Chapter Five

Requirements Validation

Introduction

- Requirements validation is concerned with checking the requirements document for consistency, completeness and accuracy.
- It is the final stage of requirements engineering.
- **Aim**: to "validate" the requirements,
  - i.e., check the requirements to certify that they represent an acceptable description of the system which is to be implemented.
- The process involves:
  - ✓ system stakeholders,
  - ✓requirements engineers and
  - ✓system designers who analyze the requirements for problems, omissions and ambiguities.

Requirements Analysis Vs Requirements Validation

- These activities have much in common.
  - They involve:
    - ✓ Analyzing the requirements,
    - ✓Judging if they are an appropriate description of stakeholder needs and
    - ✓Checking for requirements problems.
- However, there are important differences between these activities so that it makes sense to consider them separately.

Requirements Analysis

- ✓ Is concerned with 'raw' requirements as elicited from system stakeholders.
- ✓ The requirements are usually incomplete and are expressed in an informal and unstructured way.
- ✓ **Focus** on making sure that the requirements meet stakeholder needs rather than the details of the requirements description.
- ✓ It should mostly be **concerned with** answering the question 'have we got the right requirements?'

Requirements Validation

- ✓Is concerned with checking a final draft of a requirements document, which includes all system requirements and where known incompleteness and inconsistency has been removed.
- ✓The document and the requirements should follow defined quality standards.
- ✓The validation process should be more concerned with the way in which the requirements are described.
- ✓It should mostly (but not exclusively) be concerned with answering the question 'have we got the requirements right?'

Cont'd…

- Some stakeholder-related problems are inevitably discovered during requirements validation and these must be corrected.

- **Examples of requirements problems discovered during validation might be:**
  - lack of conformance to quality standards.
  - poorly worded requirements which are ambiguous.
  - errors in models of the system or the problem to be solved.
  - requirements conflicts which were not detected during the analysis process.

- These problems should be solved before the requirements document is approved and used as a basis for system development.

What kind of problems are these and how to fix them?

On the next slide

Cont'd…

✓In some cases, the problems will simply be documentation problems and

- can be fixed by improving the requirements description.

✓In other cases, however, the problems result from flaws in the requirements elicitation, analysis and modelling.

- You may have to re-enter some of the earlier requirements engineering process activities.

➢The main- problem of requirements validation is that there is no existing document which can be a basis for the validation.

- A design or a program may be validated against the specification.
- However, there is no way to demonstrate that a requirements specification is 'correct' with respect to some other system representation.

Cont'd…

- **Specification validation**, therefore, really means ensuring that the requirements document represents a clear description of the system for design and implementation and is a final check that the requirements meet stakeholder needs.

