Requirements Engineering & Prioritization in Embedded Software Development:

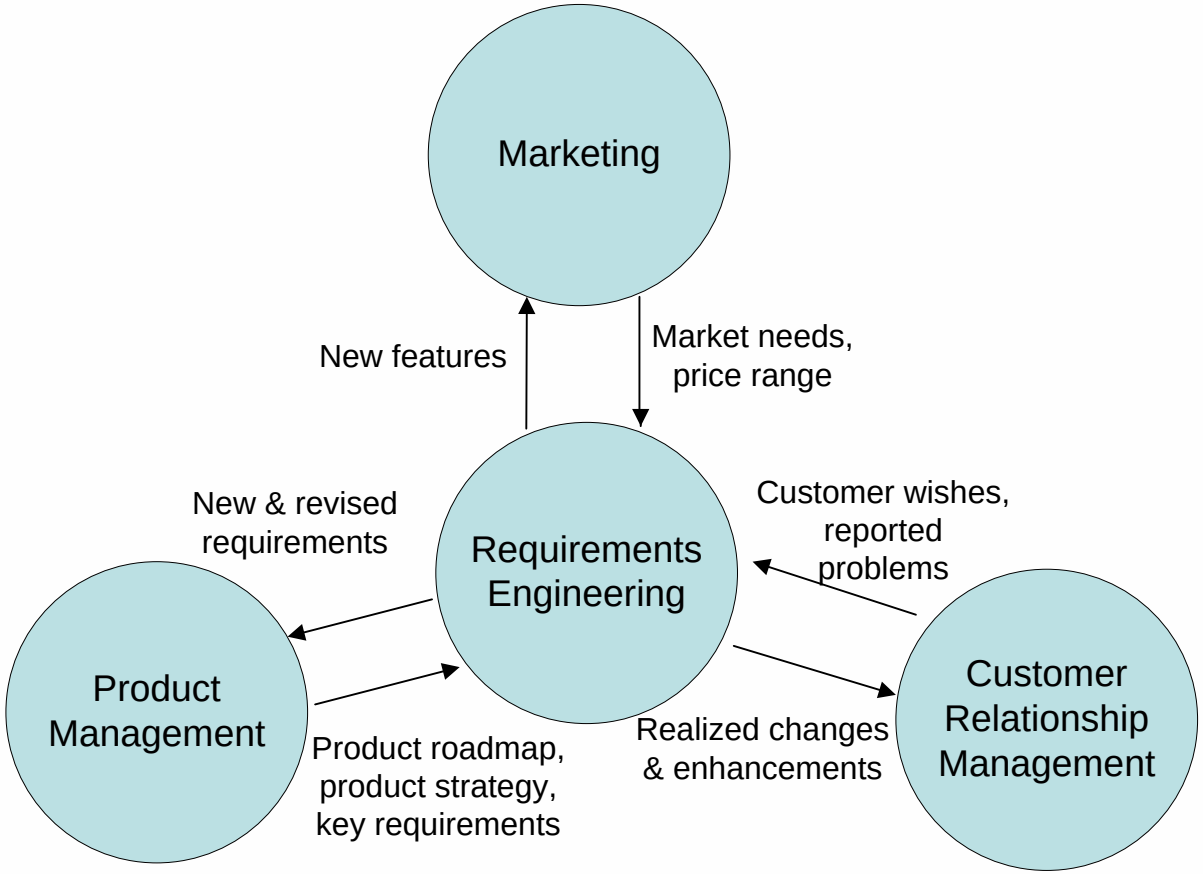
the SPRINT-SMEs Approach

Dr. Vassilis C. Gerogiannis
Associate Professor, Department of Project Management, Technological Education Institute of Larissa, Larissa, Greece (gerogian@teilar.gr)

SPRINT-SMEs: overall description

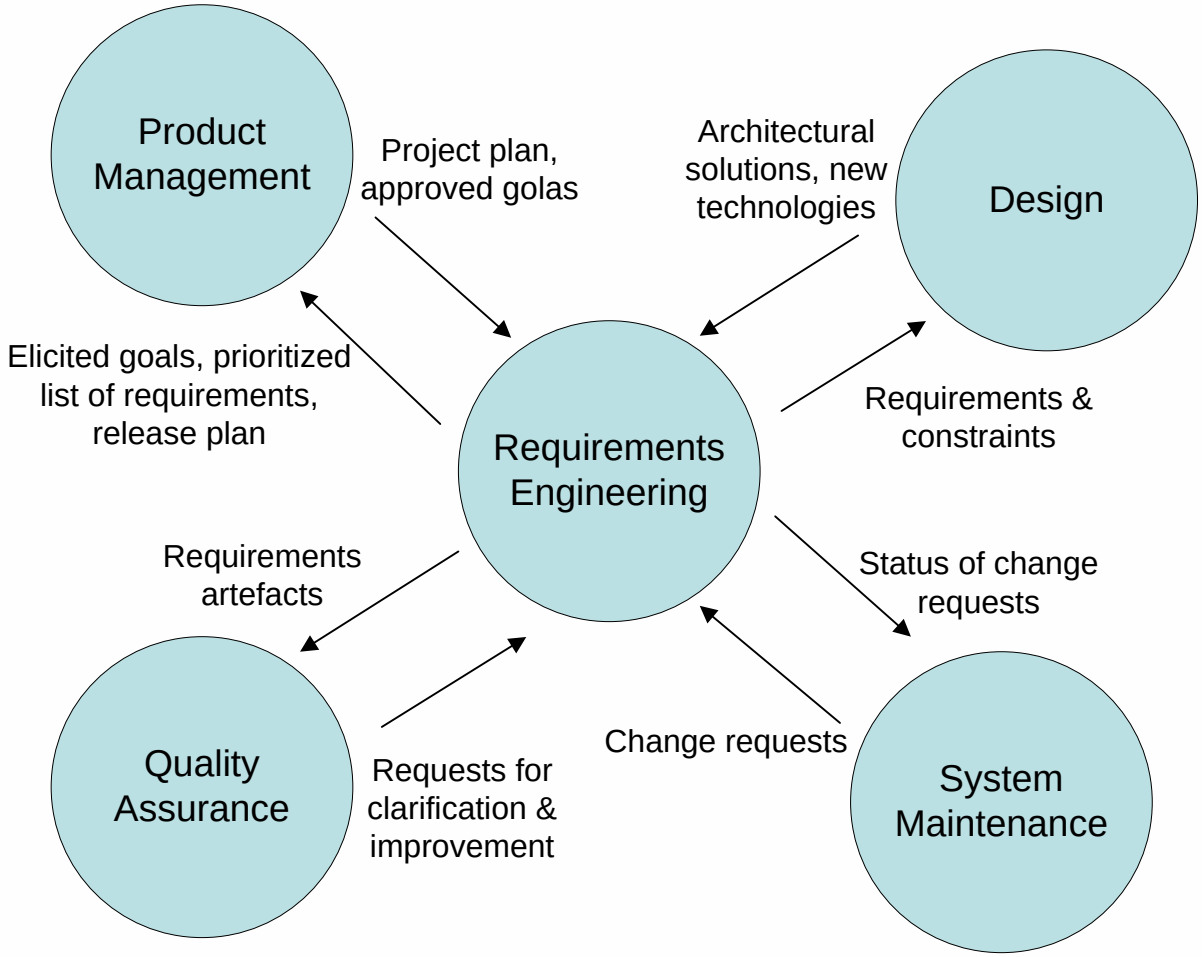
- SPRINT-SMEs (Dec 2012 - Noe 2015) project is funded by the R&D program ARCHIMEDES III (Greek Ministry of Education)
- Research in Software PRocess ImprovemeNT Methodologies for Greek Small & Medium sized Software Development Enterprises (http://sprint.teilar.gr/)
- SPRINT-SMEs objective: develop a practical framework for the improvement of software processes in SW SMEs
- SPRINT-SMEs emphasizes on SPIRE projects (Software Process Improvement in Requirements Engineering)
- Requirements Engineering (RE) is challenging for most SW SMEs
- RE aims at preventing project failures and delivering high quality SW products consistent with end-customers' needs

