Recall The Team Skills

1. ~~Analyzing the Problem~~
2. Understanding User and Stakeholder Needs
3. Defining the System
4. Managing Scope
5. Refining the System Definition
6. **Building the Right System**
   - From Use Cases to Implementation
   - From Use Cases to Test Cases
   - Tracing Requirements
   - **Managing Change**
   - Assessing Requirements Quality

Chapter 28
Managing Change

Why Do Requirements Change?

- A requirements management process can be useful only if it recognizes and addresses the issue of change.
- There are several reasons for the inevitability of changes to the requirements.
  - Some of these reasons are internal factors and may be under our control
  - But many of them are external factors and are outside the control of the developers and the users

External Factors

- External factors are those change agents over which the project team has little or no control.
- Changes occur because of :
  - a change to the problem we attempting to solve
  - the identity of the users changed
  - The users changed their minds or their perceptions about what they wanted the system to do
  - The external environment has changed, which creates new constraints and/or new opportunities.
  - The existence of a new system causes the requirements for the system itself to change.
- We can't prevent change, but we can manage it.

Internal Factors

- We failed to ask the right people the right questions at the right time during the initial requirements gathering effort.
- We failed to create a practical process to help manage changes to the requirements that would normally have happened on an incremental basis.
- Iterating from requirements to design causes new requirements.

Unofficial Requirement Change

- Some requirement changes are "official" external changes, representing customer requests made through the appropriate channels of communications
- But many are "unofficial" (also known as "requirements leakage"). Examples:
  - Direct customer requests to programmers
  - Functionality inserted by programmers with "careful consideration" of what's good for the customer
- Up to half of the total work product of the system are invested in requirements leakage!

A Process for Managing Change

1. Recognize that change is inevitable, and plan for it.
2. Baseline the requirements.
3. Establish a single channel to control change.
4. Use a change control system to capture changes.
5. Manage change hierarchically.

Step 1: Recognize That Change Is Inevitable, and Plan for It

- The team must recognize that changing requirements for the system is inevitable and even necessary.

- Develop a plan for managing change that should include some allowance for change in the initial baseline.

Step 2: Baseline the Requirements

- In each iteration, the team should baseline the requirements for the build
  - Putting version control on vision document, software requirements and use-case models
  - Publishing it for the development team
- Once the baseline has been established, new requirements can be more easily identified and managed.
  - A request for a new requirement can be compared against the existing baseline to see where it will fit in and whether it will create a conflict with any other requirements.

Step 2: Baseline the Requirements

- If the change is accepted, we can manage the evolution of that change from the vision to the software requirements, from the software requirements to the appropriate technical design documents and models, and then to the code and the test procedures.

Step 3: Establish a Single Channel to Control Change

- Ad hoc changes to a software system can cause significant and unintended consequences
- *Every change should go through a single channel* to determine its impact on the system and to make the official decision as to whether the change is going to be made in the system or not.
  - For small projects, the official channel can be one person – e.g., the project manager
  - For larger systems, the channel should consist of a few people who share the responsibility and make the decision together: Change Control Board (CCB)