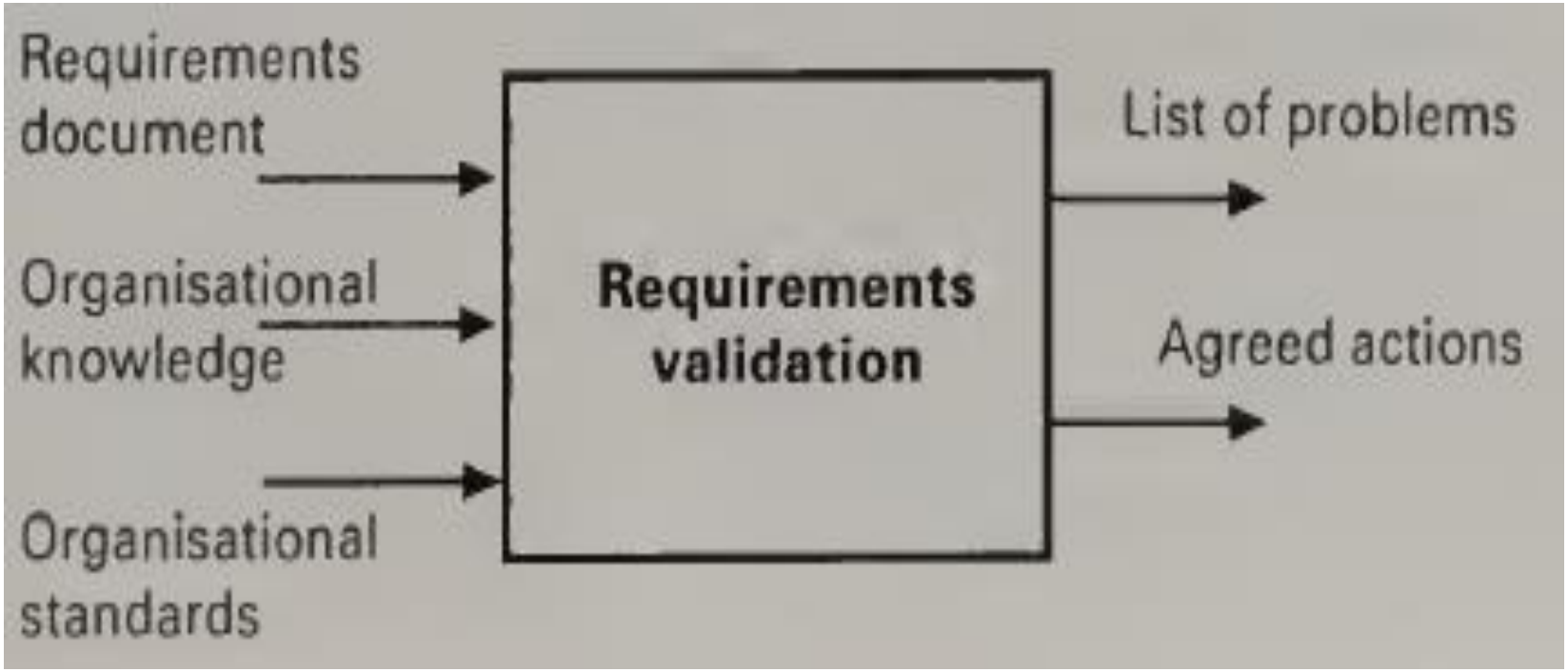

Figure 5.1 Requirements validation - inputs and outputs.

Cont'd…

- The inputs to the requirements validation process are as follows.

1. The requirements document

- The complete version of the requirement document which is formatted and organized according to organizational standards.

2. Organizational standards

- The requirements validation process should check conformance with company standards.
- standards are relevant for the requirements document should be an input to the validation process.

3. Organizational knowledge

- The people involved in requirements validation may know the organization, its particular terminology and its practices and the skills of the people involved in developing and using the system.
- This implicit knowledge is very important as the same requirements may be closely linked to organizational structure, standards and culture.

Cont'd…

- The process outputs are as follows.

1. A problem list

- This is a list of reported problems with the requirements document.
  - e.g., ambiguity, incompleteness, etc.
- In practice, however, it is often difficult to classify problems in this way.

2. Agreed actions

- This is a list of actions in response to requirements problems which have been agreed by those involved in the validation process.
- There is not necessarily a 1:1 correspondence between problems and actions.
  - ✓Some problems may result in several corrective actions; others may simply be noted but with no associated corrective action.

Cont'd…

- Requirements validation is a prolonged process as it involves people reading and thinking about a lengthy document.
- Meetings may have to be arranged and experiments carried out with prototype systems.
- It can take several weeks or sometimes months to validate the requirements for a complex system.
- This is particularly likely when stakeholders from different organizations are involved.

Techniques for Requirements Validation

- ✓Requirements reviews
  - Pre-review checking
  - Requirements checklist
- ✓Prototyping
- ✓User manual development
  - Rewriting the requirement

Requirements Reviews

- Requirements reviews are the most widely used **technique** of requirements validation.
- They involve a group of people who read and analyze the requirements, look for problems, meet to discuss the problems and agree on a set of actions to address the identified problems.

