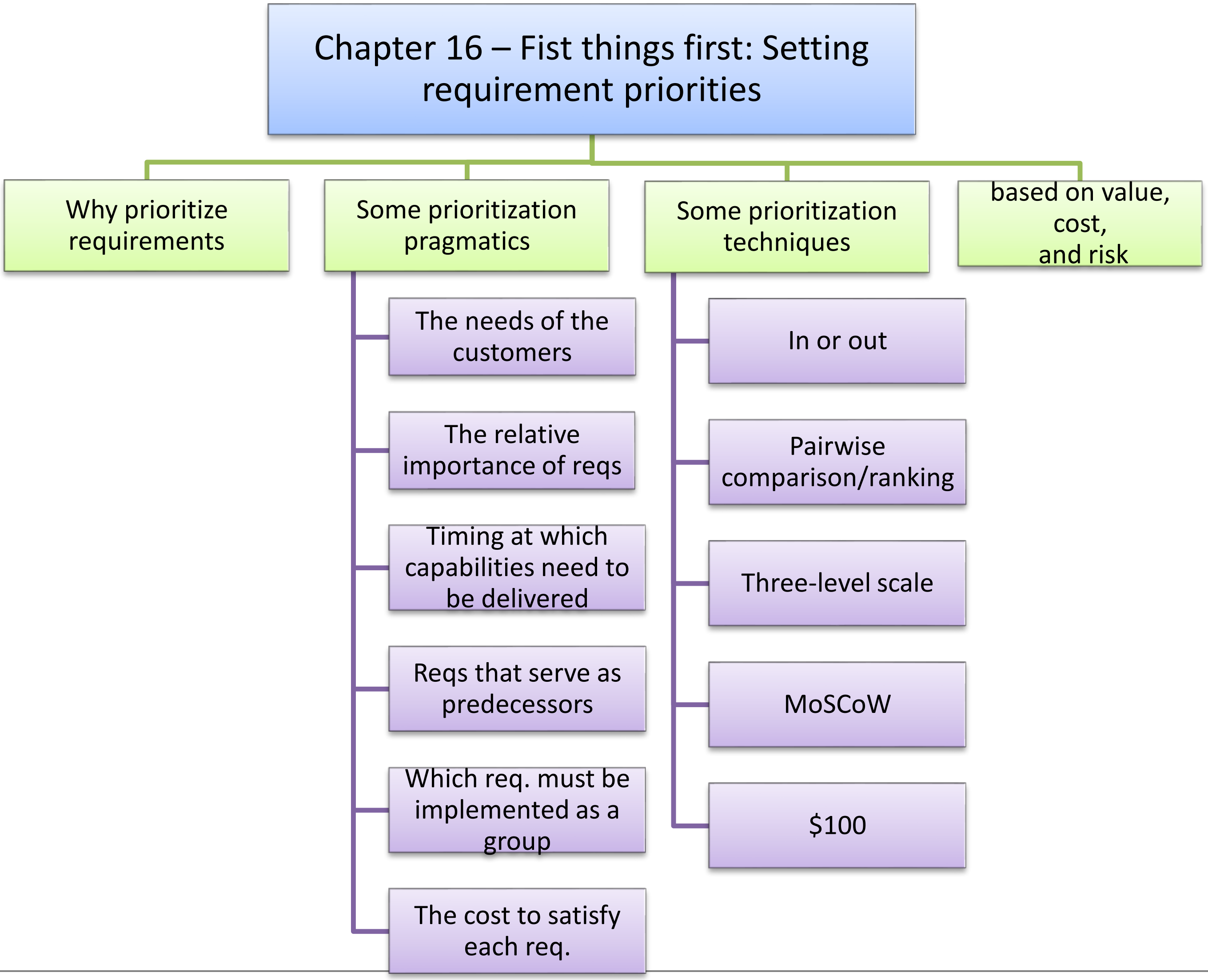
ĐẠI HỌC FPT CẦN THƠ

CHAPTER 17

Validating the requirements

Review chapter 16


Chapter 16 – Fist things first: Setting requirement priorities
Why prioritize requirements
Some prioritization pragmatics
Some prioritization techniques
based on value, cost, and risk
The needs of the customers
The relative importance of reqs
Timing at which capabilities need to be delivered
Reqs that serve as predecessors
Which req. must be implemented as a group
The cost to satisfy each req.
In or out
Pairwise comparison/ranking
Three-level scale
MoSCoW
$100

Objectives

- Help student understand detecting errors in the requirements specifications will save time and money.
- Student could enhance starting test planning and test-case development in parallel with requirements development, they will detect many errors shortly after they’re introduced.

