FPT UNIVERSITY

ĐẠI HỌC FPT CẦN THƠ

CHAPTER 7

Requirements elicitation

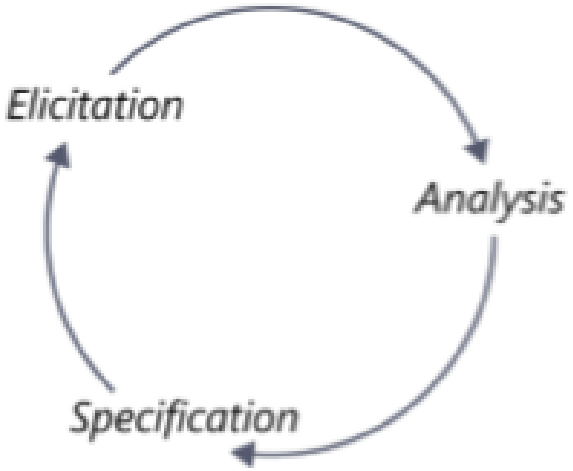
Objectives

- Student should understand the importance of requirements elicitation in requirement engineering.
- Student should enhance the list of requirements elicitation techniques and how to use them.
- Students should understand which techniques work well? Why? Which ones do not work so well? Why not?
- Which techniques that should use in agile projects, other projects.

Contents

- Requirements elicitation overview
- Requirements elicitation techniques
- Planning elicitation on your project
- Preparing for elicitation
- Performing elicitation activities
- Following up after elicitation
- Classifying customer input
- Some cautions about elicitation
- Finding missing requirement

Requirements elicitation overview



- *Elicitation is not the same as "gathering requirements."*
- *Elicitation is a collaborative and analytical process that includes activities to collect, discover, extract, and define requirements.*

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

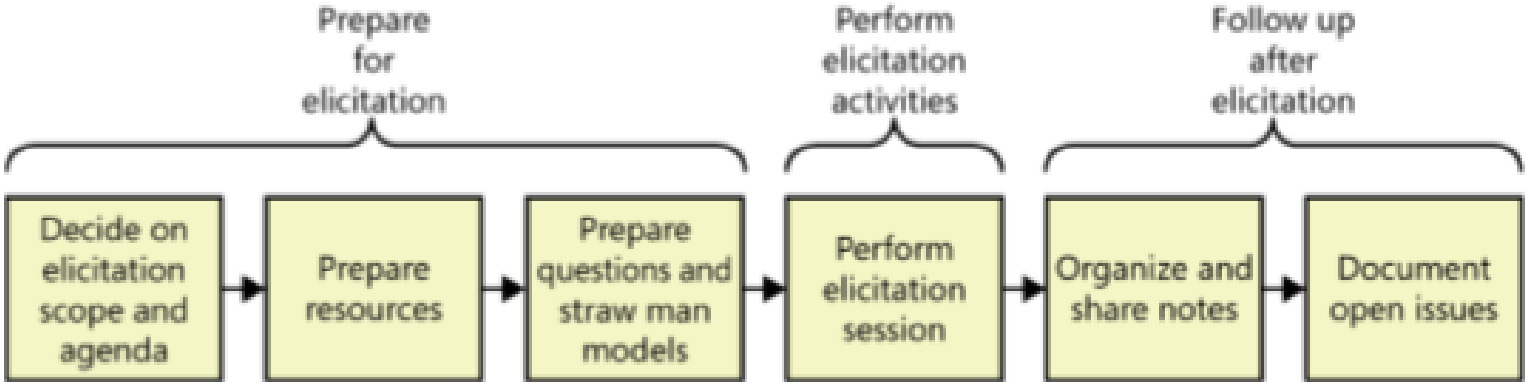


FIGURE 7-2 Activities for a single requirements elicitation session.

Requirements elicitation techniques: Interviews

- Interviews are a traditional source of requirements input for both commercial products and information systems, across all SW development approach. Some tips as follows:
  - Establish rapport: To begin an interview, introduce yourself if the attendees don't know you, review agenda, remind attendees of the session objectives,…
  - Stay in scope: Keep the discussion focused on its objectives.
  - Prepare questions and straw man models ahead of time: Prepare a list of questions to guide the conversation.
  - Suggest ideas: Rather than transcribing what customers say, a creative BA proposes ideas and alternatives during elicitation.
  - Listen actively: Practice the techniques of active listening and paraphrasing.