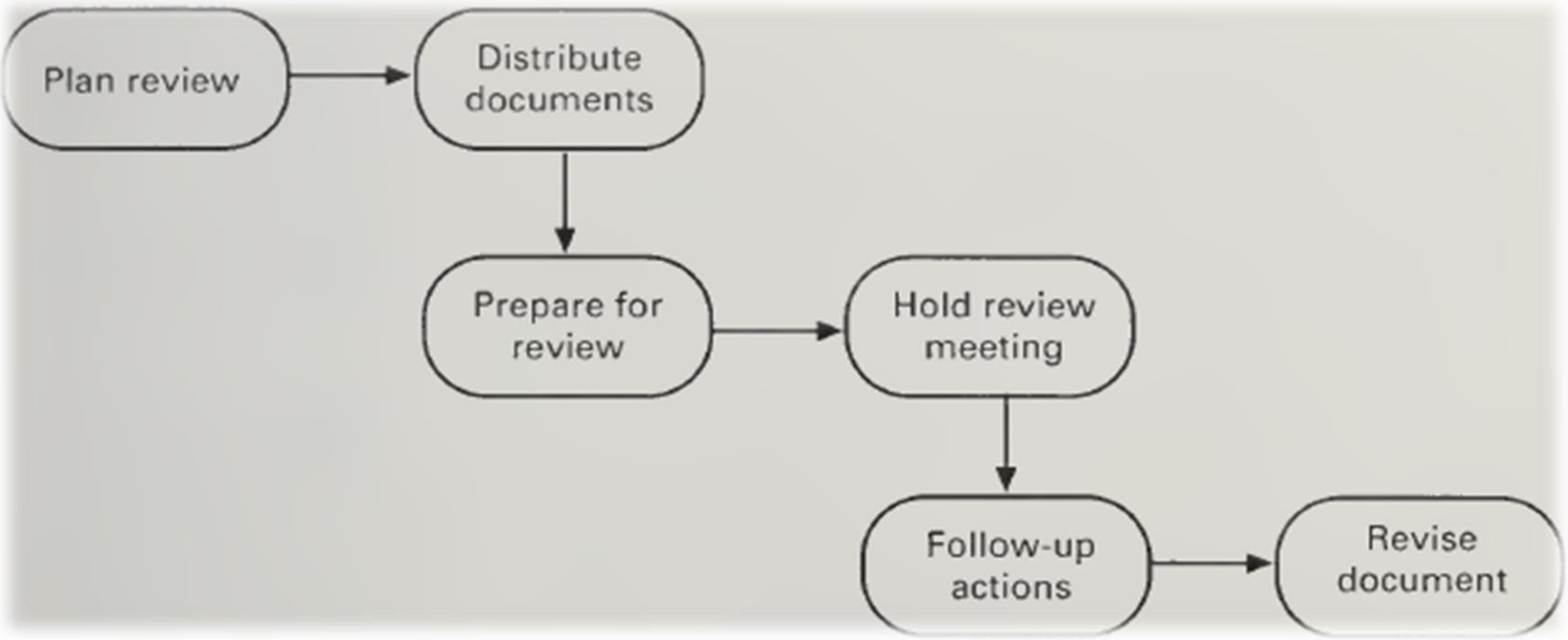

Figure 5.2 process model for the requirements review process

Cont'd…

- The principal stages in the review process are as follows.

1. Plan review

- The review team is selected and a time and place for the review meeting is chosen.

2. Distribute documents

- The requirements document and any other relevant documents are distributed to the review team members.

3. Prepare for review

- The individual reviewers read the requirements document to identify conflicts, omissions, inconsistencies, deviations from standards and any other problems.

Cont'd…

4. Hold review meeting

- The individual comments and problems are discussed and a set of actions to address the problems is agreed.

5. Follow-up actions

- The chair of the review checks that the agreed actions have been carried out.

6. Revise document

- The requirements document is revised to reflect the agreed actions.
- At this stage, it may be accepted or it may be re-reviewed.

Cont'd…

- The requirements review is a formal meeting.
- It should be chaired by someone who has not been involved in producing the requirements which are being validated.
- During the meeting, a requirements engineer presents each requirement in turn for comment by the group and identified problems are recorded for later discussion.
- One member of the group should be assigned the role of scribe to note the identified requirements problems.

Cont'd…

- In program inspections, it is normal practice to report errors to the program author for correction.

❑However, in requirements reviews, the review group make decisions on actions to be taken to correct the identified problems.

- Unlike programming errors, the problems usually require discussion and negotiation to agree on a possible solution.

Actions which might be decided for each problem are as follows.

1. Requirements clarification

- The requirement may be badly expressed or may have accidentally omitted information which has been collected during requirements elicitation.
- The author should improve the requirement by rewriting it.

Cont'd…

2. Missing information

- Some information is missing from the requirements document.

➢It is the responsibility of the requirements engineers who are revising the document to discover this information from system stakeholders or other requirements sources.

3. Requirements conflict

- There is a significant conflict between requirements.

➢The stakeholders involved must negotiate to resolve the conflict.

4. Unrealistic requirement

- The requirement does not appear to be implementable with the technology available or given other constraints on the system.

➢Stakeholders must be consulted to decide whether the requirement should be deleted or modified to make it more realistic.

Pre-review checking

- Reviews involve a lot of time and expense so it makes sense to minimize the work of reviewers.
- Before distributing the requirements document for general review, one person should carry out quick standards check to ensure that the document structure and the defined requirements are consistent with whatever standards have been defined.
- They should also process the document with whatever automatic checkers are available to remove spelling mistakes, cross-reference errors, etc.
- This is a quick and cheap way to check standards conformance as only one person is needed to carry out this type of check.

➢Furthermore, checking against a standard can quickly reveal problems with the requirements.

Cont'd…

- An analyst or an engineer who is familiar with the requirements standards but who has not been involved in the system requirements specification should be responsible for this initial document check.
- It is not necessary for the document checker to understand the requirements in detail.
  - ✓The checker should compare the structure of the requirement document to the defined standard and should highlight missing or incomplete sections.
- Individual requirements should also be checked for standards conformance.