Contents

Why validate the requirements?

- One study found that it took an average of 30 minutes to fix a defect discovered during the requirements phase. In contrast, 5 to 17 hours were needed to correct a defect identified during system testing.
- If you start your test planning and test-case development in parallel with requirements development, you'll detect many errors shortly after they're introduced.

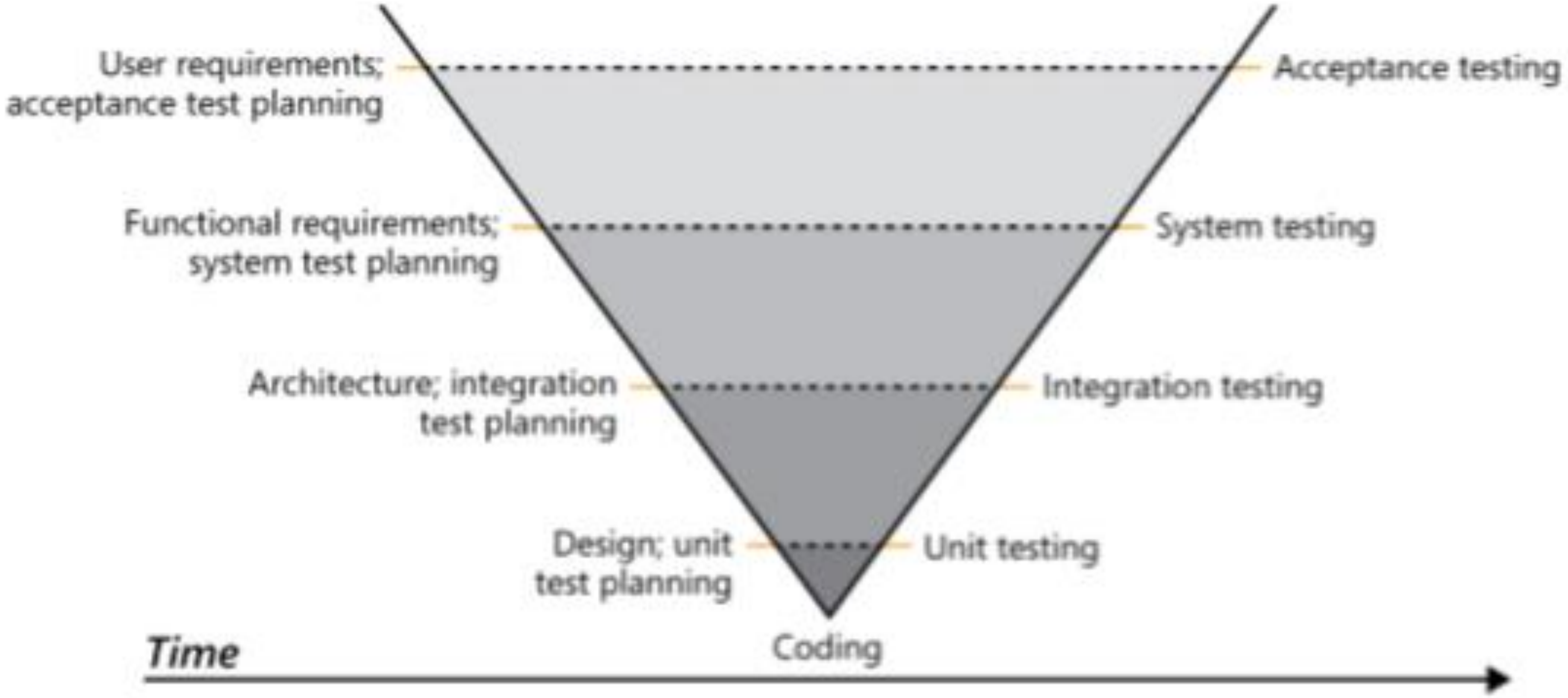

User requirements; acceptance test planning
Acceptance testing
Functional requirements; system test planning
System testing
Architecture; integration test planning
Integration testing
Design; unit test planning
Unit testing
Coding
Time

