

SWE 215: Software Requirements Engineering

Fundamentals of Requirements Management

Dr. Jameleddine Hassine

ICS Department, KFUPM

jhassine@kfupm.edu.sa

Objectives

- Learn how to manage requirements artifacts
- Learn the fundamentals of requirements traceability

Contents

- Introduction to Requirements Management
- Traceability
- Requirements Management Tools

Why Do Requirements Change?

- Change in software development is **inevitable** and **difficult to control**.
 - Everything may **change**:
 - Business
 - Context
 - Technologies
 - Markets
 - ...
- Possible responses to change
 - Add, modify, or remove requirements

Some Problems Due to Changing Requirements

- Requirements changing **towards the end of development without any impact assessment**
- **Unmatched/outdated** requirements specifications causing **confusion** and **unnecessary rework**
- **Time** spent coding, writing test cases or documentation for **requirements that no longer exist**

Requirements Management

- A **systematic approach** to **eliciting**, **organizing**, and **documenting** the requirement of the system, and a process that establishes and maintains agreement between the customer and the project team on the changing requirements of the system.

Requirements Management Activities (1)

- Includes all activities intended to **maintain the integrity and accuracy of expected requirements**
- Manage changes to agreed requirements
- Keep project plans synchronized with requirements
- Control versions of individual requirements and versions of requirements documents
- Manage relationships between requirements
- Managing the dependencies between the requirements document and other documents produced in the systems engineering process
- Track requirements status

Requirements Change Factors (1)

- Requirements errors, conflicts, and inconsistencies
 - May be detected at any phase (when requirements are analyzed, specified, validated, or implemented)
- Evolving customer/user knowledge of the system
 - When the requirements are developed, customers/users simultaneously develop a better understanding of what they really need
- Technical, schedule, or cost problems
 - Difficult to plan and know everything in advance
 - We may have to revisit the list of requirements and adapt it to the current situation

Requirements Change Factors (2)

Changing customer priorities, new needs:

- May be caused by a change in the system environment (technological, business, political...), i.e., the context
- Business and strategic goals may change
- May be caused by the arrival of a new competitor
- Laws and regulations may change
- Collaborating systems may change
- May also be caused by technology changes in the enterprise (migration to a new operating system, DBMS...)
- May be caused by organizational changes (organizational structure, business processes, employees...)

Requirements Volatility

Some requirements are usually more subject to change than others

- **Stable requirements** are concerned with the essence of a system and its application domain
 - Derived from the client's principal business activities or the domain model
 - Example: **a hospital will always have doctors, nurses, patients...**
- **Volatile requirements** are specific to the instantiation of the system in a particular environment for a particular customer at a particular time
 - Example: **in a hospital, we can think of requirements related to the policies of the government health system**

Expectations of Requirements Management

- **Identification** of individual requirements (**Unique Identification**)
- **Traceability** from highest level requirements to implementation
 - Established via **links** through a **requirements database**
 - **Links between requirements and design models, tests, code...**
 - Coverage and consistency analysis
- **Impact assessments** of proposed changes
 - Which other requirements (and other linked artifacts) will be affected by a change

Requirements Have Attributes!

- Attributes establish **context** and **background**, and go beyond the requirement description
- For **filtering, analysis, metrics...**
 - Creation date, Last update, Author, Stakeholders (Owners / Source)
 - Version number
 - Status, Priority, Importance, Stability
 - Rationale, Comments
 - Acceptance criteria
 - Subsystem / Product release number
- The more complex the project, the richer the attributes...

Requirements Status

- Help manage the requirement lifecycle
 - Their number and nature depend on the process in place
- Example of a set of statuses:
 - **Proposed**: by some stakeholder
 - **Approved**: part of baseline, committed to implement
 - **Rejected**: after evaluation
 - **Implemented**: designed and implemented
 - **Verified**: Relevant tests have passed
 - **Deleted**: Removed from list

Version Control

- Every version of a requirement needs to be uniquely identified
 - Changes need to be documented and clearly communicated
 - A **version identifier** must be updated with every change to the requirement
- Requirements documents should include
 - **A revision history:** changes, dates, by whom, why...
 - Standard markers for revisions (e.g., strikethrough or underlined text, coloring, line markers...)
- **Version control tool may be used**
 - To store and manage the revision history
 - To store justifications (to add, modify, delete, reject a requirement)



Traceability

Traceability Quotes

- Requirements traceability refers to the ability to describe and follow the **life of a requirement**, in both **forwards** and **backwards** direction (i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through all periods of ongoing refinement and iteration in any of these phases)".¹
- One cannot **manage** what cannot be traced.²
- Traceability gives essential assistance in understanding the **relationships** that exist within and across software requirements, design, and implementation.³
- Traceability information helps **assess the impact of changes** to requirements, connecting these requirements as well as requirements for other representations of the system.³
- Traceability is **often mandated** by contracts and standards, e.g., military and aerospace ¹.