Requirements elicitation techniques: Workshops

- Workshops encourage stakeholder collaboration in defining requirements. Workshops often include several types of stakeholders, from users to developers to testers. Working in group is more effective for resolving disagreements than is talking to people individually.
- The facilitator plays critical role in planning the workshop, selecting participants, and guiding them to a successful outcome.
- Workshops can be resource intensive, sometimes requiring numerous participants for several days at a time. They must be well planned to avoid the waste time. Minimize wasted time by coming into a workshop with drafts of materials prepared ahead of time.
- Few tips for workshop:
  - Establish and enforce ground rules: On time; silencing electronic devices; holding one conversation at a time; expecting everyone to contribute;…

Requirements elicitation techniques: Workshops (con't)

- Few tips for workshop (con't):
  - Fill all of the team roles: Facilitator makes sure that the following tasks are covered by people in the workshop: note taking, time keeping, scope management, meeting minutes taking,…
  - Plan an agenda: Create the plan and workshop agenda ahead of time, and send it to participants.
  - Stay in scope: The facilitator will have to reel in the elicitation participants periodically to keep them on topic.
  - Use parking lots to capture items for later consideration: The important information during elicitation discussions: quality attributes, biz rules, user interface ideas,… Organize this information on flipcharts – parking lots – so you don't lose it and to demonstrate respect for the participant who bring it up.

Requirements elicitation techniques: Workshops (con't)

- Few tips for workshop (con't):
  - Timebox discussions: Allocating a fixed time to each discussion topic helps you to avoid the trap of spending far more time than intended on the first topic and neglecting other important topics.
  - Keep the team small but include the right stakeholders: Elicitation workshops with more than 5 or 6 participants can become mired in side trips, concurrent conversations, and bickering. Consider running multiple workshops in parallel to explore the requirements of different user classes.
  - Keep everyone engaged: Sometimes certain participants will stop contributing to the discussion with many reasons, such as, their ideas are not valued by others, or they don't want to disrupt the work that the group has completed so far…. The facilitator must read their body language to try to re-engage them.

Requirements elicitation techniques: Focus groups

- 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.
- Useful for commercial products and don't have ready access to end users within your company.

Requirements elicitation techniques: Observations

- Focus on task elicitation in the system-as-is.
- Understanding a task is often easier by observing people performing it (rather than verbal or textual explanation).
- Passive observation: no interference with task performers:
  - Watch from outside, record (notes, video), edit transcripts, interpret.
- Active observation: you get involved in the task, even become a team member and you can ask the questions to understand why they do that.

Requirements elicitation techniques: Questionnaires

- Questionnaires are a way to survey large groups of users to understand their needs.
- Preparing well-written questions is the biggest challenge with questionnaires. Some tips for you:
  - Provide answer questions that cover the full set of possible response.
  - Make answer choices both mutually exclusive and exhaustive (list all possible choices).
  - Don’t phrase a question in a way that implies a “correct” answer.
  - If you use scales, use them consistently throughout the questionnaire.
  - Used closed questions with 2 or more specific choices. Opened questions allows users to respond any way they want.
  - Consider consulting with an expert in questionnaire design to ensure the right questions to the right people.
  - Always test a questionnaire before distributing it.
  - Don’t ask too many questions.

System interface analysis

- What it is? Interface analysis is an independent elicitation technique used to examine the systems to which your system connects.
- Purpose: System interface analysis reveals functional requirements regarding the exchange of data and services between system.
- How to process: Context diagrams and ecosystem maps are an obvious choice to begin finding interfaces between systems.
  - For each system that interfaces with yours, identify functionality in the other system that might lead to requirements for your system. These requirements could describe what data to pass to other system, what data is received from it, and rules about that data, such as validation criteria.
  - You might also discover existing functionality that you do *not* need to implement in your system.

User interface analysis

- What it is? User interface (UI) is an independent technique in which you study existing systems to discover user and functional requirements.
- Purpose: Discover user and functional requirements.
- How to process: One way is to interact directly with the existing system. Another way is to screen shots. If there is no existing system, you might be able to look at user interface of similar product.
- When working with packaged solutions or an existing system, UI analysis can help you identify a complete list of screens to discover potential features.
- By navigating the existing UI, you can learn about common steps users take in the system and draft use cases to review with users.
- UI analysis can reveal pieces of data that users need to see.

Document analysis

- What it is? Document analysis is a technique used to examine any existing documentation for potential SW requirements.
- Purpose: Document analysis helps to reveal that might need to be retained, as well as obsolete functionality needed to replace.
- How to process: Document analysis is used to study requirements specifications, biz processes, lesson-learned collections, and user manuals for existing or similar applications.