FIGURE 17-1 The V model of software development incorporates early test planning and test design.

Validation and verification

Verification (kiểm tra) determines whether the product of some development activity meets its requirements (doing the thing right).

Validation (sự thông qua) assesses whether a product satisfies customer needs (doing the right thing).

Requirements validation activities:

- The software requirements accurately describe the intended system capabilities and properties that will satisfy the various stakeholders' needs.
- The software requirements are correctly derived from the business requirements, system requirements, business rules, and other sources.
- The requirements are complete, feasible, and verifiable.
- All requirements are necessary, and the entire set is sufficient to meet the business objectives.
- All requirements representations are consistent with each other.
- The requirements provide an adequate basis to proceed with design and construction

Reviewing requirements

- Reviewing requirements is a powerful technique for identifying ambiguous or unverifiable requirements, requirements that aren’t defined clearly enough for design to begin, and other problems
- Informal review approaches:
  - A peer *deskcheck*, in which you ask one colleague to look over your work product.
  - A *passaround* *(vượt qua xung quanh)*, in which you invite several colleagues to examine a deliverable concurrently.
  - A *walkthrough* *(diện tập)*, during which the author describes a deliverable and solicits comments on it.
- The best-established type of formal peer review is called an *inspection* *(thanh tra)*. Inspection of requirements documents is one of the highest-leverage software quality techniques available.
- There are many types used to review requirements as follows:
  - The inspection process
  - Defect checklist
  - Requirements review tips
  - Requirements review challenges

The inspection process

Inspection is a well-defined multistage process. It involves a small team of participants who carefully examine a work product for defects and improvement opportunities. Inspections serve as a quality gate through which project deliverables must pass before they are baselined.

There are several forms of inspection, but any one of them is a powerful quality technique.

- Participants: Ensure that you have all of the necessary people in an inspection meeting before proceeding. The participants in an inspection should represent four perspectives (quan điểm).
  - The author of the work product and perhaps peers of the author: The BA who wrote the requirements document provides this perspective.
  - People who are the sources of information that fed into the item being inspected: User representatives or author of a predecessor specification.
  - People who will do work based on the item being inspected: Developer, tester, PM, user documentation writer.
  - People who are responsible for interfacing systems that will be affected by the item being inspected: These inspectors will look for problems with the external interface requirements

Try to limit the team to 7 or fewer inspectors and not include the author' manager.

