ĐẠI HỌC FPT CẦN THƠ

CHAPTER 3

Requirements engineering good practices

Objectives

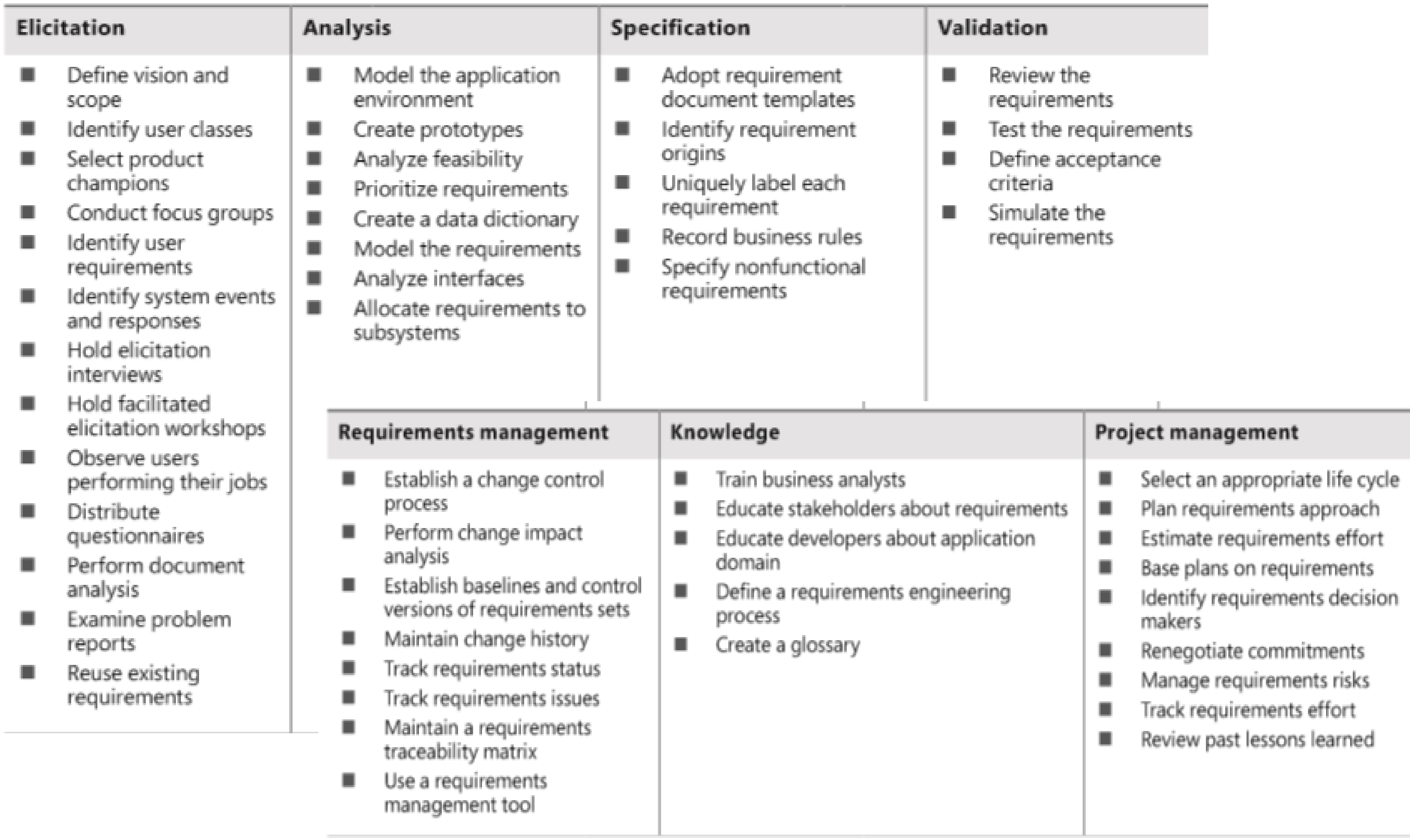
- Student should enhance every stage in the requirement development process framework and how to implement these stages.
- Student should obtain a wide range of good practice in software requirements to prepare for the next chapters.
- Need to help student understand these practices aren't suitable for every situation, so use good judgment, common sense, and experience. Even the best practices need to be selected, applied, and adapted thoughtfully to appropriate situations by skilled business analysts.
- Student must enhance that different practices might be most appropriate for understanding the requirements for different portions of a given project.
- Use cases and user interface prototypes might help for the client side, whereas interface analysis is more valuable on the server side…etc.

