Co-funded by the
Erasmus+ Programme
of the European Union

SOFIA UNIVERSITY
"ST. KLIMENT OHRIDSKI
EST. 1888

ICT-TEX

ICT-TEX course on Digital skills

Topic 4: Introduction to Software Engineering

The course is developed under Erasmus+ Program Key Action 2:
Cooperation for innovation and the exchange of good practices Knowledge Alliance

ICT IN TEXTILE AND CLOTHING HIGHER EDUCATION AND BUSINESS

Project Nr. 612248-EPP-1-2019-1-BG-EPPKA2-KA

The information and views set out in this publication are those of the authors and do not necessarily reflect the official opinion of the European Union. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein.

Co-funded by the
Erasmus+ Programme
of the European Union

4.1. Software Engineering

Co-funded by the Erasmus+ Programme of the European Union

Contents

4.1. Software Engineering

- Software Engineering
- Software
- Software processes
- Software development processes
- Software engineering method – definition
- Software process model
- Software process models examples

4.2. Requirements Engineering

- Requirements Engineering
- Requirements
- Requirements classification
- Non-functional requirements classification
- Metrics for specifying non-functional requirements
- Requirements engineering process
- Requirements engineering process and stakeholders
- Software Requirements Specification or (SRS)
- Software requirements specification structure
- Users of software requirements specification

Software Engineering

- Definition of Software Engineering

"Software engineering is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use."

- Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, 2016, pp. 21

Software Engineering

- Two important aspects of the definition of Software Engineering
  - **Software** (definition, types of software products, application in different areas, reuse, importance and so on.)
  - **Engineering** (the process of production of the software, including additional activities such as the development of tools, theories, and means to support the software development process; risk project management, and so on.)

Co-funded by the
Erasmus+ Programme
of the European Union

Software Engineering

- **Software Engineering** represents a systematic approach to the development of **high quality software**, as well as to its **marketing**, **operation** and **maintenance**.

- **Software Engineering** is a discipline that deals with all aspects of the design and development of **high-quality** software.

Software

Definition of Software

"Software is: (1) instructions (computer programs) that when executed provide desired features, function, and performance; (2) data structures that enable the programs to adequately manipulate information, and (3) descriptive information in both hard copy and virtual forms that describes the operation and use of the programs."

- Pressman, R., Maxim, B. Software Engineering: A Practitioner's Approach. 9th edition, Published by McGraw-Hill Education, ISBN: 9781260548006, 2019, pp.5

Back to Contents

Co-funded by the
Erasmus+ Programme
of the European Union

Application of software

The classification by R. Pressman, presents the *application of software as*:

- System software
- Application software
- Engineering/scientific software
- Embedded software
- Product-line software
- Web/mobile applications
- Artificial intelligence software

Pressman, R., Maxim, B. Software Engineering: A Practitioner's Approach. 9th edition, Published by McGraw-Hill Education, ISBN: 9781260548006, 2019, pp.7

Co-funded by the
Erasmus+ Programme
of the European Union

Software as a product

The *software* that is intended to be sold (to derive financial benefits and so on) is called a **(software) product**.

Software products can be two main types:

- Generic software products
- Customized software products

Co-funded by the Erasmus+ Programme of the European Union

Essential attributes of good software

The product characteristics of good software can be:

- Acceptability
- Dependability and security
- Efficiency
- Maintainability

Co-funded by the
Erasmus+ Programme
of the European Union

Software processes

- *"Software Process as a framework for the activities, actions, and tasks required to build high-quality software."*
- *Software Processes* describe and control the life cycle of a software product.
- The software product life cycle can include:
  - The Beginning. The emergence of the idea of creating a product and
  - The End. The moment when its use is discontinued.

Pressman, R., Maxim, B. Software Engineering: A Practitioner's Approach. 9th edition, Published by McGraw-Hill Education, ISBN: 9781260548006, 2019, pp.20

Co-funded by the
Erasmus+ Programme
of the European Union

Main activities at every software process

Specification - what the software should do and what its scope is.

Design and implementation - writing code, programming the software product is produced according to the specifications. It can also be called *Development process*.

Validation - check if the software is what the user wants.

Evolution - a change in the software as a result of changes in the requirements for it.

Co-funded by the
Erasmus+ Programme
of the European Union

Software development processes

Software Development Processes means the set of activities related to the development of the software product from the beginning to the end of its life cycle.

Software Development Process is the entire process of setting the task, planning, implementation, and evaluation of a software and hardware application, including the tools used, methods, and necessary staff

Co-funded by the Erasmus+ Programme of the European Union

Software engineering process framework

A Generic Process framework for Software Engineering - applicable to all software projects (no matter size or complexity)

Includes main Umbrella and framework process activities:

- Communication
- Planning
- Modeling
- Construction
- Deployment

Co-funded by the Erasmus+ Programme of the European Union

Software engineering process – Umbrella Activities

- The *Umbrella Activities* applicable during the development process of software projects
- Support the management, quality assurance, and risk management.

Umbrella Activities classification by Pressman, R., Maxim, B. Software Engineering: A Practitioner's Approach. 9th edition, Published by McGraw-Hill Education, ISBN: 9781260548006, 2019, pp.11

Back to Contents

Co-funded by the
Erasmus+ Programme
of the European Union

Software engineering process – Umbrella Activities

The main *Umbrella Activities* include:

- Software project tracking and control.
- Risk management
- Software quality assurance
- Technical reviews
- Measurement.
- Software configuration management
- Reusability management
- Work product preparation and production.

