Chapter 4 – Requirements Engineering

Lecture 3

California State University, Fresno – Department of Computer Science

Req. Elicitation and Analysis Process

- Req. Discovery
  - Interact w/ stakeholders to discovery req.
- Req. Classification and Organization
  - Collect, group and organize req.
- Req. prioritization and negotiation
  - Req. from different stakeholders may conflict. Differences need to be resolved and agreement on compromised prioritized req. is required.
- Req. Spec.
  - Document req. as spec.

4.5.1 Requirements discovery

- The process of gathering information about the required and existing systems and distilling the user and system requirements from this information.
- Interaction is with system stakeholders from managers to external regulators.
- **Sources of info include documentations, stakeholders, specifications of similar software**
- Systems normally have a range of stakeholders.

Stakeholders in the MHC-PMS

- ✧ Patients whose information is recorded in the system.
- ✧ Doctors who are responsible for assessing and treating patients.
- ✧ Nurses who coordinate the consultations with doctors and administer some treatments.
- ✧ Medical receptionists who manage patients' appointments.
- ✧ IT staff who are responsible for installing and maintaining the system.

Stakeholders in the MHC-PMS

- A medical ethics manager who must ensure that the system meets current ethical guidelines for patient care.
- Health care managers who obtain management information from the system.
- Medical records staff who are responsible for ensuring that system information can be maintained and preserved, and that record keeping procedures have been properly implemented.
- Requirements may be also elicited from domain experts or *other **systems** interacting with the system being specified.*
- *In summary: 3 resources of requirement: stakeholders, domain experts, system (experts).*

Interviewing

- Formal or informal interviews with stakeholders are part of most RE processes.
- Types of interview
  - Closed interviews based on pre-determined list of questions
  - Open interviews where various issues are explored with stakeholders.
- Effective interviewing
  - Be open-minded, avoid pre-conceived ideas about the requirements and are willing to listen to stakeholders.
  - Prompt the interviewee to get discussions going using a springboard question, a requirements proposal, or by working together on a prototype system.

Interviews in practice

- Normally a mix of closed and open-ended interviewing.
- Interviews are good for getting an *overall understanding* of what stakeholders do and how they might interact with the system.
- **Interviews are not good for understanding domain/application requirements**
  - Requirements engineers cannot understand specific domain terminology;
  - Some domain knowledge is so familiar that people find it hard to articulate or think that it isn't worth articulating.
- Interviews are not good for understanding **organizational** req. and constraints.
  - Because of subtle **power** relationships between the different people in the org.

Scenarios

- Scenarios are real-life examples of how a system can be used.
- They should include
  - A description of the starting situation;
  - A description of the normal flow of events;
  - A description of what can go wrong;
  - Information about other concurrent activities;
  - A description of the state when the scenario finishes.

Scenario for collecting medical history in MHC-PMS

Initial assumption: The patient has seen a medical receptionist who has created a record in the system and collected the patient's personal information (name, address, age, etc.). A nurse is logged on to the system and is collecting medical history.

Normal: The nurse searches for the patient by family name. If there is more than one patient with the same surname, the given name (first name in English) and date of birth are used to identify the patient.

The nurse chooses the menu option to add medical history.

The nurse then follows a series of prompts from the system to enter information about consultations elsewhere on mental health problems (free text input), existing medical conditions (nurse selects conditions from menu), medication currently taken (selected from menu), allergies (free text), and home life (form).

Scenario for collecting medical history in MHC-PMS

What can go wrong: The patient's record does not exist or cannot be found. The nurse should create a new record and record personal information.

Patient conditions or medication are not entered in the menu. The nurse should choose the 'other' option and enter free text describing the condition/medication.

Patient cannot/will not provide information on medical history. The nurse should enter free text recording the patient's inability/unwillingness to provide information. The system should print the standard exclusion form stating that the lack of information may mean that treatment will be limited or delayed. This should be signed and handed to the patient.

Other activities: Record may be consulted but not edited by other staff while information is being entered.

System state on completion: User is logged on. The patient record including medical history is entered in the database, a record is added to the system log showing the start and end time of the session and the nurse involved.