Contents

Requirements engineering good practices

- The notion of best practices is debatable: who decides what is “best” and on what basis? One approach is to convene (tập hợp) a body of industry experts to analyze project from many organizations. These experts seek out practices whose effective performance is associated with successful projects and which are not at all on failed projects.
- There are more than 50 practices, grouped into 7 categories, that can help all development teams do a better on their requirements activities.
- Note that this chapter is titled “Good practices for requirements engineering”, not “Best practices.”

Requirements engineering good practices



Elicitation	Analysis	Specification	Validation
■ Define vision and scope	■ Model the application environment	■ Adopt requirement document templates	■ Review the requirements
■ Identify user classes	■ Create prototypes	■ Identify requirement origins	■ Test the requirements
■ Select product champions	■ Analyze feasibility	■ Uniquely label each requirement	■ Define acceptance criteria
■ Conduct focus groups	■ Prioritize requirements	■ Record business rules	■ Simulate the requirements
■ Identify user requirements	■ Create a data dictionary	■ Specify nonfunctional requirements	
■ Identify system events and responses	■ Model the requirements		
■ Hold elicitation interviews	■ Analyze interfaces		
■ Hold facilitated elicitation workshops	■ Allocate requirements to subsystems		
■ Observe users performing their jobs			
■ Distribute questionnaires			
■ Perform document analysis			
■ Examine problem reports			
■ Reuse existing requirements			

Requirements management	Knowledge	Project management
■ Establish a change control process	■ Train business analysts	■ Select an appropriate life cycle
■ Perform change impact analysis	■ Educate stakeholders about requirements	■ Plan requirements approach
■ Establish baselines and control versions of requirements sets	■ Educate developers about application domain	■ Estimate requirements effort
■ Maintain change history	■ Define a requirements engineering process	■ Base plans on requirements
■ Track requirements status	■ Create a glossary	■ Identify requirements decision makers
■ Track requirements issues		■ Renegotiate commitments
■ Maintain a requirements traceability matrix		■ Manage requirements risks
■ Use a requirements management tool		■ Track requirements effort
		■ Review past lessons learned