Cont'd…

- This initial check should also check that:
  - all pages in the document are numbered,
  - all diagrams and figures are labelled,
  - there are no requirements which are unfinished or labelled 'To be completed and that all required appendices in the document have been included.

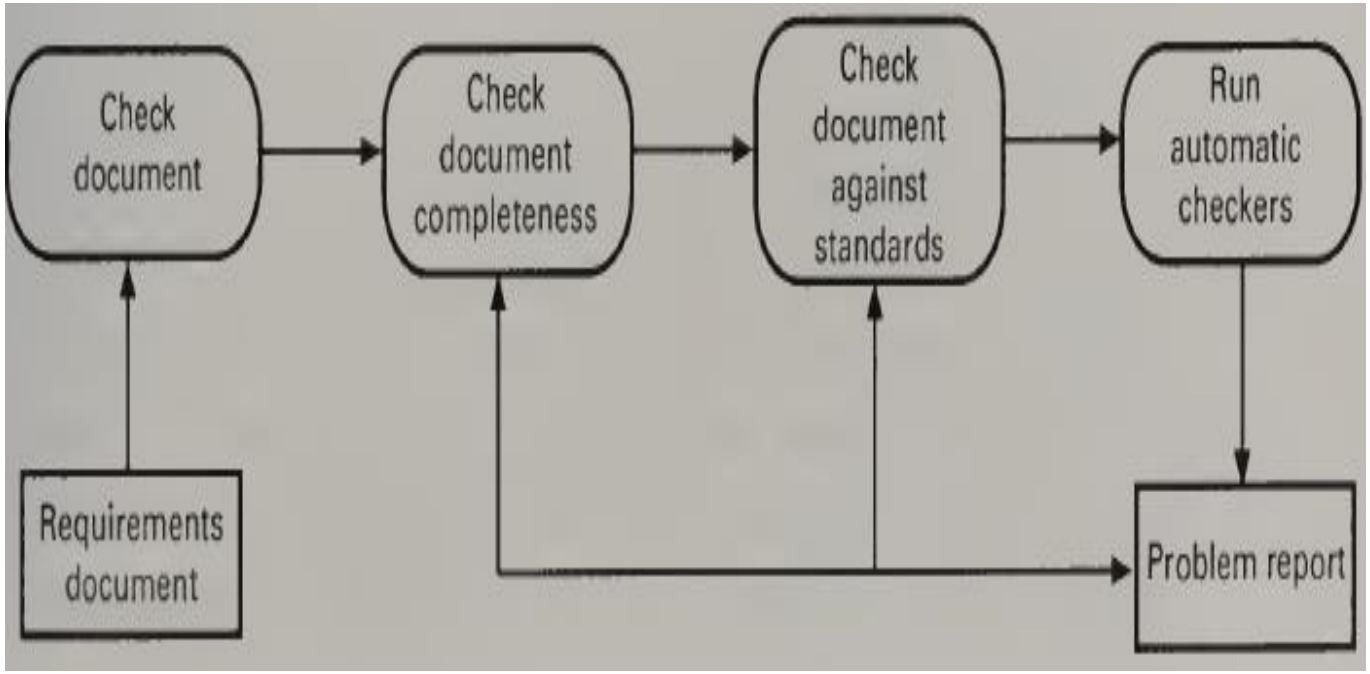

Figure 5.3 Pre-review checking

Cont'd…

- After the initial checking process is complete, there are two possible options if deviations from the standard are found.

I. **Return the document** to the requirements engineering team to correct deviations from the standards.

- This option should be chosen if there is enough time to allow for a re-issue of the document.

II. **Note the deviations from the standards** and distribute this to document reviewers.

- This saves the time and cost of creating a new version of the requirements document.
- However, the deviations from standards may make it harder for reviewers to understand the requirements.

Review team membership

- Selecting the right membership for the requirements review team is important.
- Ideally, the requirements document should be reviewed by a **multi-disciplinary team** drawn from people with different backgrounds.
- If possible, the team should include:
  - a system end-user or end-user representative,
  - a customer representative,
  - one or more domain experts,
  - engineers who will be responsible for system design and implementation and
  - requirements engineers.

Cont'd…

- The **advantages** of involving stakeholders from different disciplines are as follows.

I. People from different backgrounds bring different skills, domain knowledge and experience to the review.
   - It is therefore more probable that requirements problems will be discovered.

II. If system stakeholders from different backgrounds are involved in the review process,
   - they feel involved in the requirements engineering process and develop an understanding of the needs of other stakeholders.
   - They are therefore more likely to understand why changes to requirements which they have proposed are necessary.

Review checklists

➢Requirements review checklists should be more general.