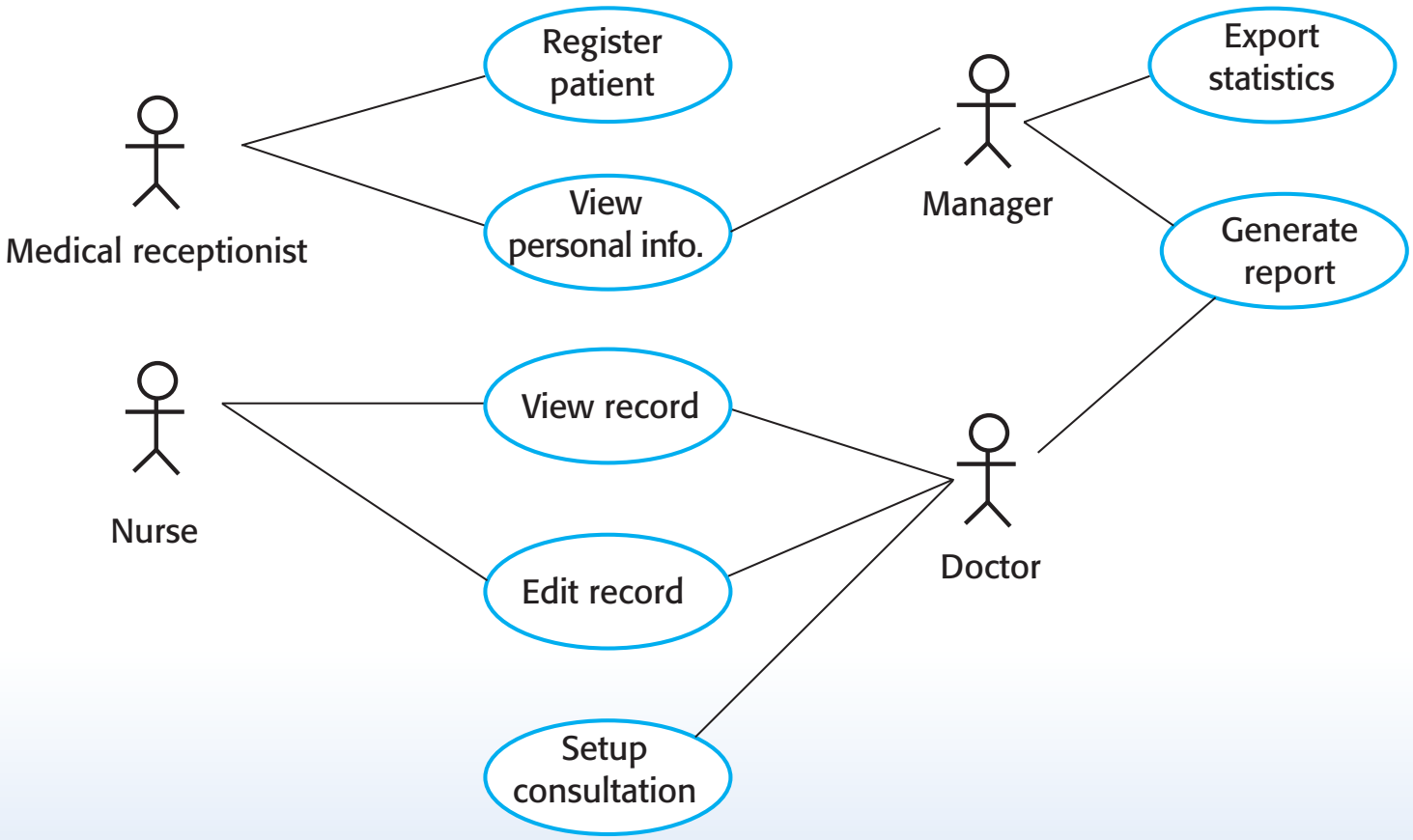
Use cases

- Use-cases are a scenario based technique in the UML which identify the actors in an interaction and which describe the interaction itself.
- A set of use cases should describe all possible interactions with the system.
- High-level graphical model supplemented by more detailed tabular description (see Chapter 5).
- Sequence diagrams may be used to add detail to use-cases by showing the sequence of event processing in the system (but at high level in req. Usually sequence diagrams are used in design to describe interactions among objects).