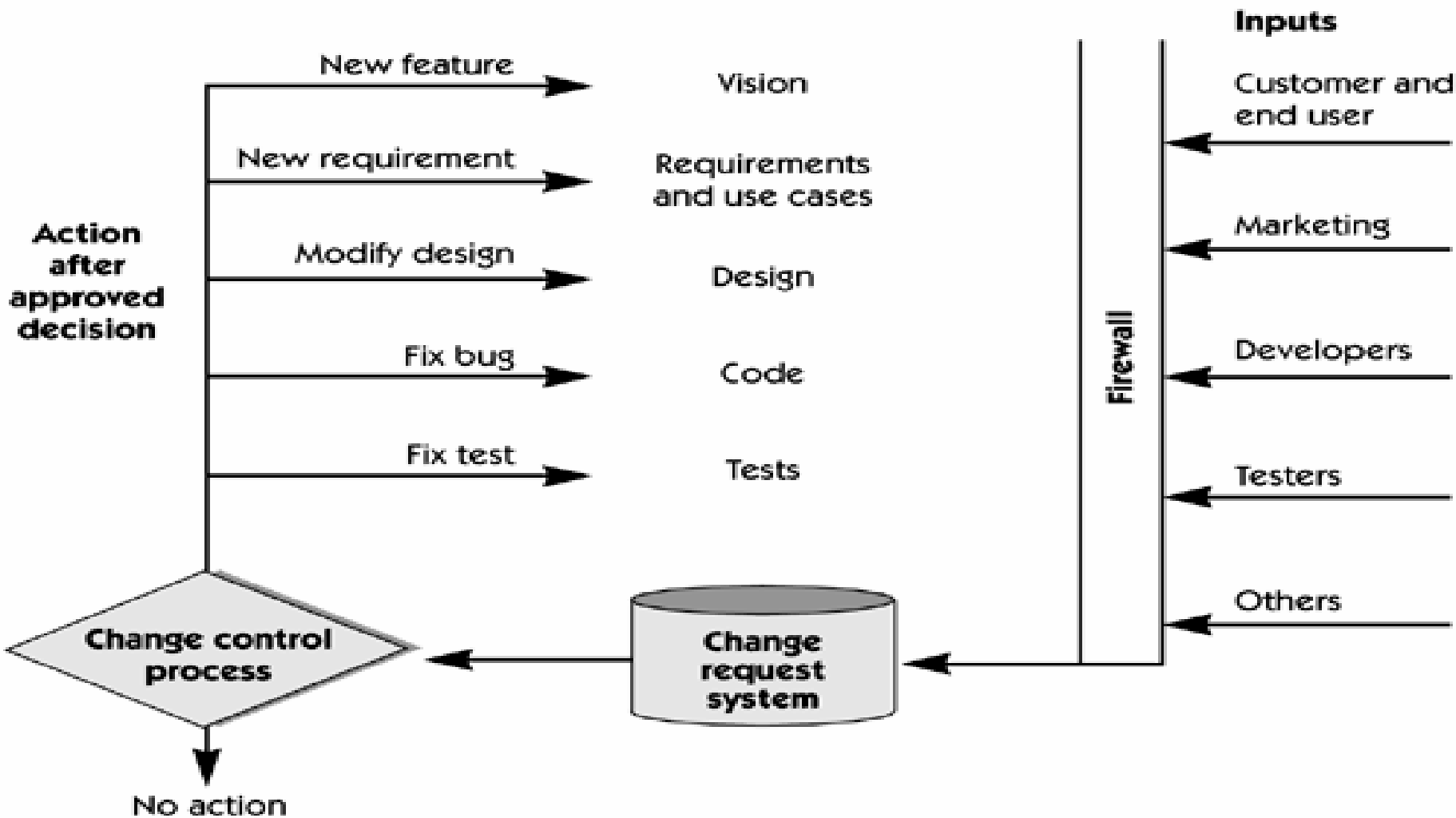
Step 4: Use a Change Control System to Capture Changes

- The team should implement a system for capturing all change requests.
- When considering whether to approve a change request, the CCB must consider the following factors:
  - The impact of the change on the cost and functionality of the system
  - The impact of the change on customers and other external stakeholders not well represented on the CCB: other project contractors, component suppliers, and so on
  - The potential for the change to destabilize the system

Step 4: Use a Change Control System to Capture Changes

- Once a change has been approved, the next step is to decide where to insert the change.
  - For example, we need to determine whether to change a requirement or to change a test being proposed.
  - Subsequent changes will ripple through in the hierarchy