- They should not be concerned with individual requirements but with the quality properties of the requirements document as a whole and with the relationships between requirements.

- The following table 5.1 lists some general quality properties of requirements and requirements documents which may be used to derive requirements checklists.

Cont'd…

Table 5.1 Requirements quality attributes

Review check	Description
Understandability	• Can readers of the document understand what the requirements mean? • This is probably the most important attribute of a requirements document - if it can't be understood, the requirements can't be validated.
Redundancy	• Is information unnecessarily repeated in the requirements document? • Sometimes, of course, repeating information adds to understandability. • There must be a balance struck between removing all redundancy and making the document harder to understand.

Cont'd…

Review check	Description
Completeness	Does the checker know of any missing requirements or is there any information missing from individual requirement descriptions?
Ambiguity	• Are the requirements expressed using terms which are clearly defined? • Could readers from different backgrounds make different interpretations of the requirements?
Consistency	Do the descriptions of different requirements include contradictions? Are there contradictions between individual requirements and overall system requirements?

Cont'd…

Review check	Description
Organization	• Is the document structured in a sensible way? • Are the descriptions of requirements organized so that related requirements are grouped? • Would an alternative structure be easier to understand?
Conformance to standards	• Does the requirements document and individual requirements conform to defined standards? • If there is a departure from the standards, is it justified?
Traceability	• Are requirements unambiguously identified, include links to related requirements and to the reasons why these requirements have been included? • Is there a clear link between software requirements and more general systems engineering requirements?

Cont'd…

- Organizations should derive their own set of requirements review questions based on their experience and their local standards.
- These may include more specialized questions which are tailored to the types of system which they develop.

❑The following Table 5.2 gives some examples of possible questions which might be associated with the quality attributes described in Figure 5.1.

- Checklists should be expressed in a fairly general way and should be understandable by people such as end-users who are not system experts.

❑As a general rule, checklists should not be too long.

- If checklists have more than 10 items, checkers cannot remember all items.

Cont'd…

- Checklists can be distributed and used to remind people what to look for when reading the requirements document.
- Alternatively, they can be used more systematically where, for each requirement, an indication is given that the checklist item has been considered.
- This may be done on paper or you can manage this type of checklist completion by using a simple database or a spreadsheet.

Prototyping

- People find it very difficult to visualize how a written statement of requirements will translate into an executable software system.

❑If you develop a prototype system to demonstrate requirements,

- stakeholders and other end-users find it easier to discover problems and suggest how the requirements may be improved.

➢If a prototype has been developed for requirements elicitation,

✓it makes sense to use this later in the requirements engineering process for validation.

➢However, if a prototype system is not already available,

✓it is **not likely to be cost-effective** to develop a prototype system only for requirements validation.

Cont'd…

Elicitation Prototypes Vs Validation Prototypes

- Prototypes for validation must be more complete than elicitation prototypes.

- **Elicitation prototypes:**
  - Can simply include those requirements which are particularly difficult to describe or understand.
  - They may leave out well-understood requirements.
  - Usually have missing functionality and may not include changes agreed during the requirements analysis process.

Cont'd….

- It is therefore usually necessary to continue development of the prototype during the requirements validation process as shown in Figure 5.3.
- This figure also shows the other process activities which should go on in parallel with prototype development.