Use cases

- No hard and fast distinction between scenarios and use cases.
  - A use case may be considered as a single scenario
  - A set of scenarios can be combined to a use case (a normal scenario plus exceptional scenarios)
- Textual description is needed for each use case
  - Brief description
  - Step-by-step description
- **Not efficient** on eliciting constraints or high-level business and non-functional requirements

Figure 4.15 Use cases for the MHC-PMS



Setup Consultation Use Case Description

- Setup consultation allows two or more doctors, working in different offices, to view the same record at the same time. One doctor initiates the consultation by choosing the people involved from a drop-down menu of doctors who are online. The patient record is then displayed on their screens but only the initiating doctor can edit the record. In addition, a text chat window is created to help coordinate actions. It is assumed that a phone conference for voice communication will be separately set up.

Ethnography

- Ethnography is an observational tech. that can be used to understand operational processes and help derive support req. for these processes.
- It helps discover implicit system req. that reflect the **actual** ways that people work, rather than the formal processes defined by the org.
- People do not have to explain or articulate their work.
- Social and organisational factors of importance may be observed.
- Ethnographic studies have shown that work is usually richer and more complex than suggested by simple system models.

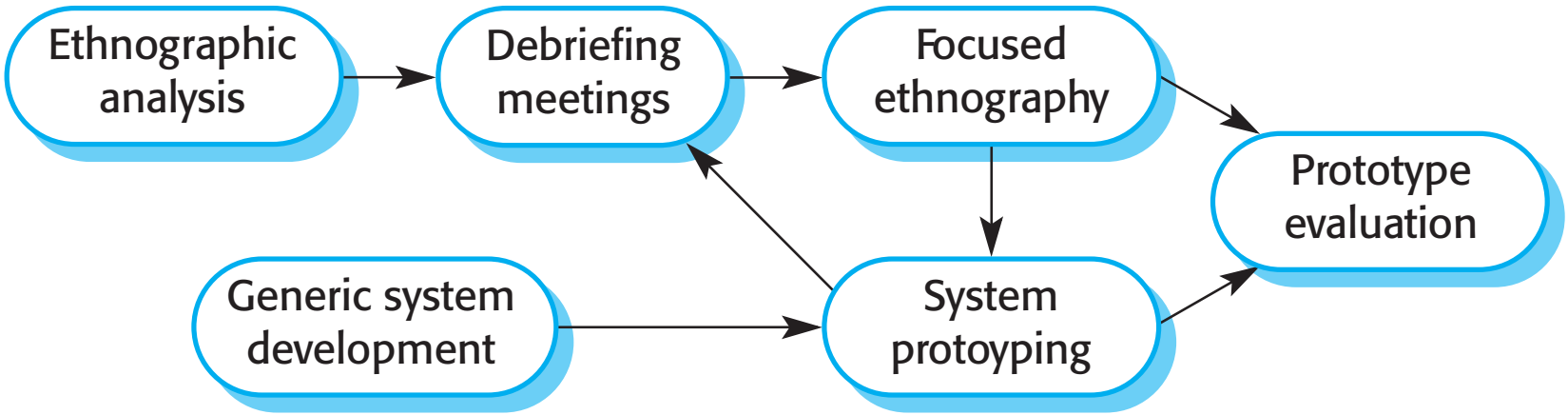
Scope of ethnography

- Requirements that are derived from the way that people **actually work** rather than the way which process definitions suggest that they ought to work.
- Requirements that are derived from cooperation and awareness of other people's activities.
  - Awareness of what other people are doing leads to changes in the ways in which we do things.
- Ethnography is effective for understanding **existing** processes but cannot identify **new** features that should be added to a system.
- Also, it focuses on people, so not suitable to discover organizational requirements or domain requirements.

Focused ethnography

- Developed in a project studying the air traffic control process
- Combines ethnography with prototyping
  - Reduce # of prototyping refinement
- Prototyping focuses the ethnography by identifying problems and questions that can then be discussed with the ethnographer.
- The problem with ethnography is that it studies existing practices which may have some historical basis which is no longer relevant.