Interrelations between Requirements Engineering & other organisational processes



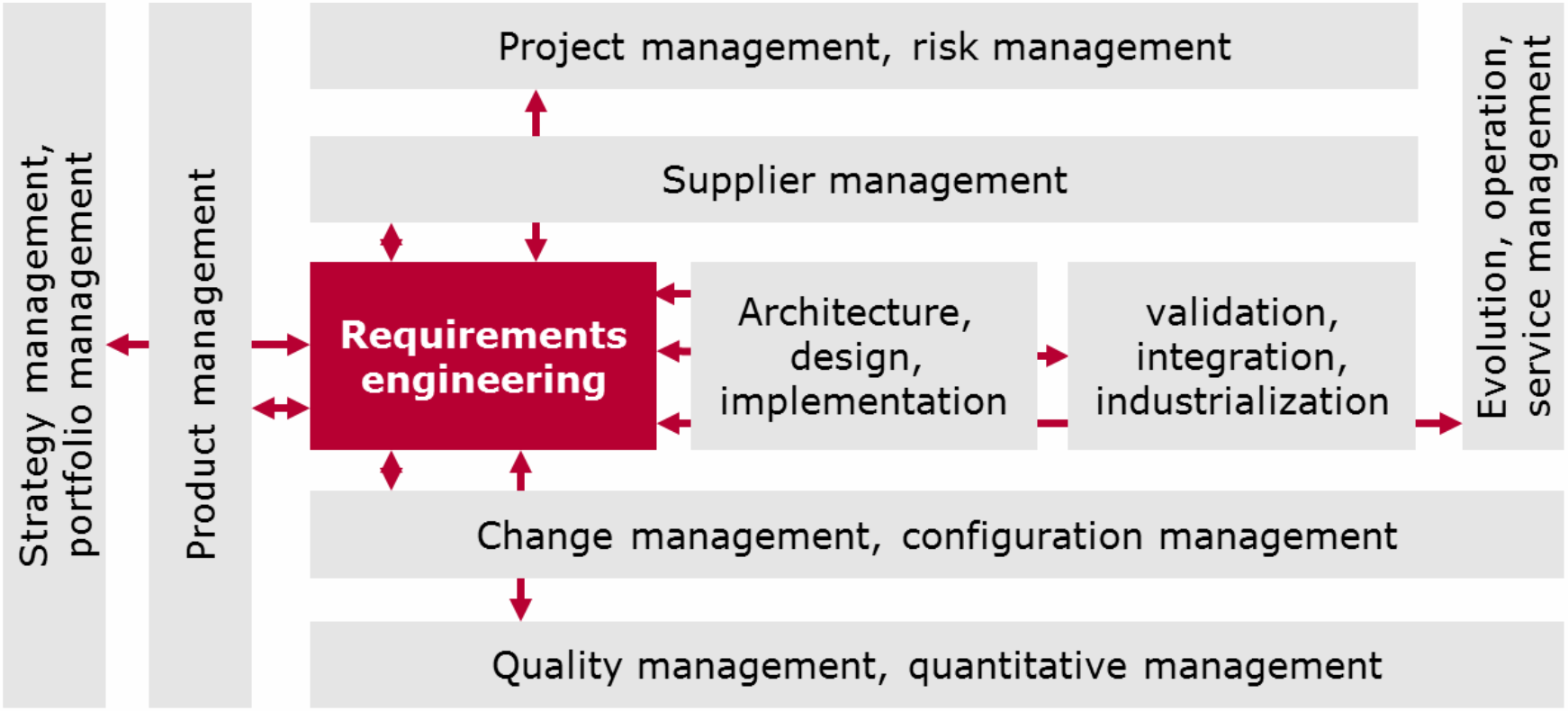
Source: “Requirements Engineering, Fundamentals, Principles & Techniques”, K. Pohl, Springer, 2010

Interrelations between Requirements Engineering & other development activities (1/2)



Source: “Requirements Engineering, Fundamentals, Principles & Techniques”, K. Pohl, Springer, 2010

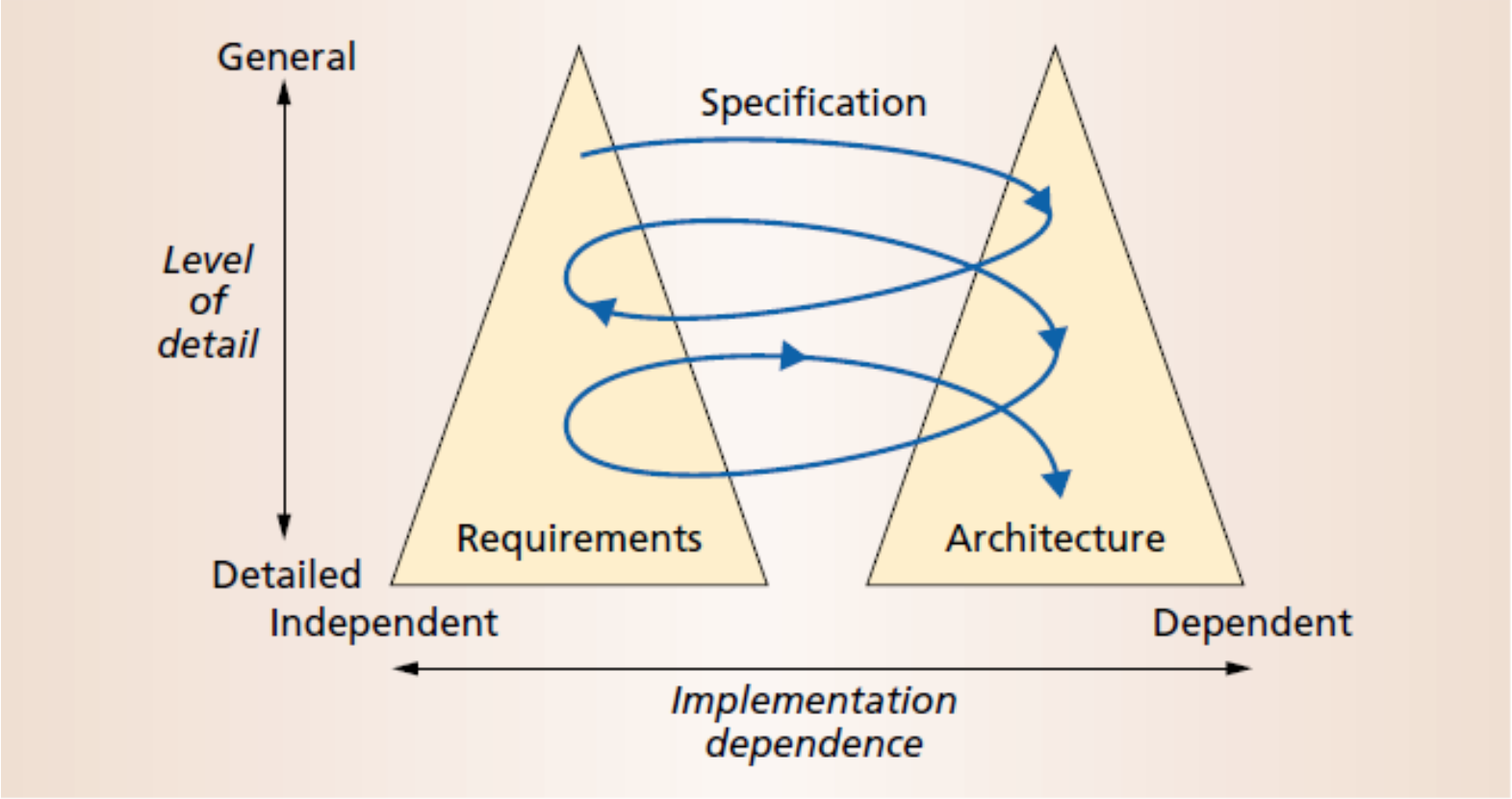
Interrelations between Requirements Engineering & other development activities (2/2)



