SENG 350
- Software Architecture & Design

Shuja Mughal

Architectural Quality

Fall 2024

University of Victoria

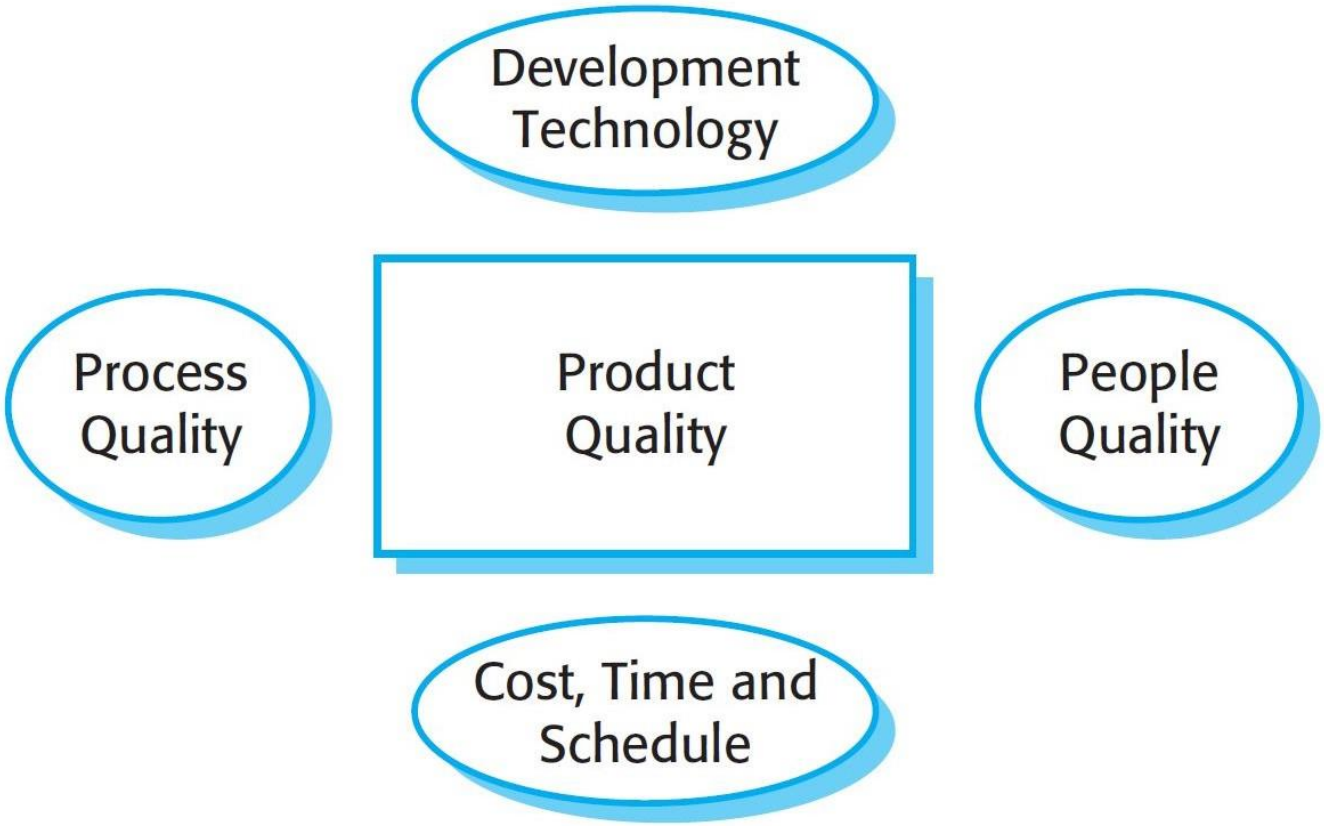
Previously

- Root Cause Analysis
- Goal Question Metric

Architecture and product quality

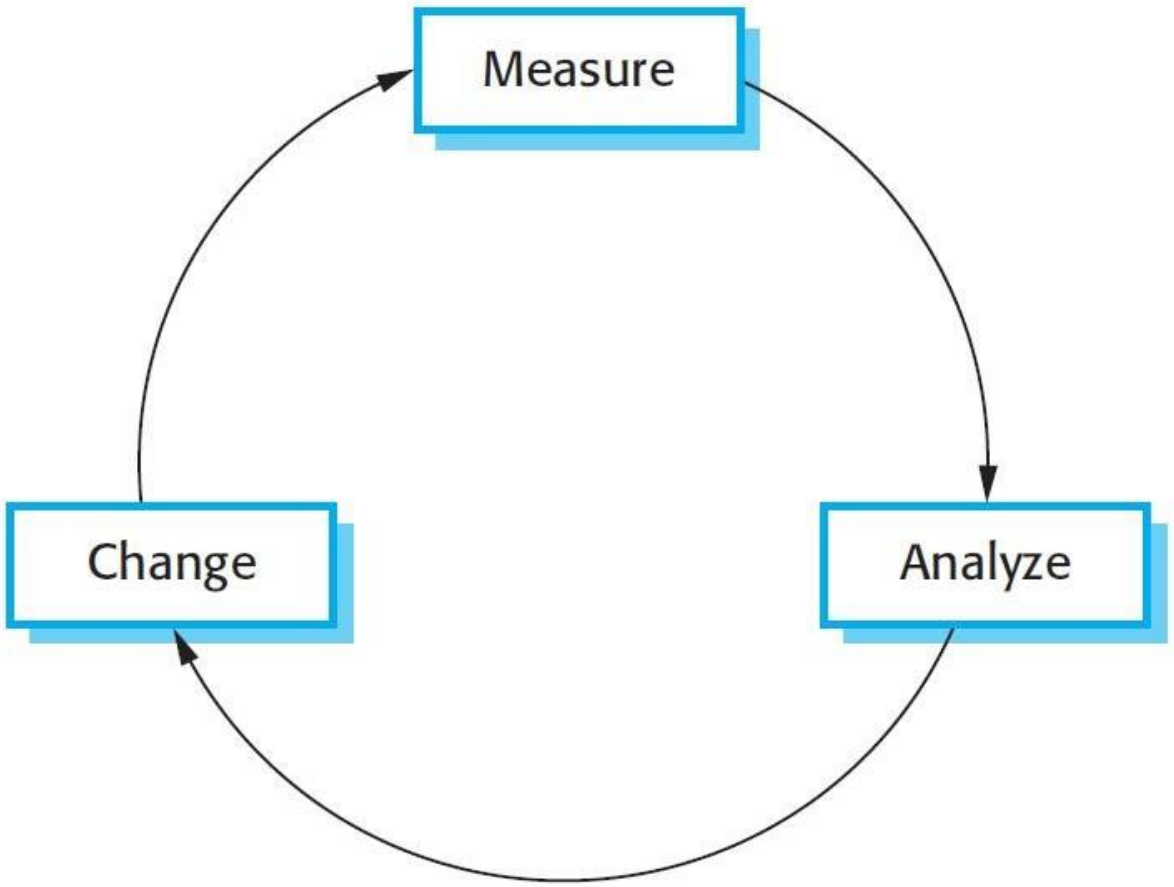
- Architecture quality and product quality are closely related, and architectural improvement benefits arise because the quality of the product depends on its architecture.
- A good architecture is usually required to produce a good product.