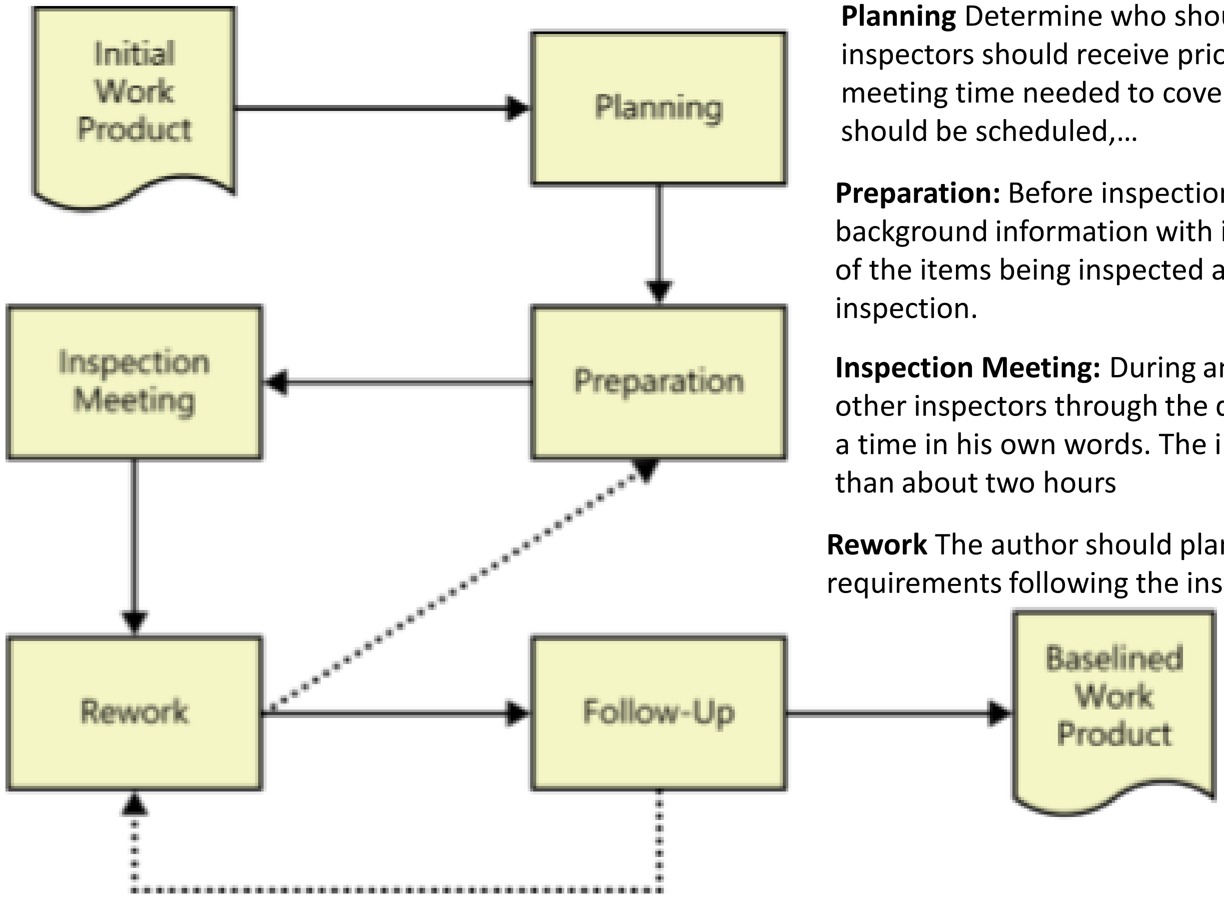
The inspection process

- Inspection roles: Some of the inspection team members perform the following specific roles during the inspection:
  - Author: The author created or maintains the work product being inspected. The author of a requirements document is usually the business analyst who elicited customer needs and wrote the requirements. However, the author takes a more passive role during an inspection, so he/she can listen to the comments from other inspectors, respond to—but not debate—their questions, and think. This way the author can often spot (phát hiện) errors that other inspectors don’t see.
  - Moderator: The moderator plans the inspection with the author, coordinates the activities, and facilitates the inspection meeting.
  - Reader: One inspector is assigned the role of reader. During the inspection meeting, the reader paraphrases the requirements and model elements being examined one at a time. The other participants then point out potential defects and issues that they see.
  - Recorder: The recorder uses standard forms to document the issues raised and the defects found during the meeting. The recorder should review aloud (nói to) or visually share (by projecting or sharing in a web conference) what he wrote to confirm its accuracy

The inspection process

- Entry criteria: You're ready to inspect a requirements document when it satisfies specific prerequisites. These *entry criteria* set some clear expectations for authors to follow while preparing for an inspection. The moderator uses the entry criteria as a checklist before deciding to proceed with the inspection. Following are some suggested inspection entry criteria for requirements documents:
  - The document conforms to the standard template and doesn't have obvious spelling, grammatical, or formatting issues.
  - Line numbers or other unique identifiers are printed on the document to facilitate referring to specific locations.
  - All open issues are marked as TBD (to be determined) or accessible in an issue-tracking tool.
  - The moderator didn't find more than three major defects in a ten-minute examination of a representative sample of the document.
- Inspection stages: An inspection is a multistep process showed in the next slide.