Back to Contents

Umbrella Activities classification by Pressman, R., Maxim, B. Software Engineering: A Practitioner's Approach. 9th edition, Published by McGraw-Hill Education, ISBN: 9781260548006, 2019, pp.11

Co-funded by the Erasmus+ Programme of the European Union

Software engineering method – definition

- *The method of software engineering* is a structured approach to software development that aims to facilitate the creation of cost-effective high-quality software.

Co-funded by the Erasmus+ Programme of the European Union

Software engineering methods

- Software engineering methods include:
  - **Model description** - describes models of the software to be developed. For example: (Data flow model; Structural model; Object model; Incremental and Iterative model; Prototyping model, etc.)
  - **Rules** - restrictions applicable to system models
  - **Recommendations** - tips for good design practices.
  - **Description of the process** - what activities should be performed and in what order.

Back to Contents

Co-funded by the Erasmus+ Programme of the European Union

Software process model

- The abstract presentation of the software process from a certain perspective represents the **Model of the Software Process.**
- *A "Software Process Model is a simplified representation of a software process. Each process model represents a process from a particular perspective and thus only provides partial information about that process"*

- Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, 2016, pp. 45

Co-funded by the Erasmus+ Programme of the European Union

Software process model description

Description of Software Process Models

- Process flow – a description of the sequence of activities, actions, and tasks that are performed concerning sequence and time: (Linear process flow, Iterative process flow, Evolutionary process flow, Parallel process flow)
- Workflow - as a sequence of activities that are performed
- Data flow - the flow of information between the various stages in the process is considered
- Role/action - who does what.

Co-funded by the Erasmus+ Programme of the European Union

Software process models examples

Waterfall Model

Evolutionary models - Incremental and Iterative Development, Prototype models, Spiral Model

V Model

Agile - Scrum, Extreme Programing - XP, Kanban, Feature-Driven Development (FDD), Dynamic systems development method (DSDM) and so on.

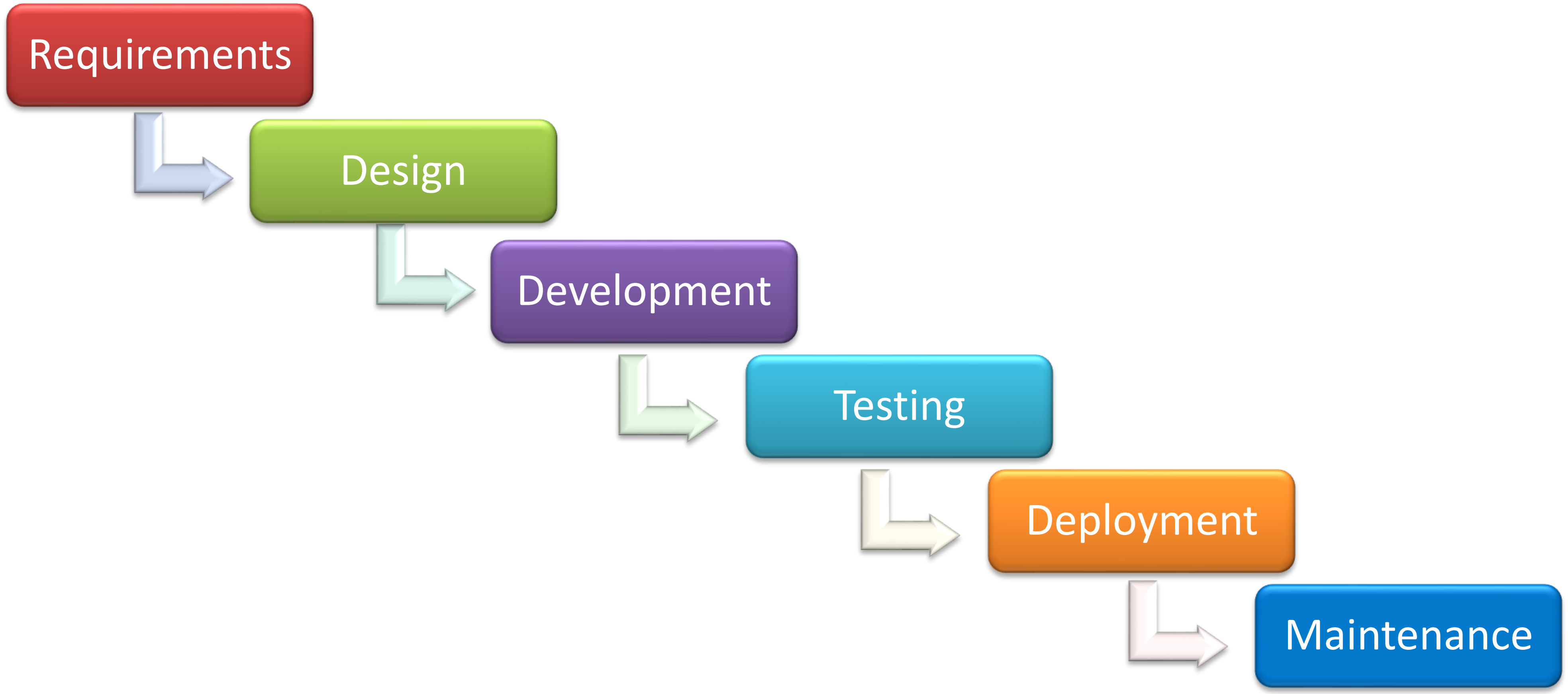
Co-funded by the Erasmus+ Programme of the European Union

Waterfall model

- The oldest method of software development is based on the so-called Waterfall model. It consists of a sequence of steps (stages), each stage having to be completed before the next one starts. At the end of each stage, as a result, we have certain documents, data, code, and other artifacts.