Factors affecting software product quality



University of Victoria

The architecture improvement cycle



University of Victoria

Goal-Question-Metric Paradigm

Goals

- What is the organization trying to achieve? The objective of process improvement is to satisfy these goals.

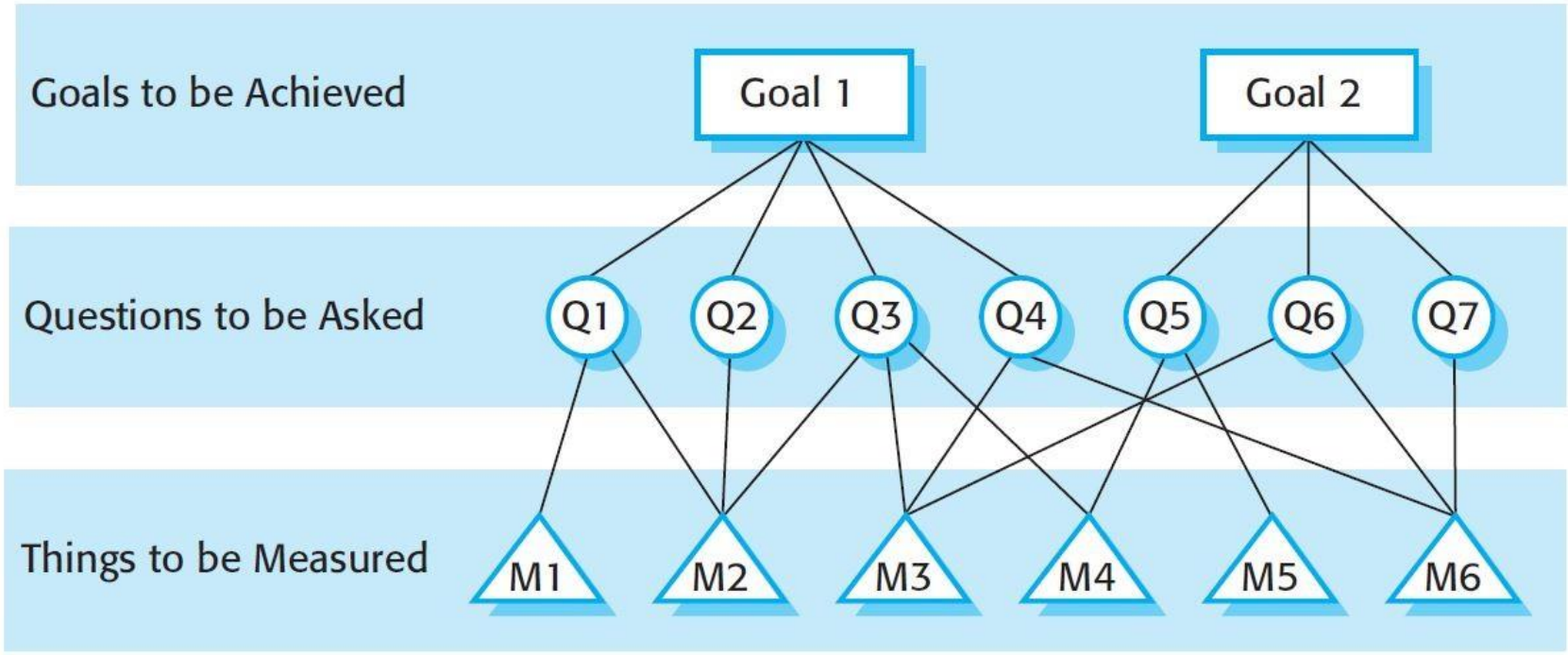
Questions

- Questions about areas of uncertainty related to the goals. You need process knowledge to derive these.