Source: https://vector.com/vc_requirements_engineering_en.html

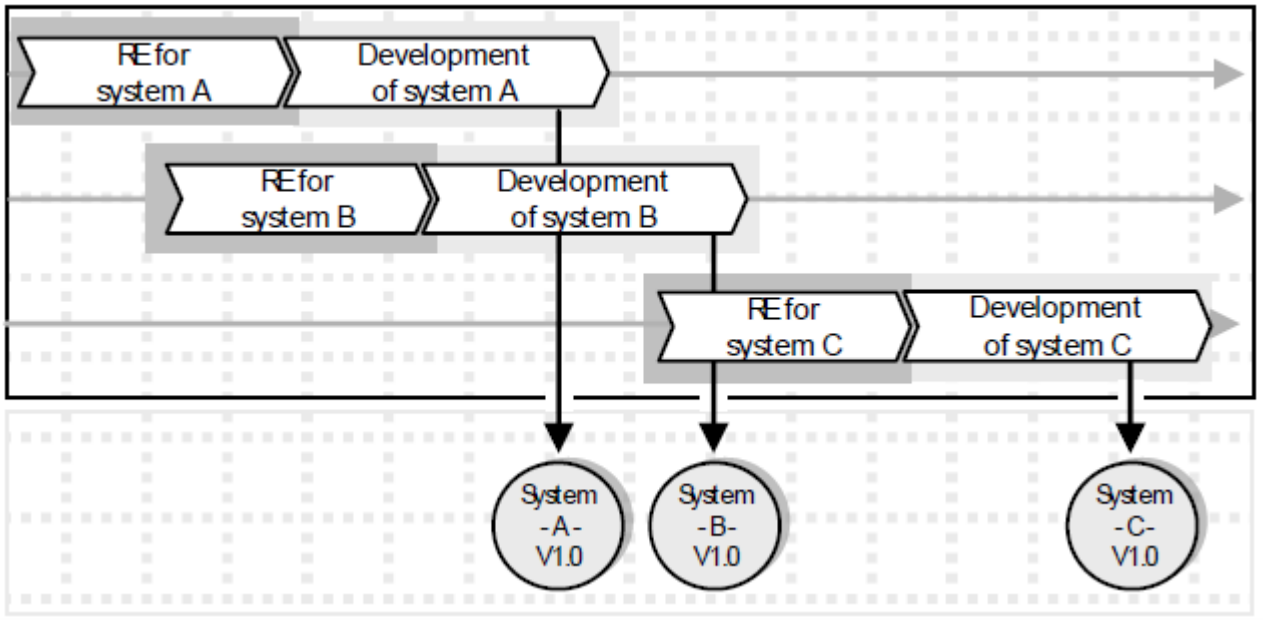
Requirements Engineering in embedded SW development

Interactions between Requirements Engineering & Architectural Design



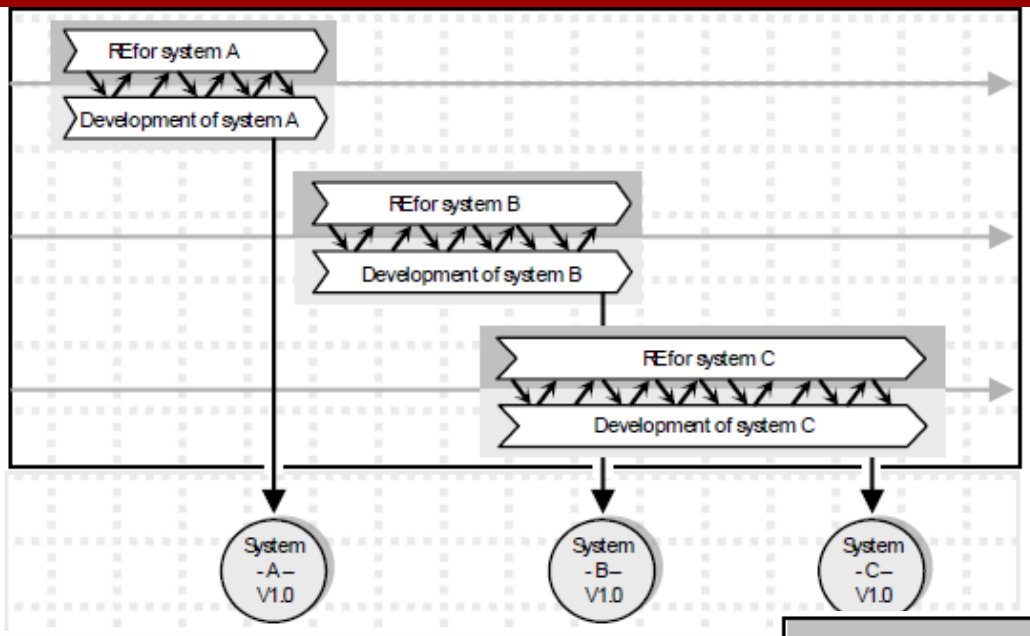
Source: “Weaving Together Requirements and ‚Architectures”, B. Nuseibeh, IEEE Computer, Vol. 34, No. 3, 2001, pp. 115-117