Figure 4.16 Ethnography and prototyping for requirements analysis



Requirements validation

- ✧ Concerned with demonstrating that the requirements define the system that the customer **really wants**.
- ✧ Requirements error costs are high so validation is very important
  - Fixing a requirements error after delivery may cost up to 100 times the cost of fixing an implementation error.
  - Why?

Requirements checking

- Validity. Does the system provide the functions which best support the customer's needs?
- Consistency. Are there any requirements conflicts?
- Completeness. Are all functions required by the customer included?
- Realism. Can the requirements be implemented given available budget and technology
- Verifiability. Can the requirements be checked?

Requirements validation techniques

- ✧ Requirements reviews
  - Systematic manual analysis of the requirements.
- ✧ Prototyping
  - Using an executable model of the system to check requirements. Covered in Chapter 2.
- ✧ Test-case generation
  - Developing tests for requirements to check testability.

Requirements reviews

- Regular reviews should be held while the requirements definition is being formulated.
- Both client and contractor staff should be involved in reviews.
- Reviews may be formal (with completed documents) or informal. Good communications between developers, customers and users can resolve problems at an early stage.

Review checks

- Verifiability
  - Is the requirement realistically testable?
- Comprehensibility
  - Is the requirement properly understood?
- Traceability
  - Is the origin of the requirement clearly stated?
- Adaptability
  - Can the requirement be changed without a large impact on other requirements?

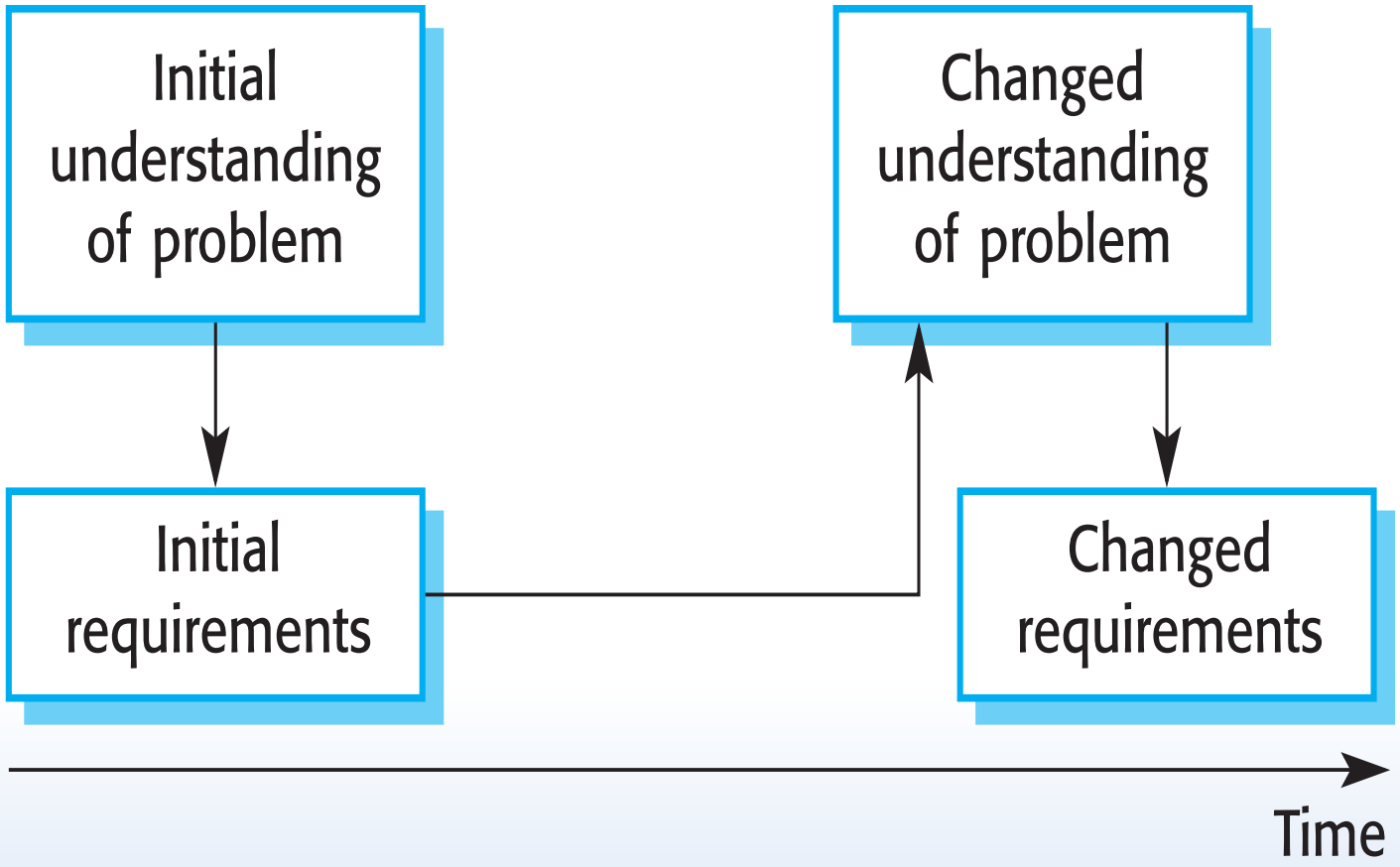
4.7 Requirements management planning: Changing requirements