The inspection process: Inspection stages



Planning Determine who should participate, what materials the inspectors should receive prior to the inspection meeting, the total meeting time needed to cover the material, and when the inspection should be scheduled,…

Preparation: Before inspection meeting, the author should share background information with inspectors so they understand the context of the items being inspected and know the author's objectives for the inspection.

Inspection Meeting: During an inspection meeting, the reader leads the other inspectors through the document, describing one requirement at a time in his own words. The inspection meeting shouldn't last more than about two hours

Rework The author should plan to spend some time reworking the requirements following the inspection meeting.

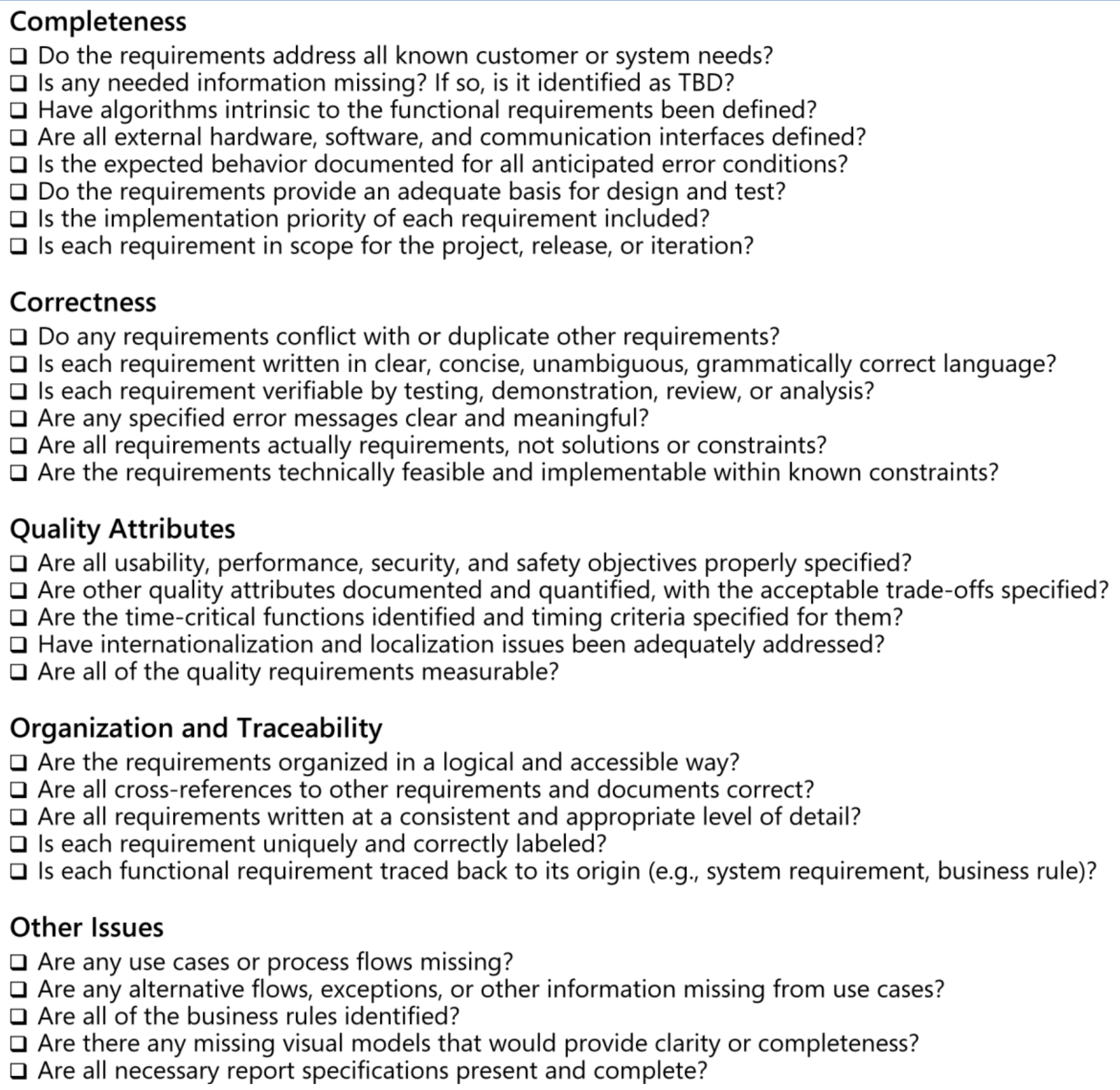
Follow-up In this final inspection step, the moderator works with the author to ensure that all open issues were resolved and that errors were corrected properly.