Co-funded by the Erasmus+ Programme of the European Union

Waterfall model



Co-funded by the
Erasmus+ Programme
of the European Union

Waterfall model stages

The stages in the Waterfall model are as follows:

- Requirements
- Design
- Development
- Testing
- Deployment
- Maintenance

Co-funded by the
Erasmus+ Programme
of the European Union

Waterfall model stages

Requirements

- What the software should do and what its scope is. Intensive communication with the client and stakeholders. The specification must be elaborated in detail and focused on the hardware, software, requirements of the contracting authority, and with priority to implementation. At the end of this stage, the *Software Requirement Specification document* (SRS) is prepared.

Co-funded by the
Erasmus+ Programme
of the European Union

Waterfall model stages

Design stage

- The design creates a detailed product design, models at different levels of abstraction, and different parts or aspects of the project. As a result, a description of the software architecture is used, various system components and connections between them are identified, the necessary hardware and software resources for goal setting, etc. A schedule for the implementation of the project is made, they are distributed and the edition of their implementation is determined. The risk and the necessary resources (time, people, equipment, money) are assessed

Co-funded by the Erasmus+ Programme of the European Union

Waterfall model stages

Development stage

- At this stage, the coding of the model in selected programming languages is used.

Testing

- This is a process of checking the written code. Testing can be performed on modules as well as on the whole system. The process is very long and accompanies implementation and integration. Testing is a key step in correcting errors and omissions. Thanks to testing, it is possible to adapt the entire system to the specification. When errors are detected - return to the previous stages.

Co-funded by the
Erasmus+ Programme
of the European Union

Waterfall model stages

Maintenance stage

- At this stage, the components of the system are assembled into a new system and its properties are revealed. It is mandatory to test the new system by the developer and the contracting authority - whether it meets the specification. Upon successful completion of the tests, the new system reaches its end users and starts working.

Co-funded by the Erasmus+ Programme of the European Union

Waterfall model stages

Deployment stage

- This stage represents the activity of making changes and improvements in integrated software by establishing good communication with the client and the users of the system. The modification of the software is in a corrective, adaptive direction and with subsequent improvement of the product, by improving existing or adding new functionalities.

