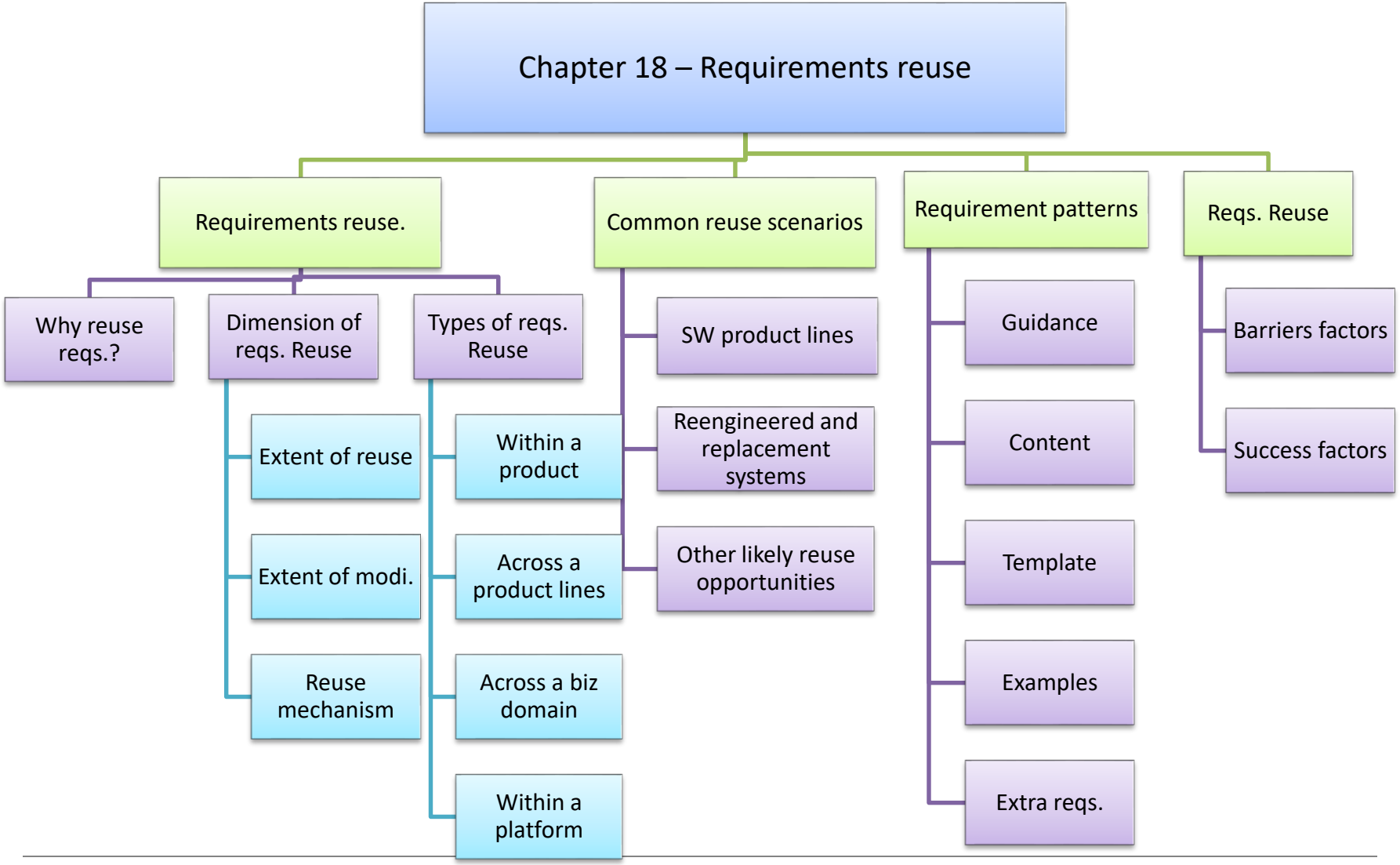
FPT UNIVERSITY

ĐẠI HỌC FPT CẦN THƠ

CHAPTER 19

Beyond requirements development

Review chapter 18



Objectives

- Exploring some approaches for bridging (thu hẹp) the gap between requirements development and a successful product release.
- Student should understand the influence of requirements on project plans, designs, code, and test.

Contents

1. The effects of requirements on software development.
2. Estimating requirements effort.
3. From requirements to project plans.
4. From requirements to designs and code.
5. From requirements to tests.
6. From requirements to success.