FIGURE 17-2 Inspection is a multistep process. The dotted lines indicate that portions of the inspection process might be repeated if reinspection is necessary because of extensive rework.

Defect checklist

- Completeness
- Correctness
- Quality attributes
- Organization and Traceability
- Other issues

Read page 339 for more information



Completeness

- ❑ Do the requirements address all known customer or system needs?
- ❑ Is any needed information missing? If so, is it identified as TBD?
- ❑ Have algorithms intrinsic to the functional requirements been defined?
- ❑ Are all external hardware, software, and communication interfaces defined?
- ❑ Is the expected behavior documented for all anticipated error conditions?
- ❑ Do the requirements provide an adequate basis for design and test?
- ❑ Is the implementation priority of each requirement included?
- ❑ Is each requirement in scope for the project, release, or iteration?

Correctness

- ❑ Do any requirements conflict with or duplicate other requirements?
- ❑ Is each requirement written in clear, concise, unambiguous, grammatically correct language?
- ❑ Is each requirement verifiable by testing, demonstration, review, or analysis?
- ❑ Are any specified error messages clear and meaningful?
- ❑ Are all requirements actually requirements, not solutions or constraints?
- ❑ Are the requirements technically feasible and implementable within known constraints?

Quality Attributes

- ❑ Are all usability, performance, security, and safety objectives properly specified?
- ❑ Are other quality attributes documented and quantified, with the acceptable trade-offs specified?
- ❑ Are the time-critical functions identified and timing criteria specified for them?
- ❑ Have internationalization and localization issues been adequately addressed?
- ❑ Are all of the quality requirements measurable?

Organization and Traceability

- ❑ Are the requirements organized in a logical and accessible way?
- ❑ Are all cross-references to other requirements and documents correct?
- ❑ Are all requirements written at a consistent and appropriate level of detail?
- ❑ Is each requirement uniquely and correctly labeled?
- ❑ Is each functional requirement traced back to its origin (e.g., system requirement, business rule)?

Other Issues

- ❑ Are any use cases or process flows missing?
- ❑ Are any alternative flows, exceptions, or other information missing from use cases?
- ❑ Are all of the business rules identified?
- ❑ Are there any missing visual models that would provide clarity or completeness?
- ❑ Are all necessary report specifications present and complete?

Requirements review tips

- Plan the examination: The consumers of the requirements specification won’t be reading it front-to-back like a book; reviewers don’t have to, either. Invite certain reviewers to focus on specific sections of documents.
- Start early: Begin reviewing sets of requirements when they are perhaps only 10 percent complete.
- Allocate sufficient time: Give reviewers sufficient time to perform the reviews, both in terms of actual hours to review (effort) and calendar time.
- Provide context: Give reviewers context for the document and perhaps for the project if they are not all working on the same project.
- Set review scope: Tell reviewers what material to examine, where to focus their attention, and what issues to look for.
- Limit re-reviews: Tell reviewers what material to examine, where to focus their attention, and what issues to look for.
- Prioritize review areas: Prioritize for review those portions of the requirements that are of high risk or have functionality that will be used frequently.

