7 Requirements Elicitation

Contents

- Requirements elicitation techniques
- Planning elicitation on your project
- Preparing for elicitation
- Performing elicitation activities
- Following up after elicitation
- Classifying customer input
- How do you know when you’re done?
- Some cautions about elicitation
- Assumed and implied requirements
- Finding missing requirements

Guide

- The heart of requirements development is *elicitation.*
  - Elicit: to succeed in getting information or a reaction from someone, especially when this is difficult.
    by *Longman Dictionary of Contemporary English*

- Elicitation is a collaborative and analytical process that includes activities to collect, discover, extract, and define requirements.
- The output of requirements development is a common understanding of the needs held by the diverse project stakeholders.
- Elicitation participants should resist the temptation to design the system until they understand the problem.

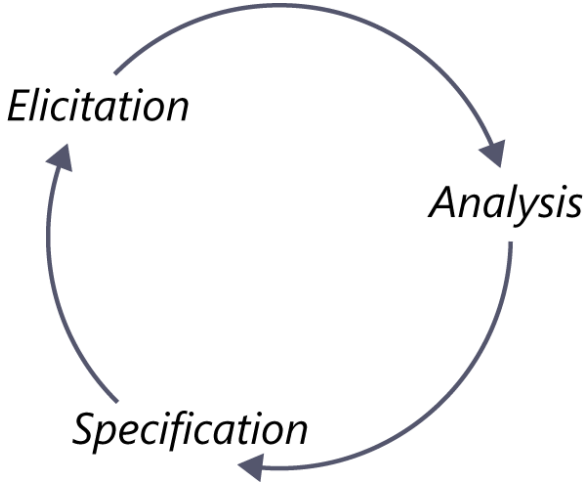


Fig 7-1 The cyclic nature of requirements elicitation, analysis, and specification.

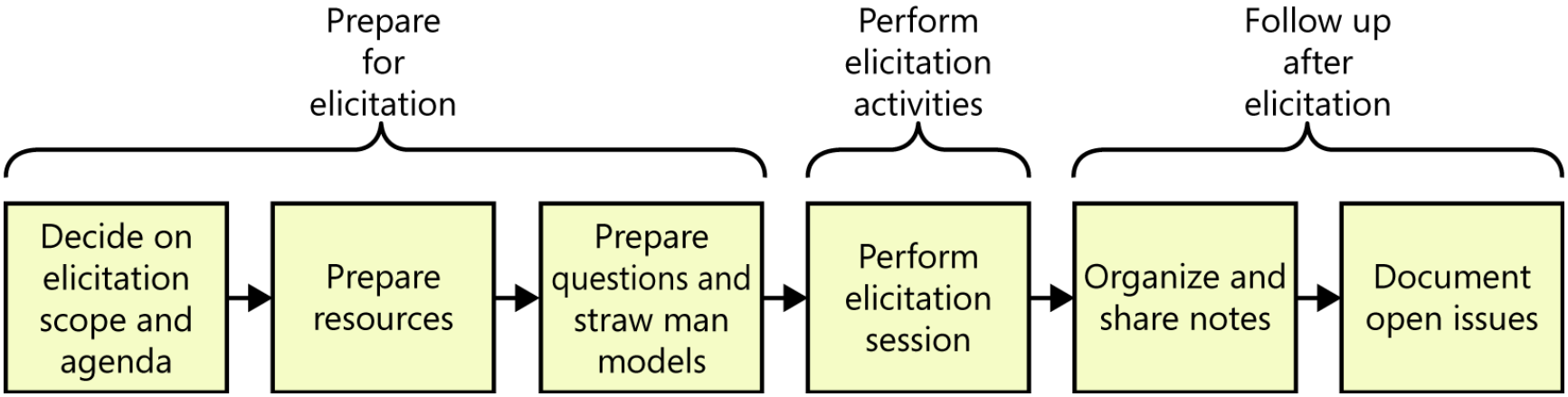


Fig 7-2 Activities for a single requirements elicitation session

1 Elicitation techniques

- Interviews
  - The most obvious way to find out what the users of a software system need is to ask them.
  - Establish friendly relationship
  - Stay in scope
  - Prepare questions and straw man models ahead of time
  - Suggest ideas
  - Listen actively

Elicitation techniques 2

- Workshops
  - *a structured meeting* in which a carefully selected group of stakeholders and content experts work together to define, create, refine, and reach closure on deliverables (such as models and documents) that represent user requirements
  - Establish and enforce ground rules
  - Fill all of the team roles
  - Plan an agenda
  - Stay in scope
  - Use parking lots to capture items for later consideration
  - Timebox discussions
  - Keep the team small but include the right stakeholders
  - Keep everyone engaged

Elicitation techniques 3

- Focus groups
  - *A focus group* is a representative group of users who convene in a facilitated elicitation activity to generate input and ideas on a product's functional and quality requirements.
  - Focus group sessions must be interactive, allowing all users a chance to voice their thoughts.
  - Focus groups are useful for exploring users' attitudes, impressions, preferences, and needs.