A requirements development process framework

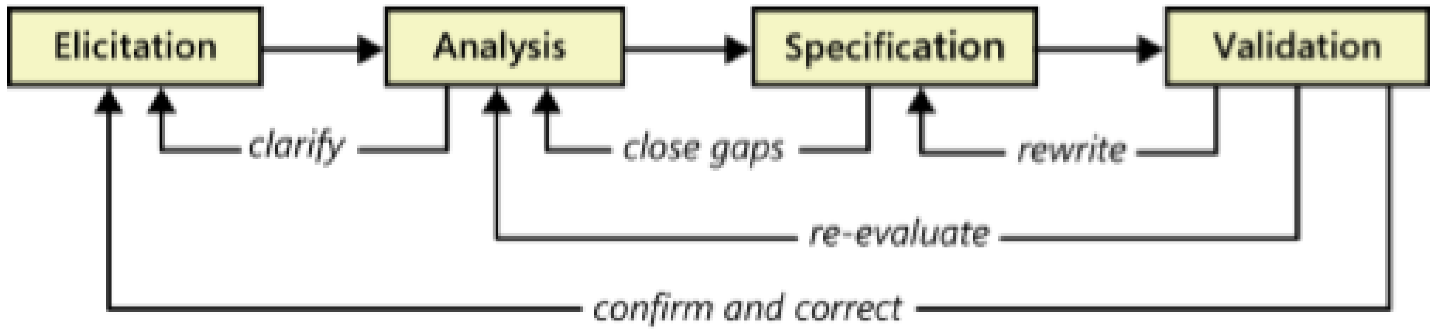
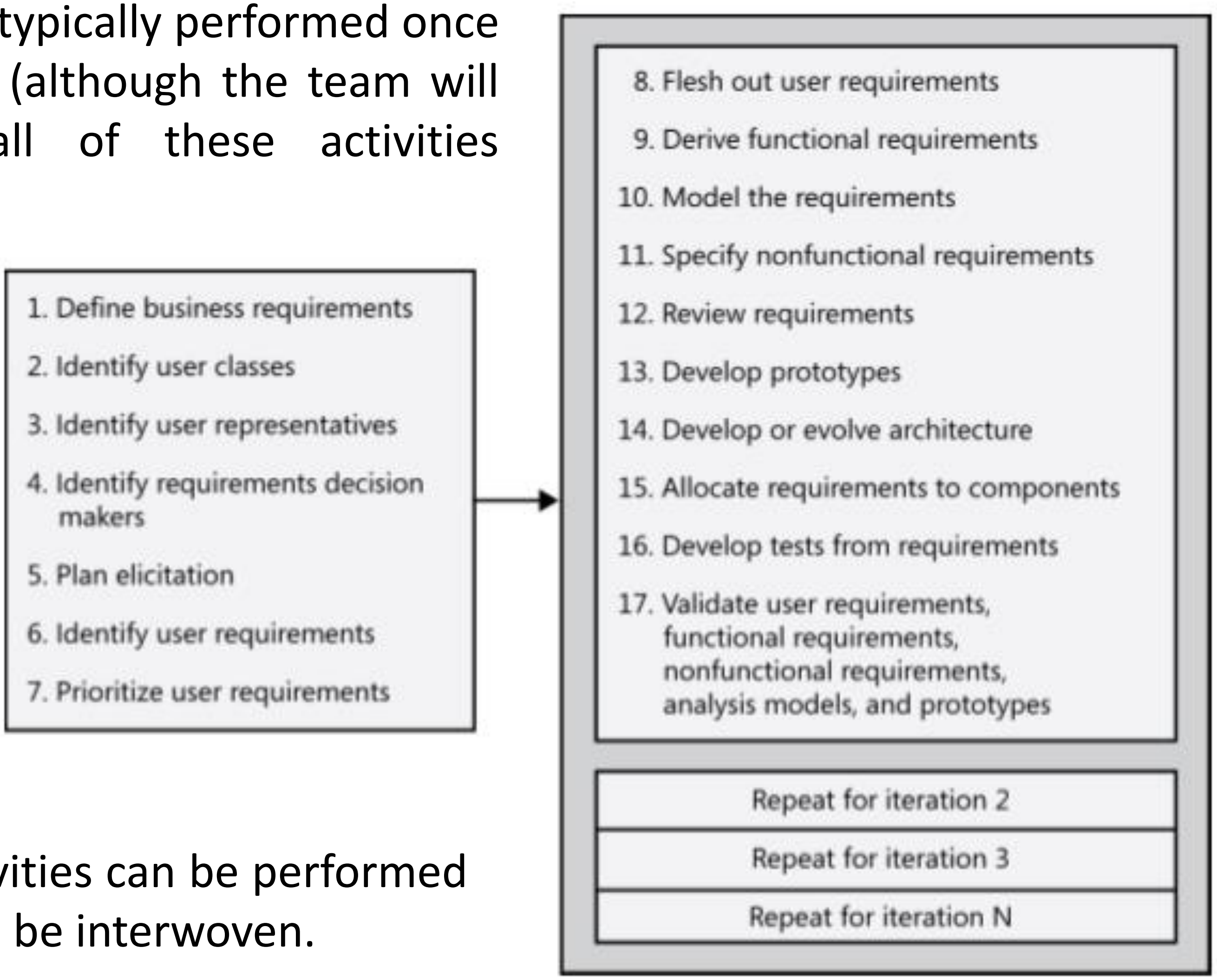


FIGURE 3-1 Requirements development is an iterative process.

- Don’t expect to perform these activities in a simple linear. In practice, they are interwoven, incremental, and iterative, as above figure.
- This iterative process continues throughout requirements development and possibly – as with agile projects – throughout the full project duration.