Requirements review challenges

- **Large requirements documents:** The prospect (triển vọng) of thoroughly examining a several-hundred-page requirements document is daunting (khó khăn). To avoid overwhelming the review team, perform incremental reviews throughout requirements development.
- **Large inspection teams:**
  - Make sure every participant is there to find defects, not to be educated or to protect a position (quan điểm).
  - Understand which perspective (such as customer, developer, or tester) each inspector represents.
  - Establish several small teams to inspect the requirements in parallel and combine their defect lists, removing any duplicates.
- **Geographically separated reviewers:** Using videoconferencing can be an effective solution. Web conferencing tools allow reviewers to ensure that they are all looking at the same material during the discussion.
- **Unprepared reviewers:** One of the prerequisites to a formal review meeting is that the participants have examined the material being reviewed ahead of time, individually identifying their initial sets of issues.

Testing the requirements

- You can begin deriving conceptual tests from user requirements early in the development process. Use the tests to evaluate functional requirements, analysis models, and prototypes. The tests should cover the normal flow of each use case, alternative flows, and the exceptions you identified during elicitation and analysis.
- Ideally, a BA will write the functional requirements and a tester will write the tests from a common starting point: the user requirements.

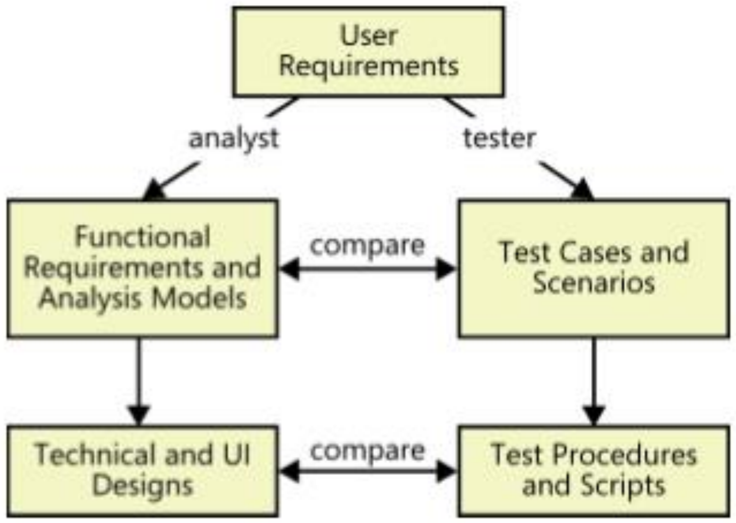


FIGURE 17-5 Development and testing work products are derived from a common source.

These conceptual tests are independent of implementation. For example, consider a use case called "View a Stored Order" for the Chemical Tracking System. Some conceptual tests are:

- User enters order number to view, order exists, user had placed the order. Expected result: show order details.
- User enters order number to view, order doesn't exist. Expected result: Display message "Sorry, I can't find that order."
- User enters order number to view, order exists, user hadn't placed the order. Expected result: Display message "Sorry, that's not your order."