- The business and technical environment of the system always changes after installation.
  - New hardware may be introduced, it may be necessary to interface the system with other systems, business priorities may change (with consequent changes in the system support required), and new legislation and regulations may be introduced that the system must necessarily abide by.
- The people who pay for a system and the users of that system are rarely the same people.
  - System customers impose requirements because of organizational and budgetary constraints. These may conflict with end-user requirements and, after delivery, new features may have to be added for user support if the system is to meet its goals.

Changing requirements

- Large systems usually have a diverse user community, with many users having different requirements and priorities that may be conflicting or contradictory.
  - The final system requirements are inevitably a compromise between them and, with experience, it is often discovered that the balance of support given to different users has to be changed.

Figure 4.17 Requirements evolution



Requirements management

- Requirements management is the process of managing changing requirements during the requirements engineering process and system development.
- New requirements emerge as a system is being developed and after it has gone into use.
- You need to ***keep track of individual requirements and maintain links between dependent requirements so that you can assess the impact of requirements changes.*** You need to establish a formal process for making change proposals and linking these to system requirements.

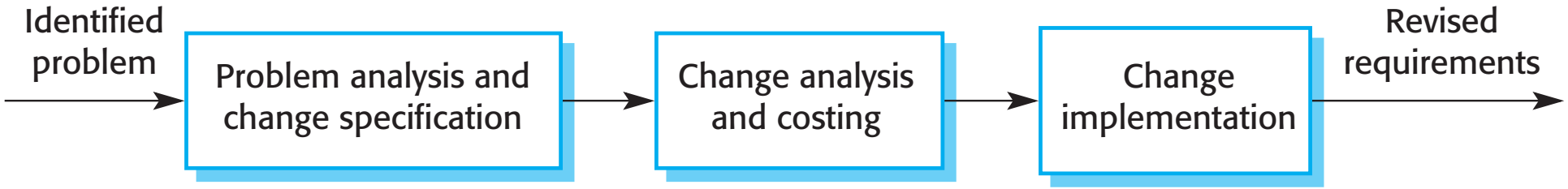
Requirements management planning

- Establishes the level of requirements management detail that is required.
- Requirements management decisions:
  - *Requirements identification* Each requirement must be uniquely identified so that it can be cross-referenced with other requirements.
  - *A change management process* This is the set of activities that **assess** the impact and cost of changes. Will discuss this process in more detail in the following section.
  - *Traceability policies* These policies define the relationships between each requirement and between the requirements and the system design that should be recorded.
  - *Tool support* Tools that may be used range from specialist requirements management systems to spreadsheets and simple database systems. (Req. Storage, Change Management, Traceability Management)
    - https://www.capterra.com/requirements-management-software/

Requirements change management

- Deciding if a requirements change should be accepted
  - *Problem analysis and change specification*
    - During this stage, the problem or the change proposal is analyzed to check that it is valid. This analysis is fed back to the change requestor who may respond with a more specific requirements change proposal, or decide to withdraw the request.
  - *Change analysis and costing*
    - The effect of the proposed change is assessed using traceability information and general knowledge of the system requirements. Once this analysis is completed, a decision is made whether or not to proceed with the requirements change.
  - Change implementation
    - The requirements document and, where necessary, the system design and implementation, are modified. Ideally, the document should be organized so that changes can be easily implemented.

Figure 4.18 Requirements change management



Appendix

- Requirements validation using
  - Walkthrough
  - Inspection

Walkthroughs

- A walkthrough team consists of from four to six members
- It includes representatives of
  - The team responsible for the current workflow (manager + a team representative)
  - The team responsible for the next workflow
  - Client representative
  - The SQA group (chair)
- The walkthrough is preceded by **preparation**
  - Lists of items
    - Items not understood
    - Items that appear to be incorrect

Walkthrough

- The walkthrough team is chaired by the SQA representative
- In a walkthrough we detect faults, not correct them
  - A correction produced by a committee is likely to be of low quality
  - The cost of a committee correction is too high
  - Not all items flagged are actually incorrect
  - A walkthrough should not last longer than 2 hours
  - There is no time to correct faults as well

Walkthrough

- document-driven vs. participant-driven
  - Document-driven is led by authors
  - Participant-driven let everyone list and discuss what potential errors are.
- Verbalization leads to fault finding
- A walkthrough should never be used for performance appraisal

Inspections

- An inspection has five **formal** steps
  - Overview by document authors (of req., spec., design, code, or plan)
  - Preparation, aided by statistics of fault types
    - Types and frequencies of faults are needed
  - Inspection
  - Rework
  - Follow-up
    - Ensure every issue has been resolved

Inspections

- Faults are recorded by severity
  - Example:
    - Major or minor
- Faults are recorded by fault type
  - Examples of design faults:
    - Not all specification items have been addressed
    - Actual and formal arguments do not correspond

Inspections

- For a given workflow, we compare current fault rates with those of previous products
- We take action if there are a disproportionate number of faults in an artifact
- Redesigning from scratch is a good alternative
- We carry forward fault statistics to the next workflow
- We may not detect all faults of a particular type in the current inspection

Inspections

- IBM inspections showed up
  - 82% of all detected faults (1976)
  - 70% of all detected faults (1978)
  - 93% of all detected faults (1986)
- Switching system
  - 90% decrease in the cost of detecting faults (1986)
- JPL
  - Four major faults, 14 minor faults per 2 hours (1990)
  - Savings of $25,000 *per inspection*
- The number of faults decreased exponentially by phase (1992)

Inspections

- Warning
  - Fault statistics should never be used for performance appraisal
  - “Killing the goose that lays the golden eggs”

Comparison of Inspections and Walkthroughs

- Walkthrough
  - Two-step, informal process
    - Preparation
    - Analysis
- Inspection
  - Five-step, formal process
    - Overview
    - Preparation
    - Inspection
    - Rework
    - Follow-up

Strengths and Weaknesses of Reviews

- Reviews can be effective
  - Faults are detected early in the process
- Reviews are less effective if the process is inadequate
  - Large-scale software should consist of smaller, largely independent pieces
  - The documentation of the previous workflows has to be complete and available online

Metrics for Inspections

- Inspection rate (e.g., design pages inspected per hour)
- Fault density (e.g., faults per KLOC inspected)
- Fault detection rate (e.g., faults detected per hour)
- Fault detection efficiency (e.g., number of major, minor faults detected per hour)

- Does a 50% increase in the fault detection rate mean that
  - Quality has decreased? Or
  - The inspection process is more efficient?

Key points

- ✧ You can use a range of techniques for requirements elicitation including interviews, scenarios, use-cases and ethnography.
- ✧ Requirements validation is the process of checking the requirements for validity, consistency, completeness, realism and verifiability.
- ✧ Business, organizational and technical changes inevitably lead to changes to the requirements for a software system. Requirements management is the process of managing and controlling these changes.