Requirements Elicitation & Documentation

Kuldar Taveter
Marlon Dumas

University of Tartu
kuldar.taveter@ut.ee
marlon.dumas@ut.ee

• Easterbrook S., Requirements Engineering course, University of Toronto
Requirements Elicitation

• Starting point: there is a “problem” that needs solving or an “opportunity” that we want to exploit
  • e.g. dissatisfaction with the current state of affairs
  • e.g. a new business opportunity
  • e.g. a potential saving of cost, time, resource usage, etc.

• The requirements analyst must identify the “problem”/”opportunity”
  • Which problem needs to be solved? (identify problem Boundaries)
  • Where is the problem? (understand the Context/Problem Domain)
  • Whose problem is it? (identify Stakeholders)
  • Why does it need solving? (identify the stakeholders’ Goals)
  • How might a software system help? (collect some Scenarios)
  • When does it need solving? (identify Development Constraints)
  • What might prevent us solving it? (identify Feasibility and Risk)
Elicitation Techniques

• Analyzing existing documents & data
• “Brainstorming” possible requirements
  – within the dev. team
• Prepared interviews (one-on-one)
  – Typical for mini-projects
• Focus groups or workshops (one-on-many)
  – useful for larger projects
• Prototyping/mockups
• Meetings with the customer/users
  – E.g. for checkpoints, or showing prototypes
• …
Background Reading

• **Sources of information:**
  – company reports, organization charts, policy manuals, job descriptions, reports, written documentation of existing systems, etc.

• **Advantages:**
  – Helps the analyst to get an understanding of the organization before meeting the people who work there
  – Helps to prepare for other types of fact finding
    • e.g. by being aware of the business objectives of the organization.
  – may provide detailed requirements for the current system

• **Disadvantages:**
  – written documents often do not match up to reality
  – Can be long-winded with much irrelevant detail

• **Appropriate for**
  – Whenever you not familiar with the organization being investigated
Interviews

• **Types:**
  – Semi-structured - agenda of (fairly open) questions
  – Open-ended - no pre-set agenda

• **Advantages**
  – Rich collection of information
  – Good for uncovering opinions, feelings, goals, as well as hard facts
  – Can probe in depth, & adapt follow-up questions to what the person tells you

• **Disadvantages**
  – Large amount of qualitative data can be hard to analyze
  – Hard to compare different respondents
  – Interviewing is a difficult skill to master

*Source: Adapted from Goguen and Linde, 1993, p154.*
Interviewing Tips

• Ask if you can record the interview
  – Make sure the tape recorder is visible
  – Say that they can turn it off at any time.
  – In any case, take notes!
• Try to cover the most relevant “W” questions
  – Who would be the typical users of the system?”
  – Why will each user type use the system? What problem it solves?
  – What should the system do for each user?
  – When (in which circumstances) will each user use the system?
• Follow up interesting leads
  – e.g. “Could we pursue what you just said a little further?”
• Ask open-ended questions towards the end
  – e.g. “Is there anything else you would like to add?”
Prototyping (mockups): Lecture on next Wednesday by Liis Todesk from Fraktal, Ltd.
Meetings

- Used for summarization and feedback
  - E.g. meet with stakeholders towards the end of each stage:
    - to discuss the results of the information gathering stage
    - to conclude on a set of requirements
    - to agree on a design etc.
  - Use the meeting to confirm what has been learned, talk about findings

- Every meeting should have a clear objective:
  - E.g. presentation, problem solving, conflict resolution, progress analysis, gathering and merging of facts, training, planning,...

- Plan the meeting carefully:
  - Schedule the meeting and arrange for facilities
  - Prepare an agenda and distribute it well in advance
  - Keep track of time and agenda during the meeting
  - Follow up with a summary to be distributed to meeting participants
Combine Different Techniques

- Background reading (e.g., Internet?)
- (Initial) Meeting
  - Hard Data analysis
  - Brainstorming
- Interviews
- Meeting
- Prototyping
  - Meeting
  - ...
Requirements Documentation

- Natural Language (NL) feature descriptions (plus supporting tables and graphs)
- Structured natural language / Scenarios
  - e.g., user stories, use case descriptions
- Semi-formal notations
  - e.g., UML use case diagrams, class diagrams, state diagram, sequence diagrams, goal models, etc.
You have understood

- Why the system will be built, and
- For whom it is useful.

