AB6 Framework
The Analyst’s Basics Framework for CORPORATE INFORMATICS

Course for Business Information Systems Analysts and Product Owners

15/04/2019– Version: 1.02

© 2018, 2019 Axel Vanhooren
World of Systems

YOU ARE HERE
We live in a world of

- STRUCTURES
- SYSTEMS
- PROCESSES

and in a world of INFORMATION

ranging from words, diagrams and pictures on paper or screen to thoughts and electrical nerve impulses.

War is a matter of vital importance to the state; a matter of life or death, the road either to survival or to ruin. Hence, it is imperative that it be studied thoroughly.

The Art of War, Sun Tzu
Contributors

Special thanks go to Frédéric Naisse for his kind contribution.

Versions

15/04/2019
15/03/2019
15/02/2019
10/01/2019
18/12/2018
30/10/2018

Notes:
The dates are mentioned on the pages. The dates can be used to find the pages that are new or which have significantly changed. A search on the new date allows to find them easily.
<table>
<thead>
<tr>
<th></th>
<th>Table of Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>Core Ideas</td>
</tr>
<tr>
<td>3</td>
<td>Fundamental Topics</td>
</tr>
<tr>
<td>3.1</td>
<td>Systems</td>
</tr>
<tr>
<td>3.2</td>
<td>Organisation &amp; Company</td>
</tr>
<tr>
<td>3.3</td>
<td>Information</td>
</tr>
<tr>
<td>3.4</td>
<td>Processes</td>
</tr>
<tr>
<td>3.5</td>
<td>Problem Solving</td>
</tr>
<tr>
<td>4</td>
<td>Information Component</td>
</tr>
<tr>
<td>5</td>
<td>Business Knowledge and Expertise</td>
</tr>
<tr>
<td>6</td>
<td>History</td>
</tr>
<tr>
<td>7</td>
<td>General Subjects and Views</td>
</tr>
<tr>
<td>8</td>
<td>Information Exploitation</td>
</tr>
<tr>
<td>9</td>
<td>Information Systems</td>
</tr>
<tr>
<td>10</td>
<td>Engineering</td>
</tr>
<tr>
<td>11</td>
<td>Management</td>
</tr>
<tr>
<td>12</td>
<td>Work Environment</td>
</tr>
<tr>
<td>13</td>
<td>Organisation</td>
</tr>
<tr>
<td>14</td>
<td>Roles and Collaboration</td>
</tr>
<tr>
<td>15</td>
<td>The Analyst</td>
</tr>
<tr>
<td>16</td>
<td>Projects</td>
</tr>
<tr>
<td>17</td>
<td>Information Systems Development Framework</td>
</tr>
<tr>
<td>18</td>
<td>Analysis</td>
</tr>
<tr>
<td>19</td>
<td>Architecture</td>
</tr>
<tr>
<td>20</td>
<td>Design</td>
</tr>
<tr>
<td>21</td>
<td>Programming</td>
</tr>
<tr>
<td>22</td>
<td>Testing</td>
</tr>
<tr>
<td>23</td>
<td>Deployment</td>
</tr>
<tr>
<td>24</td>
<td>Techniques</td>
</tr>
</tbody>
</table>
FOREWORD

The AB6 Framework aims to spread, to strengthen and to boost the discipline of engineering information solutions in corporate environments.

Companies seek to automate and optimise their operations. Innovation is as important, not among the least, to stay ahead of the competition. IT has also the potential to transform companies and to drive business achievements. But the critical conditions here fore are far from ideal.

1) IT initiatives evolved from the development of smaller applications meant to be used by a limited number of users and having a specific function to the development of much larger systems in complex heterogeneous environments. Meanwhile, IT, as a discipline, had to develop under pressures and in turbulent circumstances. The problems corporate IT encounters today and its evolution are symptoms of a lack of thoughtfulness and maturity.

2) The present driving idea is to develop software applications wanted by the business community. Responding to their demand should create customer’s satisfaction. The focus in this way of thinking is put on the demand, on the business stakeholders, on the developers, on programming and on technologies.

Lower goals lead lower results. Striving for higher goals lead to better and greater achievements. But this requires a paradigm shift. A new belief system, a more solid thinking framework, a holistic view and approach and higher goals are essential.

There are two possible ways to evolve to another belief system. We can mature through repeating mistakes until the right lessons are learned through crisis's. This is rather a painful method and may take time. Or, we can evolve by thinking with an open, analytical and critical mind. AB6 hopes to contribute to the latter.

It is hoped that AB6 acts as a trigger and forms a basis for engaging in analysis, reflection and critical investigation to solve issues and obstacles and to renew and consolidate the discipline of Informatics, ISE, Systems Analysis, Business Analysis and other variants.

Axel Vanhooren
Freelance Consultant Business Informatics
Belgium

LATEST VERSION @
Elevate the Goals

ENTERPRISE BUILDER

Engineering and building that part of the company that exploits information with the intention of supporting and strengthening the company and satisfying customer’s information needs.

BUSINESS DRIVER

• Maximising resource utilisation
• Increasing production volumes and business results

ADAPTOR / IMPROVER

• Adapting to fit
• Reducing negative effects: limits, risks, costs
• Improving the system internally: controls, optimisation, structure, ...

ENABLER

• Putting technologies at work
• Enabling the business community to work better
• Automating traditional ways of working
• Introducing new technologies
• Modernisation
• Demand-driven software development
Cross-roads in Software Development

Company-oriented Systems ≠ End-User-oriented Systems

Main concern: Allowing the company to function

Examples: Airport Management, ERP, accounting systems, production planning systems, supply chains, cross-organisational systems, ...