A representative requirements development process

- The first 7 steps are typically performed once early in the project (although the team will need to revisit all of these activities periodically).
- The remaining steps are performed for each release or development iteration.
- Many of these activities can be performed iteratively, they can be interwoven.



1. Define business requirements
2. Identify user classes
3. Identify user representatives
4. Identify requirements decision makers
5. Plan elicitation
6. Identify user requirements
7. Prioritize user requirements
8. Flesh out user requirements
9. Derive functional requirements
10. Model the requirements
11. Specify nonfunctional requirements
12. Review requirements
13. Develop prototypes
14. Develop or evolve architecture
15. Allocate requirements to components
16. Develop tests from requirements
17. Validate user requirements, functional requirements, nonfunctional requirements, analysis models, and prototypes
Repeat for iteration 2
Repeat for iteration 3
Repeat for iteration N

Requirements elicitation

- Define product vision and project scope:
  - The vision statement gives all stakeholders a common understanding of the product's outcome.
  - The scope defines the boundary between what's in and what's out for a specific release or iteration.
- Identify user classes and their characteristics:
  - To avoid overlooking the needs of any user community, identify the various groups of users for your product.
  - Describe aspects of their job tasks, attitudes, location, or personal characteristics that might influence product design.
- Select a product champion for each user class:
  - Identify an individual who can accurately serve as the literal voice (tiếng nói thật) of the customer for each user class.
  - The product champion presents the needs of user class and makes decisions on its behalf.

Requirements elicitation

- Conduct focus groups with typical users:
  - Convene groups of representative users of your previous products or of similar products. Collect their input on both functionality and quality characteristics for the product under development.
  - Focus groups are particularly valuable for commercial product development.
- Work with user representatives to identify user requirements:
  - Explore with your user representatives the tasks they need to accomplish with the SW and the value they're trying to achieve.
  - User requirements can be use cases, user stories, or senarios.
- Identify system events and responses:
  - List the external events that the system can experience and its expected response to each event.

Requirements elicitation

- Hold elicitation interviews:
  - Interviews can be performed one-on-one or with a small group of stakeholders.
- Hold facilitated elicitation workshops:
  - Facilitated requirements-elicitation workshops that permit collaboration between analysts and customers are a powerful way to explore user needs and to draft requirements documents.
- Observe users performing their jobs:
  - Watching users perform their biz tasks establishes a context for their potential use of a new application.
  - Simple process flow diagrams can depict the steps and decisions involved and show how different user groups interact.

Requirements elicitation

- Distribute questionnaires:
  - Questionnaires are a way to survey large groups of users to determine what they need. If questions are well written, this help you quickly determine analytical information about needs.
- Perform document analysis:
  - Existing documentation can help reveal how system currently work or what they are supposed to do. Documentation includes any written information about current systems, biz processes, requirements specifications, competitor research, and COST (commercial off-the-shelf) package user manuals.
- Examine problem reports of current systems for requirement ideas:
  - Problem reports and enhancement requests from users provide a rich source of ideas for capabilities to include in a later release or in a new product.
- Reuse existing requirements:
  - If customers request functionality similar to that already present in an existing product, see whether the requirements are flexible enough to permit reusing or adapting the existing SW components.

Requirements analysis

Requirements analysis involves refining the requirements to ensure that all stakeholders understand them and scrutinizing them for errors, omissions, and other deficiencies.

- Model the application environment:
  - The context diagram is simple analysis model that shows how the new system fits into its environment.
  - It defines the boundaries and interfaces between the system being developed and external entities, such as users, HW devices, and other systems.
- Create user interface and technical prototypes:
  - When the developers or users aren't certain about the requirements, construct a prototype – a partial, possible, or preliminary implementation – to make the concepts and possibilities more tangible.

Requirements analysis

- Analyze requirement feasibility:
  - The BA work with developers to evaluate the feasibility of implementing each requirement at acceptable cost and performance in the intended operating environment.
  - This allows stakeholders understand the risks associated with implementing each requirement, including conflicts and dependencies with other requirements.
- Prioritize the requirements:
  - Prioritizing requirements ensures that the team implements the highest value or most timely functionality first.
  - Apply an analytical approach to determine the relative implementation priority of product features, use cases, user stories, or functional requirements.

