FACOLTÀ DI INGEGNERIA
UNIVERSITÀ DEGLI
STUDI DI PALERMO

Corso di Laurea Specialistica
Ingegneria Gestionale

Sistemi ICT per il Business Networking

Requirements Engineering

Docente: **Vito Morreale** (vito.morreale@eng.it)

UP Phases

- 1. **Inception** — approximate vision, business case, scope, vague estimates.
- 2. **Elaboration** — refined vision, iterative implementation of the core architecture, resolution of high risks, identification of most requirements and scope, more realistic estimates.
- 3. **Construction** — iterative implementation of the remaining lower risk and easier elements, and preparation for deployment.
- 4. **Transition** — beta tests, deployment.

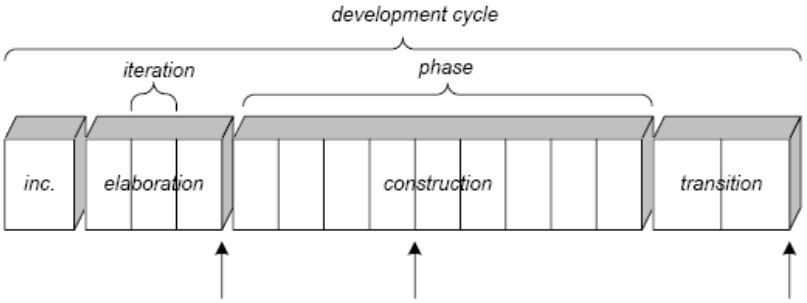
UP Phases

- This is **not the old "waterfall"** or sequential lifecycle of first defining all the requirements, and then doing all or most of the design.
- Inception is not a requirements phase; rather, it is a kind of feasibility phase, where just enough investigation is done to support a decision to continue or stop.
- Elaboration is not the requirements or design phase; rather, it is a phase where the core architecture is iteratively implemented, and high risk issues are mitigated.

Elaboration phase

- **Elaboration ends when** the high risk issues have been resolved, the architectural core or skeleton is complete, and "most" requirements are understood.
- **At the end of elaboration**, it is possible to more realistically estimate the remaining effort and duration for the project.

Phases and iterations

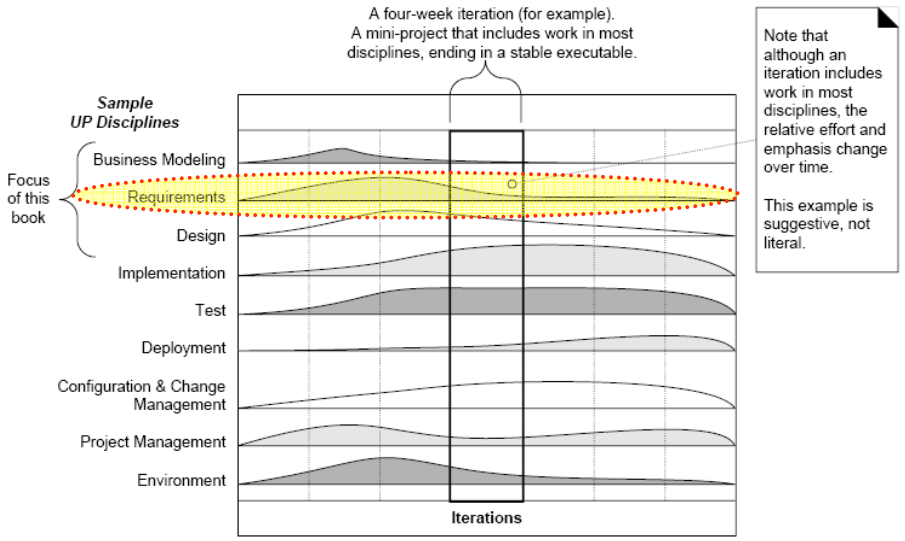


milestone	*release*	*increment*	*final production release*
An iteration end-point when some significant decision or evaluation occurs.	A stable executable subset of the final product. The end of each iteration is a minor release.	The difference (delta) between the releases of 2 subsequent iterations.	At this point, the system is released for production use.

UP Disciplines

- The UP describes **work activities** (e.g. writing a use case) within **disciplines**
- **Discipline**: a set of activities (and related **artifacts**) in one subject area (e.g. the activities within requirements analysis)
- **Artifact**: any work product (e.g. code, Web graphics, database schema, text documents, diagrams, models, ...).