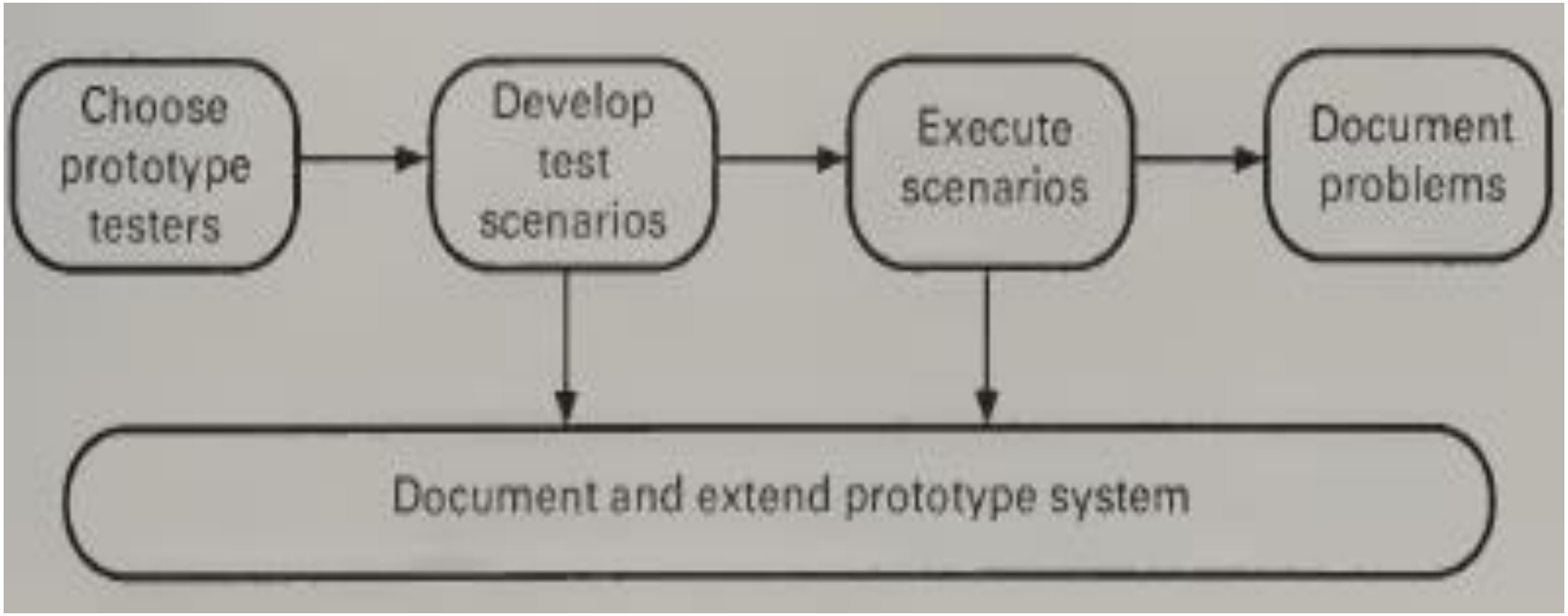

Figure 5.3 Prototyping for requirements validation.

Cont'd…

1. Choose prototype testers

- Choosing the right people to act as prototype testers is very important.
- The best testers are users who are fairly experienced and who are openminded about the use of new systems.
- If possible, end-users who do different jobs should be involved so that different areas of system functionality will be covered.

2. Develop test scenarios

- This means that careful planning is required to draw up a set of test scenarios which provide broad coverage of the requirements.

Cont'd…

3. Execute scenarios

- Requirements engineers should spend some time observing how end-users make use of the system.
- This can reveal particular problem areas and the coping strategies which users develop for dealing with system features which they find useful but awkward to use.

4. Document problems

- To be effective, problems which users encounter must be carefully documented.
- It's usually best to define some kind of electronic or paper problem report form which users fill in when they encounter a problem or want to suggest some change to the system.

User manual development

- Rewriting the requirements in a different way is a very effective validation technique.
- To rewrite the requirements you must understand the requirements and the relationships between them.
- Developing this understanding reveals conflicts, omissions and inconsistencies.
- One possible alternative form of the requirements which may be produced using the validation process is a draft of the end-user documentation for the system.

Cont'd…

❑**The user manuals should include the following information.**

1. A description of the functionality which has been implemented and how to access that functionality through the user interface.
2. It should be made clear which parts of the system are not implemented.
   - Manual writers should not leave users to find this out when problems arise.
3. A description of how to recover from difficulties.
   - Users are working with an experimental system and it is inevitable that things will go wrong.
   - Information about how they can get back to a known system state and restart use of the system should be included.

Cont'd…

4. If users have to install the prototype themselves, installation instructions should be provided.
   - If there is insufficient time or resources to build a prototype system for validation, some of the benefits of this approach can be gained by writing a draft user manual.

- The manual should be written by systematically translating the functionality described in the requirements into descriptions, written in end-user terms, of how to use them.
- If it is difficult to explain a function to end-users or to explain how to express system functionality, this suggests that there may be a requirements problem.

Cont'd…

- In some cases, the prototype user manual can be a basis for the final user documentation.
- However, this may not always be possible.
- The system development may be cancelled or the system may change to such an extent that a completely new user manual must to be written.
- The cost of writing the draft manual should be included in the requirements validation costs.

Model Validation

- Part of the requirements specification for a system may consist of one or more system models.

➢The validation of these models has three **objectives**.

1. To demonstrate that each individual model is **self-consistent**.
   - That is, the model should include all information which is necessary and there should be no conflicts between the different parts of the model.
2. If there are several models of the systems, to demonstrate that these are internally and externally consistent.
   - That is, entities which are referenced in more than one model should be defined to be the same in each model,
   - comparable items should have the same names and the model interfaces should be consistent.

Cont'd…

3. To demonstrate that the models accurately reflect the real requirements of system stakeholders.
   - This is the most difficult model validation task.
   - It involves making convincing arguments that the system defined in the model is the system which stakeholders really need.