Requirements analysis

- Create a data dictionary:
  - Definitions of data items and structures associated with the system reside in the data dictionary.
  - That helps everyone working on the project to use consistent data definitions.
- Model the requirements:
  - An analysis model is a diagram that depicts requirements visually, in contrast to the textual representation of a list of functional requirements.
  - Model can reveal incorrect, inconsistent, missing, and superfluous (không cần thiết) requirements.
  - Such models include data flow diagrams, entity-relationship diagrams, state-transition diagrams, state tables, dialog maps, decision trees, and others.

Requirements analysis

- Analyze interfaces between your system and the outside world:
  - All SW systems have connections to others parts of the world through external interfaces. Information systems have user interfaces and often exchange data with other SW systems.
  - Embedded systems involve interconnections between SW and HW components. Network-connected applications have communication interfaces.
- Allocate requirements to subsystems:
  - The requirements for complex product that contains multiple subsystems must be apportioned (chia ra) among the various SW, HW, and human subsystems and components.

Requirements specification

The essence of requirements specification is to document requirements of different types in a consistent, accessible, and reviewable way that is readily understandable by the intended audiences.

- Adopt requirement document templates:
  - Adopt standard templates for documenting requirements in your organization.
  - The templates provide a consistent structure for recording various groups of requirements-related information.
- Identify requirement origins:
  - To ensure that all stakeholders know why every requirement is needed, trace each one back to its origin.
  - This might be a use case or some other customer input, a high-level system requirement, or a biz rule.

Requirements specification

- Uniquely label each requirement:
  - Define a convention for giving each requirement a unique identifying label. The convention must be robust enough to withstand (chống lại) additions, deletions, and changes made in the requirements over time.
- Record business rules:
  - Biz rules include corporate policies, government regulations, standards, and computational algorithms.
  - Document your biz rules separately from a project's requirements because they typically have an existence beyond the scope of a specific project.
- Specify nonfunctional requirements:
  - It's possible to implement a solution that does not exactly what it's supposed to do but does not satisfy the users' quality expectations.
  - To avoid that problem, you need to go beyond the functionality discussion to understand the quality characteristics that are important to success. These characteristics are performance, reliability, usability, modifiability, and many others.

Requirements validation

Validation ensures that the requirements are correct, demonstrate the desired quality characteristics, and will satisfy customer needs.

- Review the requirements:
  - Peer review of requirements, particularly the type of rigorous review called inspection, is one of the highest-value SW quality practices available.
  - The small teams of reviews (analyst, customers, developers, and testers) carefully examine the written requirements, analysis models, and related information for defects.
- Test the requirements:
  - Tests constitute an alternative view of the requirements.
  - Writing tests requires you to think about how to tell if the expected functionality was correctly implemented.

Requirements validation

- Define acceptance criteria:
  - Ask users to describe how they will determine whether the solution meets their needs.
  - Acceptance criteria include a combination of SW passing a defined set of acceptance tests based on user requirements, demonstration satisfaction of specific nonfunctional requirements, tracking open defects and issues,…
- Simulate the requirements:
  - Commercial tools are available that allow a project team to simulate a proposed system either in place of or to augment written requirements specifications.

Requirements management

After you have the initial requirements for a body of work in hand, you must cope with the inevitable changes that customers, managers, marketing, the development team, and others request during development. You need to control all changes!

- Establish a requirements change control process:
  - Changes are inevitable!
  - Your change process should define how requirements changes are proposed, analyzed, and resolved. In the project management, a Change Control Board (CCB) evaluates proposed requirements changes, decides which ones to accept, and set implementation priorities or target releases.
- Perform impact analysis on requirements changes:
  - Impact analysis is an important element of the change process that helps the CCB make informed biz decisions.
  - Evaluate each proposed requirement change to assess the effect it will have on the project.

Requirements management

- Establish baselines and control versions of requirements sets:
  - A baseline defines a set of agreed-upon requirements, typically for a specific release or iteration.
  - After the requirements have been baselined, changes should be made only through the project's change control process. Every change must have unique version.
- Maintain a history of requirements changes:
  - Retain a history of the changes made to individual requirement. Sometimes you need to revert to an earlier version of a requirement or want to know how a requirement came to be in its current form.
  - Record the dates that requirements were changed, the changes that were made, who made each change, and why.