Elicitation techniques 4

- Observations
  - Sometimes you can learn a lot by observing exactly how users perform their tasks.
  - Observations are time consuming, not suitable for every user or every task.
  - Observations can be silent or interactive.

Elicitation techniques 5

- Questionnaires
  - Questionnaires are a way to survey large groups of users to understand their needs.
  - Inexpensive, easily across geographical boundaries.
  - Results of questionnaires can be used as an input to other elicitation techniques, like workshops.
  - Preparing well-written questions is the biggest challenge with questionnaires.

Elicitation techniques 6

- System interface analysis
  - Interface analysis is an independent elicitation technique that entails examining the systems to which your system connects.
  - System interface analysis reveals functional requirements regarding the exchange of data and services between systems.
  - Context diagrams and ecosystem maps are an obvious choice to begin finding interfaces for further study.

Elicitation techniques 7

- User interface analysis
  - User interface (UI) analysis is an independent elicitation technique in which you study existing systems to discover user and functional requirements.
  - screen shots

Elicitation techniques 8

- Document analysis
  - Document analysis entails examining any existing documentation for potential software requirements.
  - requirements specifications
  - business processes
  - lessons-learned collections
  - user manuals for existing or similar applications
  - corporate or industry standards
  - regulations

2 Planning elicitation

- Elicitation objectives
- Elicitation strategy and planned techniques
- Schedule and resource estimates
- Documents and systems needed for independent elicitation
- Expected products of elicitation efforts
- Elicitation risks

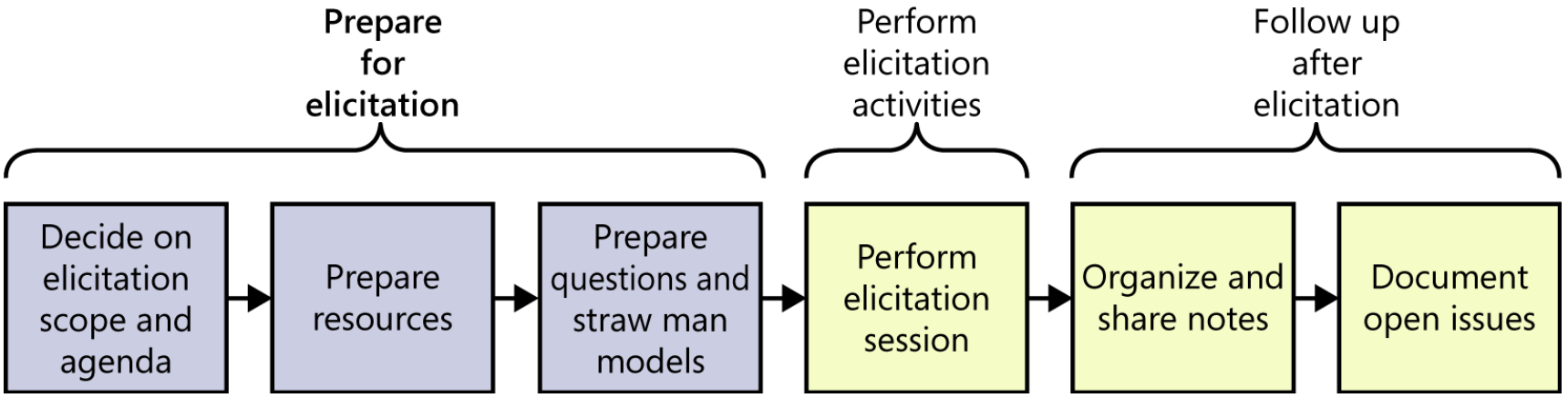
Combination of techniques

- Many BAs like interviews and workshops, and do not think to use other techniques that might reduce resource needs or increase the quality of the information discovered.
- Rarely will a BA get the best results by using only one elicitation technique on a project.
- Elicitation techniques apply across the spectrum of development approaches.
- The selection of elicitation techniques should be based on the characteristics of the project.