Requirements Engineering is NOT only applied at early development phases



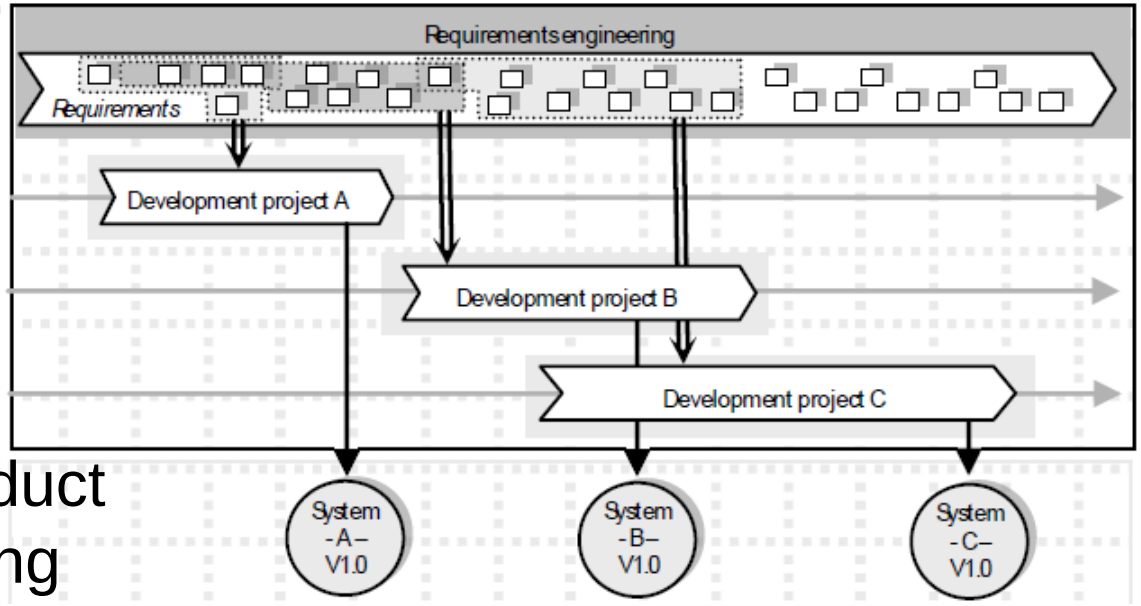
Source: “Requirements Engineering, Fundamentals, Principles & Techniques”, K. Pohl, Springer, 2010

Continuous Requirements Engineering



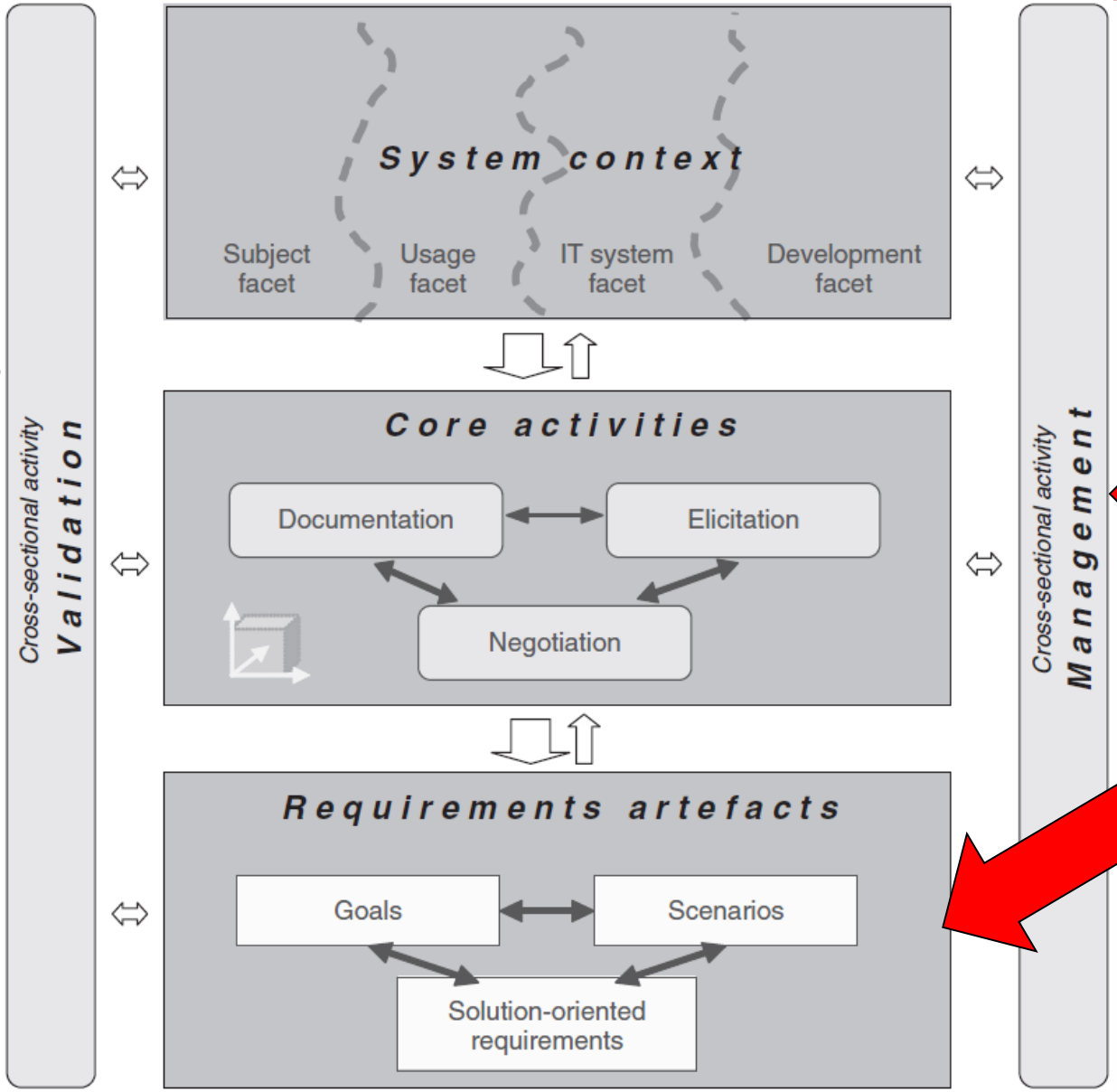
Cross-lifecycle Requirements Engineering

Source: “Requirements Engineering, Fundamentals, Principles & Techniques”, K. Pohl, Springer, 2010



Cross-project & cross-product Requirements Engineering

Requirements Engineering (RE) framework



Source: "Requirements Engineering, Fundamentals, Principles & Techniques", K. Pohl, Springer, 2010

In SPRINT-SMEs we put emphasis on :

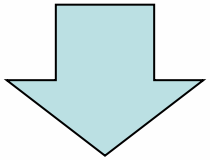
- **Requirements Prioritization**
- **Solutions (Commercial Off-The-Shelf - COTS) evaluation & selection**

Requirements Prioritization: basic concepts

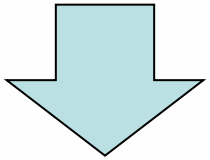
- There are usually more requirements than you can implement given the available time and resource constraints
- The priority of a requirement documents the requirement's importance with regard to one or several prioritization criteria
- The result of the prioritization is typically a (partial) ordering of requirements with regard to the prioritization criteria
- This information can be valuable for managing resources, design / implementation and/or release planning
- "*Prioritization means balancing the business benefit of each requirement against its cost and any implications it has for the architechtural foundation and future evolution of the product* " (Wiegers, 1999)

Requirements Prioritization as a process: issues to be considered / steps to be followed

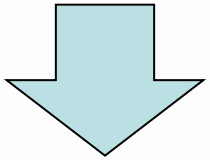
- *Step 1*: What is the context of requirements prioritization (in which development activity requirements should be prioritized)



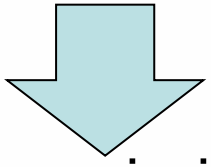
- *Step 2*: Determine involved stakeholders