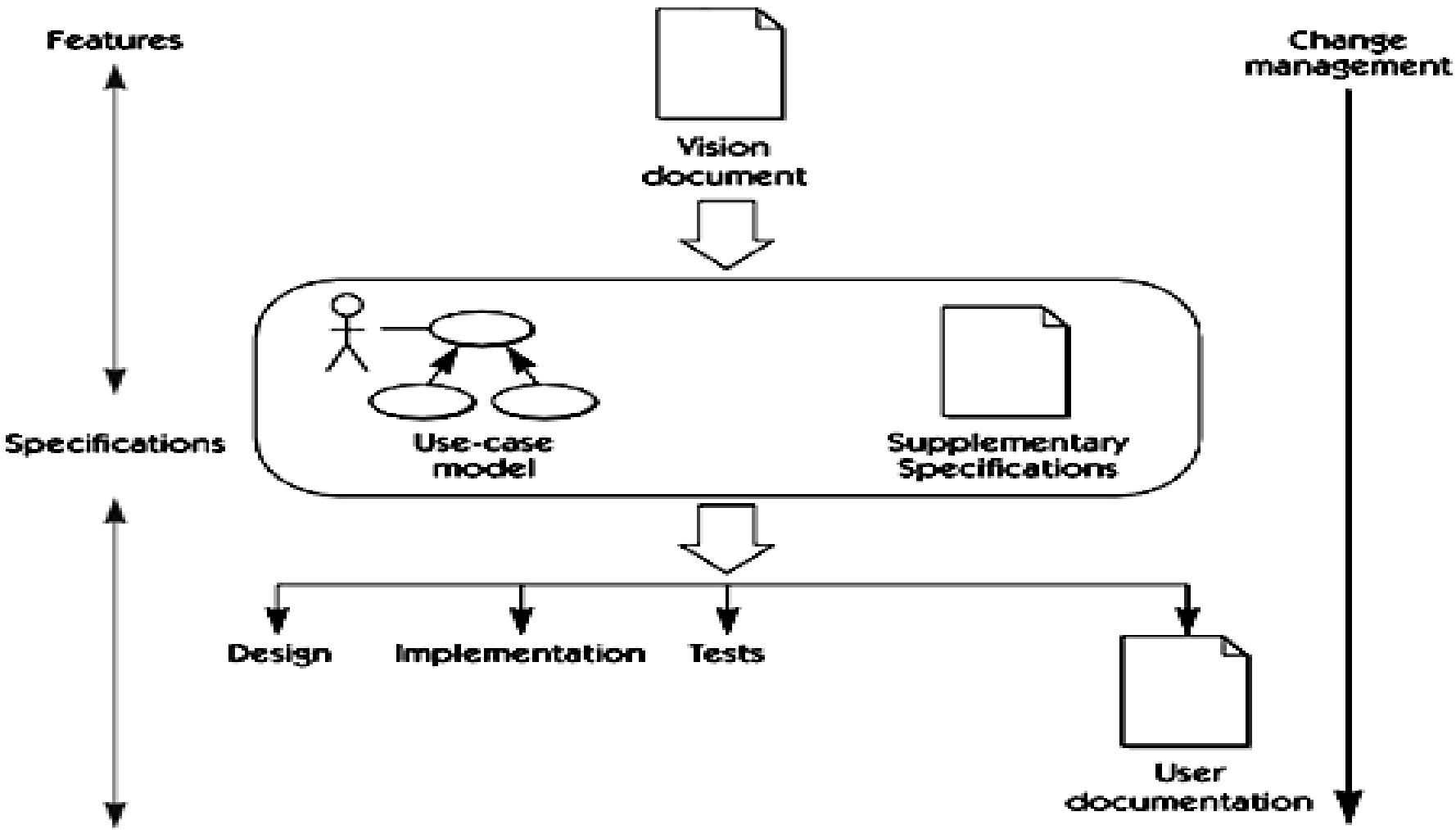
Change Request Flow



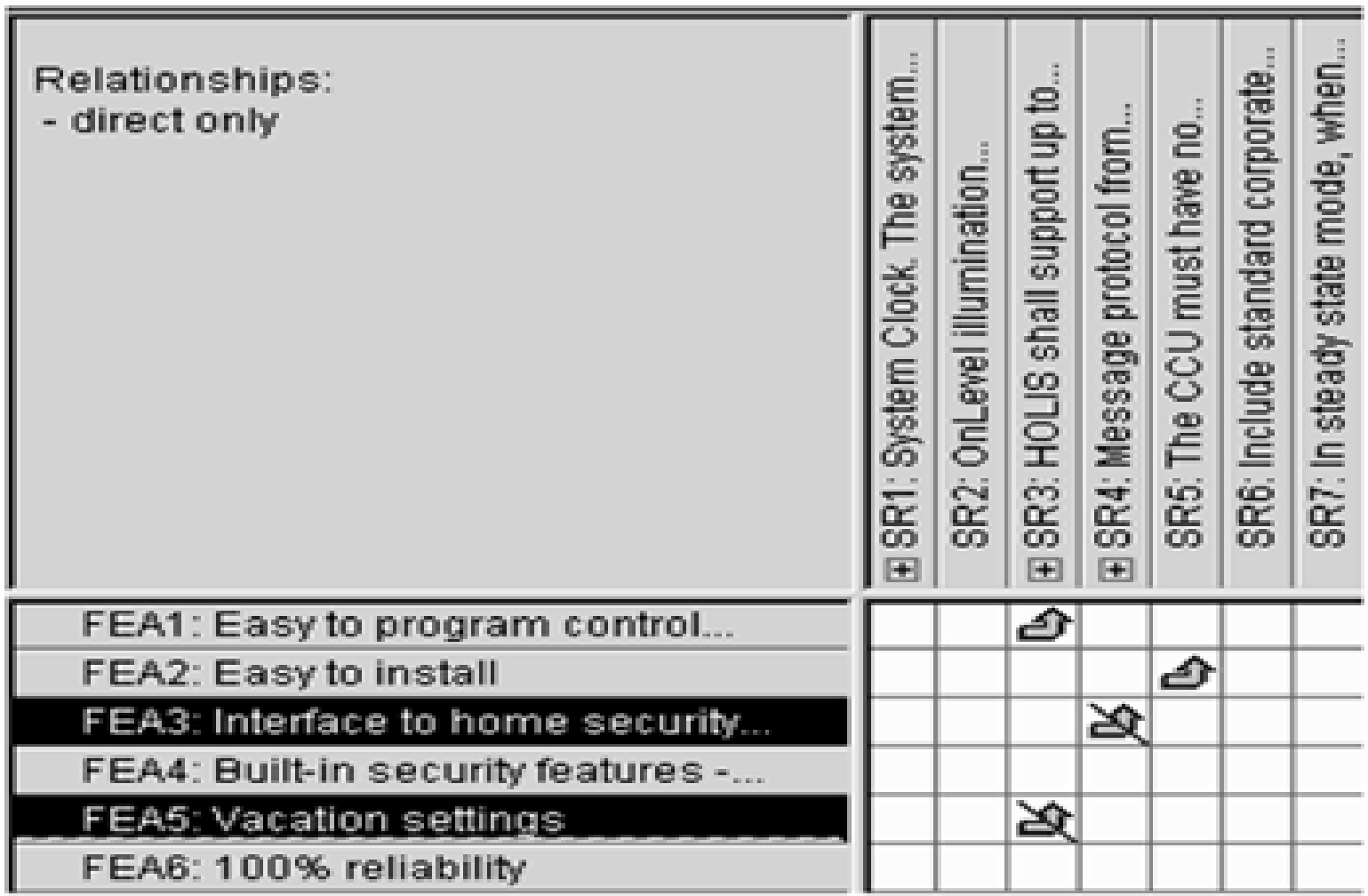
Step 5: Manage Change Hierarchically

- A change to one requirement can have a ripple effect in other related requirements, the design, or other subsystems
  - This fact may not be obvious to the requester, who casually asks the programmer to make a "quick and easy" change to the system.
- If the requirements is well structured and the system is well encapsulated, changes should be limited to certain areas instead of being spread everywhere.
- A programmer doesn't have the authority to introduce new features and requirements directly into the code on the user's behalf.

Hierarchical Ripple Effect



Impact Analysis by Traceability Link



Requirements Configuration Management

- Some elements of the preceding change review and approval process are referred to as change control, version control, or configuration management in some organizations.

Requirements Configuration Management

- The benefits of requirements configuration management are:
  - Prevents unauthorized and potentially destructive changes to the requirements
  - Preserves the revisions to requirements documents
  - Facilitates the retrieval and/or reconstruction of previous versions of documents
  - Supports a managed, organized baseline "release strategy" for incremental improvements or updates to a system
  - Prevents simultaneous update of documents or conflicting and uncoordinated updates to different documents at the same time.

Key Points

- A process to manage requirements can be useful only if it recognizes and addresses the issue of change.
- Internal change factors include failing to ask the right people the right questions at the right time and failing to create a practical process to help manage changes to requirements.
- In order to have a reasonable probability of success, requirements leakage must be stopped or at least reduced to a manageable level.