You have explained this in the Wiki.

- what is going to be built and have broken the functionality down into functional requirements

After reading the requirements, I understand who will be the users of the application and what will the user be able to do with the system.
Example: Home Access Control

Objective: Design an electronic system for:

- Home access control
  - Locks and lighting operation
- Intrusion detection and warning
<table>
<thead>
<tr>
<th>Identifier</th>
<th>Priority</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQ1</td>
<td>5</td>
<td>The system shall keep the door locked at all times, unless commanded otherwise by authorized user. When the lock is disarmed, a countdown shall be initiated at the end of which the lock shall be automatically armed (if still disarmed).</td>
</tr>
<tr>
<td>REQ2</td>
<td>2</td>
<td>The system shall lock the door when commanded by pressing a dedicated button.</td>
</tr>
<tr>
<td>REQ3</td>
<td>5</td>
<td>The system shall, given a valid key code, unlock the door and activate other devices.</td>
</tr>
<tr>
<td>REQ4</td>
<td>4</td>
<td>The system should allow mistakes while entering the key code. However, to resist “dictionary attacks,” the number of allowed failed attempts shall be small, say three, after which the system will block and the alarm bell shall be sounded.</td>
</tr>
<tr>
<td>REQ5</td>
<td>2</td>
<td>The system shall maintain a history log of all attempted accesses for later review.</td>
</tr>
<tr>
<td>REQ6</td>
<td>2</td>
<td>The system should allow adding new authorized persons at runtime or removing existing ones.</td>
</tr>
<tr>
<td>REQ7</td>
<td>2</td>
<td>The system shall allow configuring the preferences for device activation when the user provides a valid key code, as well as when a burglary attempt is detected.</td>
</tr>
<tr>
<td>REQ8</td>
<td>1</td>
<td>The system should allow searching the history log by specifying one or more of these parameters: the time frame, the actor role, the door location, or the event type (unlock, lock, power failure, etc.). This function shall be available over the Web by pointing a browser to a specified URL.</td>
</tr>
<tr>
<td>REQ9</td>
<td>1</td>
<td>The system should allow filing inquiries about “suspicious” accesses. This function shall be available over the Web.</td>
</tr>
</tbody>
</table>
### Example NL Requirements

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Priority</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQ1</td>
<td>5</td>
<td>The system shall keep the door locked at all times, unless commanded otherwise by authorized user. When the lock is disarmed, a countdown shall be initiated at the end of which the lock shall be automatically armed (if still disarmed).</td>
</tr>
<tr>
<td>REQ2</td>
<td>2</td>
<td>The system shall lock the door when commanded by pressing a dedicated button.</td>
</tr>
<tr>
<td>REQ3</td>
<td>5</td>
<td>The system shall, given a valid key code, unlock the door and activate other devices.</td>
</tr>
<tr>
<td>REQ4</td>
<td>4</td>
<td>The system should allow mistakes while entering the key code. However, to resist “dictionary attacks,” the number of allowed failed attempts shall be small, say three, after which the system will block and the alarm bell shall be sounded.</td>
</tr>
<tr>
<td>REQ5</td>
<td>2</td>
<td>The system shall maintain a history log of all attempted accesses for later review.</td>
</tr>
<tr>
<td>REQ6</td>
<td>2</td>
<td>The system should allow adding new authorized persons at runtime or removing existing ones.</td>
</tr>
<tr>
<td>REQ7</td>
<td>2</td>
<td>The system shall allow configuring the preferences for device activation when the user provides a valid key code, as well as when a burglary attempt is detected.</td>
</tr>
<tr>
<td>REQ8</td>
<td>1</td>
<td>The system should allow searching the history log by specifying one or more of these parameters: the time frame, the actor role, the door location, or the event type (unlock, lock, power failure, etc.). This function shall be available over the Web by pointing a browser to a specified URL.</td>
</tr>
<tr>
<td>REQ9</td>
<td>1</td>
<td>The system should allow filing inquiries about “suspicious” accesses. This function shall be available over the Web.</td>
</tr>
</tbody>
</table>

*’shall’*: mandatory (?)