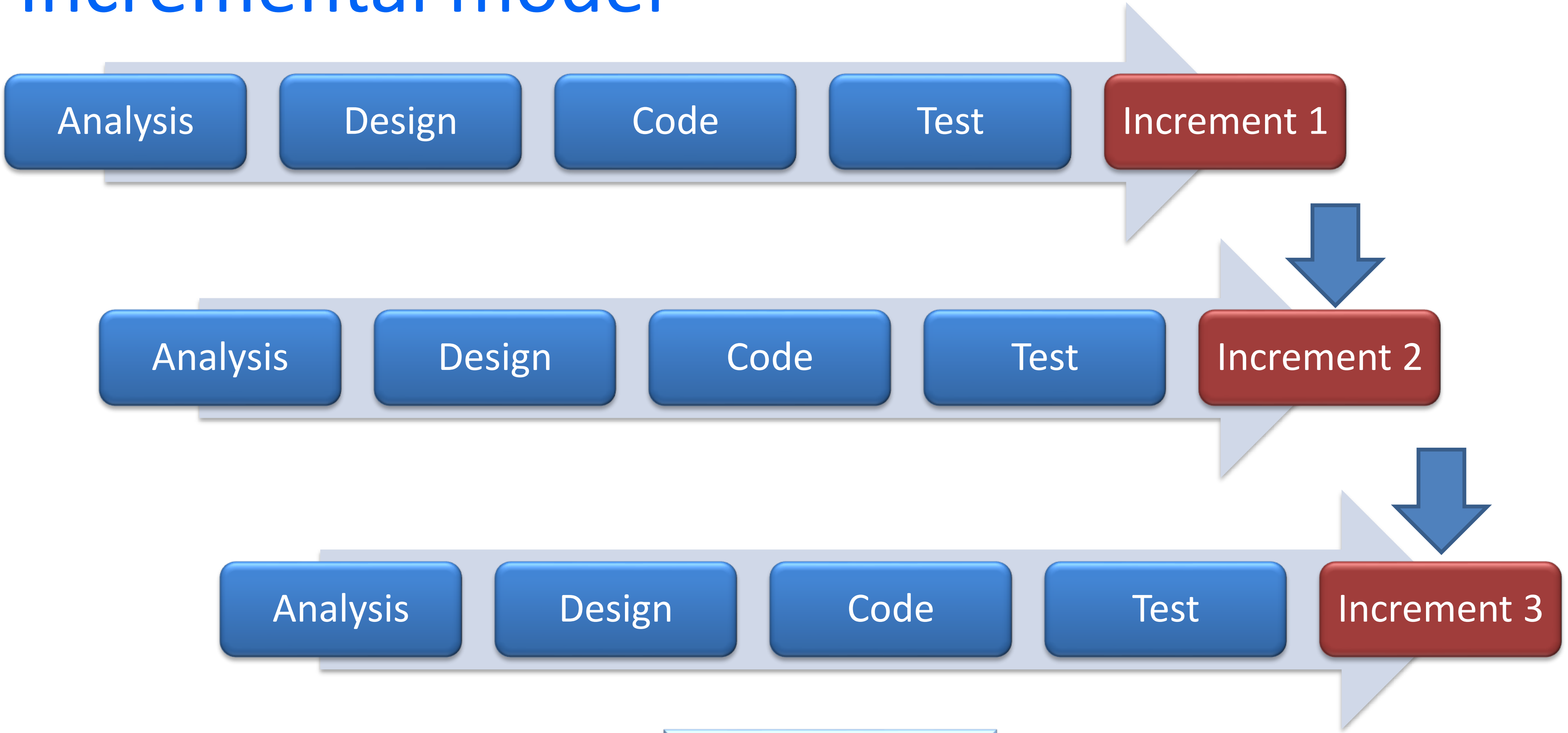
Back to Contents

Co-funded by the
Erasmus+ Programme
of the European Union

SOFIA UNIVERSITY
"ST. KLIMENT OHRIDSKI
EST. 1888

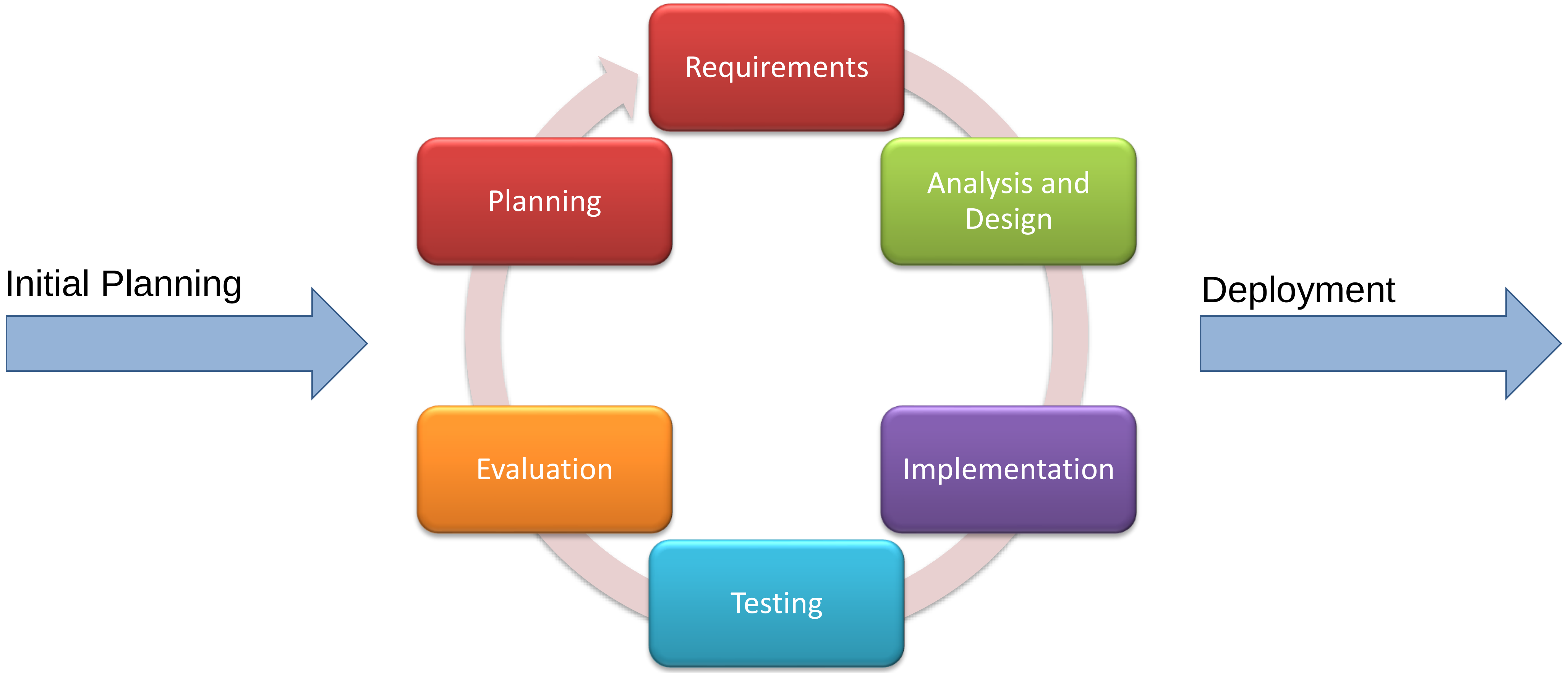
ICT-TEX

Incremental model



Back to Contents

Iterative model



Back to Contents

Co-funded by the Erasmus+ Programme of the European Union

SOFIA UNIVERSITY "ST. KLIMENT OHRIDSKI" EST. 1888

ICT-TEX

Agile model – 12 Principles

Customer Satisfaction	Welcome Change	Deliver Frequently	Working Together
Motivated Team	Face-to-Face	Working Software	Constant Pace
Good Design	Simplicity	Self Organization	Reflect and Adjust

Back to Contents

Co-funded by the Erasmus+ Programme of the European Union

Agile models

Learn more about:	At Web page:
Scrum	https://www.scrum.org/resources/what-is-scrum
Scaled Agile Framework (SAFe)	https://www.atlassian.com/agile/agile-at-scale/what-is-safe
Dynamic Systems Development technique (DSDM)	https://www.geeksforgeeks.org/dynamic-systems-development-method-dsdm/
Crystal	https://airfocus.com/glossary/what-is-the-crystal-agile-framework/

Back to Contents

Agile models

Learn more about:	At Web page:
Extreme Programming (XP)	https://ronjeffries.com/xprog/what-is-extreme-programming/
Kanban	https://kanbanize.com/kanban-resources/getting-started/what-is-kanban
Lean	https://medium.com/kayvan-kaseb/using-lean-in-software-development-1b01bbb98d6e
Feature-Driven Development (FDD)	https://www.lucidchart.com/blog/why-use-feature-driven-development