Metrics

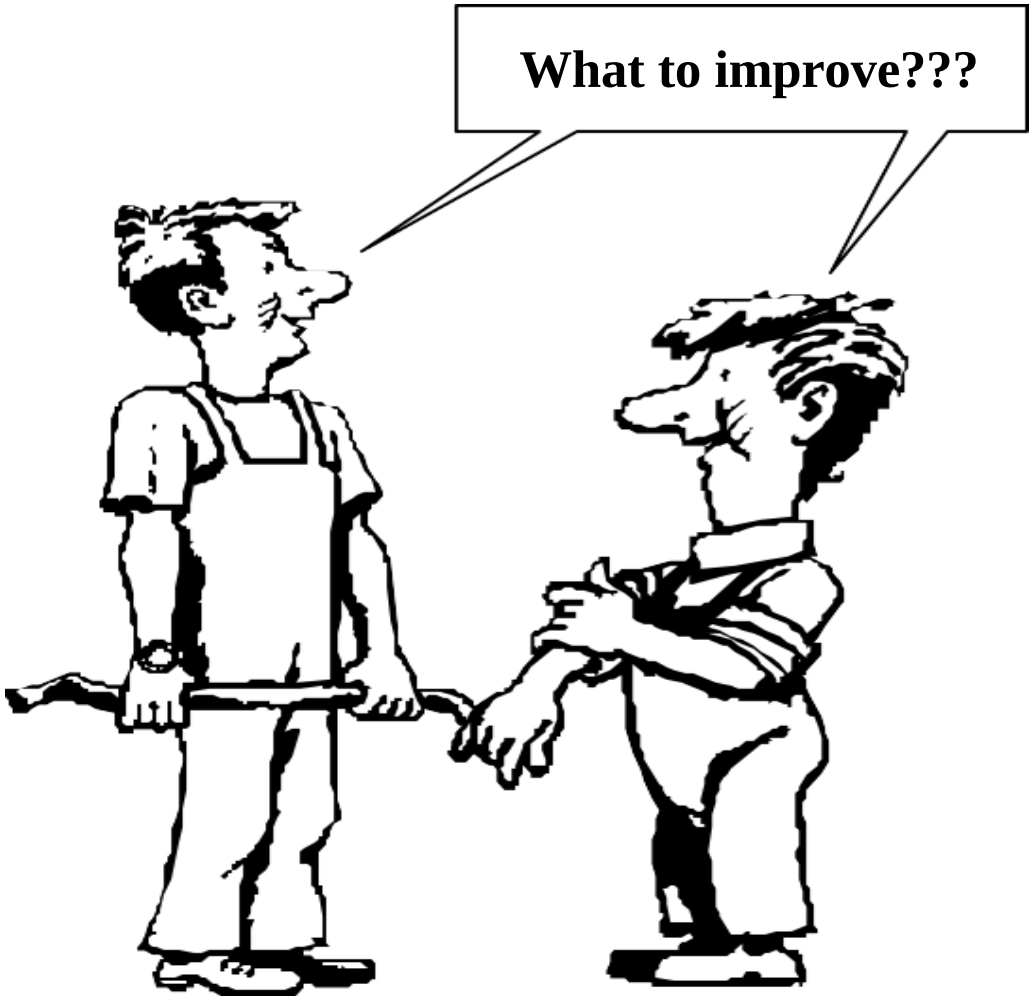
- Measurements to be collected to answer the questions.

The GQM paradigm



University of Victoria

Are we improving the right thing?