The effects of requirements on software development

- We'll look at several ways in which requirements influence project plans, designs, code, and tests, as shown in Figure 19-1.

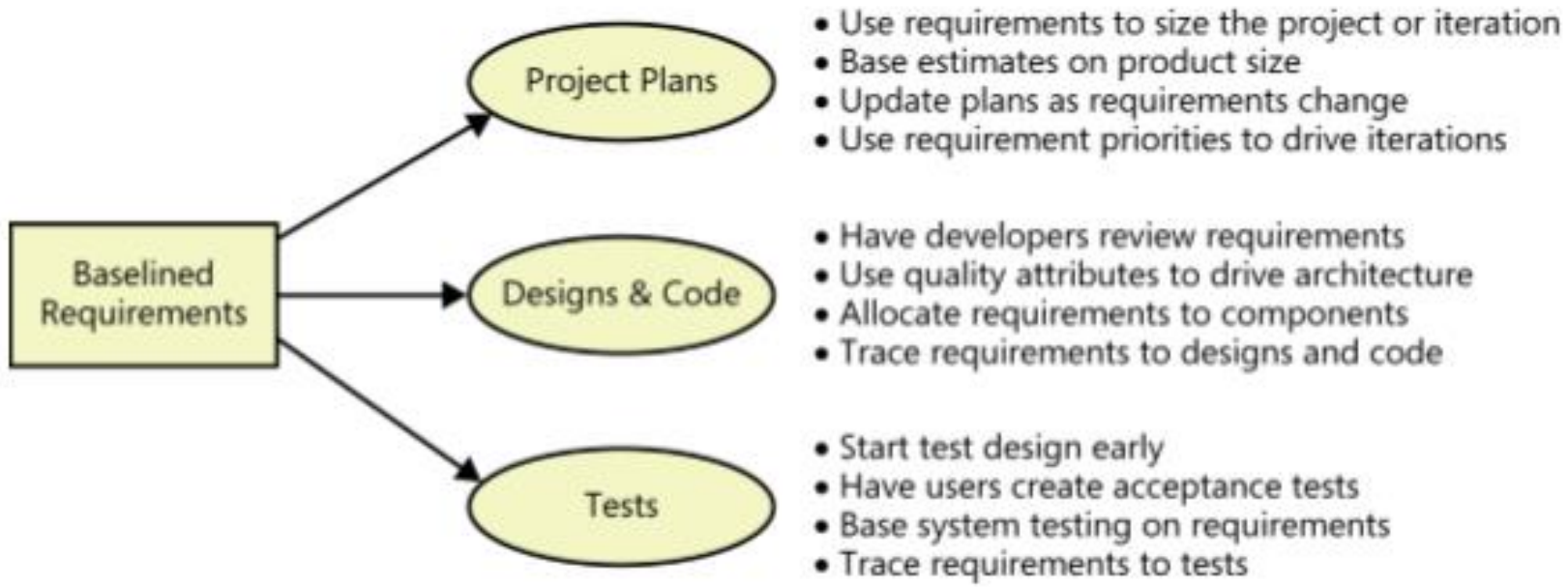


FIGURE 19-1 Requirements drive project planning, design, coding, and testing activities.

Estimating requirements effort

- One of the earliest project planning activities is to judge how much of the project's schedule and effort should be devoted to requirements activities.
- Requirements engineering activity is distributed throughout the project in different ways, depending on whether the project is following a sequential (waterfall), iterative, or incremental development life cycle (Agile projects).
- Small projects typically spend 15 to 18 percent of their total effort on requirements work (Wiegers 1996), but the appropriate percentage depends on the size and complexity of the project.
- Despite the fear (mặc dù lo sợ) that exploring requirements will slow down a project, considerable evidence shows that taking time to understand the requirements actually accelerates development.

From requirements to project plans

- Because requirements are the foundation for the project's intended work, you should base (bố trí) estimates, project plans, and schedules on those requirements. Remember, the most important project outcome is a system that meets its business objectives.
- Estimating project size and effort from requirements
  - The number of individually testable requirements (Wilson 1995)
  - Function points (Jones 1996b; IFPUG 2010)
  - Story points (Cohn 2005; McConnell 2006) or use case points (Wiegers 2006)
  - The number, type, and complexity of user interface elements
  - Estimated lines of code needed to implement specific requirements
- Requirements and scheduling
  - Estimated product size.
  - Known productivity of the development team, based on historical performance.
  - A list of the tasks needed to completely implement and verify a feature or use case.
  - Reasonably stable requirements, at least for the forthcoming development iteration.
  - Experience, which helps the project manager adjust for intangible factors (các yếu tố vô hình) and the unique aspects (khía cạnh độc đáo) of each project.