Co-funded by the Erasmus+ Programme of the European Union

4.2. Requirements Engineering

Back to Contents

Co-funded by the Erasmus+ Programme of the European Union

Requirements Engineering

Requirements engineering is *"the term for the broad spectrum of tasks and techniques that lead to an understanding of requirements. From a software process perspective, requirements engineering is a major software engineering action that begins during the communication activity and continues into the modeling activity. Requirements engineering establishes a solid base for design and construction. It must be adapted to the needs of the process, the project, the product, and the people doing the work."*

- *Pressman, R., Maxim, B. Software Engineering: A Practitioner's Approach. 9th edition, Published by McGraw-Hill Education, ISBN: 9781260548006, 2019, pp.103*

Back to Contents

Co-funded by the Erasmus+ Programme of the European Union

Requirements Engineering

"Requirements engineering is the subset of systems engineering concerned with discovering, developing, tracing, analyzing, qualifying, communicating and managing requirements that define the system at successive levels of abstraction."

- *Dick J., Hull E., Jackson K. Introduction. In: Requirements Engineering. Springer, Cham. https://doi.org/10.1007/978-3-319-61073-3_1, ISBN: 978-3-319-61073-3, (2017) , pp.9*

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements Engineering

Requirements Engineering is the process of establishing the services that the customer requires from the software system, as well as the constraints under which it operates and is created.

Requirements Engineering is a discipline of software technology that encompasses the activities of the specification of software products or systems.

Co-funded by the Erasmus+ Programme of the European Union

Requirements

The *requirements* themselves are a description of the services and limitations that the system has, which are generated during the Requirements Engineering Process.

The *requirements* are a specification of what needs to be implemented. They describe how the system should behave or a specific property or attribute of the system. They are also a requirement for the system development process.

Co-funded by the Erasmus+ Programme of the European Union

Requirements

- The requirements help us determine:
  - What the benefits of the system would be,
  - What it should do, and
  - Some general understanding of its capabilities.
- They can also help identify potential difficulties in implementing the system at an early stage.
- They affect the entire life cycle of a system.

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements

- **Well-defined** requirements contribute to a good and successful product.
- **Poorly defined** requirements lead the product to failure.
- In different cases, the requirements may have different levels of detail. From the general that the system will offer a service, to a specific description in detail.

Co-funded by the Erasmus+ Programme of the European Union

Requirements classification

The Requirements can be divided into the following categories

Depending on the group that uses or describes them

Depending on what they describe

Back to Contents

Co-funded by the Erasmus+ Programme of the European Union

Requirements classification

Depending on the group that uses or describes them are:

- User requirements
- Business requirements
- System requirements

Co-funded by the Erasmus+ Programme of the European Union

Requirements classification

User requirements

- These are the requirements of the user (client) to the system. They are usually described in natural language or user terminology. They include diagrams of the services that the system will provide, as well as the corresponding operational constraints. They are written for the clients.

Requirements classification

Business requirements

- These are the requirements of the business to the system. These are the costs or budget for development, the time for development, the technologies or processes used, and others.

Back to Contents

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements classification

System requirements

- These are detailed system requirements, described in non-contradictory language. They should be understandable to both technical staff and customers. Structured document with a detailed description of system functions, services, and operational constraints. These types of requirements determine what needs to be implemented and can be part of the contract for the development of the system.

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements classification

Depending on what the requirements describe are:

- Functional requirements
- Non-functional requirements
- Domain requirements

Requirements classification

Functional requirements

- These are requirements that describe the behavior of the system. They describe the different services that the system provides, the way the system should respond to specific inputs and behavior, and in specific situations.
  - User functional requirements can only describe the basic idea of a given service, but
  - System functional requirements must describe the system at the best possible level of detail.

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements classification

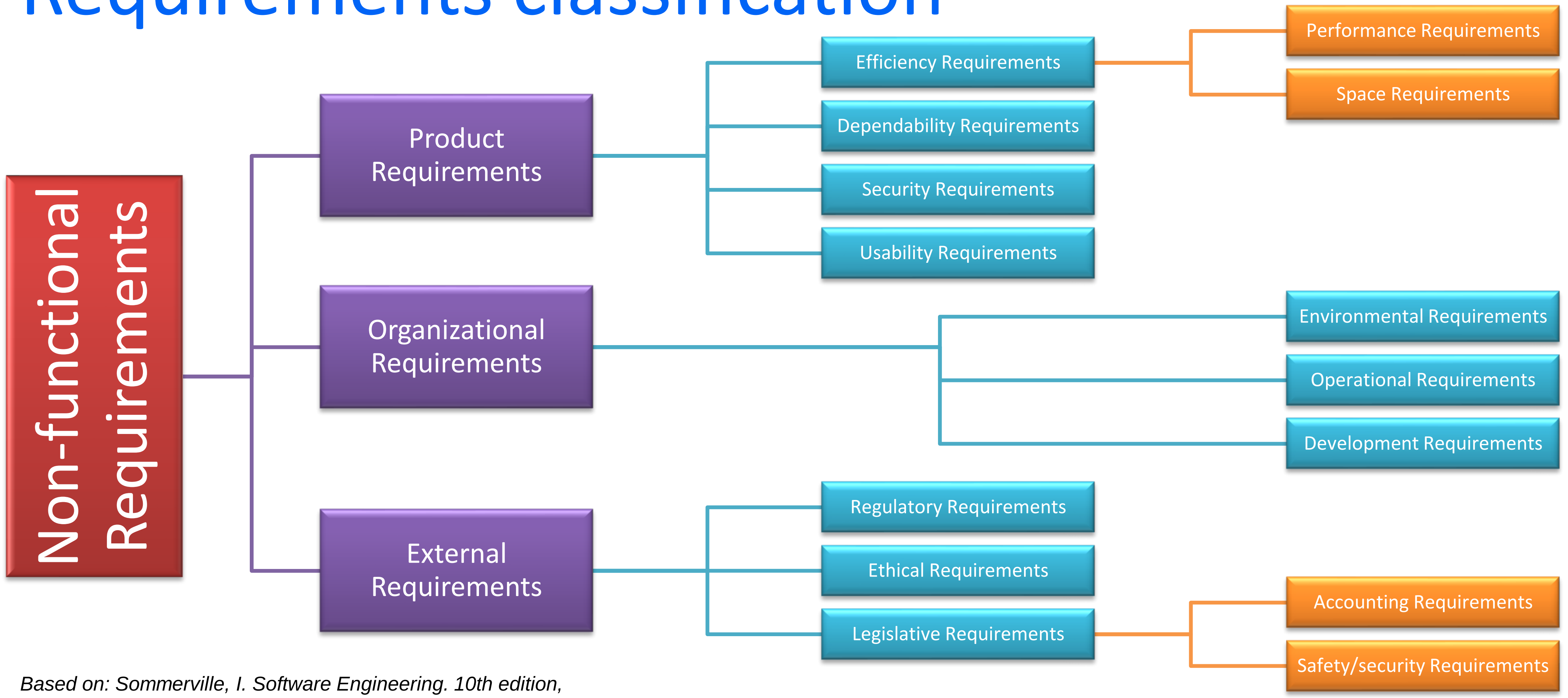
Non-functional requirements (quality attribute requirements)