*’should’*: optional (?)
<table>
<thead>
<tr>
<th>Identifier</th>
<th>Priority</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQ1</td>
<td></td>
<td>The system shall keep the door locked at all times, unless commanded otherwise by authorized user. When the lock is disarmed, a countdown shall be initiated at the end of which the lock shall be automatically armed (if still disarmed).</td>
</tr>
<tr>
<td>REQ2</td>
<td>2</td>
<td>The system shall lock the door when commanded by pressing a dedicated button.</td>
</tr>
<tr>
<td>REQ3</td>
<td>5</td>
<td>The system shall, given a valid key code, unlock the door and activate other devices.</td>
</tr>
<tr>
<td>REQ4</td>
<td>4</td>
<td>The system should allow mistakes while entering the key code. However, to resist “dictionary attacks,” the number of allowed failed attempts shall be small, say three, after which the system will block and the alarm bell shall be sounded.</td>
</tr>
<tr>
<td>REQ5</td>
<td>2</td>
<td>The system shall maintain a history log of all attempted accesses for later review.</td>
</tr>
<tr>
<td>REQ6</td>
<td>2</td>
<td>The system should allow adding new authorized persons at runtime or removing existing ones.</td>
</tr>
<tr>
<td>REQ7</td>
<td>2</td>
<td>The system shall allow configuring the preferences for device activation when the user provides a valid key code, as well as when a burglary attempt is detected.</td>
</tr>
<tr>
<td>REQ8</td>
<td>1</td>
<td>The system should allow searching the history log by specifying one or more of these parameters: the time frame, the actor role, the door location, or the event type (unlock, lock, power failure, etc.). This function shall be available over the Web by pointing a browser to a specified URL.</td>
</tr>
<tr>
<td>REQ9</td>
<td>1</td>
<td>The system should allow filing inquiries about “suspicious” accesses. This function shall be available over the Web.</td>
</tr>
</tbody>
</table>
### Example NL Requirements

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Priority</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQ1</td>
<td>5</td>
<td>The system shall keep the door locked at all times, unless commanded otherwise by authorized user. When the lock is disarmed, a countdown shall be initiated at the end of which the lock shall be automatically armed (if still disarmed).</td>
</tr>
<tr>
<td>REQ2</td>
<td>2</td>
<td>The system shall lock the door when commanded by pressing a dedicated button.</td>
</tr>
<tr>
<td>REQ3</td>
<td>5</td>
<td>The system shall, given a valid key code, unlock the door and activate other devices.</td>
</tr>
<tr>
<td>REQ4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>REQ5</td>
<td>2</td>
<td>For <strong>REQ3</strong>, the customer may suggest these test cases:</td>
</tr>
<tr>
<td>REQ6</td>
<td>2</td>
<td>- Test with the valid key of a current tenant on his or her apartment (pass)</td>
</tr>
<tr>
<td>REQ7</td>
<td>2</td>
<td>- Test with the valid key of a current tenant on someone else’s apartment (fail)</td>
</tr>
<tr>
<td>REQ8</td>
<td>1</td>
<td>- Test with an invalid key on any apartment (fail)</td>
</tr>
<tr>
<td>REQ9</td>
<td>1</td>
<td>- Test with the key of a removed tenant on his or her previous apartment (fail)</td>
</tr>
<tr>
<td>REQ10</td>
<td>1</td>
<td>- Test with the valid key of a just-added tenant on his or her apartment (pass)</td>
</tr>
<tr>
<td>REQ11</td>
<td>1</td>
<td>The system should allow filing inquiries about “suspicious” accesses. This function shall be available over the Web.</td>
</tr>
</tbody>
</table>
# Example User Stories