- *Step 3*: Determine the artefacts to be prioritized



- *Step 4*: Select the prioritization criteria



- *Step 5*: Select and apply a prioritization technique

Prioritization in Requirements Engineering activities

- *Elicitation:* Which requirements should be ellaborated next, which stakeholders should be considered first
- *Documentation:* Determine the order in which the requirements shall be documented
- *Negotiation:* Conflicting requirements are prioritized with respect to their influence to the project success
- *Validation:* Determine the order in which requirements should be validated and detected defects should be resolved
- *Management:* Define which change requests should be processed first
- Particularly, in **embedded software**:
  - Requirements (and their prioritization) influence and partially determine the system architecture
  - Design/architectural decisions influence the selection of requirements

A level of *automation* is required to easily apply prioritization iteratively in different context / activities

Prepare and organize Requirements Prioritization

- Determine the stakeholders to be involved in the prioritization:
  - Users/customers, project manager, product manager, representatives of the development & the quality assurance team etc.
  - Each stakeholder may have a different perspective/importance weight
- Determine the artefacts to be prioritized (goals, scenarios, solution-oriented requirements):
  - Avoid priotitizing requirements from different abstaction levels in one activity. Requirements artefacts at lower abstraction often inherit priorities of corresponding higher-level requirements
  - Prioritization can be problematic (time consuming & not reliable) when the number of prioritization items is too large

The needs and the complexity of the prioritization process should be systematically managed

Select the Prioritization Criteria/Factors

Criterion/Factor	Sub-criteria / Sub-factor
Importance of the requirement	Urgency of implementing a requirement Contribution of a requirement to the system acceptance Importance of a requirement with regard to the impact on the architectural design Importance of a requirement with regard to the marketing position of the business
Cost of requirement implementation	Complexity of the requirement Familiarity/knowledge/experience related with the implementation of the requirement Developers expertise diversity required for the implementation of a requirement Effort required for the implementation of a requirement Degree of reuse associated with a requirement implementation Extend of documentation associated with a requirement and its implementation
Potential damage/disadvantage from not implementing the requirement	Contract penalties that would result from neglecting a requirement Decrease of sales potential that would result from neglecting a requirement Loss of the business prestige that would result from neglecting a requirement
Risk associated with the requirement	Degree of cooperation among developers required when realizing the requirement Required time for design and implementation of requirement Requirement that can be realized fast Risk of exceeding the time schedule and/or deadlines of a release plan Risk of not satisfying the customer Risk of low system performance Risk of project failure
Volatility of the requirement	

Prioritization effort/complexity increases as more criteria/factors are considered

Select the Prioritization Technique

Number/Complexity of requirements		Low number of reqs / Low complexity of prioritization	Large number of reqs / Low complexity of prioritization	Low number of reqs / High complexity of prioritization	Large number of reqs / High complexity of prioritization
Technique	**Effort**				
Ranking	Low	★★★	★★★	★★	★
Top ten	Very low	★★★	★★★	★★	★
One – criterion classification	Low	★★★	★★	★★	★
Kano classification	Low	★★	★★	★★★	★★
Wiegers' prioritization matrix	Medium	★	★	★★★	★★
Cost-value approach (AHP)	High	★	★	★★★	★★

Source: "Requirements Engineering, Fundamentals, Principles & Techniques", K. Pohl, Springer, 2010

- Techniques become impractical as soon as the number of requirements / prioritization criteria increases
- Scalability problems limit severely the applicability of the techniques
- Recent surveys show that ad-hoc prioritization and priority grouping of requirements are the dominant methods

Sources:

- Berntsson-Svensson et al. ,"Quality Requirements in Industrial Practice – an extended interview study at eleven companies", IEEE Transactions on Software Engineering, vol. 38 no. 4, pp. 923-935, 2012
- Berntsson-Svensson et al. ,"Prioritization of quality requirements: State of practice in eleven companies", Requirements Engineering Conference, pp. 69-78, 2011

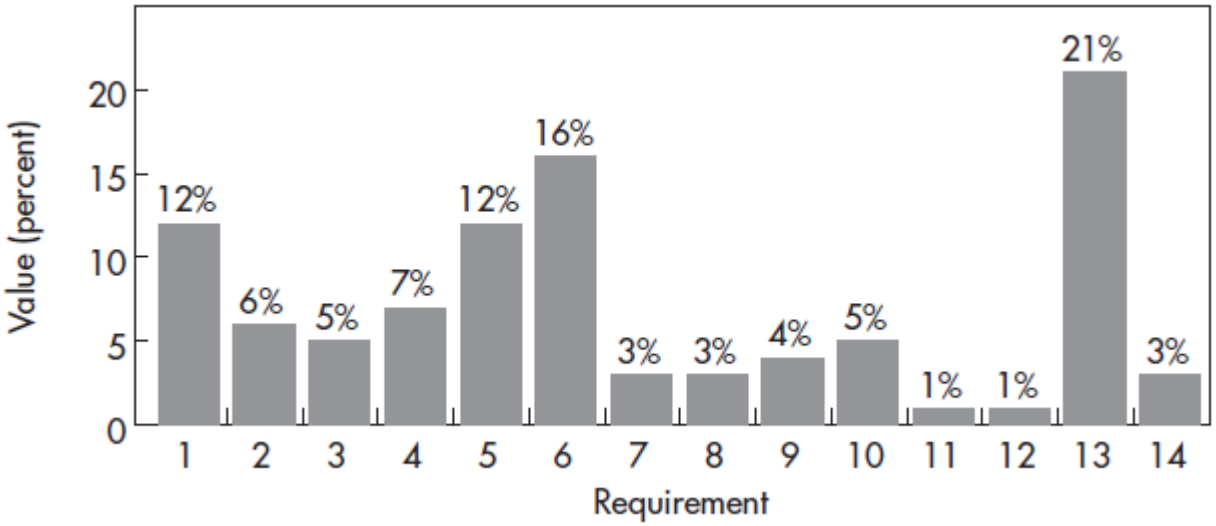
Example 1: Wiegers' Prioritization Example

Source: Wiegers, K. E., First Things First: Prioritizing Requirements http://www.processimpact.com/articles/prioritizing.html

Relative Weights:	2	1			1		0.5		
Feature	***Relative Benefit***	***Relative Penalty***	***Total Value***	***Value %***	***Relative Cost***	***Cost %***	***Relative Risk***	***Risk %***	***Priority***
1. Query status of a vendor order	5	3	13	8.4	2	4.8	1	3.0	1.345
2. Generate a Chemical Stockroom inventory report	9	7	25	16.2	5	11.9	3	9.1	0.987
3. See history of a specific chemical container	5	5	15	9.7	3	7.1	2	6.1	0.957
4. Print a chemical safety datasheet	2	1	5	3.2	1	2.4	1	3.0	0.833
5. Maintain a list of hazardous chemicals	4	9	17	11.0	4	9.5	4	12.1	0.708
6. Modify a pending chemical request	4	3	11	7.1	3	7.1	2	6.1	0.702
7. Generate an individual laboratory inventory report	6	2	14	9.1	4	9.5	3	9.1	0.646
8. Search vendor catalogs for a specific chemical	9	8	26	16.9	7	16.7	8	24.2	0.586
9. Check training database for hazardous chemical training record	3	4	10	6.5	4	9.5	2	6.1	0.517
10. Import chemical structures from structure drawing tools	7	4	18	11.7	9	21.4	7	21.2	0.365
Totals	**54**	**46**	**154**	**100**	**42**	**100**	**33**	**100**	--

Relative weights of criteria are estimated rather ad-hoc. How they can be determined in a systematic way ?