Planning elicitation on your project

Early in a project, the BA should plan the project's approach to requirements elicitation. Your plan should address the following items:

- **Elicitation objectives:** for the entire project and for each planned elicitation activity.
- **Elicitation strategy and planned techniques:** Describe which techniques to use with different stakeholder groups. You might use some combination of questionnaires, workshops, customer visits, interviews, observations,…
- **Schedule and resource estimates:** Identify both customer and development participants for the various elicitation activities, along with estimates of the effort and calendar time required.
- **Documents and systems needed for independent elicitation:** All material needed have to document carefully.
- **Expected products of elicitation efforts:** Create a list of use cases, an SRS, an analysis of questionnaire results, or quality attribute specifications helps ensure that you target the right stakeholders, topics, and details during elicitation.
- **Elicitation risks:** identify factors that could impede your ability to complete the elicitation activities and plan to mitigate the impact of risks.

Preparing for elicitation

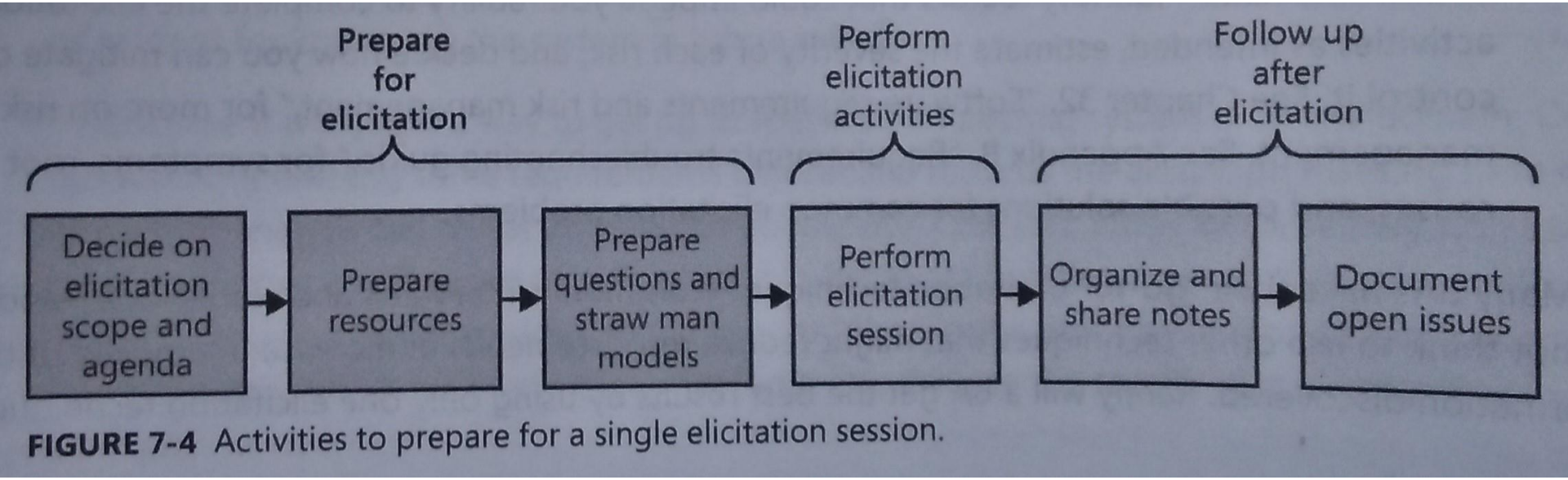


FIGURE 7-4 Activities to prepare for a single elicitation session.

- Plan session scope and agenda
- Prepare resources
- Learn about the stakeholders
- Prepare questions
- Prepare straw man models

Preparing for elicitation

- Plan session scope and agenda: Decide on the scope of the elicitation session, taking into account how much time is available. The agenda should detail what topics will be covered, the available time for each topic, and targeted objectives. Share the session agenda with stakeholders in advance.
- Prepare resources: Schedule the physical resources needed, such as rooms, projectors, teleconference numbers, and videoconferencing equipment. Also, schedule the participants, being sensitive to time zone differences if you are not in the same location.
- Learn about the stakeholders: Identify the relevant stakeholders for the session and learn about their cultural and regional preferences for meetings. If there are some foreigners as participants, you need to provide them with supporting documentation, such as slides, ahead of time so they can read ahead or follow along.