- If models are expressed using notations which are supported by CASE tools, some of this checking can be automated.
- CASE tools can check individual models for consistency and can carry out some cross-model checks.

➢ However, some consistency checks involving several different models cannot be automated but may be considered in model reviews.

- An example of the type of check which cannot be automated is where an entity defined in one model is referenced in another but the reference is to the wrong entity.

Cont'd…

- Checking that the model reflects the real needs of stakeholders can be difficult.
  - You need to get the stakeholders themselves involved in the model validation process.
- Non-technical people do not intuitively understand data-flow diagrams, event diagrams or object models.
  - They prefer working with natural language descriptions.
- One way to get round this problem is to paraphrase or rewrite the model in natural language.
- Advantage of making this transformation:
  - stakeholders such as end-users, organizational management and regulators can understand and comment on the detailed system specification.
  - It is an effective way to detect errors, inconsistencies and incompleteness in the model.

Cont'd…

- For example, in a data-flow diagram, you might use a template with the following fields to describe each transformation:

1. **transformation name**
2. **transformation inputs and input sources**
   - Gives the name of each input to the transformation and lists where that input comes from.
3. **transformation function**

- Explain what the transformation is supposed to do to convert inputs to outputs.

4. **transformation outputs**

- Gives the name of each output and lists where the output goes to.

5. **control any exception or control information which is included in the model.**

Cont'd…

- Only a few people in an organization can understand formal models so a translation to natural language is essential to validate that they reflect the real requirements of the system.