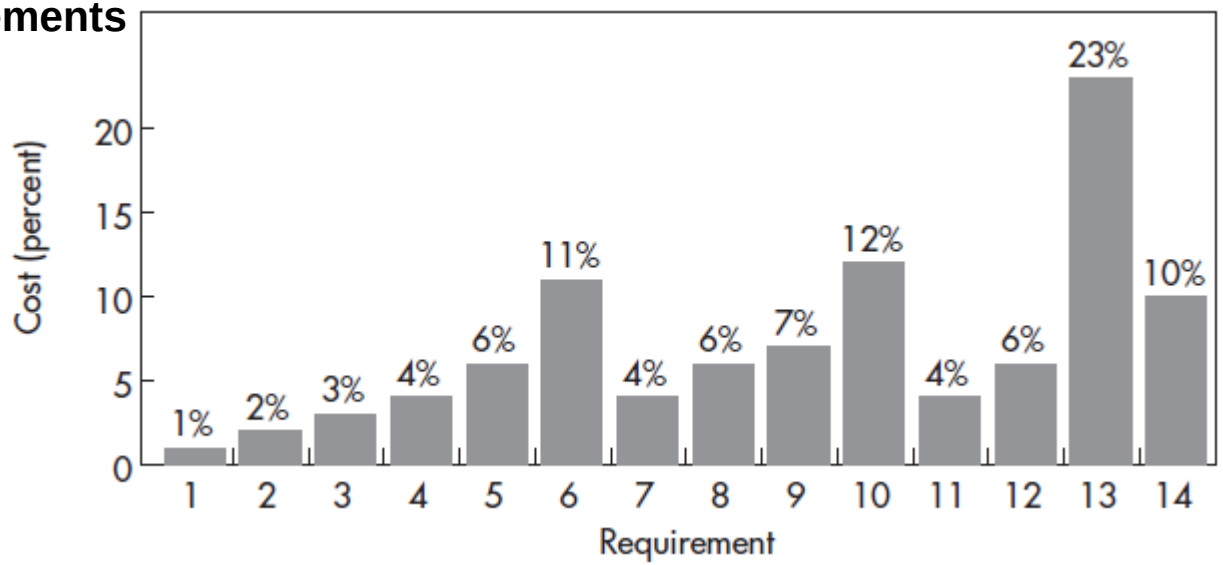
Example 2: Cost-Value (AHP) Prioritization Example (1/2)



Value distribution of Requirements

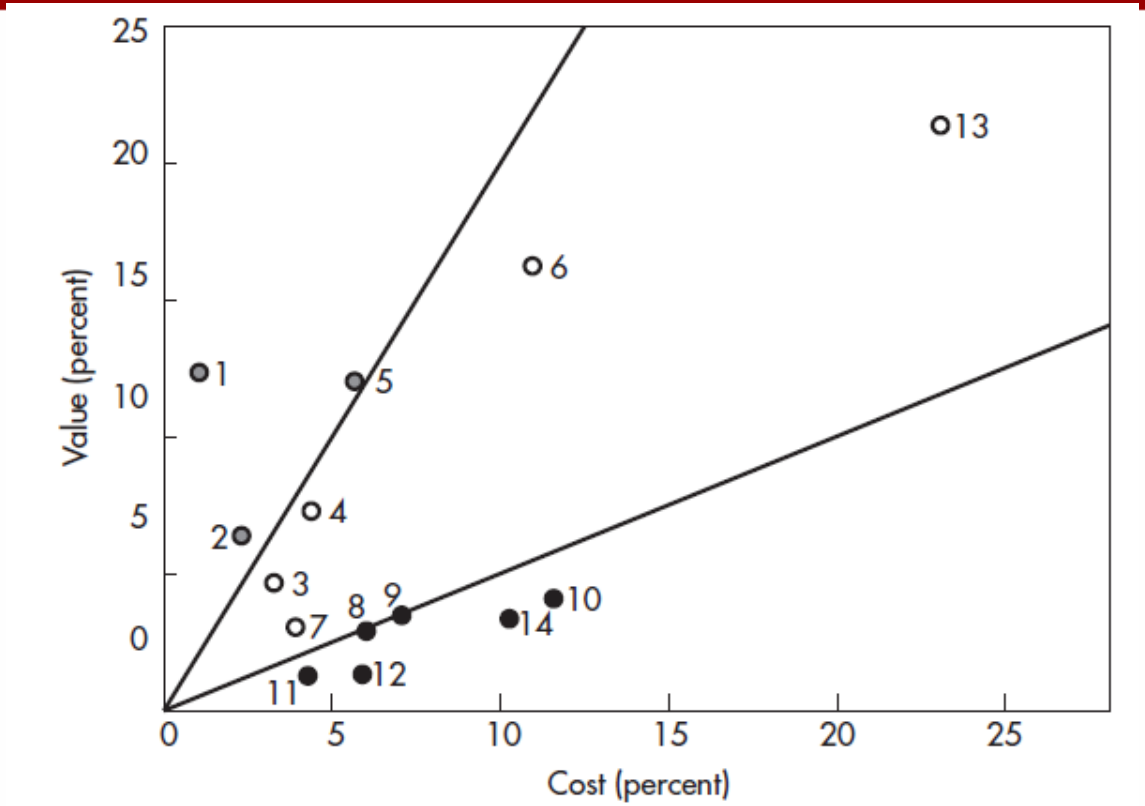
Requirements are compared pair-wise with respect to value and cost

Estimated cost of Requirements



Source: Karlsson, J. & Ryan, K. (1997). A Cost-Value Approach for Prioritizing Requirements, *IEEE Software September/October1997*, 67-74.

Example 2: Cost-Value (AHP) Prioritization Example (2/2)