- These are requirements describing limitations in the system or development process. These can be requirements for performance, reliability, security, performance, capacity, or other resources, technologies, and processes used, etc.

Domain requirements

- These are requirements produced by the application area of the system that reflects its specifics.

Requirements classification



Based on: Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, 2016

Back to Contents

Co-funded by the
Erasmus+ Programme
of the European Union

SOFIA UNIVERSITY
"ST. KLIMENT OHRIDSKI
EST. 1888

ICT-TEX

Non-functional requirements classification

They are divided into three main categories:

- Product requirements
- Organizational requirements
- External requirements

Back to Contents

Co-funded by the
Erasmus+ Programme
of the European Union

Non-functional requirements classification

Product requirements

- Specify or constrain the runtime behavior of the software.
- Examples for non-functional product requirements are:
  - ✓Performance requirements
  - ✓Usability requirements
  - ✓Efficiency requirements
  - ✓Dependability requirements
  - ✓Security requirements, and Space requirements.

Co-funded by the Erasmus+ Programme of the European Union

Non-functional requirements classification

Organizational requirements

- These requirements are "*all requirements that are derived from factors external to the system and its development process.*"

-*Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, 2016, pp. 108*

Back to Contents

Non-functional requirements classification

External requirements

- These requirements are non-functional "*system requirements derived from policies and procedures in the customer's and developer's organizations.*"

  -*Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, 2016, pp. 108-109*

- Examples for non-functional organizational requirements:
  - Environmental requirements;
  - Operational requirements;
  - Development requirements.

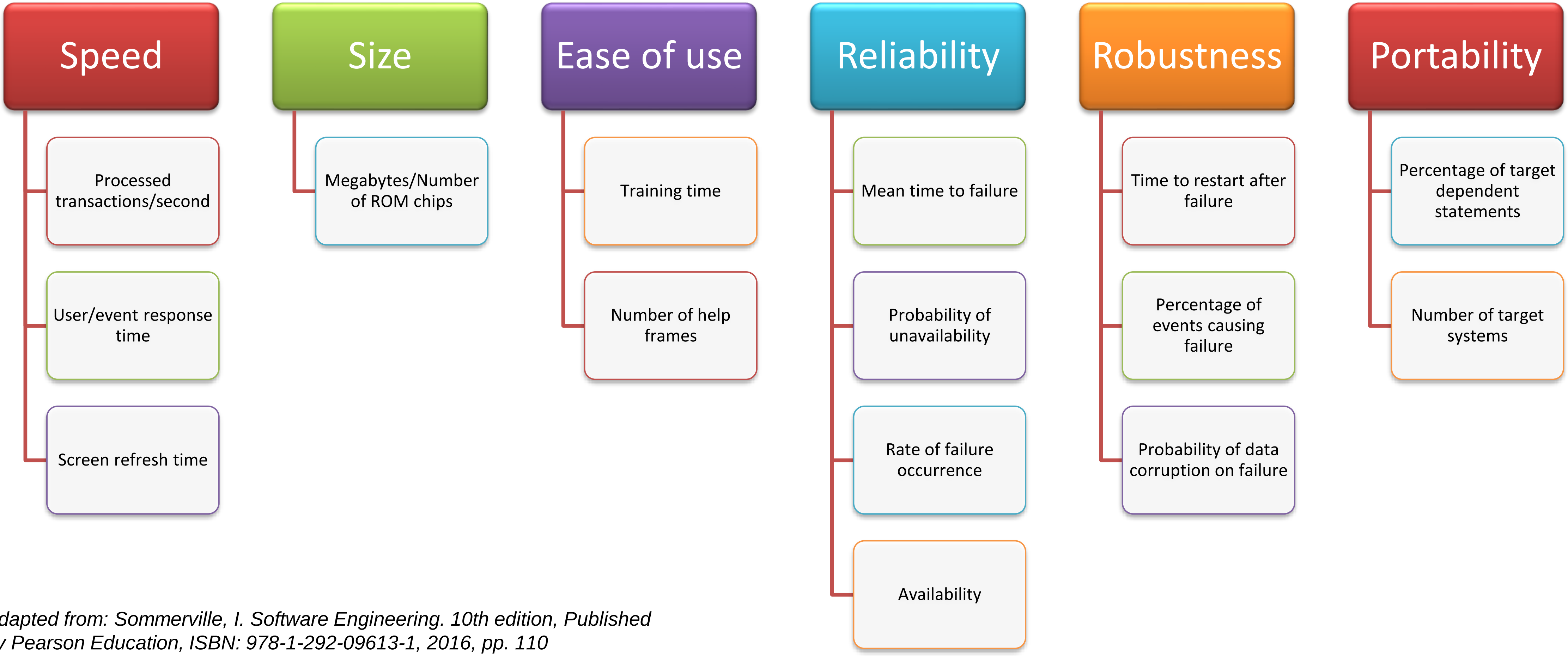
Non-functional requirements classification