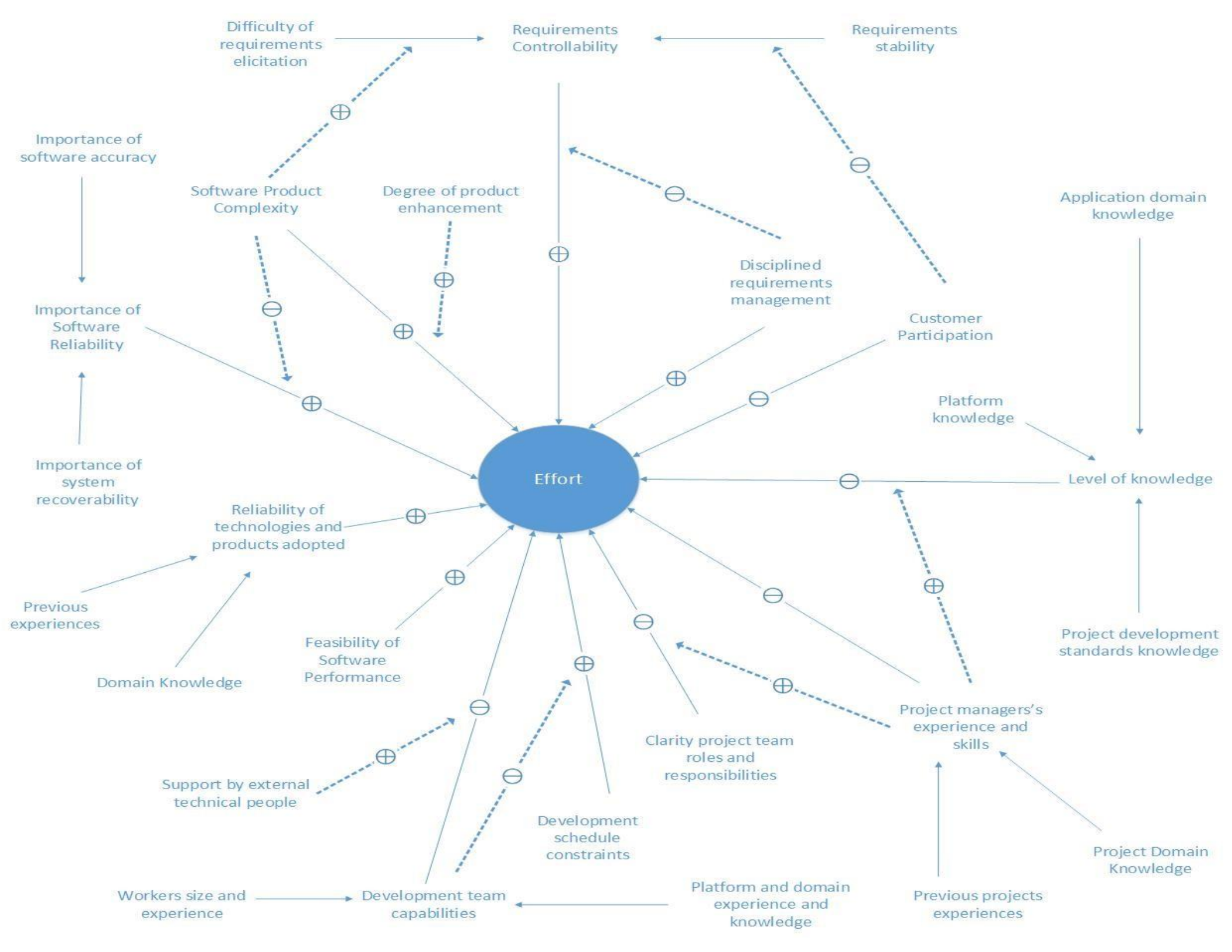
University of Victoria

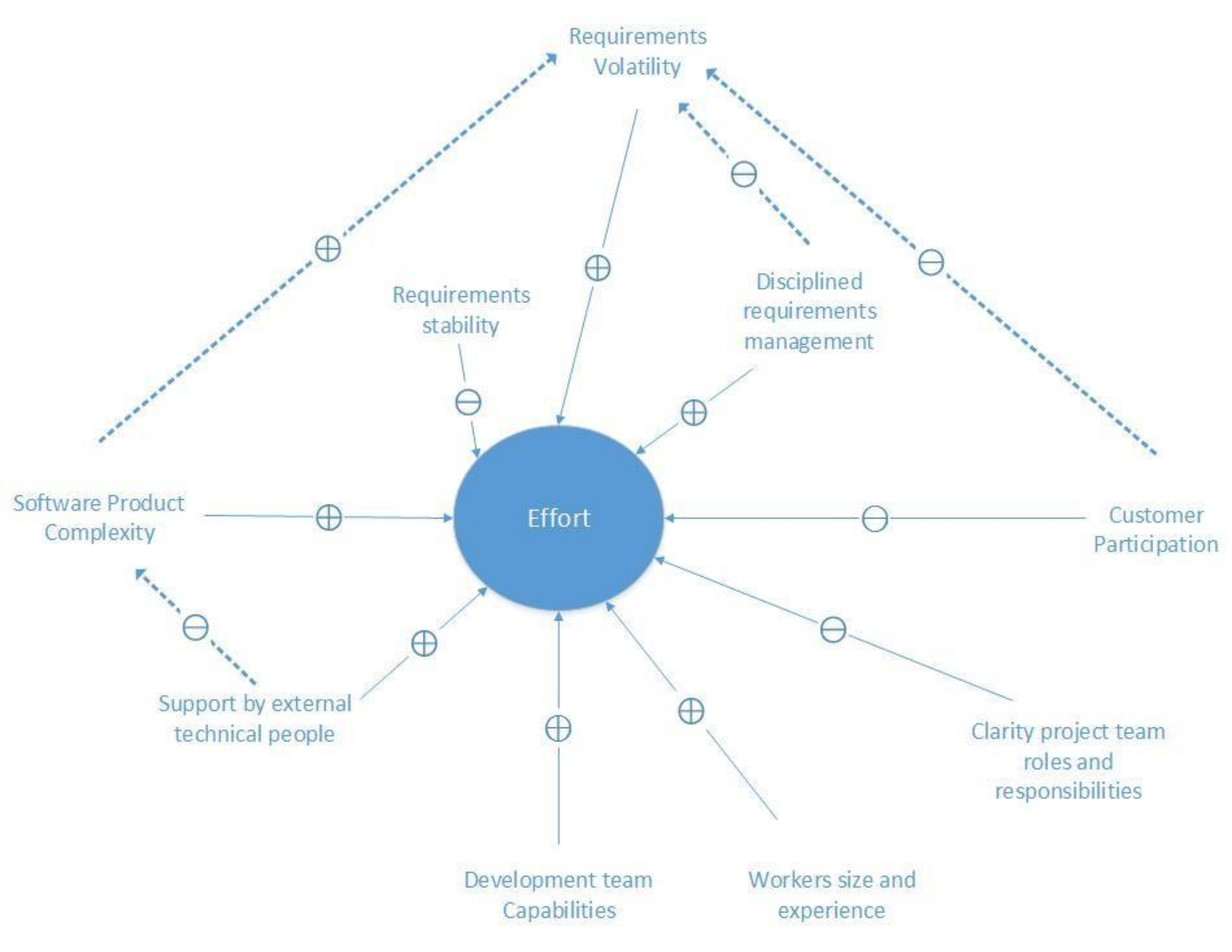
Casual Model

Example - CoBRA® cost estimation model

- Cost estimation, benchmarking, and risk assessment (CoBRA) is a hybrid effort estimation method.
- It represents a model-based, parametric estimation method that implements a define-your-own-model approach.

University of Victoria


Difficulty of requirements elicitation
Requirements Controllability
Requirements stability
Importance of software accuracy
Software Product Complexity
Degree of product enhancement
Application domain knowledge
Disciplined requirements management
Importance of Software Reliability
Customer Participation
Platform knowledge
Importance of system recoverability
Effort
Level of knowledge
Reliability of technologies and products adopted
Previous experiences
Domain Knowledge
Feasibility of Software Performance
Project development standards knowledge
Project managers's experience and skills
Clarity project team roles and responsibilities
Support by external technical people
Development schedule constraints
Project Domain Knowledge
Workers size and experience
Development team capabilities
Platform and domain experience and knowledge
Previous projects experiences


Requirements Volatility
Requirements stability
Disciplined requirements management
Software Product Complexity
Effort
Customer Participation
Support by external technical people
Clarity project team roles and responsibilities
Development team Capabilities
Workers size and experience

Factors

1. Requirement Volatility.
2. Software Product Complexity.
3. Development Team Capabilities.
4. Disciplined Requirements Management.
5. Customer Participation.
6. Clarity project team.
7. Requirements Stability.
8. Worker team size.
9. Support by External People.

Equation for Factors

Requirement Volatility = (Number of changes in requirements/ Total changes in requirements)

RV = NCR / TCR;

F1 = RV * Cost of change in one Requirement;

F1 = RV * C;

Where C = Can be 5 $;

Software Product Complexity = (LOC of module/Total LOC)

$$SPC = \sum_{i=1}^{n} \left(\frac{LOC\ of\ module\ i}{Total\ LOC}\right);$$

F2 = SPC * Cost of Per Line of Code;

F2 = SPC * C;

Where C = Can be 0.5 $;

Development Team Capabilities = (Experience of Employee/Total team experience)

$$DTC = \sum_{i=1}^{n} \left(\frac{\text{Experience of Employee } i}{Total\ Team\ Experience}\right)$$

F3 = DTC * Cost per hour of employee

F3 = DTC * C

Where C = Can be 25 $;

University of Victoria

Equation for Factors

Disciplined Requirement = (Number of Requirements formally managed/Total Requirement)

DR = NRFM / TR;