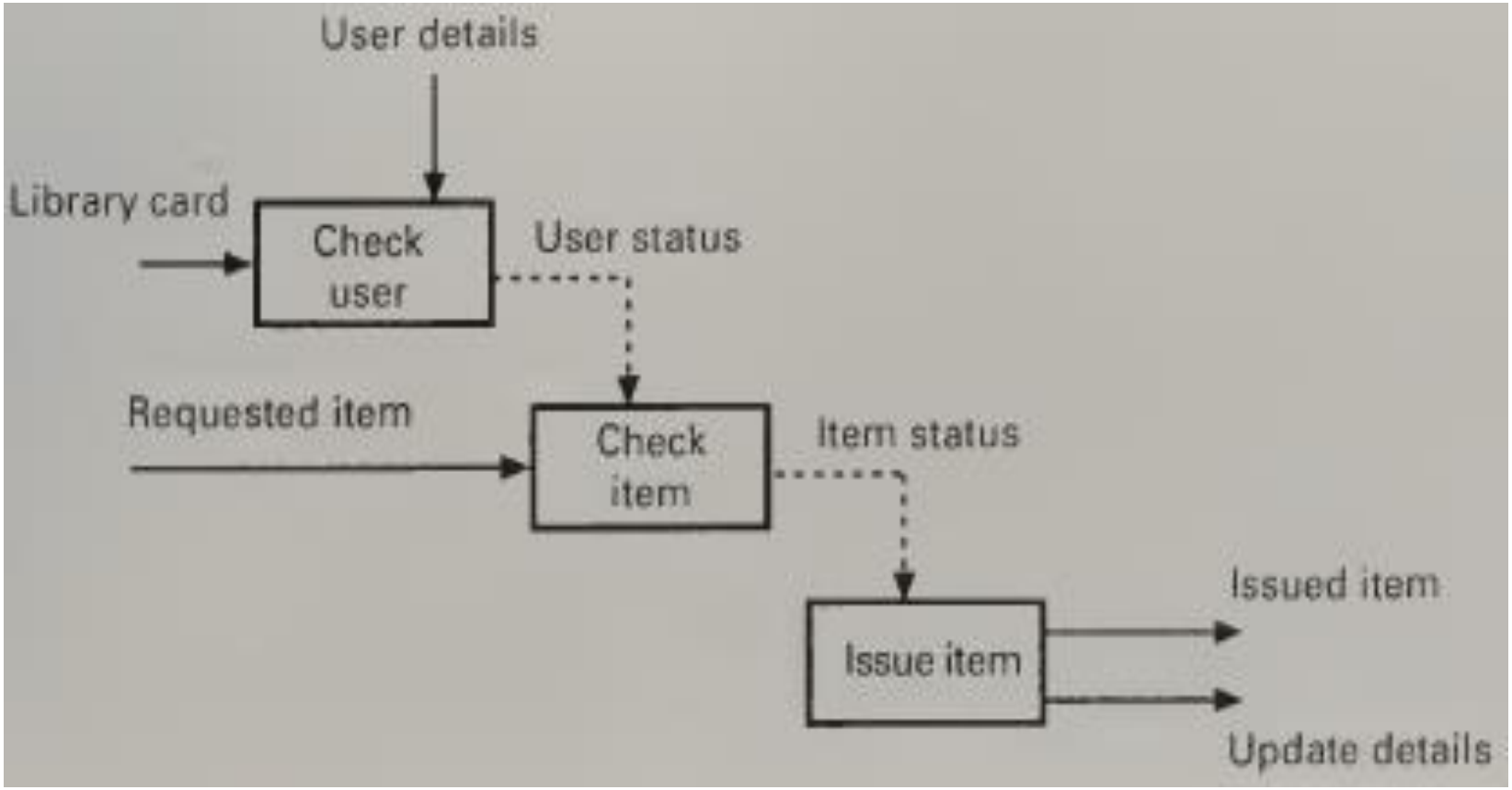

Figure 5.4 Data-flow diagram for the issue function.

Cont'd…

Check user	
Inputs and sources	User's library card from end-user
Transformation function	Checks that the user is a valid library user
Transformation outputs	The user's status
Control information	User details from the database
Check item	
Inputs and sources	The requested item from the end-user
Transformation function	Checks if an item is available for issue
Transformation outputs	The item's status
Control information	The user's status from Check user
Issue item	
Inputs and sources	*None*
Transformation function	Issues an item to the library user. Items are stamped with a return date.
Transformation outputs	The item issued to the end user
	Database update details
Control information	Item status – items only issued if available

Figure 5.5 Paraphrased description of '**Issue**' data-flow diagram.

Requirements Testing

- A desirable attribute of requirement is that it should be testable.
  - That is, it should be possible to define one or more tests that may be carried out in the finished system which will clearly demonstrate that the requirement has been met.
- The execution of the implemented requirement may be simulated using these tests.
- While the actual tests of a system are carried out after implementation, proposing possible tests is an effective way of revealing requirements problems such as incompleteness and ambiguity.

❑If there are difficulties in deriving test cases for a requirement, this implies that there is some kind of requirement problem.

- There may be missing information in the requirement or the requirement description may not make clear exactly what is required.

Cont'd…

- Each functional requirement in the requirements document should be analyzed and a test should be defined which can objectively check if the system satisfies the requirement.
- The objective of proposing test cases for requirements is to validate the requirement not the system.

❑To define test cases for a requirement, you can ask the following questions about that requirement.

1. What usage scenario might be used to check the requirement?
    - This should define the context in which the test should be applied.

Cont'd…

2. Does the requirement, on its own, include enough information to allow a test to be defined?

    2. If not, what other requirements must be examined to find this information?

    - If you need to look at other requirements, you should record these.
    - There may be dependencies between requirements which are important for traceability.

3. Is it possible to check the requirement using a single test or are multiple test cases required?

    - If you need several tests, it may mean that there is more than one requirement embedded in a single requirement description.

4. Could the requirement be re-stated so that the required test cases are fairly obvious?

Cont'd…

- A test record form should be designed and filled in for each requirement which is 'tested'.

- This should include at least the following information.

1. The requirement's identifier

- There should be at least one for each requirement.

2. Related requirements

- These should be referenced as the test may also be relevant to these requirements.

3. Test description

- A brief description of the test which could be applied and why this is an objective requirements test.

- This should include system inputs and the corresponding outputs which are expected.

Cont'd…

4. Requirements problems

- A description of requirements problems which made test definition difficult or impossible.

5. Comments and recommendations

- These are advice on how to solve requirements problems which have been discovered.

Cont'd…

- When designing requirements tests, the test designer need not be concerned with practicalities such as testing costs, avoiding redundant tests, detailed test data definition, etc.
- It isn't necessary to propose real tests which will be applied to the final system.
- The tester can make any assumptions that he wishes about the way in which the system satisfies other requirements and the ways in which the test may actually be carried out.

➢However, wherever possible, it makes sense to try and design tests which can be used as system tests.

- These are applied after implementation as part of the system verification and validation process.
- By reusing requirements tests in this way, the overall costs of test planning may be reduced.

Cont'd…

There are problems, however, in designing tests for some types of requirements.

1. System requirements

- These are requirements which apply to the system as a whole.
- In general, these are the most difficult requirements to validate irrespective of the method used as they may be influenced by any of the functional requirements.
- Tests, which are not executed, cannot test for non-functional system-wide characteristics such as usability.

Cont'd…

2. Exclusive requirements

- These are requirements which exclude specific behavior.
- For example, a requirement may state that system failures must never corrupt the system database.
- It is not possible to test such a requirement exhaustively.

3. Some non-functional requirements

- Some non-functional requirements, such as reliability requirements, can only be tested with a large test set.
- Designing this test set does not help with requirements validation.

Thank You !
?