External requirements

- Examples for non-functional external requirements:
  - ✓Regulatory requirements
  - ✓Ethical requirements
  - ✓Legislative requirements
  - ✓Accounting requirements
  - ✓Safety/security requirements

Back to Contents

Non-functional requirements classification



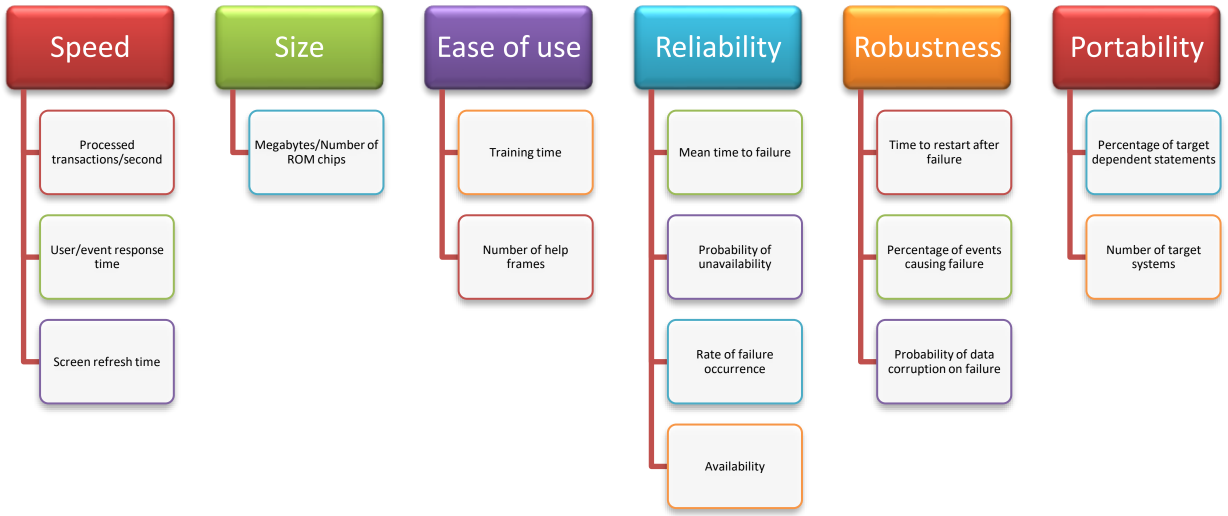
- Speed
  - Processed transactions/second
  - User/event response time
  - Screen refresh time
- Size
  - Megabytes/Number of ROM chips
- Ease of use
  - Training time
  - Number of help frames
- Reliability
  - Mean time to failure
  - Probability of unavailability
  - Rate of failure occurrence
  - Availability
- Robustness
  - Time to restart after failure
  - Percentage of events causing failure
  - Probability of data corruption on failure
- Portability
  - Percentage of target dependent statements
  - Number of target systems

Adapted from: Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, 2016, pp. 110

Back to Contents

Metrics for specifying non-functional requirements

- The table presents metrics for specifying non-functional system properties.
- It gives the possibility for measurement of the properties (characteristics) when the system is being tested to verify that the system has met its non-functional requirements.


Speed
Processed transactions/second
User/event response time
Screen refresh time
Size
Megabytes/Number of ROM chips
Ease of use
Training time
Number of help frames
Reliability
Mean time to failure
Probability of unavailability
Rate of failure occurrence
Availability
Robustness
Time to restart after failure
Percentage of events causing failure
Probability of data corruption on failure
Portability
Percentage of target dependent statements
Number of target systems

Adapted from: Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, 2016, pp. 110

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements engineering process

The engineering processes of the requirements may differ, but all of them have the following characteristic activities:

- *Identification of requirements*
- *Analysis and specification of requirements*
- *Validation of requirements*
- *Requirements management*

Co-funded by the Erasmus+ Programme of the European Union

Requirements engineering process

Identification of requirements

- The technical staff and the clients take part in this activity. The main goal is to describe in maximum detail what the software system expects to do. In addition to customers, this information may be collected based on other similar systems or information about the area in which the product will operate.

Requirements engineering process

Analysis and specification of requirements

- Usually, customer requirements are in colloquial (non-formal) language and there can be many ambiguities and inconsistencies. Clients and external sources of information can again participate in the analysis process, and this time the goal is to detail and specify the requirements, as well as to clear the contradictions if any.

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements engineering process

Analysis and specification of requirements

- In the analysis of the requirements, the priority of the requirements is determined and they are grouped according to different criteria. The requirements analysis results in many documents that describe the requirements in technical language using diagrams or unambiguous terminology.