	Interviews	Workshops	Focus groups	Observations	Questionnaires	System interface analysis	User interface analysis	Document analysis
Mass-market software	x		x		x			
Internal corporate software	x	x	x	x		x		x
Replacing existing system	x	x		x		x	x	x
Enhancing existing system	x	x				x	x	x
New application	x	x				x		
Packaged software implementation	x	x		x		x		x
Embedded systems	x	x				x		x
Geographically distributed stakeholders	x	x			x			

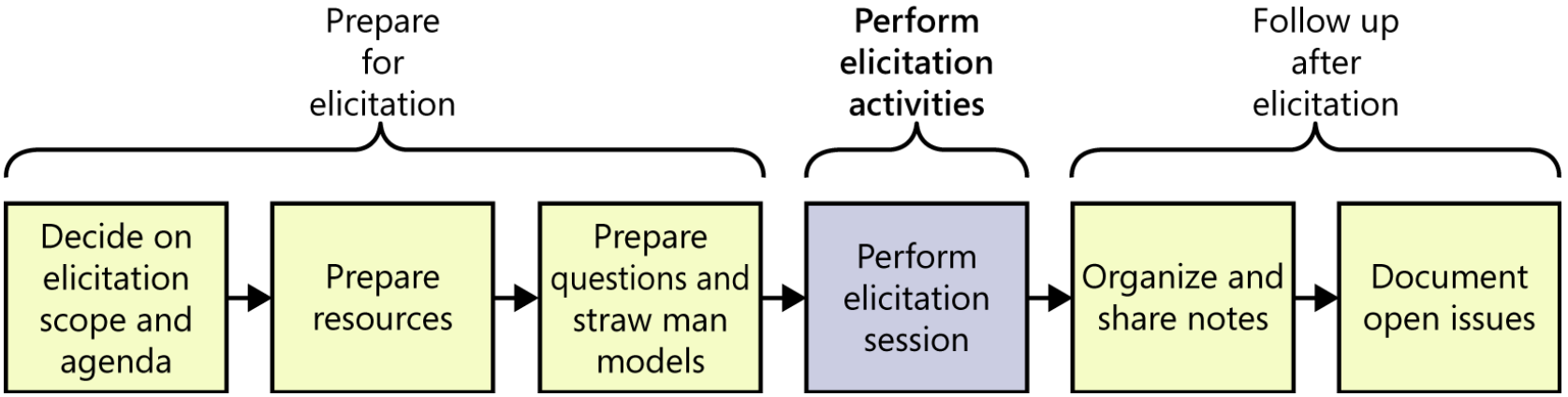
Fig 7-3 Suggested elicitation techniques by project characteristic

3 Preparing for elicitation



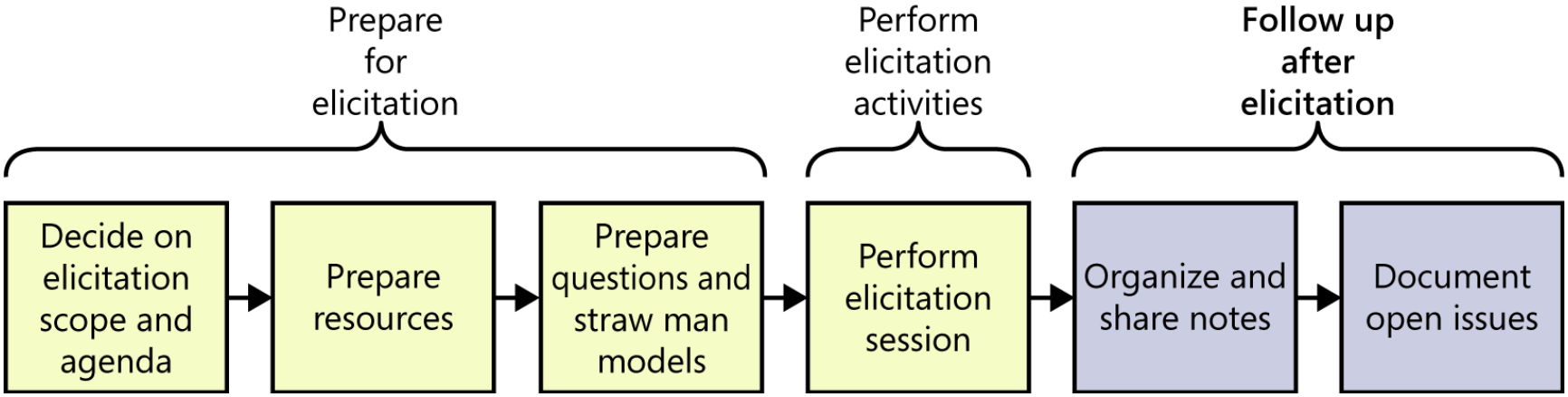
- Plan session scope and agenda
- Prepare resources, like rooms, projectors, whiteboards
- Learn about the stakeholders
- Prepare questions
- Prepare straw man models, eg. use cases, process flows