<table>
<thead>
<tr>
<th>Identifier</th>
<th>User Story</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST-1</td>
<td>As an authorized person (tenant or landlord), I can keep the doors locked at all times. (… so I am safe)</td>
</tr>
<tr>
<td>ST-2</td>
<td>As an authorized person (tenant or landlord), I want the lock be automatically locked after a defined period of time.</td>
</tr>
<tr>
<td>ST-3</td>
<td>As an authorized person (tenant or landlord), I can lock the doors on demand.</td>
</tr>
<tr>
<td>ST-4</td>
<td>As an authorized person (tenant or landlord), I can unlock the doors. (Test: Allow a small number of mistakes, say three.)</td>
</tr>
<tr>
<td>ST-5</td>
<td>As a landlord, I can at runtime manage authorized persons.</td>
</tr>
<tr>
<td>ST-6</td>
<td>As an authorized person (tenant or landlord), I can view past accesses.</td>
</tr>
<tr>
<td>ST-7</td>
<td>As a tenant, I can configure the preferences for activation of various devices.</td>
</tr>
<tr>
<td>ST-8</td>
<td>As a tenant, I can file complaint about “suspicious” accesses.</td>
</tr>
</tbody>
</table>

Note: ‘Why’ part is missing in the examples above.
User Stories

As a tenant, I can unlock the doors to enter my apartment.

- user-role (benefactor)
- capability (functionality)
- business-value (motivation/rationale)

**who - what - why**

- Similar to NL requirements, but focus on the user benefits, instead on system characteristics (alone).
- Unfortunately, third element (business-value) is often omitted
- Preferred tool in agile methods.
NL Requirements vs. User Stories

• Traditional requirement – “shall” statements:
  • “The system shall provide a user configurable interface for all user and system manager functions”
  • “The user interface shall be configurable in the areas of:
    • Screen layout
    • Font
    • Background and text color

• Corresponding “User Story”:
  • “As a system user or system manager, …
  • … I want to be able to configure the user interface for screen layout, font, background color, and text color, …
  • … so that I can use the system in the most efficient manner”
User Story

• <Actor/Role> As a user

• I want to narrow down people search results by location

• so I can find the right person more quickly

Acceptance test:

Given I am on the search screen
And ‘Paula’ is on the same indexed page with ‘Tartu’

When I search for ‘Paula’

Then I see ‘Tartu’ in the location section of the search results
Organizing user stories by goal models

The Notation For Goal Model

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Goal</td>
</tr>
<tr>
<td></td>
<td>Quality Goal</td>
</tr>
<tr>
<td></td>
<td>Role</td>
</tr>
<tr>
<td></td>
<td>Relationship between goals</td>
</tr>
<tr>
<td></td>
<td>Goal and quality goal relationship</td>
</tr>
</tbody>
</table>

User Stories

As an app user I want to be able to see list of all lost items from database

As an app user I want to be able to see list of found items from database

As an app user when clicking on item in a list of found/lost items I want to see all available item information on a screen (picture + all the info)
Completeness

• How do I know I have nailed down the main requirements?

• Tip (useful for typical data-driven apps):
  – Identify the main entities in your domain
    • Putting them in a “domain model” helps
  – For each entity, ask yourself if users need to:
    • Search an entity within a collection
    • Create a new entity
    • Read one existing entity
    • Update one or more entities
    • Delete one or more entities
Detailed Functional Requirements
Use Case Diagrams and Descriptions

Use Case Description:
• Name of Use Case
• Actors associated with Use Case
• Pre-conditions
• Post-conditions
• Normal Flow of Events (Basic Scenario)
• Alternative Flow of Events (Alternative Scenarios)
• …
# Schema for Detailed Use Cases

<table>
<thead>
<tr>
<th>Use Case UC-#:</th>
<th>Name / Identifier</th>
<th>[verb phrase]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Related Requirements:</td>
<td>List of the requirements that are addressed by this use case</td>
<td></td>
</tr>
<tr>
<td>Initiating Actor:</td>
<td>Actor who initiates interaction with the system to accomplish a goal</td>
<td></td>
</tr>
<tr>
<td>Actor’s Goal:</td>
<td>Informal description of the initiating actor’s goal</td>
<td></td>
</tr>
<tr>
<td>Participating Actors:</td>
<td>Actors that will help achieve the goal or need to know about the outcome</td>
<td></td>
</tr>
<tr>
<td>Preconditions:</td>
<td>What is assumed about the state of the system before the interaction starts</td>
<td></td>
</tr>
<tr>
<td>Postconditions:</td>
<td>What are the results after the goal is achieved or abandoned; i.e., what must be true about the system at the time the execution of this use case is completed</td>
<td></td>
</tr>
</tbody>
</table>

## Flow of Events for Main Success Scenario:

1. The initiating actor delivers an action or stimulus to the system (the arrow indicates the direction of interaction, to- or from the system)
2. The system’s reaction or response to the stimulus; the system can also send a message to a participating actor, if any
3. ...