F4 = DR * Cost of one Disciplined Requirement;

F4 = DR * C;

Where C = Can be 10 $;

Customer Participation = (Number of changes in requirements/ Total changes in requirements)

CP = 1 - NCR / TCR

F5 = DR * Cost of change in one Requirement;

F5 = DR * C;

Where C = Can be 5 $;

Clarity of Roles and Responsibility = (Delay in time / Total time)

CRR = DT / TT

F6 = DR * Cost of Resource waiting till that time;

F6 = DR * C;

Where C = Can be 200 $;

Equation for Factors

Requirement Stability = (Number of changes in requirements/ Total changes in requirements)

RS = NCR / TCR;

F7 = RS * Cost of change in one Requirement;

F7 = RS * C;

Where C = Can be 10 $;

Worker Team Size = (Number of Human Resources Involved/ Total Human Resource)

WTS = NHRI / THR

F8 = WTS * Cost per hour of employee;

F8 = WTS * C;

Where C = Can be 25 $;

Support by External Team = (Number of External Team Involved/ Total External Team)

SBET = NETI / TET

F9 = SBET * Cost per hour of external employee;

F9 = SBET * C;

Where C = Can be 35 $;

University of Victoria

Effort

Effort =

(Requirement Volatility + Software Product Complexity + Development Team Capabilities + Disciplined Requirements Management + Worker team size + Support by External People)

–

(Customer Participation + Requirements Stability + Clarity of Roles and Responsibility)

Results for Simulation

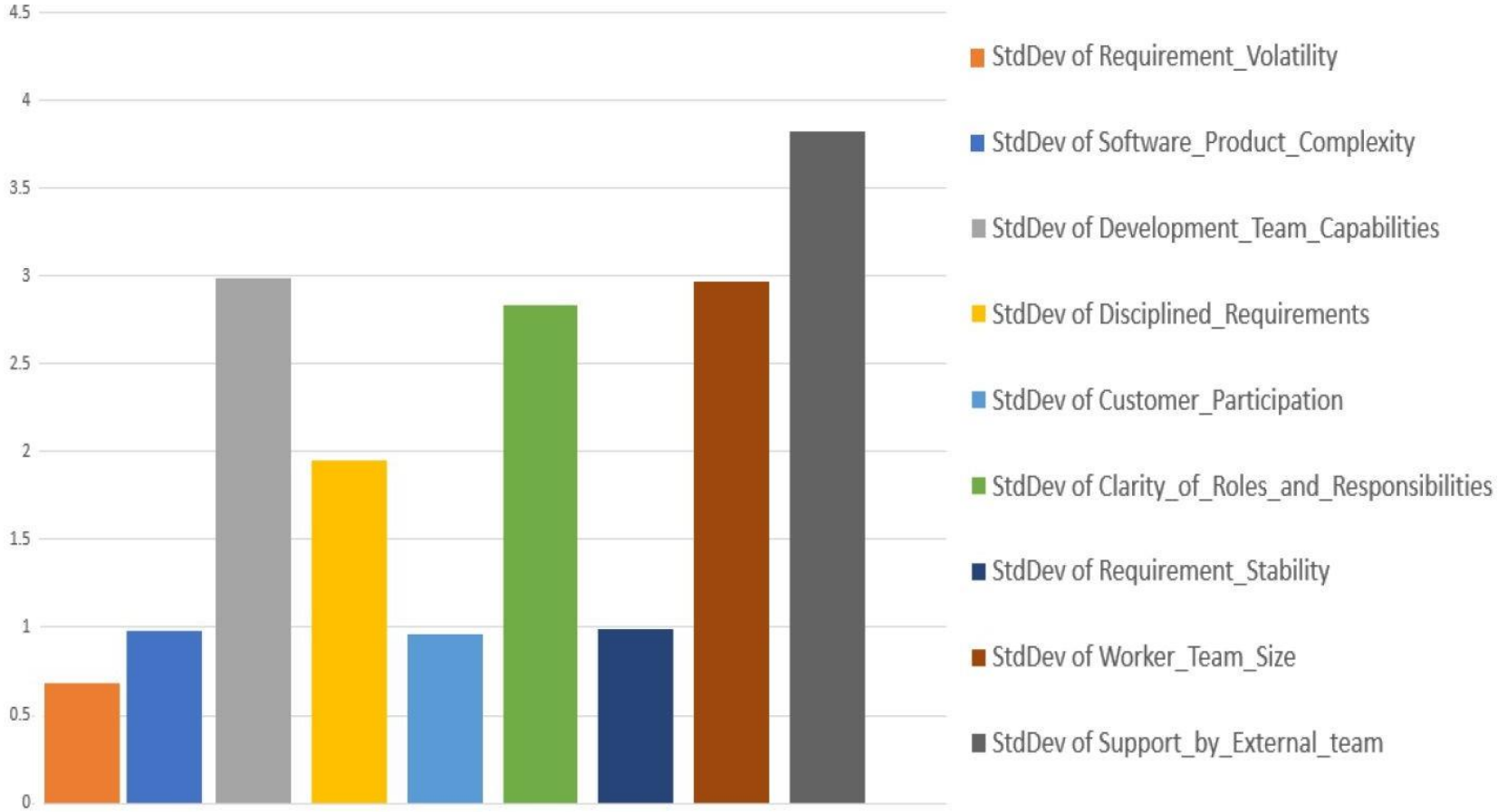
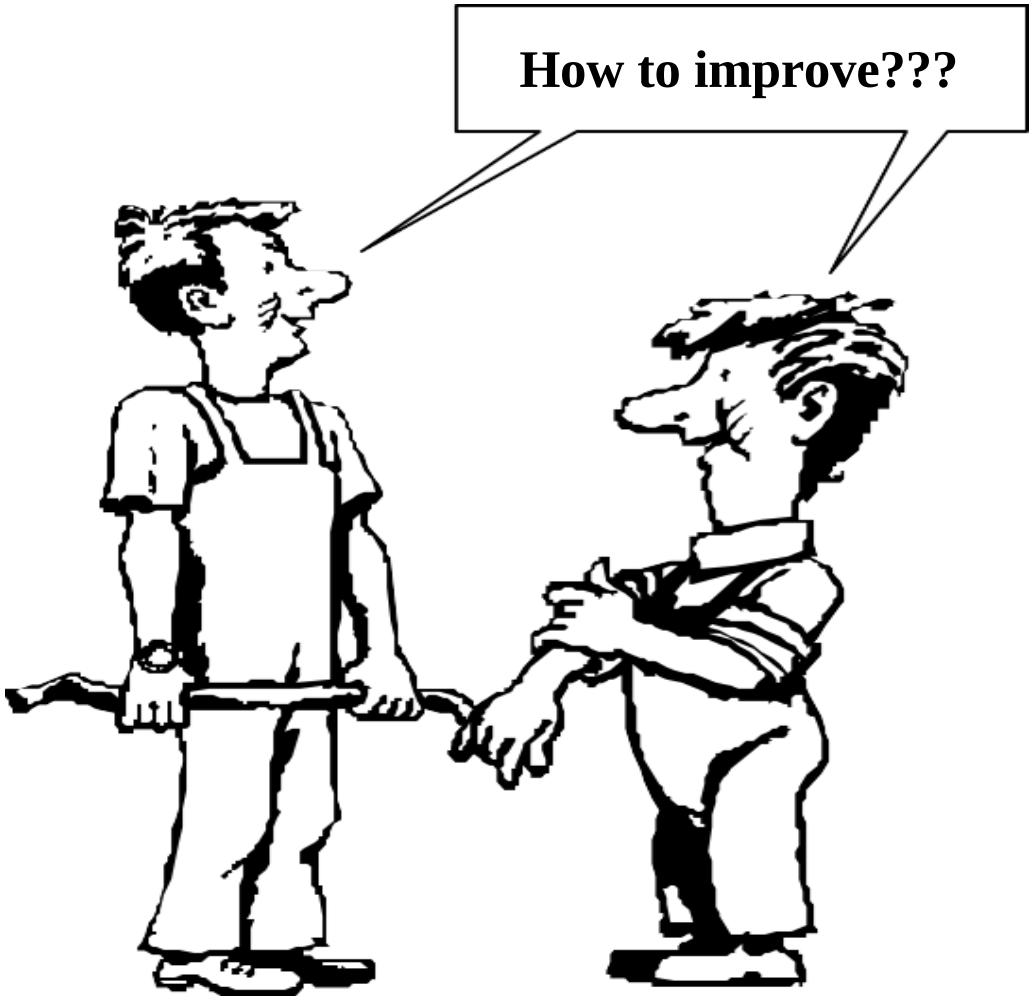