Requirements management

- Track the status of each requirement:
  - Create a repository with one record for each discrete requirement of any type that affects implementation.
  - Store key attributes about each requirement, including its status (such as proposed, approved, implemented, or verified), so you can monitor the number of requirements in each status category at any time.
- Track requirements issues:
  - When busy people are working on a complex project, it's easy to lose sight of the many issues that arise, including questions about requirements that need resolution, gaps to eradicate, and issues arising from requirements reviews.
  - Issue-tracking tools can keep these items from falling through the cracks.

Requirements management

- Maintain a requirements traceability matrix:
  - Requirements traceability matrix is helpful for confirming that all requirements are implemented and verified.
  - It’s also useful during maintenance when a requirement has to be modified.
  - Populate this matrix during development, not at the end.
- Use a requirements management tool:
  - Commercial requirements management tools let you store various types of requirements in a database.
  - This tool helps you implement and automate many of the other requirements management practices described in this section.

Knowledge

Biz analysis is a specialized and challenging role, with its own body of knowledge. Training can increase the proficiency and comfort level of those who serve as analysts, but it can't compensate for absent interpersonal skills or a lack of interest in the role.

- Train business analysts:
  - All team members who will perform BA tasks should be trained in requirements engineering.
  - That gives them a solid foundation on which to build through their own experiences and advanced training.
- Educate stakeholders about requirements
  - Users who will participate in SW development should receive one or two days of education about requirements so they understand terminology, key concepts and practices, and why this is such an important contributor to project success.

Knowledge

- Educate developers about the application domain:
  - To give developers a basic understanding of the application domain, arrange a seminar on the customer's biz activities, terminology, and adjectives for the product being created. This can reduce confusion, miscommunication, and rework.
- Define a requirements engineering process:
  - Document steps your organization follows to elicit, analyze, specify, validate, and manage requirements. Providing guidance on how to perform the key steps will help analysts to a consistently good job.
- Create a glossary:
  - A glossary that defines specialized terms from the application domain will minimize misunderstandings. Include synonyms, acronyms or abbreviations, terms that can have multiple meanings, and terms that have both domain-specific and everyday meanings.

Project management

- Select an appropriate software development life cycle:
  - Your organization should define several development life cycles that are appropriate for various types of projects and different degrees of requirements uncertainty. Each PM should select and adapt the life cycle that best suits her project.
- Plan requirements approach:
  - Each project team should plan how it will handle its requirements development and management activities. An elicitation plan helps ensure that you identify and obtain input from appropriate stakeholders at the right stages of the project using the most appropriate techniques.
- Estimate requirements effort:
  - Stakeholders often want to know how long it's going to take to develop the requirements for a project and what percentage of their total effort should be devoted to requirements development and management.

Project management

- Base project plans on requirements:
  - Develop plans and schedules for your project iteratively as the scope and detailed requirements become clear. Begin by estimating the effort needed to develop the user requirements from the initial product vision and product scope.
- Identify requirements decision makers:
  - SW development involves making many decisions. Conflicting users inputs must be resolved, commercial package components must be selected, changes requests must be evaluated,… So we need to have decision makers.
- Renegotiate project commitments when requirements change:
  - A project team makes commitments to deliver specific set of requirements within a particular schedule and budget. If all changes impact to schedule, budget and scope of the project you need to renegotiate with stakeholders.

Project management

- Analyze, document, and manage requirements-related risks:
  - Unanticipated events and conditions can be wreak havoc on an unprepared project. Identify and document risks related to requirements as part of the project's risk-management activities.
- Track the effort spent on requirements:
  - To improve your ability to estimate the resources needed for requirements work on future projects, record the effort your team expends on requirements development and management activities.
- Review lessons learned regarding requirements on other projects:
  - A learning organization conducts periodic retrospective (nhìn lại) to collect lessons learned from completed projects for from earlier iterations of current project.

QS STARS RATED FOR EXCELLENCE 2012

FPT Education

FPT UNIVERSITY

THE END
THANK YOU!