From requirements to designs and code

- The boundary between requirements and design is not a sharp line (sắc nét) but a gray, fuzzy area. Try to avoid *inadvertent (cẩu thả)* design, needless (vô ích) or unintended restrictions on the design. Include designers in requirements reviews to make sure the requirements can serve as a solid foundation for design.
- Architecture and allocation:
  - Architecture is especially critical for systems that include both software and hardware components and for complex software-only systems.
  - An essential step is to allocate the high-level system requirements to the various subsystems and components. An analyst, system engineer, or architect decomposes the system requirements into functional requirements for both software and hardware subsystems.
  - Allocation of system capabilities to subsystems and components must be done from the top down.

From requirements to designs and code

- Software design:
  - A variety of software designs will satisfy most products' requirements. These designs will vary in (khác nhau về) their performance, efficiency, robustness, and the technical methods employed. If you leap directly from requirements into code, you're essentially designing the software mentally and on the fly.
  - The time you spend translating requirements into designs is an excellent investment in building high-quality, robust products.
  - You needn't develop a complete, detailed design for the entire product before you begin implementation, but you should design each component before you code it.
- User interface design:
  - User interface design is an extensively studied domain that goes well beyond the scope of this book.
  - A display-action-response (DAR) model is a useful tool for documenting the UI elements that appear in screens and how the system responds to user actions.
  - Example in the next slide.

From requirements to designs and code

FIGURE 19-4 *High-fidelity webpage design.*

From requirements to designs and code

UI Element: Submit a Pearl Page at PearlsFromSand.com	
UI Element Description	
ID	submit.html
Description	Page where users can submit their own life lessons to be posted on the Pearls from Sand blog
UI Element Description	
Precondition	**Display**
Always	"Home" link "About the Book" link "About the Author" link "Blog" link "Submit a Pearl" link (inactive, different color because it's the current page) "Buy the Book" link "Contact" link "Name" text field "City" text field "State or Province" drop-down list "Email" text field "Title" text field "Pearl Category" drop-down list "Your Story" text field "I agree" check box, cleared "Submit" button "Pearl Submission Guidelines" link "Pearl Submission Terms" link
User just submitted a pearl	"Name," "City," "State or Province," and "Email" fields are populated with values from previous pearl. "Title," "Pearl Category," "Your Story," and "I agree" fields are reset to default values.

From requirements to designs and code

UI Element Behaviors		
Precondition	**User Action**	**Response**
Always	User clicks on navigation links: "Home," "About the Book," "About the Author," "Buy the Book," "Contact," "Pearl Submission Guidelines," "Pearl Submission Terms"	Corresponding page is displayed
Always	User clicks on either "Blog" link	Pearls from Sand blog opens in new browser tab
Always	User types or pastes text into a text field	User's text is displayed in field; for "Your Story" field, count of remaining characters is displayed
Always	User clicks on "I agree" check box	Check box toggles on/off
One or more invalid entries	User clicks on "Submit" link	Error message appears for any invalid text entry or length or for required fields that are blank
All fields have valid entries; "I agree" check box is selected	User clicks on "Submit" link	Pearl is submitted; pearl counter is incremented; email with pearl info is sent to Submitter and Administrator; successful submission acknowledgment message is displayed.
"I agree" box not checked	User clicks on "Submit" link	System displays error message on this page

FIGURE 19-5 Display-action-response (DAR) model for the webpage shown in Figure 19-4.