Preparing for elicitation

- Prepare questions: Before going to elicitation session you need to have a set of prepared questions.
  - Asking *why* question several times can move the discussion from a presented solution to a solid understanding of the problem that need to be solved.
  - Asking *open-ended* questions to help you understand the users' current biz processes and to see how the new system could improve their performance.
- Prepare straw man models:
  - The most useful models are use cases and process flows because they closely align with how people think about doing their jobs.
  - A straw man serves as a starting point that helps you learn about the topic and inspires your users to think of ideas. It is easier to revise a draft model than to create one from scratch.

Performing elicitation activities

- Educate stakeholders:
  - Explain the exploration techniques you'll be using, such as use cases, process flows, and how they can help stakeholders provide better requirements.
- Take good notes:
  - Assign someone who is not actively participating in the discussion to be the scribe, responsible taking accurate notes.
  - Session notes should contain an attendee list, invitees (absents), decisions made, actions to be take,…
- Exploit the physical space:
  - Most rooms have 4 walls, so use them during facilitation to draw diagrams or create lists.
  - If there are many participants in multiple locations, use the online conferences, such as teleconferences, videoconferences,…

Following up after elicitation

- Organizing and sharing the notes:
  - Review and update your notes soon after the session is completed, while the content is still fresh in your mind.
  - Editing the elicitation notes is a risk. Soon after each interview or workshop, share the consolidated notes with the participants and ask them to review to ensure that they accurately represent the session.
- Documenting open issues:
  - During elicitation activities, you might encounter items to be further explored at a later date, record them in an issue-tracking tool.
  - For each issue, record any relevant notes related to resolve the issue, progress already made, an owner, and due date.

