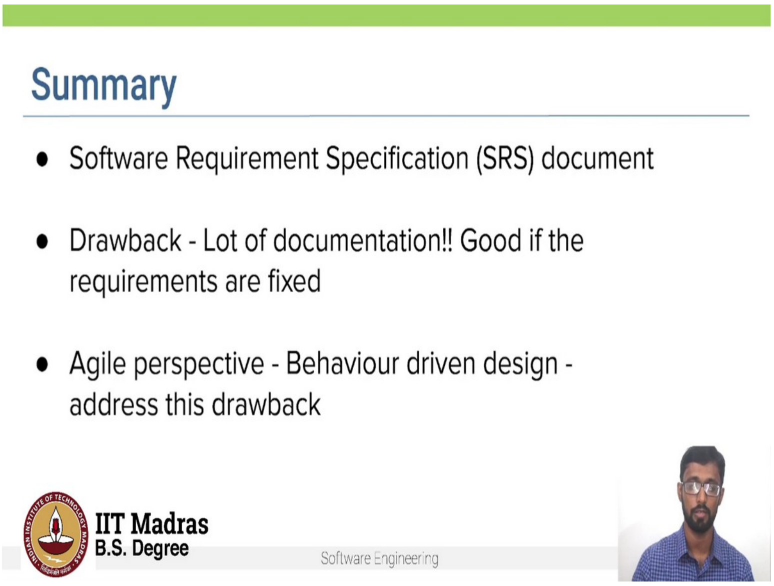
Software Engineering

So, let us look at some advantages of using an SRS. First, it helps to form an agreement between the customers and developers. So, after the SRS is prepared, the SRS document is also given to customers, the customer read it and after several iterations finally the SRS document is accepted. So, this helps the customers to form an expectation of what they will get from the developers and for the developers about what the customer expects from the software.

Second, it helps to reduce future reworks and SRS forces stakeholders to rigorously think about all the requirements before the start of the design and develop and this helps to reduce changes in later states. Third, an SRS provides a basis for estimating costs and schedules. So, in subsequent weeks, we will look at important tasks in managing a software project. So, estimating cost and schedule is an important software management task.

And, an SRS helps in estimating the size of the software which is nothing but a function of all the requirements of the system. And, based on this estimation, other estimation such as effort required and cost are estimated. And based on these estimates, the project manager can also create a schedule for the development of the software. And finally, an SRS can facilitate future extensions and it can serve as a basis for planning future enhancements of the software system.

(Refer Slide Time: 9:32)



Summary

- Software Requirement Specification (SRS) document
- Drawback - Lot of documentation!! Good if the requirements are fixed
- Agile perspective - Behaviour driven design - address this drawback

So, in this lecture, we looked at what a software requirements specification document is, we looked at different parts and benefits of the SRS. However, there is a drawback of SRS which is a lot of documentation is required. And, this is okay if the requirements are fixed. In the next lecture we will look at the Agile perspective, which is the Behavior driven design, which can help in addressing this drawback of SRS in the planning document perspective.