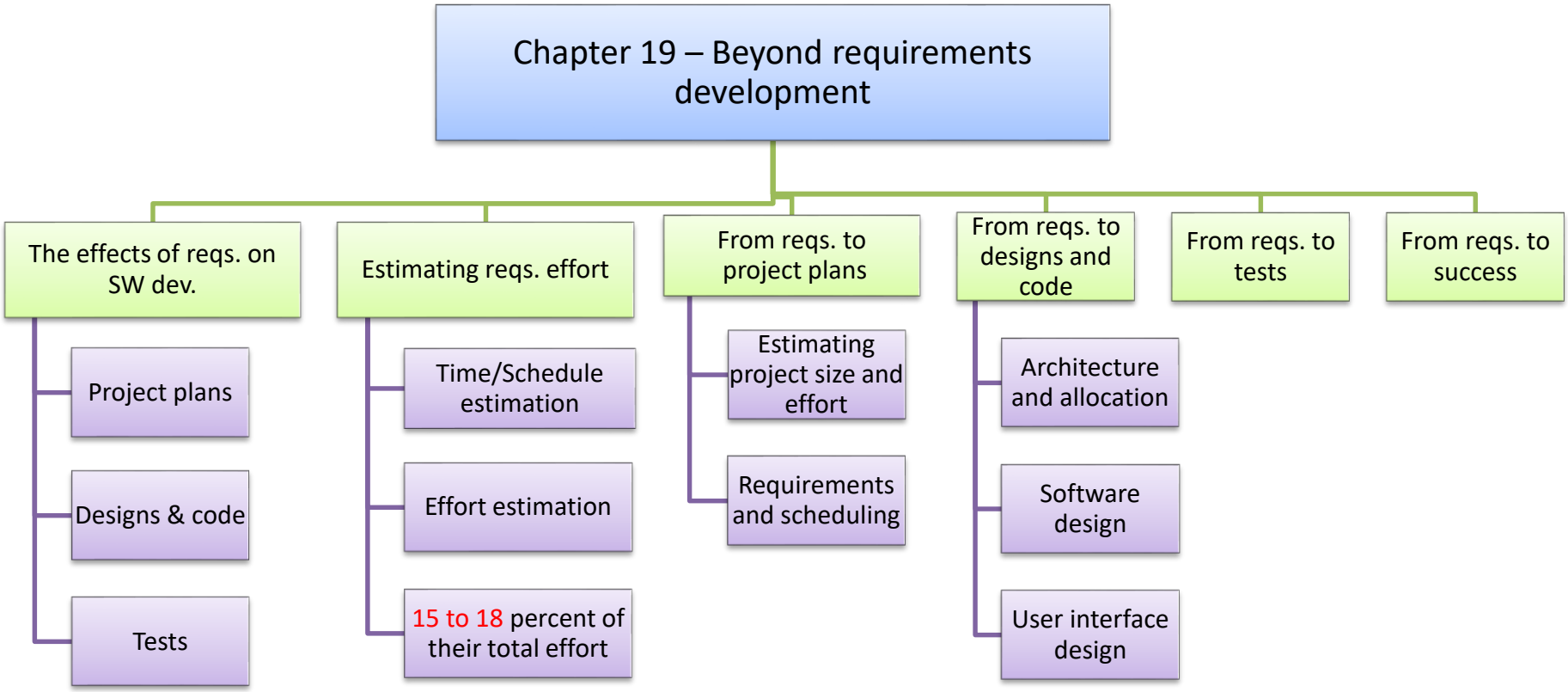
From requirements to tests

- From requirements to tests:
  - Requirements analysis and testing fit together beautifully.
  - As consultant Dorothy Graham (2002) points out, "Good requirements engineering produces better tests; good test analysis produces better requirements."
  - Agile development teams typically write acceptance tests in lieu of (thay cho) precise requirements (Cohn 2004).
  - The testers or quality assurance staff should determine how they'd verify the implementation of each requirement. Possible methods include:
    - Testing (executing the software to look for defects).
    - Inspection (examining the code to ensure that it satisfies the requirements).
    - Demonstration (showing that the product works as expected).
    - Analysis (reasoning through (lập luận thông quan) how the system should work under certain circumstances).

From requirements to success

- From requirements to success:
  - A more successful team had a practice of listing all the requirements that were planned for a specific release.
  - The project's quality assurance group evaluated each release by executing the tests for those requirements.
  - A requirement that didn't satisfy its test criteria was counted as a defect.
  - The QA group rejected the release if more than a predetermined number of requirements weren't met or if specific high-impact requirements weren't satisfied.
  - This project was successful largely because it used its documented requirements to decide when a release was shippable (sẵn sàng bàn giao).
  - The ultimate deliverable from a software development project is a solution that meets the customers' needs and expectations.
  - Requirements are an essential step on the path from business need to satisfied customers.

Review chapter 19



THE END
THANK YOU!