Figure 12: Standard Deviation of the Factors

University of Victoria

How to Improve the Architecture?

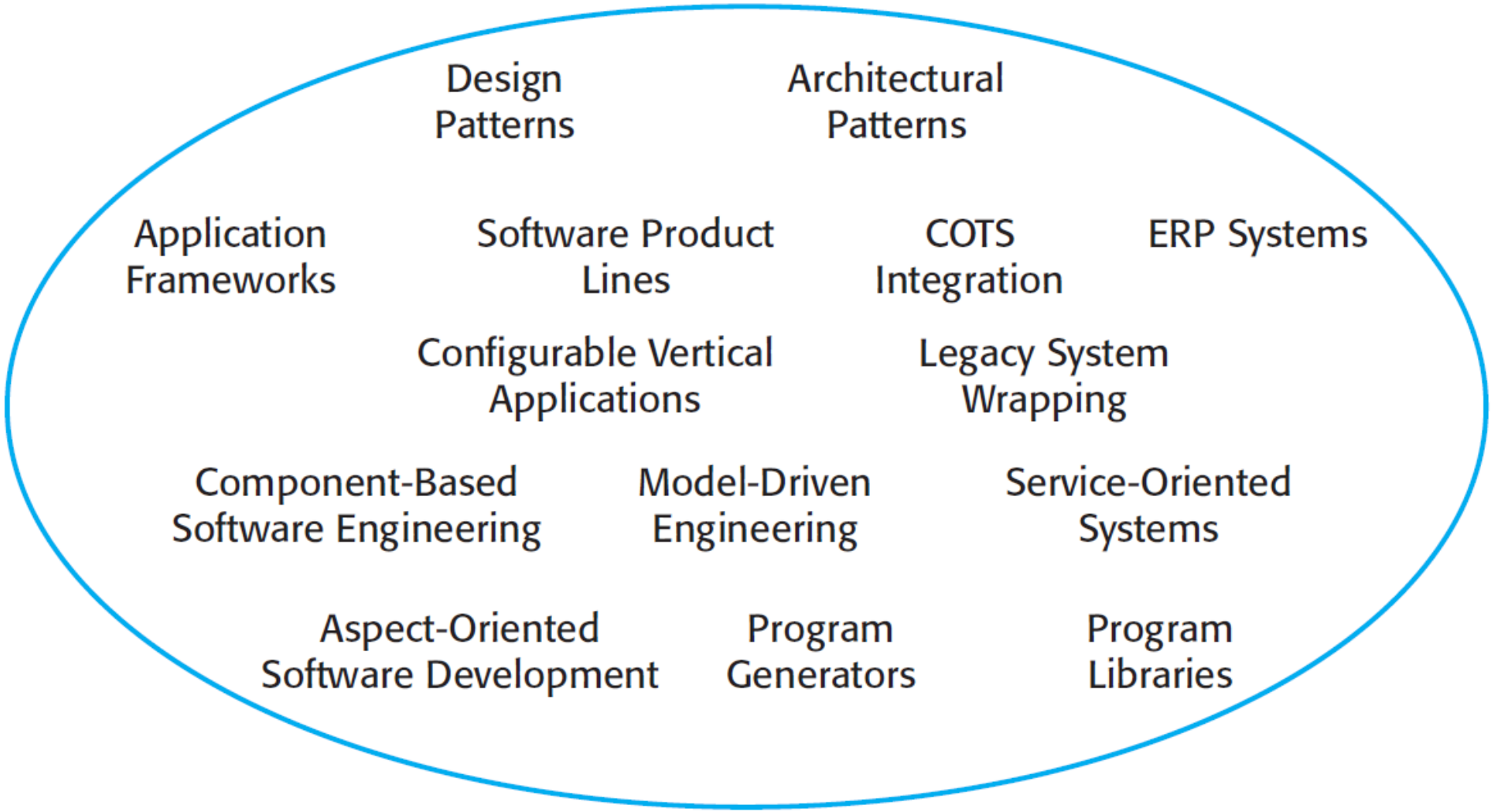


University of Victoria

The reuse landscape

- Although reuse is often simply thought of as the reuse of system components, there are many different approaches to reuse that may be used.
- Reuse is possible at a range of levels from simple functions to complete application systems.
- The reuse landscape covers the range of possible reuse techniques.