4 Performing elicitation activities



- Educate stakeholders
- Take good notes
  - Assign someone for taking accurate notes
- Exploit the physical space

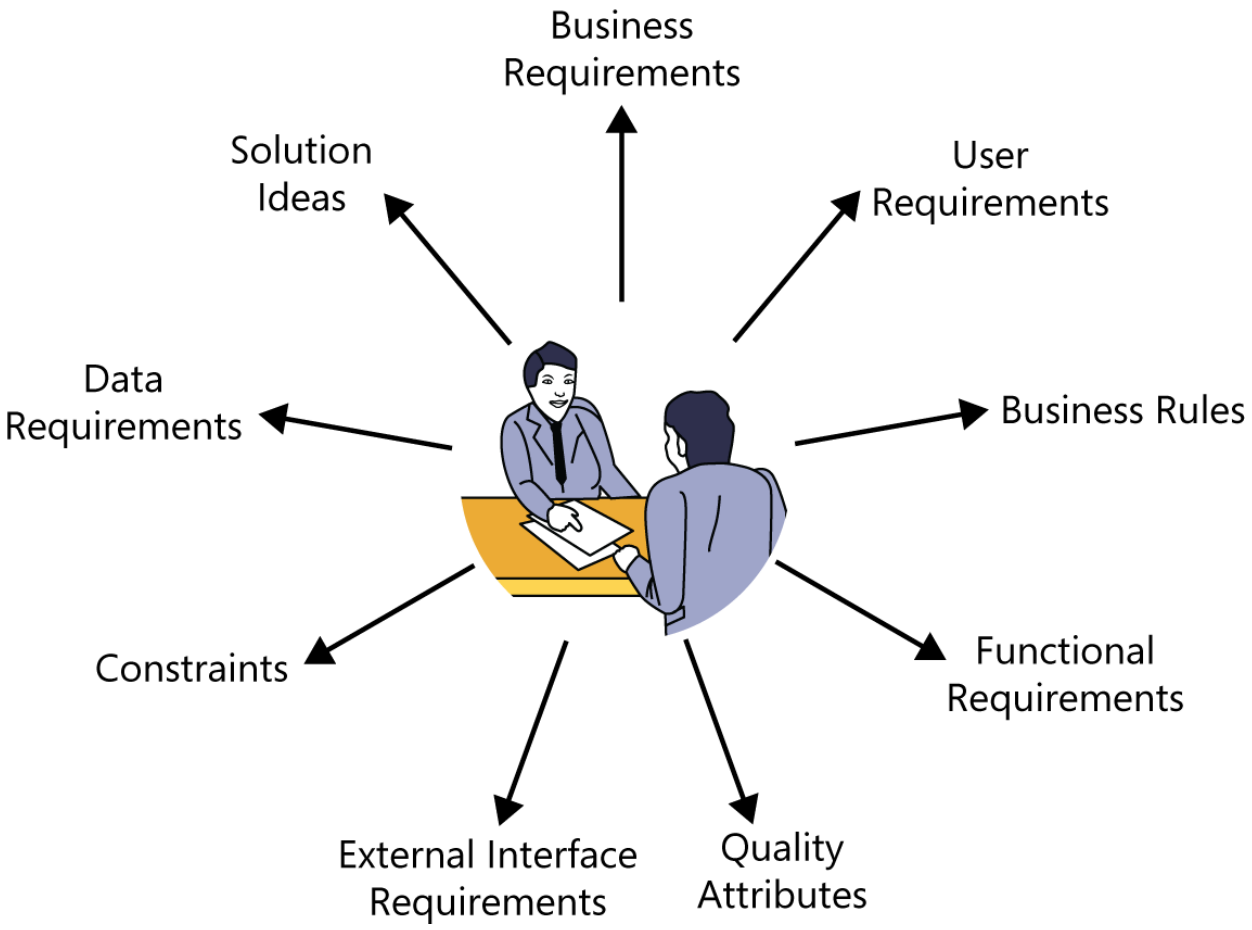
5 Following up after elicitation



- Consolidate your input from multiple sources.
- Review and update your notes soon after the session is complete, while the content is still fresh in your mind.
- Share the consolidated notes with the participants and ask them to review them to ensure that they accurately represent the session. Early review is essential to successful requirements development.

- Documenting open issues.

6 Classifying customer input



Classifying customer input 2

- **Business requirements** Anything that describes the financial, marketplace, or other business benefit that either customers or the developing organization wish to gain from the product is a business requirement. (Ch 5)
  - Increase market share in region X by Y percent within Z months
  - Save $X per year on electricity now wasted by inefficient units

- **User requirements** General statements of user goals or business tasks that users need to perform are user requirements, most typically represented as use cases, scenarios, or user stories. (Ch 8)
  - I need to print a mailing label for a package.
  - As the lead machine operator, I need to calibrate the pump controller first thing every morning.

Classifying customer input 3

- **Business rules** When a customer says that only certain users can perform an activity under specific conditions, he might be presenting a business rule. (Ch 9)
  - A new client must pay 30 percent of the estimated consulting fee and travel expenses in advance.
  - Time-off approvals must comply with the company's HR vacation policy.

- **Functional requirements** Functional requirements describe the observable behaviors the system will exhibit under certain conditions and the actions the system will let users take. (Ch 11)
  - If the pressure exceeds 40.0 psi, the high-pressure warning light should come on.
  - The user must be able to sort the project list in forward and reverse alphabetical order.

Classifying customer input 4

- **Quality attributes** Statements that describe how well the system does something are quality attributes. (Ch 14)
  - The mobile software must respond quickly to touch commands.
  - The shopping cart mechanism has to be simple to use so my new customers don't abandon the purchase.

- **External interface requirements** Requirements in this category describe the connections between your system and the rest of the universe. (Ch 10)
  - The manufacturing execution system must control the wafer sorter.
  - The mobile app should send the check image to the bank after I photograph the check I'm depositing.

Classifying customer input 5

- **Constraints** Design and implementation constraints legitimately restrict the options available to the developer. (Ch 14)
  - Files submitted electronically cannot exceed 10 MB in size.
  - The browser must use 256-bit encryption for all secure transactions.

- **Data requirements** Customers are presenting a data requirement whenever they describe the format, data type, allowed values, or default value for a data element; the composition of a complex business data structure; or a report to be generated. (Ch 13)
  - The ZIP code has five digits, followed by an optional hyphen and four digits that default to 0000."
  - An order consists of the customer's identity, shipping information, and one or more products, each of which includes the product number, number of units, unit price, and total price.

Classifying customer input 6

- **Solution ideas** Many “requirements” from users are really solution ideas. Someone who describes a specific way to interact with the system to perform some action is suggesting a solution.
  - Then I select the state where I want to send the package from *a drop-down list*.
  - The phone has to allow the user to swipe with a finger to navigate between screens.

7 When you're done?

- No simple signal tell that you are done. You'll never be entirely done. ☹ However …

- The users can't think of any more use cases or user stories. Users tend to identify user requirements in sequence of decreasing importance.
- Users propose new scenarios, but they don't lead to any new functional requirements. A "new" use case might really be an alternative flow for a use case you've already captured.
- Users repeat issues they already covered in previous discussions.
- Suggested new features, user requirements, or functional requirements are all deemed to be out of scope.
- Proposed new requirements are all low priority.
- The users are proposing capabilities that might be included "sometime in the lifetime of the product" rather than "in the specific product we're talking about right now."
- Developers and testers who review the requirements for an area raise few questions.

8 Some cautions about elicitation

- **Balance stakeholder representation** Collecting input from too few representatives or hearing the voice of only the loudest, most opinionated customer is a problem.
- **Define scope appropriately** During requirements elicitation, you might find that the project scope is improperly defined, being either too large or too small.
- **Avoid the requirements-versus-design argument** It's often stated that requirements are about *what* the system has to do, whereas *how* the solution will be implemented is the realm of design. Requirements elicitation should indeed focus on the *what*, but there's a gray area, not a sharp line, between analysis and design.
- **Research within reason** The need to do exploratory research sometimes disrupts elicitation.

9 Assumed and implied requirements

- *Assumed requirements* are those that people expect without having explicitly expressed them.
  - What you assume as being obvious might not be the same as assumptions that various developers make.

- *Implied requirements* are necessary because of another requirement but aren't explicitly stated.
  - Developers can't implement functionality they don't know about.

10 Finding missing requirements

- Decompose high-level requirements into enough detail to reveal exactly what is being requested.
- Ensure that all user classes have provided input.
- Trace system requirements, user requirements, event-response lists, and business rules to their corresponding functional requirements to make sure that all the necessary functionality was derived.
- Check boundary values for missing requirements.
- Represent requirements information in more than one way.
- Sets of requirements with complex Boolean logic (ANDs, ORs, and NOTs) often are incomplete.
- Create a checklist of common functional areas to consider for your projects.
- A data model may reveal missing functionality.