## Flow of Events for Extensions (Alternate Scenarios):

What could go wrong? List the exceptions to the routine and describe how they are handled

1a. For example, actor enters invalid data
2a. For example, power outage, network failure, or requested data unavailable
...

The arrows on the left indicate the direction of interaction: → Actor’s action; ← System’s reaction
Use Case 1: Unlock

<table>
<thead>
<tr>
<th>Use Case UC-1: Unlock</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Related Requirem’ts:</strong></td>
</tr>
<tr>
<td><strong>Initiating Actor:</strong></td>
</tr>
<tr>
<td><strong>Actor’s Goal:</strong></td>
</tr>
<tr>
<td><strong>Participating Actors:</strong></td>
</tr>
</tbody>
</table>

- The set of valid keys stored in the system database is non-empty.
- The system displays the menu of available functions; at the door keypad the menu choices are “Lock” and “Unlock.”

| **Postconditions:** | The auto-lock timer has started countdown from autoLockInterval. |

**Flow of Events for Main Success Scenario:**

→ 1. **Tenant/Landlord** arrives at the door and selects the menu item “Unlock”
   2. **include::AuthenticateUser (UC-7)**

← 3. **System** (a) signals to the **Tenant/Landlord** the lock status, e.g., “disarmed,” (b) signals to **LockDevice** to disarm the lock, and (c) signals to **LightSwitch** to turn the light on

← 4. **System** signals to the **Timer** to start the auto-lock timer countdown

→ 5. **Tenant/Landlord** opens the door, enters the home [and shuts the door and locks]
## Subroutine «include» UC-7

<table>
<thead>
<tr>
<th>Use Case UC-7: AuthenticateUser (sub-use case)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Related Requirements:</strong> REQ3, REQ4</td>
</tr>
<tr>
<td><strong>Initiating Actor:</strong> Any of: Tenant, Landlord</td>
</tr>
<tr>
<td><strong>Actor’s Goal:</strong> To be positively identified by the system (at the door interface).</td>
</tr>
<tr>
<td><strong>Participating Actors:</strong> AlarmBell, Police</td>
</tr>
<tr>
<td><strong>Preconditions:</strong></td>
</tr>
<tr>
<td>• The set of valid keys stored in the system database is non-empty.</td>
</tr>
<tr>
<td>• The counter of authentication attempts equals zero.</td>
</tr>
<tr>
<td><strong>Postconditions:</strong> None worth mentioning.</td>
</tr>
</tbody>
</table>

### Flow of Events for Main Success Scenario:

1. **System** prompts the actor for identification, e.g., alphanumeric key
2. **Tenant/Landlord** supplies a valid identification key
3. **System** (a) verifies that the key is valid, and (b) signals to the actor the key validity

### Flow of Events for Extensions (Alternate Scenarios):

2a. **Tenant/Landlord** enters an invalid identification key

1. **System** (a) detects error, (b) marks a failed attempt, and (c) signals to the actor
2. **Tenant/Landlord** supplies a valid identification key

1a. **System** (a) detects that the count of failed attempts exceeds the maximum allowed number, (b) signals to sound **AlarmBell**, and (c) notifies the **Police** actor of a possible break-in

3. Same as in Step 3 above
You have listed the **most important** non-functional requirements.

There is **at least seven** *measurable* non-functional requirements present.

You have noted the requirements down in the Wiki.

Reading the requirements, I understand how you are going **to measure whether the system satisfies a given requirement**.

Non-functional requirement include the **initial architectural choices listing** what technologies you are going to use in order to complete the project.
Example NFRs

• **Usability requirements**
  – Usefulness – users can do what they need/want
  – **Ease-of-use** – users can do it easily
  – **Learnability** – new users can quickly learn how to do it

• **Performance requirements**
  – **Efficiency / time+space bounds**
  – Workloads, response time, throughput… e.g. ”the system must handle 1,000 transactions per second”

• **reliability**
  – Availability (downtime)
  – Integrity of information maintained and supplied to the system

• **security**
  – e.g. permissible information flows, or who can do what

• **Operating requirements**
  – physical constraints (size, weight),
  – personnel availability & skill level
  – accessibility for maintenance
  – environmental conditions…

• **Lifecycle requirements**
  – “Future-proofing”
    • **Maintainability**, Enhanceability, Portability
  – limits on development
    • e.g development time limitations,
    • resource availability
    • methodological standards
Usability requirements

Usability = Usefulness + Ease of use

• Ease of learning (learnability)
  – How easy is the system to learn for various groups of users?

• Task efficiency
  – How efficient is it for the frequent user?

• Ease of remembering
  – How easy is it to remember for the occasional user?

• Subjective satisfaction
  – How satisfied is the user with the system?

• Understandability
  – How easy is it to understand what the system does?

<table>
<thead>
<tr>
<th>Response time (sec)</th>
<th>Rating</th>
<th>Quality score</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 2</td>
<td>Exceeds expectation</td>
<td>3</td>
</tr>
<tr>
<td>2-5</td>
<td>Within the target range</td>
<td>2</td>
</tr>
<tr>
<td>6-10</td>
<td>Minimally acceptable</td>
<td>1</td>
</tr>
<tr>
<td>&gt;10</td>
<td>Unacceptable</td>
<td>0</td>
</tr>
</tbody>
</table>
Usability requirements

- **Q1**: It should be easy for novice users to do tasks Q and R.
- **Q2**: Novice users should perform tasks Q and R in a short time.
- **Q3**: Experienced users complete tasks Q, R, and S quicker than novice users
- **Q4**: Recording breakfast shall be easy using keyboard

- **Problem counts**
  - **Q1**: At most 1 of 20 novices shall encounter critical problems during tasks Q and R.

- **Task time**
  - **Q2**: Novice users shall perform tasks Q and R in 15 minutes.
  - **Q3**: Experienced users complete tasks Q, R and S in 2 minutes

- **Keystroke counts**
  - **Q4**: Recording breakfast shall be possible within 5 keystrokes per guest
Efficiency

- **Efficiency** - the capability of the software to provide the required performance relative to the amount of resources used, under stated conditions
Efficiency

- **Efficiency** - the capability of the software to provide the required performance relative to the amount of resources used, under stated conditions.

Which statement is objectively defined?

1: Product shall be able to process a lot of payment transactions in a short time even during peak load.

2: Product shall be able to process 100 payment transactions per second in peak load.
Efficiency

• **Efficiency** - the capability of the software to provide the required performance relative to the amount of resources used, under stated conditions

Which statement is objectively defined?

1: Scrolling one page up or down in a 200 page document shall take at most 1s. Searching for a specific keyword shall take at most 5s.

2: Scrolling one page up or down in a large document shall take an efficient time. Searching for a specific keyword shall take a reasonable time
Project Plan

First version of the project plan is present in the Wiki/Issue tracker. It contains at minimum:

- **Roles in project.** You have understood each team member roles and assigned responsibilities.
First version of the project plan is present in the Wiki/Issue tracker. It contains at minimum:

- **Communication means.** You have agreed both among yourselves and with the customer how and when and via which channels you are going to communicate.
Communication Genres

• **Early stages – meeting(s)**
  – Team members need to build up their trust and confidence in their co-workers
  – Decision making

• **Intermediate stages (design) – teleconferencing**
  – Activities executed in parallel
  – Some points needs to be clarified

• **Implementation stages - emails**
  – Everyone knows his role, work can progress
  – Face to face meetings – helps coordination and maintain motivation
You have created

- The list of all tasks you currently foresee need to be delivered in order to complete the project.
- Tasks are planned into iterations or given priorities, assigned to person responsible and contain initial estimation about the amount of work involved.
- The descriptions of the tasks in the project plan is understandable.

The tasks to be completed in the second iterations must be detailed in project plan. Further iterations might not be clear enough to break them down to tasks according to your current knowledge about requirements.

- The tasks in the second iteration are defined at the proper level of granularity (90% of tasks take less than 16 hours, no task is larger than 24 hours).

Make sure you know what is the definition of DONE for each task…