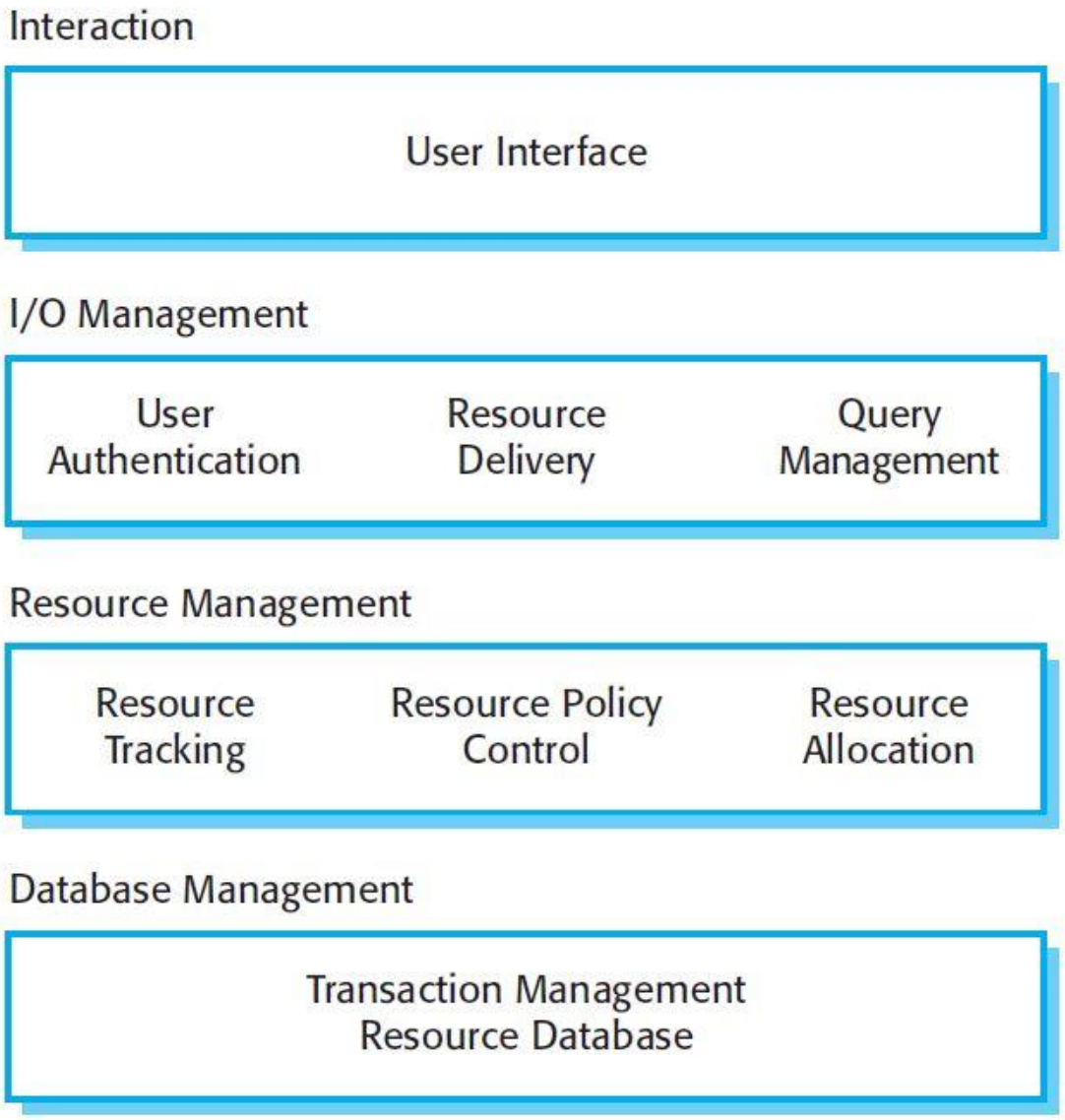
The reuse landscape



Key factors when planning reuse

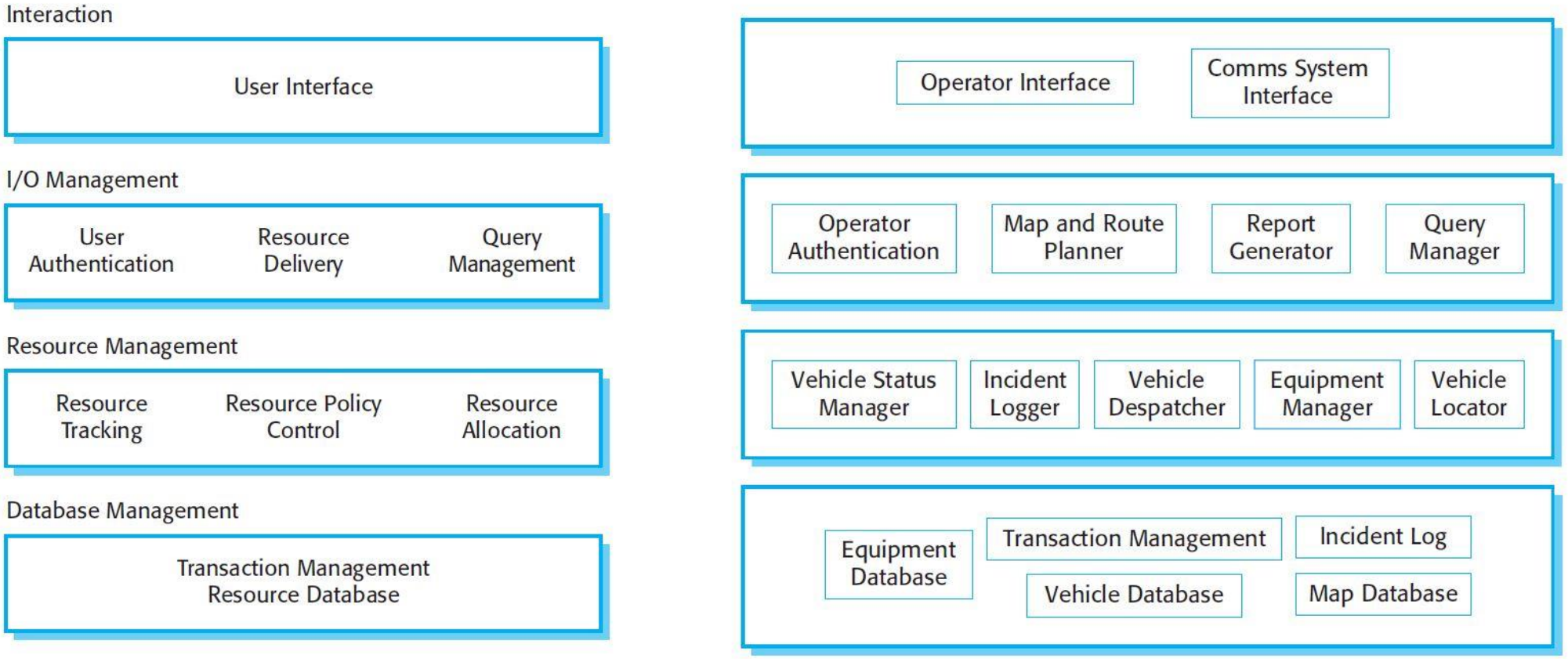
- The development schedule for the software
- The expected software lifetime
- The background, skills, and experience of the development team
- The criticality of the software and its non-functional requirements
- The application domain
- The execution platform for the software.

The architecture of a resource allocation system

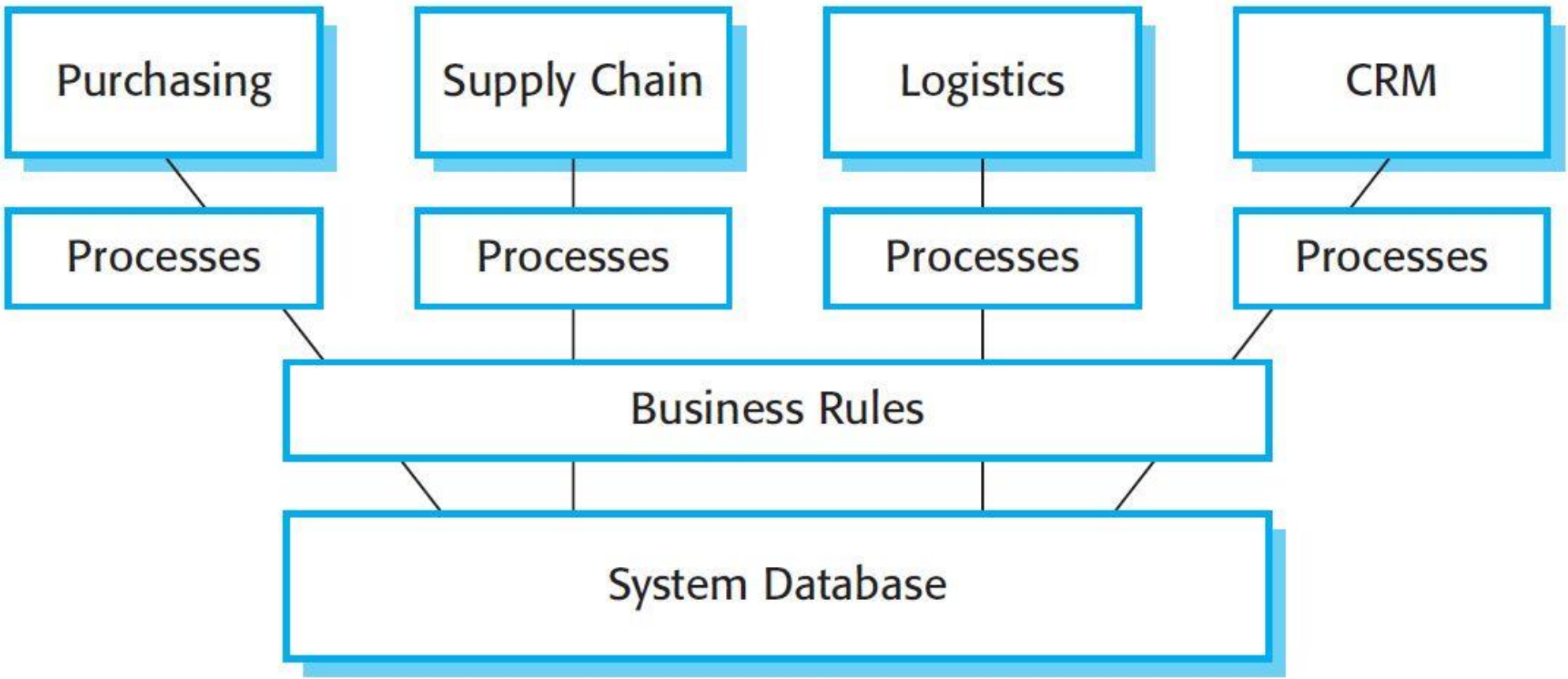


University of Victoria

The product line architecture of a vehicle dispatcher



The architecture of an ERP system



Activity

- Make a group of 2.
- Think of a problem you faced in Software Engineering.
- Use the root cause analysis.
- Find the cause and the effect of the problem.
- Make sure there are a minimum of 3 fishbone diagrams per problem.
- Create a Casual model for it.
- Submit them to teams.