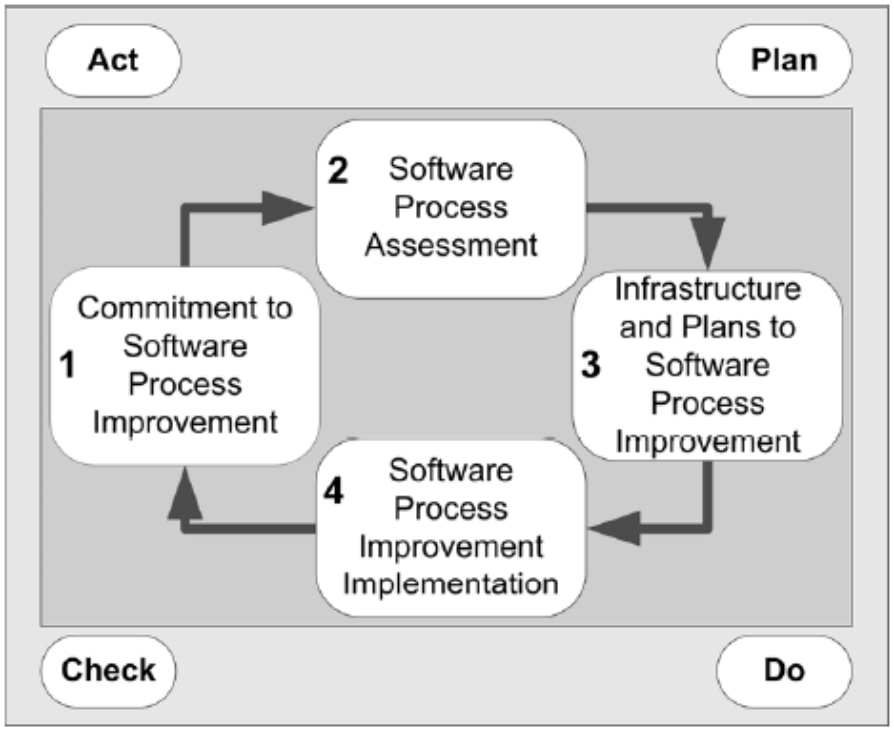


Cost-Value Diagram for Requirements

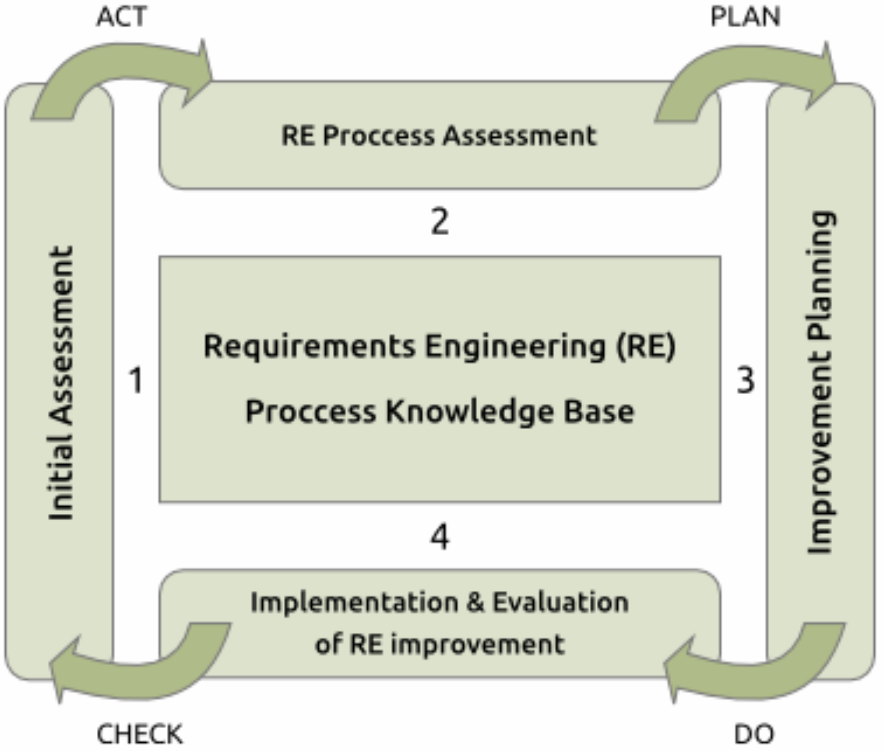
Source: Karlsson, J. & Ryan, K. (1997). A Cost-Value Approach for Prioritizing Requirements, *IEEE Software September/October1997*, 67-74.

- **What about scalability in case that the number of requirements and / or criteria is increased ?**
- **What about possible dependencies (cause-effect relationships) between the prioritization criteria ?**
- **What about prioritization criteria having impact on the software architectural process ?**

SPRINT-SMEs suggests inductive improvement (bottom-up) for RE processes (1/2)



Source: Basili, V.R., 1985. Quantitative Evaluation of Software Methodology. University of Maryland.



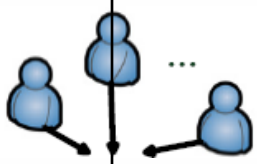
SPRINT-SMEs suggests inductive improvement (bottom-up) for RE processes (2/2)

- Prescriptive (top-down) approaches (CMMI and ISO/IEC 15504-SPICE) follow the “one size fits all” paradigm
- They define Key Process Areas required to be evaluated, regardless the characteristics/needs of an individual sw organization under assessment
- A typical sw process assessment/improvement project often demands large amount of resources and investment costs (e.g., it can last between 18 and 24 months)
- A prescriptive approach can be difficult to be followed by a SW SME
- In SPRINT-SMEs, we follow a “lightweight” approach that allows the consideration of certain process areas which are of interest for a specific organization (e.g. Requirements prioritization, COTS selection etc.)

Inductive improvement for a Requirements Prioritization process
Steps of SPRINT SMEs approach

Stakeholders/Experts are pooled/asked to determine the relevant prioritization factors

(step 1)



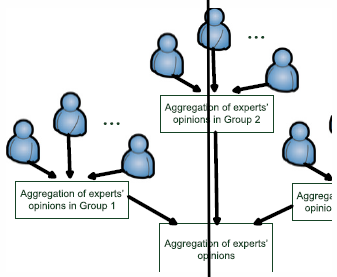
Each stakeholder determines, using linguistic terms, the impact-strength (positive/enabling or negative/impeding influence) of a prioritization factor on another

(step 2)

factor A (Requirement Volatility) has a NEGATIVE HIGH influence on factor B (Architectural Implementation)

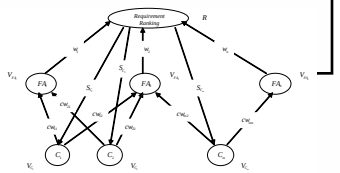
(step 3)

A model (in the form of a Fuzzy Cognitive Map - FCM) is constructed for each stakeholder



Individual FCMs are aggregated into a final FCM that depicts the prioritization factors/criteria, their dependencies and their effect on requirements ranking

(step 4)



A list of atomized requirements are evaluated (values in [0, 1]) with respect to the criteria depicted in the FCM

(step 5)

$$R = f\left(\sum_{i=1}^{n} w_i V_{FA_i} + R^{old}\right)$$

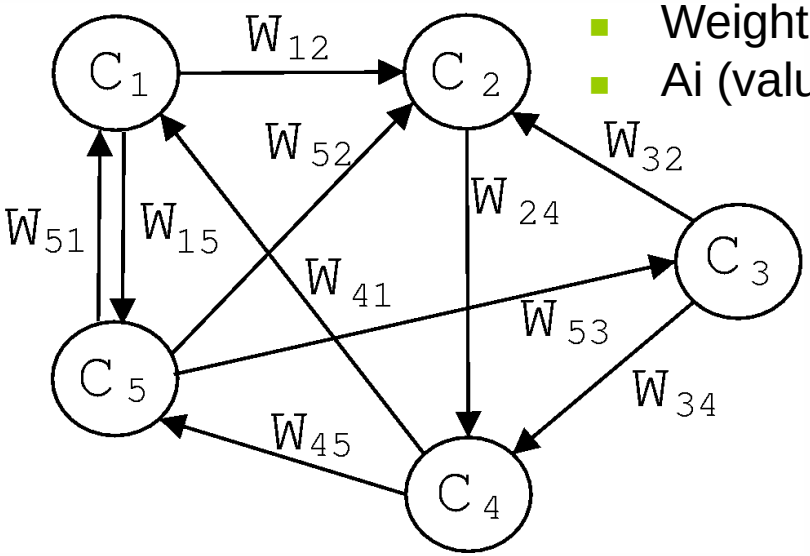
The FCM is dynamically analyzed (simulation) and a ranking index for each requirement is automatically produced