Classifying customer input

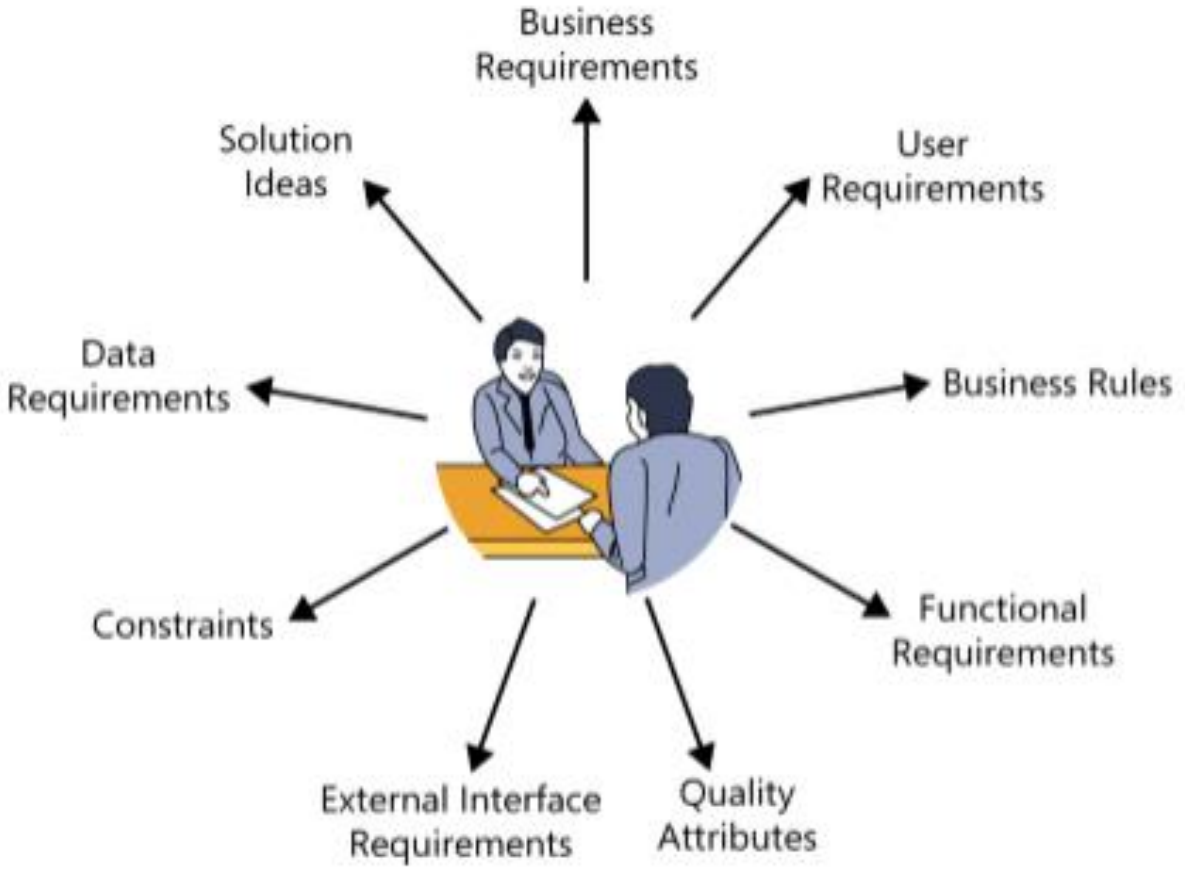


FIGURE 7-7 Classifying customer input.

Classifying customer input

- Biz requirements: Anything related to financial, marketplace, or other biz benefit that either customers or the developing organization wish to gain from the product is a biz requirements.
  - "Increase market share in region X by Y percent within Z months".
  - "Save $X per year on electricity now wasted by inefficient unites."
- User requirements: General statements of user goals or biz tasks that user need to perform are user requirements represented as use cases, scenarios, or user stories.
  - "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

- Biz rules:
  - "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: describe the observable behaviors the system will exhibit under certain conditions and actions the system will led user take.
  - "if 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

- Quality attributes: describe how well the system does something. Key words are fast, easy, user-friendly, reliable, and secure.
  - "The mobile SW must respond quickly to touch commands."
  - "The shopping cart mechanism has to be simple to use so many new customers don't abandon the purchase."
- External interface requirements: describe the connections between your system and the rest of the universe.
  - "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

- Constraints: Design and implementation constraints restrict the options available to the developer. Devices with embedded SW often must respect physical constraint such as size, weight, and interface connections.
  - "File submitted electronically cannot exceed 10 MB in size."
  - "The browser must use 256-bit encryption for all secure transactions."
- Data requirements: describe the format, data type, allowed values, or default value for a data element,…
  - "The ZIP code has 5 digits, followed by an optional hyphen and 4 digits that default to 0000."
  - "An order consists of the customer's identify, 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

- Solution ideas: Many "requirements" form 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 the finger to navigate between screens.".
- In the first EX, the phase *"from a drop-down list"* indicates that this is a solution idea because it's describing a specific user interface control.

How do you know when you're done?

- No simple signal will indicate when you've completed requirements elicitation.
- Perhaps you are done if:
  - 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.

How do you know when you're done?

- ...
  - 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.

Some cautions about elicitation

- Few cautions will decrease the learning curve:
  - Balance stakeholder representation:
    - Collecting input too few representatives or hearing the voice of only the loudest, most opinionated customer is problem. It can lead to overlooking requirements that are important to certain user classes or to including requirements.
    - The best balance involves a few product champions who can speak for their respective user classes.
  - Define scope appropriately:
    - During requirements elicitation, you might find that the project scope improperly defined, being either too large or too small.
    - If too large, you'll accumulate more requirements than are needed. If too small, the customers present their needs beyond the limited scope.

Some cautions about elicitation

- Few cautions will decrease the learning curve:
  - 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.
    - ***How*** helps to clarify and refine the understanding of what users needs. Analysis models, screen sketches, and prototypes help to make the needs expressed during elicitation more tangible and to reveal errors and omissions.
  - Research within reason:
    - The need to do exploratory research sometimes disrupts elicitation.
    - An idea or a suggestion arises, but extensive research is required to assess whether it should be considered for product. In this case, it's better to use prototype to research.

Assumed and implied requirements

- You will never document 100% of the requirements for a system. But the requirements you ***don't*** specify a risk that the project might deliver a solution different from what stakeholders expect. 2 factors behind missed expectations are ***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.

Finding missing requirements

- Missing requirements lead to a common type of requirement defect. The following techniques will help you to detect requirement defect.
  - 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.

Finding missing requirements

- Missing requirements lead to a common type of requirement defect. The following techniques will help you to detect requirement defect.
  - Sets of requirements with complex Boolean logic (ANDs, ORs, and NOTs) often are incomplete. If a combination of logical conditions has no corresponding functional requirement, the developer has to deduce what the system should do or chase down an answer.
  - Create a checklist of common functional areas to consider for your projects, such as error logging, backup and restore, access security, reporting, printing, preview capabilities,…
  - A data model can reveal missing functionality. All data entities that the system will manipulate must have corresponding functionality to create them, read them from external source, update current values, and/or delete them.

THE END
THANK YOU!