UP Disciplines



Requirements

- **Requirement**: 'a statement of a system service or constraint'. Capabilities and conditions to which the system **must** conform.
- **Service statement**
  - a **business rule** that must be obeyed at all times (e.g. 'salaries are paid on Wednesdays')
  - a **computation** that the system must carry out (e.g. 'calculate salesperson commission based on the sales in the last month using a particular formula')
- **Constraint statement**
  - a **restriction** on the system's behavior ('only direct managers can see the salary information of their staff')
  - a **restriction** on the system's development ('we must use those development tools')

Requirements determination

- To **identify**, **analyze** and **negotiate** requirements with the customers
- This **does not refer to** the waterfall attitude of attempting to fully define and stabilize the requirements in the first phase of a project, ...
- ... but rather "**a systematic approach** to finding, documenting, organizing, and tracking the changing requirements of a system" ...
- ... **in the context** of **inevitably changing and unclear** stakeholder's wishes
- Important terms:
  - **Changing**: the UP embraces change in requirements as a fundamental driver
  - **Finding**: skillful elicitation via techniques such as use case writing and requirements workshops

Types of Requirements: FURPS+

- **Functional**: features, capabilities, security
- **Usability**: human factors, help, documentation
- **Reliability**: frequency of failure, recoverability, predictability
- **Performance**: response times, accuracy, availability, resource usage
- **Supportability**: adaptability, maintainability, internationalization, configurability

FURPS

- Others:
  - **Implementation**: resource limitations, languages, tools, hardware ...
  - **Interface**: constraints imposed by interfacing with external systems
  - **Operations**: system management in its operational setting
  - **Packaging**
  - **Legal**: licensing, ...
  - ...

+

Types of Requirements

- **Important**. There are several systems of requirements categorization and quality attributes published in books and by standards organizations, such as ISO 9126 (which is similar to the FURPS+ list), and several from the Software Engineering Institute (SEI)
- **In common usage**, requirements are categorized as **functional** (behavioral) or **non-functional** (everything else)

Requirements determination

- Several **techniques**:
  - Structured and non-structured interviews
  - Questionnaires
  - Analysis of existing documents
  - Rapid prototyping
  - ...
- Attention to overlapping and contradicting requirements
- **Output**: requirements document (text, diagrams, tables, no formal models) ← a contract between developers and customers

Requirements artifacts

- **Functional requirements**:
  - Use-Case Model
  - system features list of the Vision artifact, which summarizes high-level requirements that are elaborated in other documents
- **Other requirements**:
  - in the use cases they relate to
  - Supplementary Specifications artifact
- **Glossary**: terms used in the requirements
  - encompasses the concept of the data dictionary

Development case

Discipline	Artifact Iteration-*	Incep. 11	Elab. El. .En	Const. CL.Cn	Trans. T1..T2
Business Modeling	Domain Model		s		
Requirements	Use-Case Model	s	r		
	Vision	s	r		
	Supplementary Specification	s	r		
	Glossary	s	r		
Design	Design Model		s	r	
	SW Architecture Document		s		
	Data Model		s	r	
Implementation	Implementation Model		s	r	r
Project Management	SW Development Plan	s	r	r	r
Testing	Test Model		s	r	
Environment	Development Case	s	r		

Requirements specification

- **Important**. Ideally, the specification models **should** (if possible) be independent from the hardware/software platform on which the system is to be deployed … Remember: it concerns with WHAT !!!

Some Problems with the Waterfall Lifecycle: Requirements Inflexibility

- The only constant is **change**:
  - the stakeholders **change their minds** or cannot fully envision what they want until they see a concrete system
  - the market changes
  - …
- **Waterfall**: lack of responsiveness to changing user wishes or market opportunities
- **Iterative development**: not all requirements are specified before design and implementation, and requirements are not stabilized until after at least several iterations

Requirements Inflexibility: Mitigation with iterative processes

1. A **subset of core requirements** is defined
   - The team chooses a subset of those to design and implement (based usually on **highest risk or business value**)
2. Stakeholders meet in a second one- or two-day **requirements workshop**, intensively review the partial system, and clarify and modify their requests.
3. Iteration of **incrementally implementing** the system
4. Stakeholders meet in a **third requirements workshop**, and refine again
5. A somewhat **realistic plan** and **estimate of the remaining work** is possible
6. ...

References

- **Understanding requirements** (Capitolo libro “Applying UML and patterns”)

Other references

- *The Rational Unified Process — An Introduction* by Philippe Kruchten
- *The Unified Software Development Process* by Jacobson, Booch, and Rumbaugh
- *Software Engineering Body of Knowledge (SWEBOK)*, available at www.swebok.org
- The *SEI* (www.sei.cmu.edu) has several proposals related to quality requirements
- The *ISO 9126*, *IEEE Std 830*, and *IEEE Std 1061* are standards related to requirements and quality attributes, and available on the Web at various sites