(step 6)

Fuzzy Cognitive Maps (FCMs): basic terms

FCMs have been proposed by Kosko (1986) as an extension of Cognitive Maps (used for representation of social knowledge)

Bart Kosko, *Fuzzy Cognitive Maps*, International Journal of Man-Machine Studies, 24(1986) 65-75 (first introduction of FCMs)

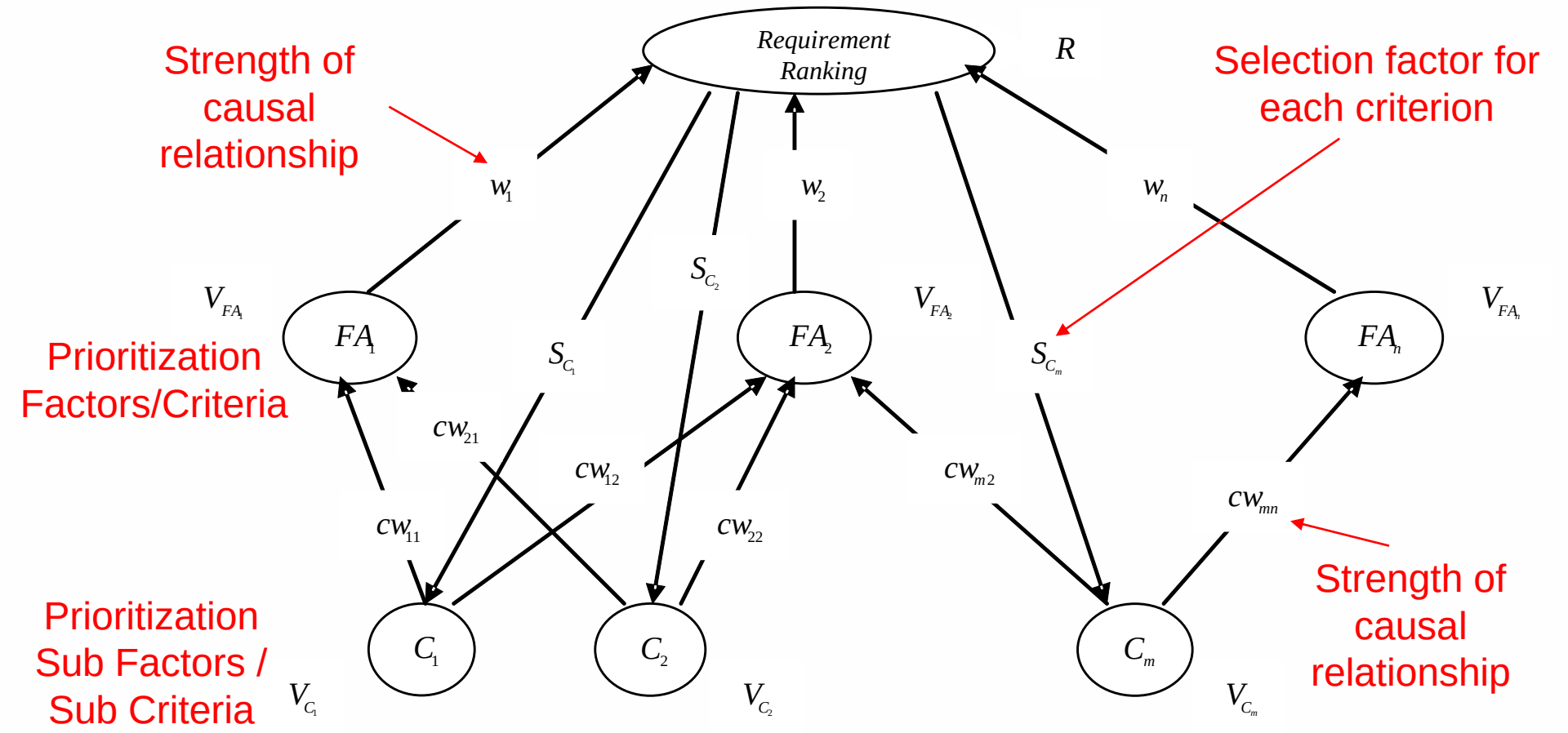


- Weighted arcs = **causal relations among the nodes**
- Ai (value of concept Ci) lie in [0,1], while W_{ij} lie in [-1,1]

$$\begin{cases} W_{ij} > 0, \text{ expresses positive causality,} \\ W_{ij} < 0, \text{ expresses negative causality,} \\ W_{ij} = 0, \text{ expresses no relation,} \end{cases}$$

$$A_i(t+1) = f\left(A_i(t) + \sum_{\substack{j=1 \\ j \neq i}}^{n} A_j(t) w_{ji}\right)$$

A FCM for Requirements Prioritization/Ranking



$$R = f\left(\sum_{i=1}^{n} w_i V_{FA_i} + R^{old}\right) \quad V_{FA_i} = f\left(\sum_{j=1}^{m} cw_{ji} V_{C_j} S_{C_j} + V_{FA_i}^{old}\right) \quad V_{C_j} = f\left(R \cdot S_{C_j} + V_{C_j}^{old}\right)$$

Conclusions

- Requirements Engineering is crucial for the development of sw intensive systems, in general, and for embedded sw systems, in particular
- In SPRINT SMEs, we follow an inductive approach for Requirements Engineering and Prioritization that is based on identifying the individual needs and priorities of a SW SME
- Development of the requirements prioritization/component selection tool is on going
- We have strong interest in validating our approach in industrial case-studies of sw development
- We have strong interest in cooperating with SW SMEs and establishing links with projects performing R&D at the same field (MODUS project)

SPRINT-SMEs Research Team

Project Coordinator:

Vassilis C. Gerogiannis, Associate Professor, Dept. of Project Management, Technological Education Institute (TEI) of Larissa, Greece

Main Research Team:

- Ioannis Stamelos, Associate Professor, Dept. of Informatics, Aristotle University of Thessaloniki, Greece
- Pandelis Ipsilandis, Professor, Dept. of Project Management, Technological Education Institute of Larissa (TEI), Greece
- Anthony Karageorgos, Assistant Professor,Technological Education Institute of Larissa (TEI), Greece
- Leonidas Anthopoulos, Assistant Professor, Dept. of Project Management, Technological Education Institute (TEI) of Larissa, Greece
- George Kakarontzas, Lecturer, Dept. of Computer Science and Telecommunications, Technological Education Institute (TEI) of Larissa, Greece
- Dimitris Tselios, Lecturer, Dept. of Project Management, Technological Education Institute (TEI) of Larissa, Greece

External Research Team:

- Fotis Liotopoulos, Ph.D. in Computer Science and Electrical & Computer Engineering
- Aggelos Thanos, Ph.D. in Computer Science (Programming Languages for Parallel Systems)
- Stamatia Bibi, Ph.D. (2008) in Software Engineering
- Androklis Mavridis, MSc in Software Engineering

Thank you for your attention ! ! !

gerogian@teilar.gr

http://sprint.teilar.gr/