Co-funded by the Erasmus+ Programme of the European Union

Requirements engineering process

Validation of requirements

- This activity is interested in whether the system described by the discovered and analyzed requirements is what the user really wants. Requirement level error can cost much more than any implementation error.
- What is monitored in this activity is:
  - ✓Validity
  - ✓Consistency
  - ✓Completeness
  - ✓Realism
  - ✓Verifiability

Requirements engineering process

Validation of requirements

- *Validity* - whether these are the functions that the user needs.
- *Consistency* - whether there are no conflicts between the individual requirements.
- *Completeness* - whether all possible requirements are included.

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements engineering process

Validation of requirements

- *Realism* - is it possible for the requirements to be implemented within the given budget and technology. Usually, prototypes of single modules in the system are made in this activity to see how feasible what is required is.
- *Verifiability* - whether the requirements can be verified. Sample test scenarios are usually done here. Verifiability is very important because it allows us to verify that the resulting system does what we want

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements engineering process

Requirements management

- In the life cycle of a system, it is often necessary to make changes to it, either as a result of changing customer requirements or as a result of the limitations of the technologies used.
- Any sufficiently large change in the way the system operates should be reflected in the documents describing the requirements for it.
- The activity of requirements management is precisely to ensure that the documents describing the requirements are in sync with the current desired state of the system.

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements engineering process and stakeholders

In the Requirements Engineering Process are involved many people (stakeholders) who have different roles

- Clients
- Programmers
- Testers
- Requirements analyzers
- Management and others

Requirements engineering process and stakeholders

Clients

- Customers are the people who demand the creation of the system. This includes the management of the company (if we have a product for the market), the customer who makes the order (if a product is made for a specific customer), and the users of the product.
- What they all have in common is that they understand the product on their own terms. They know what they want the product to do, but they have no technical knowledge of how it can be realized, nor the limitations imposed by the technology used.

Co-funded by the Erasmus+ Programme of the European Union

Requirements engineering process and stakeholders

Programmers

- These are the people who develop the system.
- They know the technology and its limitations and capabilities very well but rarely understand customer terminology.
- This includes hardware professionals who know the capabilities of the physical machines on which the system will run.
- These are the people who care about the realism of the requirements

Co-funded by the Erasmus+ Programme of the European Union

Requirements engineering process and stakeholders

Testers

- These are the people who deal with the verifiability of the requirements.
- At the stage of product development, these will be the people who will monitor whether the resulting product meets the described requirements.

Co-funded by the Erasmus+ Programme of the European Union

Requirements engineering process and stakeholders

Requirements analyzers

- These are the people who drive the process of detecting and analyzing requirements.
- Their main role is to produce documents describing customer requirements in a language accessible to technical staff (programmers and testers), as well as to ensure realism, completeness, and consistency of requirements.

Back to Contents

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements engineering process and stakeholders

Management

- These are the people who manage the requirements analysis process.
- They control and specify the process itself - how many iterations of analysis and validation to perform or under what conditions the requirements are considered sufficient to move to the next stage of the product life cycle.

Co-funded by the Erasmus+ Programme of the European Union

Software Requirements Specification or (SRS)

- **Software Requirements Specification or (SRS)** is the software requirements document
- It presents an official document between the stakeholders, which is followed and implemented by the developers of the system.
- It includes the user requirements for a system and a detailed specification of the system requirements

Co-funded by the
Erasmus+ Programme
of the European Union

Software requirements specification structure

- The possible structure of a Software Requirements Specification may include the following chapters:
  - – Preface
  - – Introduction
  - – Glossary
  - – User requirements definition
  - – System architecture
  - – System requirements specification

Co-funded by the Erasmus+ Programme of the European Union

Software requirements specification structure

- – System models
- – System evolution
- – Appendices
- – Index

More detailed information for the structure of the software requirement document you can find in the book *Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, 2016, pp. 127*

Back to Contents

Users of software requirements specification

According to Ian Sommerville the possible users of the software requirements specification documents are divided into the following groups:

- System customers
- Managers
- System engineers
- System test engineers
- System maintenance engineers

References

- Sommerville, I. Software Engineering. 10th edition, Published by Pearson Education, ISBN: 978-1-292-09613-1, (2016)
- Pressman, R., Maxim, B. Software Engineering: A Practitioner's Approach. 9th edition, Published by McGraw-Hill Education, ISBN: 9781260548006, (2019)
- Dick J., Hull E., Jackson K. Introduction. In: Requirements Engineering. Springer, Cham. https://doi.org/10.1007/978-3-319-61073-3_1, ISBN: 978-3-319-61073-3, (2017)

CONTACTS

Coordinator:
Technical University of Sofia

Project coordinator:
assoc. prof. Angel Terziev, PhD
aterziev@tu-sofia.bg

Web-site: [ICT-TEX.eu](ICT-TEX.eu)

Author:
Assistant professor Yavor Dankov
Sofia University “St. Kliment Ohridski”

Email: yavor.dankov@fmi.uni-sofia.bg
ResearchGate: https://www.researchgate.net/profile/Yavor-Dankov
Scopus: https://www.scopus.com/authid/detail.uri?authorId=57202891597

Co-funded by the Erasmus+ Programme of the European Union

KNOWLEDGE ALLIANCE
ICT-TEX
ICT IN TEXTILE AND CLOTHING HIGHER EDUCATION AND BUSINESS

These slides and the materials included in these slides (including references) are for educational purposes only. The use of slides should be done with correct citation and only for educational purposes.
The information and views set out in this publication are those of the authors and do not necessarily reflect the official opinion of the European Union. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein.