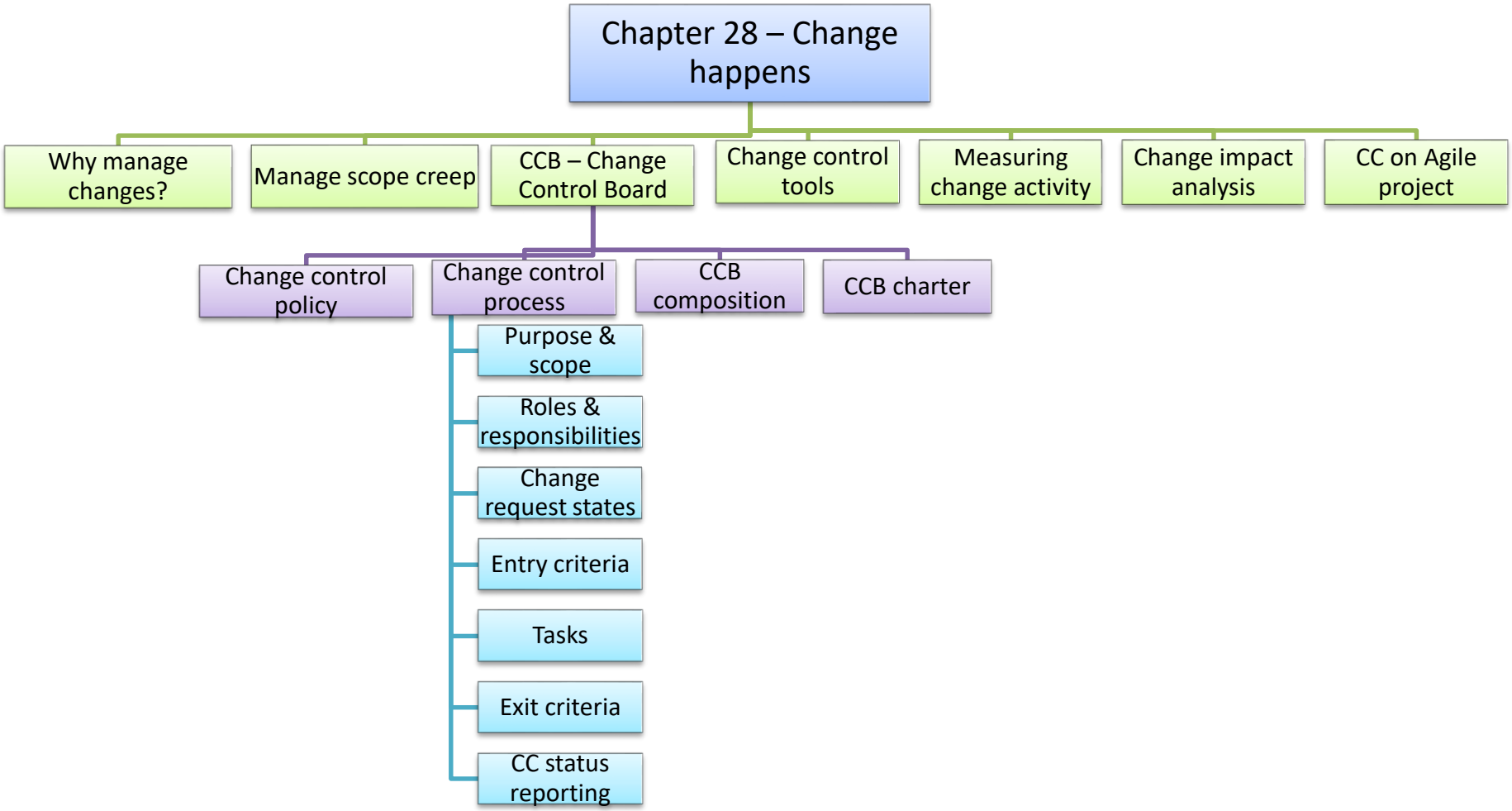
FPT UNIVERSITY

ĐẠI HỌC FPT CẦN THƠ

Chapter 29

Links in the requirements chain

Review chapter 28



Objectives

- Addressing the subject of requirements tracing (or traceability).
- Student knows that the requirements trace information documents the dependencies and logical links between individual requirements and other system elements.
- Student understands that trace information facilitates impact analysis to identify all the works related to implementation of requirement change.

Contents

- Tracing requirements
- Motivations for tracing requirements
- The requirements traceability matrix
- A requirements tracing procedure

Tracing requirements

- Why tracing requirements? →Trace links allow you to follow the life of a requirement both forward and backward, from origin through implementation.
- Types of requirements tracing: *Forward to requirements, forward from requirements, backward to requirements* and *backward from requirements.*

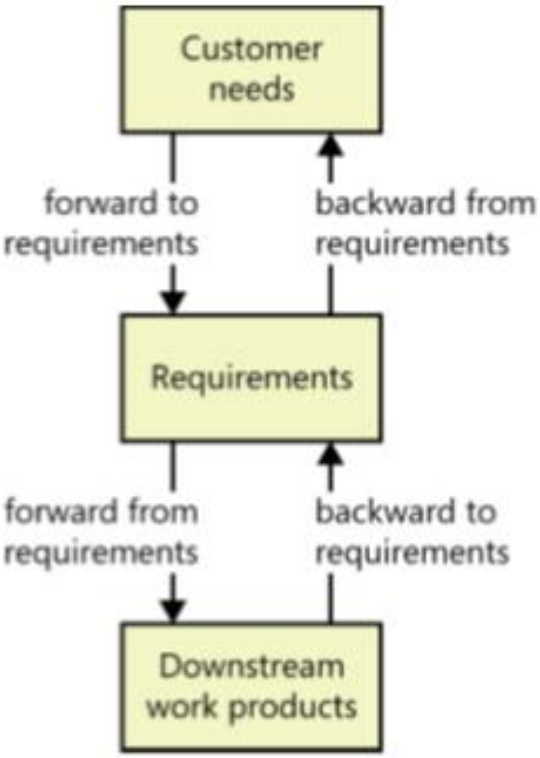


FIGURE 29-1 Four types of requirements tracing.

Tracing requirements

- There are many kinds of traceability relationships that can be defined on a project. Of course, you don't need to define and manage all these trace link types.
- Maybe you only need to trace system tests back to functional requirements or user requirements.

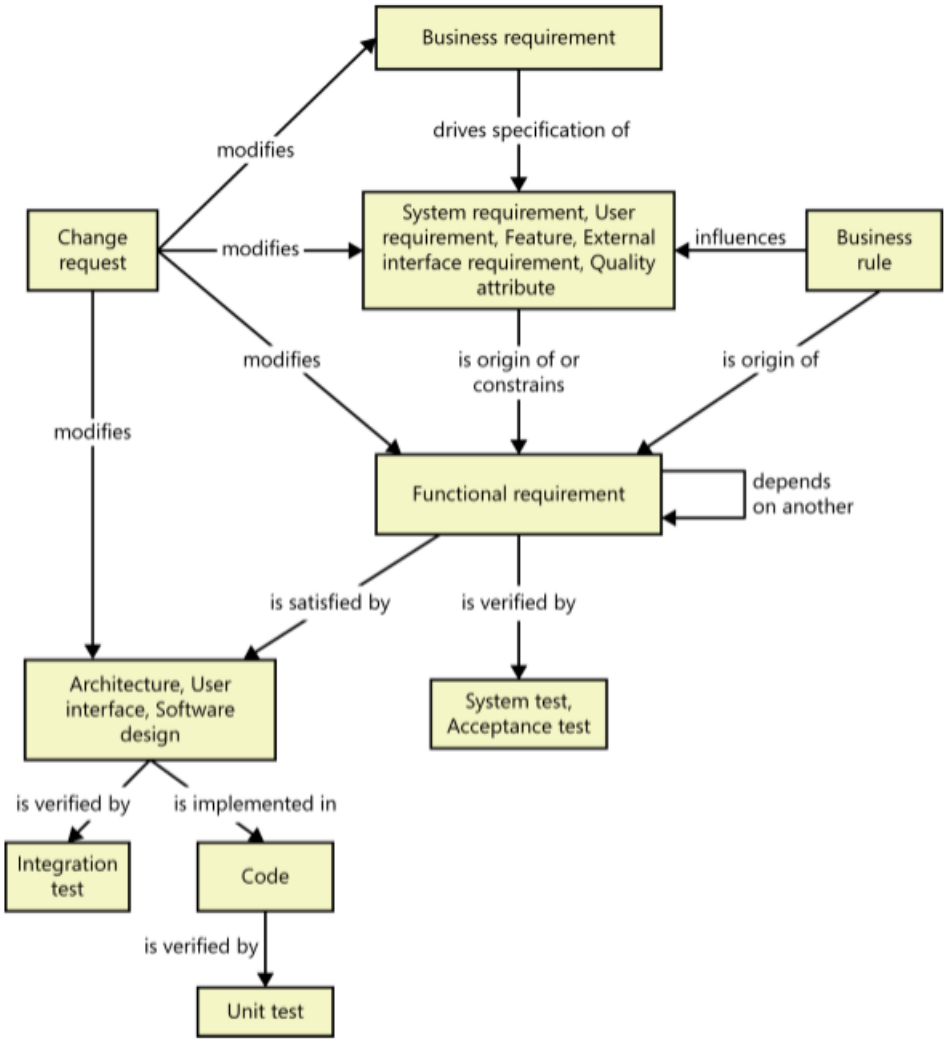


FIGURE 29-2 Some possible requirements trace links.

Motivations for tracing requirements

- At an organization level, implementing requirements tracing can improve the quality of your products, reduce maintenance costs, and facilitate reuse.
- Following are some potential benefits of implementing requirements tracing:
  - Finding missing requirements: Look for business requirements that don't trace to any user requirements, and user requirements that don't trace to any functional requirements.
  - Finding unnecessary requirements: Look for any functional requirements that don't trace back to user or business requirements and therefore might not be needed.
  - Certification and compliance: You can use trace information when certifying a safety-critical product, to demonstrate that all requirements were implemented—although that doesn't confirm that they were implemented correctly!
  - Change impact analysis: Without trace information, there's a good chance that you'll overlook a system element that would be affected if you add, delete, or modify a particular requirement.

Motivations for tracing requirements

- **Maintenance:** Reliable trace information facilitates your ability to make changes correctly and completely during maintenance. When corporate policies or government regulations change, software systems often must be updated.
- **Project tracking:** If you record the trace data during development, you'll have an accurate record of the implementation status of planned functionality.
- **Reengineering:** You can list the functions in an existing system you're replacing and trace them to where they are addressed in the new system's requirements and software components.
- **Reuse:** Trace information facilitates (tạo điều kiện) the reuse of product components by identifying packages of related requirements, designs, code, and tests.
- **Testing:** When a test fails, the links between tests, requirements, and code point developers toward likely areas to examine for the defect.

The requirements traceability matrix

- The most common way to represent the links between requirements and other system elements is in a *requirements traceability matrix*, also called a *requirements trace matrix* or a *traceability table*.
- Table 29-1 shows how each functional requirement is linked backward to a specific use case and forward to one or more design, code, and test elements.
- A design element can be something like an architectural component, a table in a relational data model, or an object class.
- Code references can be class methods, stored procedures, source code file names, or modules within a source file. Including more trace detail takes more work, but it gives you the precise locations of the related software elements.

TABLE 29-1 One kind of requirements traceability matrix

User requirement	Functional requirement	Design element	Code element	Test
UC-28	catalog.query.sort	Class catalog	CatalogSort()	search.7 search.8
UC-29	catalog.query.import	Class catalog	CatalogImport() CatalogValidate()	search.12 search.13 search.14

The requirements traceability matrix

- Table 29-2 illustrates a two-way traceability matrix. Most cells in the matrix are empty. Each cell at the intersection of two linked components contains a symbol to indicate the connection.
- Table 29-2 uses an arrow to indicate that a certain functional requirement is traced from a particular use case. For instance, FR-2 is traced from UC-1, and FR-5 is traced from both UC-2 and UC-4. This indicates that the functional requirement FR-5 is reused across two use cases, UC-2 and UC-4.

- **One-to-one** One design element is implemented in one code module.
- **One-to-many** One functional requirement is verified by multiple tests.
- **Many-to-many** Each use case leads to multiple functional requirements, and certain functional requirements are common to several use cases.

TABLE 29-2 Requirements traceability matrix showing links between use cases and functional requirements

	Use case			
Functional requirement	**UC-1**	**UC-2**	**UC-3**	**UC-4**
FR-1	↵			
FR-2	↵			
FR-3			↵	
FR-4			↵	
FR-5		↵		↵
FR-6			↵	

Traceability chain involving nonfunctional requirements

- Nonfunctional requirements such as quality attributes often do not trace directly into code.
- A *portability* requirement could restrict the language features that the programmer uses but might not result in specific code segments that enable portability.
- Other quality attributes are indeed implemented in code.
- *Security* requirements for user authentication lead to derived functional requirements that might be implemented through passwords or biometrics functionality. In those cases, you can trace the corresponding functional requirements backward to their parent nonfunctional requirement and forward into downstream deliverables as usual.

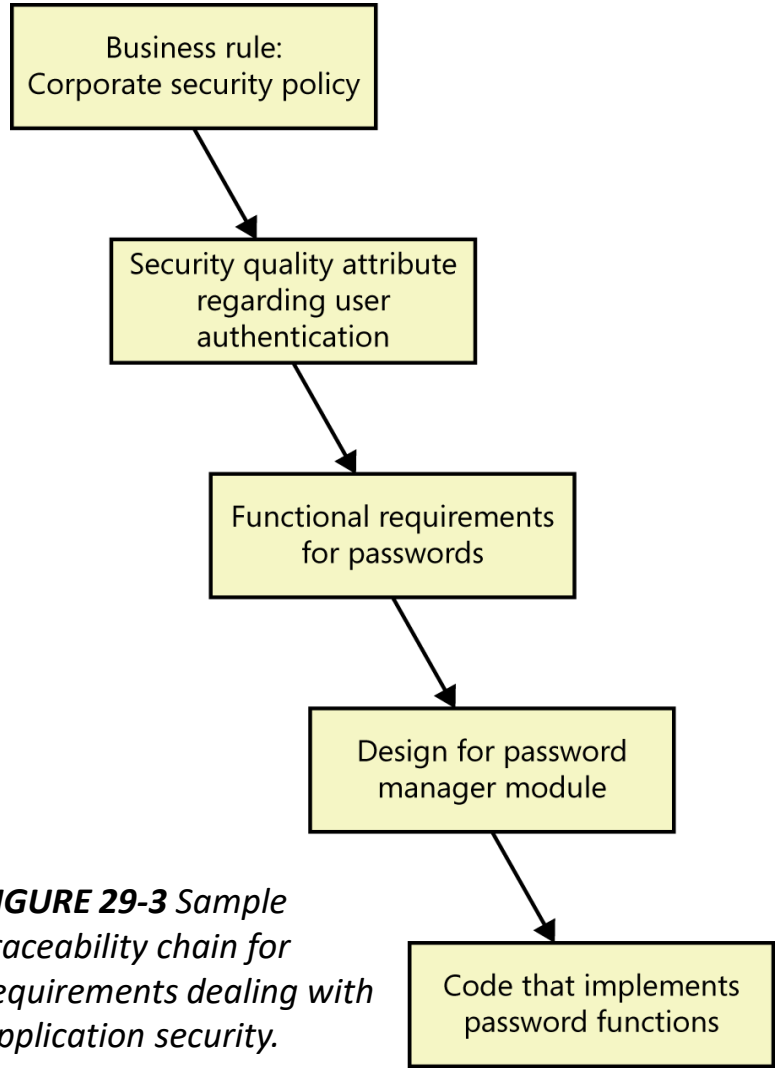


FIGURE 29-3 *Sample traceability chain for requirements dealing with application security.*

Trace link information

- Table 29-3 identifies some typical sources of knowledge about links between various types of source and target objects.
- Determine the roles and individuals who should supply each type of trace information for your project.

TABLE 29-3 Likely sources of trace link information

Link source object type	Link target object type	Information source
System requirement	Functional requirement	System engineer
User requirement	Functional requirement	Business analyst
Business requirement	User requirement	Business analyst
Functional requirement	Functional requirement	Business analyst
Functional requirement	Test	Tester
Functional requirement	Architecture element	Architect or developer
Functional requirement	Other design elements	Designer or developer
Design element	Code	Developer
Business rule	Functional requirement	Business analyst

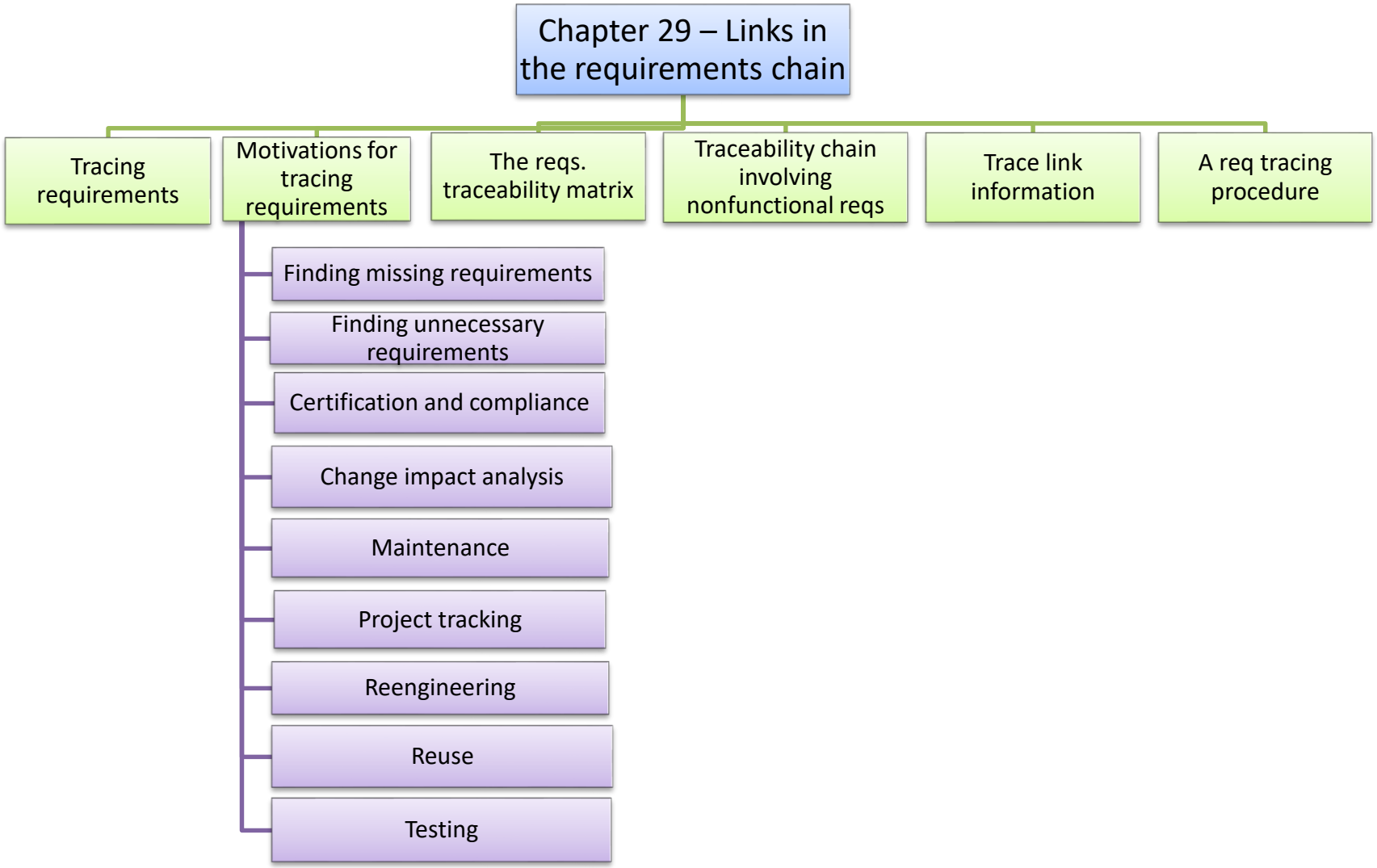
A requirements tracing procedure

- Consider following this sequence of steps when you begin to implement requirements tracing on a specific project:
  1. Educate the team and your management about the concepts and importance of requirements tracing, your objectives for this activity, where the trace data is stored, and the techniques for defining the links. Ask all participants to commit to their responsibilities.
  2. Select the link relationships you want to define from the possibilities shown in Figure 29-2. Don't try to do all of these at once! You'll be overwhelmed.
  3. Choose the type of traceability matrix you want to use: the single-matrix style shown in Table 29-1 or several matrices like the one illustrated in Table 29-2.
  4. Identify the parts of the product for which you want to maintain traceability information. Start with the critical core functions, the high-risk portions, or the portions that you expect will undergo (trải qua) the most maintenance and evolution over the product's life.

A requirements tracing procedure

- Consider following this sequence of steps when you begin to implement requirements tracing on a specific project:
  5. Identify the individuals who will supply each type of link information and the person (most likely a BA) who will coordinate the tracing activities and manage the data.
  6. Modify your development procedures to remind developers to update the links after implementing a requirement or an approved change.
  7. Define the labeling conventions you will use to give each system element a unique identifier so that the elements can be linked together.
  8. As development proceeds, have each participant provide the requested trace information as they complete small bodies of work.
  9. Audit the trace information periodically to make sure it's being kept current. If a requirement is reported as implemented and verified, but its trace data is incomplete or inaccurate, your requirements tracing process isn't working as intended.

Review chapter 29



THE END

THANK YOU!