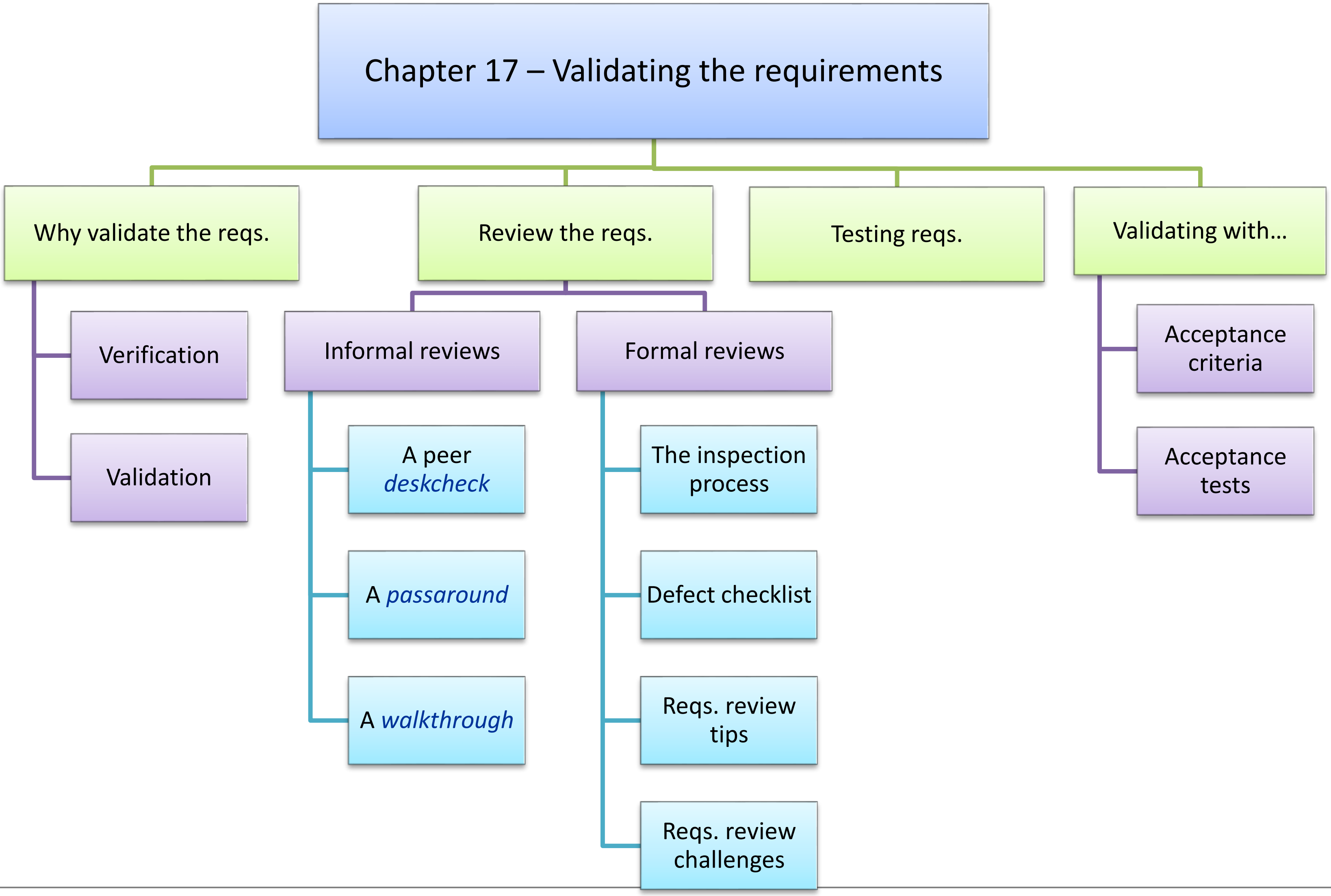
Validating requirements with acceptance criteria

- Software developers might believe that they've built the perfect product, but the customer is the final arbiter (trọng tài). Customers need to assess whether a system satisfies its predefined *acceptance criteria*.
- Acceptance criteria: Working with customers to develop acceptance criteria provides a way to validate both the requirements and the solution itself. If a customer can't express how she would evaluate the system's satisfaction of a particular requirement, that requirement is not clear enough. Acceptance criteria could also encompass dimensions such as the following:
  - Specific high-priority functionality that must be present and operating properly before the product could be accepted and used.
  - Essential nonfunctional criteria or quality metrics that must be satisfied.
  - Remaining (để lại) open issues and defects - TBD.
  - Specific legal, regulatory, or contractual conditions.

Validating requirements with acceptance criteria

- Acceptance tests:
  - Acceptance tests constitute (là) the largest portion of the acceptance criteria. Creators of acceptance tests should consider the most commonly performed and most important usage scenarios when deciding how to evaluate the software’s acceptability.
  - Focus on testing the normal flows of the use cases and their corresponding exceptions, devoting less attention (ít chú ý hơn) to the less frequently used alternative flows.
  - In the agile projects, the problem with writing *only* acceptance tests is that the requirements exist only in people’s minds. By not documenting and comparing alternate views of requirements—user requirements, functional requirements, analysis models, and tests—you can miss an opportunity to identify errors, inconsistencies, and gaps.

Review chapter 17



THE END
THANK YOU!