Benefits of Traceability

- Prevents losing knowledge
- Supports the verification process (certification, localization of defects)
- Impact analysis
- Change control
- Improved software quality (make changes correctly and completely)
- Reuse (by identifying what goes with a requirement: design, code...)
- Risk reduction (e.g., if a team member with key knowledge leaves)

Traceability Difficulties

- Various stakeholders require **different** information
- **Huge amount** of requirements traceability information must be tracked and maintained
- **Manual** creation of links is **very** demanding
 - **Likely the most annoying problem**
- Specialized **tools** must be used
- Integrating **heterogeneous** models/information from/to different sources (requirements, design, tests, code, documentation, rationales...) is not trivial
- Requires **organizational** commitment (with an understanding of the potential benefits)

Backward and Forward Traceability

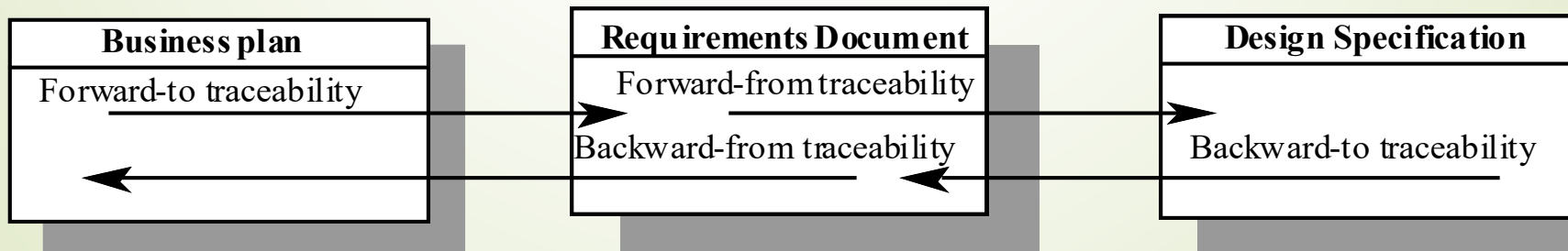
19

➤ **Backward** traceability

- To **previous** stages of development
- Links other documents (which may have preceded the requirements document) to relevant requirements
- **Help evaluate which requirements are affected by changes to users' needs**

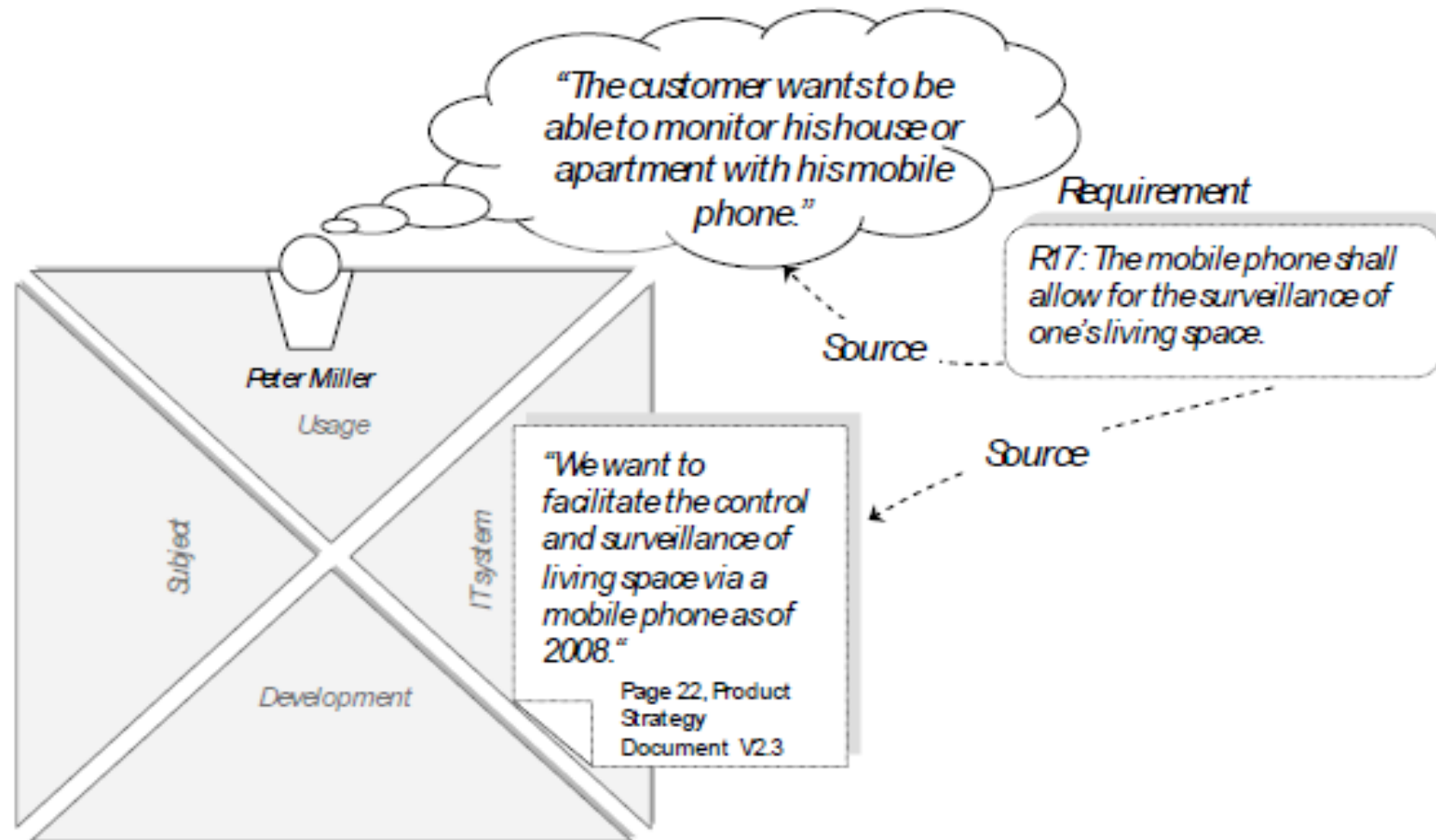
➤ **Forward** traceability

- Links requirements to the design and implementation components
- Help assure that all requirements have been satisfied

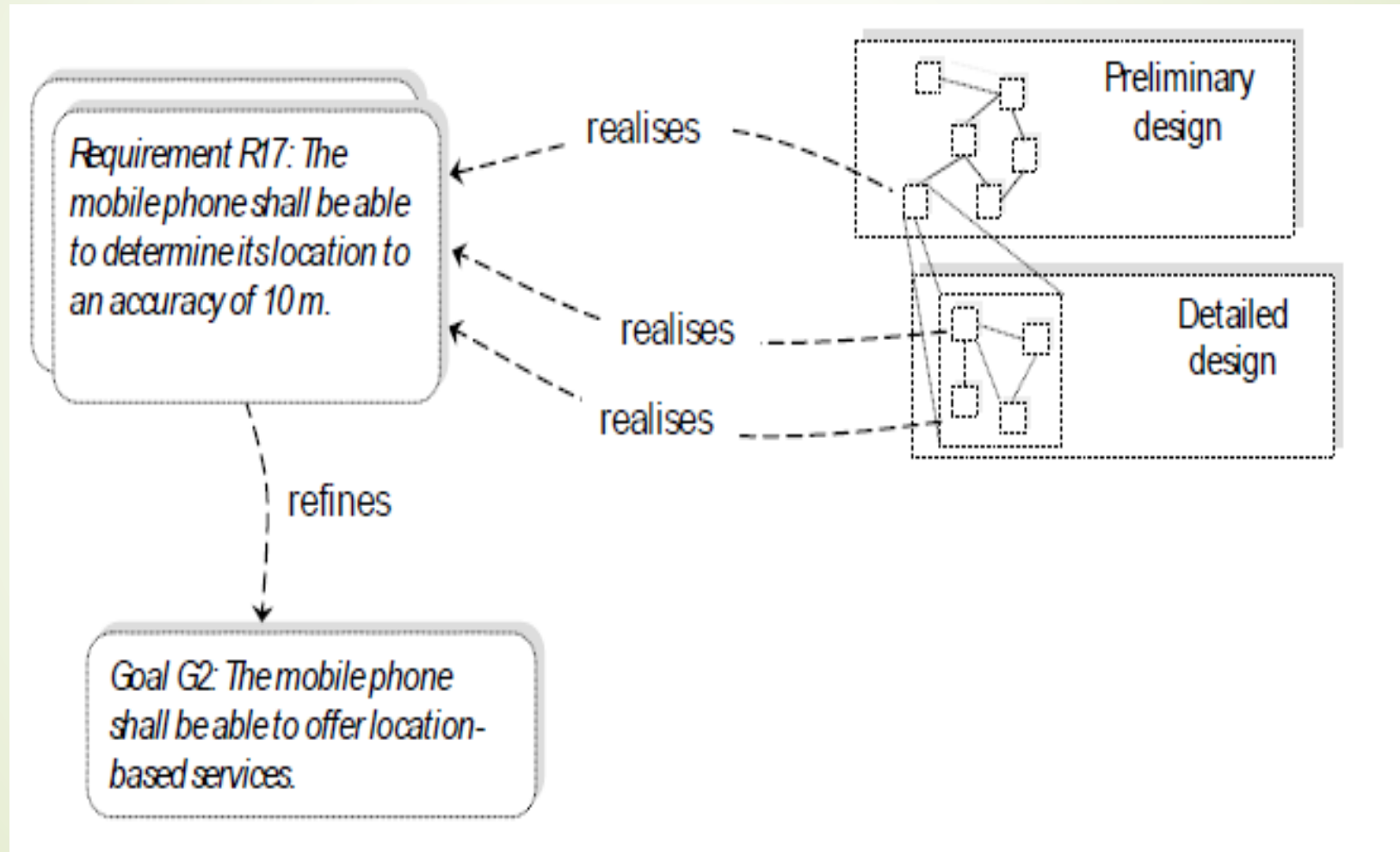


Example of backward-traceability of a requirement

20



Example of forward-traceability of requirements and traceability between requirements



Representation – Traceability Table

- Show the relationships between requirements or between requirements and other artifacts
- Table can be set up to show links between several different elements
- Backward and forward traceability
- Difficult to capture different types of links

User Requirement	Functional Requirement	Design Element	Code Module	Test Case
UC-28	catalog.query.sort	Class Catalog	catalog.sort()	search.7 search.8
UC-29	catalog.query.import	Class Catalog	catalog.import(), cat catalog.validate()	search.12 search.13 search.14

Representation – Traceability Matrix

- Define links between pairs of elements
 - E.g., requirements to requirement, use case to requirement, requirement to test case...
- Can be used to **defined relationships between pairs**
 - E.g., specifies/is specified by, depends on, is parent of, ...
- More amenable to automation than traceability table

Depends-on

	R1	R2	R3	R4	R5	R6
R1			*	*		
R2					*	*
R3				*	*	
R4		*				
R5						*
R6						

Traceability matrix for a single relationship type (ex. satisfies relationship)

		Target artefacts				
Source artefacts	<i>satisfies</i>	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
	Scenario 1	X				
	Scenario 2				X	
	Scenario 3					
	Scenario 4			X		X
	Scenario 5		X			

Traceability matrix for several relationship types

		Target artefacts				
Source artefacts		Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
	Scenario 1	<i>satisfies</i>				
	Scenario 2	<i>based_on</i>	<i>conflicts</i>		<i>satisfies</i>	
	Scenario 3		<i>satisfies</i>			
	Scenario 4	<i>conflicts</i>		<i>satisfies</i>		<i>satisfies</i>
	Scenario 5		<i>satisfies</i>		<i>based_on</i>	

Representation – Traceability List

- **Traceability matrices** become more of a problem when there are hundreds or thousands of requirements as the matrices become large and are sparsely populated
- **A simplified form** of a traceability matrix may be used where, along with each requirement description, one or more lists of the identifiers of related requirements are maintained

Requirement	Depends-on
R1	R3, R4
R2	R5, R6
R3	R4, R5
R4	R2
R5	R6

Tools to document traceability Information

- General purpose tools (e.g., **spreadsheet programs, word processors, hypertext editors**)
 - Suitable for small and short term projects
 - Not sufficient for extensive requirements tracing purposes
- **Requirements management tools**
 - Suitable for projects producing large and complex systems
 - Require investments (e.g., licenses, training end-users, system maintenance, consultation)
 - Examples: **Rational RequisitePro, DOORS**