Idea: How to build good systems that support the company’s operations or drive them? How to maximally exploit information?

Systems Analysis centric, Architecture centric, Model-based, ....

AB6 Framework

End-User-oriented Systems Feature-driven Development

Main concern: End-users’ experience of feature-usage is important. Users can choose to use or not features.

Examples: Social media platforms, Online bookstores, Airline websites, ... (particular the front-ends)

Idea: How to produce software fast and get it as quickly as possible to the end-users? How to develop good features for end-users?

Approach:

Consult with end-users

Developer-centric

Deliver

Develop

Fast feedback

Fail fast

• Both require a different approach.
• Not dichotomic. Gradual shift in emphasis between left & right column.
Improving the Systems Analysis and Engineering Competencies

DOING: “How to do”

Frameworks, Methodologies, Methods, ...

Surface

AB6 FRAMEWORK

UNDERSTANDING “WHY” (+ belief system)

ISEDMA

Belief Systems

Depth

• Ability to better identify and define initiatives correctly and to conceive better solutions

• Improved decision making

• Improved usage of frameworks, methodologies and methods and ability to adapt or conceive new ones suiting the initiatives.

• Ability to handle more complex projects and pursue higher goals
AB6 Framework

AB6 Framework proposes

- Principles
- Theories
- Concepts
- Relations
- Descriptions
- Solution elements
- Building blocks

and is

- adaptable
- improvable
- expandable

Reading AB6:

- Understand how it fits in the overall picture of AB6
- Is the statement true?
- How important is it?
- What are the implications?
- How can I use it?

IMPORTANT NOTE:

AB6 does not take new domains like Artificial Intelligence, Brain–Computer interface (BCI) (Neural-Control Interface (NCI)) and similar new domains into account. Some statements have to be reinterpreted when dealing with these specific contexts.
AB6 is not a methodology, process or structure. Neither is it a solution that can be applied as is right away.

In general, solutions ready to apply, to be strictly followed are very unlikely in Systems Analysis.

Why? Every project, situation or problem is unique. There are no standard projects, standard situations or standard problems. So, there can not be a standard recipe.

The Analyst has to determine how to work, what preconditions must be met, what techniques to be used and how to adapt to the specific circumstances.

AB6

- deepens the understanding
- organises knowledge
- provides patterns
- supports autonomous thinking
- serves as trigger for further thinking and investigation

Uncritically following or applying AB6 is the exact opposite of the intention of AB6
Variants of Analysis Disciplines

Systems Analysis

→ Business Analysis
→ Business Process Analysis
→ Information Analysis
→ Information Systems Engineering
→ Functional Analysis
→ Other related Analysis, Architecture, Engineering and Modelling disciplines

AB6 is for any kind of Analyst in the context of Business Informatics

Notes:
In AB6, the term Systems Analysis is used in the most general sense as an overarching discipline which includes all different levels and variants of analysis in disciplines like (Business) Informatics.
Systems Analysis (/and Design) exists as an intra-project discipline.
“Science is a **way of thinking** much more than it is a body of knowledge.”

Carl Sagan

“As to methods, there may be a million and then some, but **principles** are few. The man who grasps principles can successfully select his own methods. The man who tries methods, ignoring principles, is sure to have trouble.”

Harrington Emerson

“I don’t know what’s the matter with people: they don’t learn by understanding; they learn by some other way—by rote or something. Their **knowledge is so fragile!**”

Richard Feynman

"People mistakenly believe that IT failures are due to a technical problem or a software problem, and in fact it has its **roots into the culture**, **how people work together**, how they share knowledge, the **politics** of an organisation. The worse the politics, the more likely the failure."

Michael Krigsman

**Source:**
The Worst IT Project Disasters of 2013 (by Chris Kanaracus, IDG News Service)
The 6 worst IT project disasters of 2013
gmp_29008.gmr_29008.gde_29008_member_5829606858213588992
CORPORATE IT AND SYSTEMS ANALYSIS TODAY
Typical Recurrent Issues

**Information**

1. Information is not present or accessible when needed.
2. Unnecessary information is captured, while valuable information isn’t.
3. The precise meaning of some information is unknown.
4. Information is duplicated and exists in various forms.
5. Information is incoherent and not updated everywhere.
6. Data is unorganised and hard to maintain.
7. Changing information at one place impacts other organisational units.
8. Poor data quality: outdated, unreliable, vague, insufficient, old and new data are mixed, various formats and encoding standards exist, ...
9. Lack of control on data; difficulty to ensure data consistency and timeliness.
10. The company’s information pool is growing. A lot of information is unrecorded, unmanaged, uncontrolled and known by a very few ones. Chaos is looming.
11. Much of the information is not or can not be exploited.
12. Information glut, information swamp, information overload

**Processes**

1. Cross-border business processes are slow, complex, awkward and inefficient.
2. Not all processes are inventoried. No overview (and management) of processes.
3. The whole processes aren’t understood or only understood by very few.
4. Some steps are performed because they have always been done like that. It’s assumed that it is of some value to another department.
5. Business processes may interfere with each other, resulting in unclear, unforeseen situations, exceptions and conflicts.
6. Not all business processes respect a same coherent logic.
7. Business processes are executed in different ways by different people. How can we then measure these processes in a coherent way?
8. No or irrelevant process metrics are collected, while valuable aspects aren’t being measured. (not just about KPI’s, but about all “PI’s”).
9. The metrics aren’t fully exploited. Or, the right people don’t receive these measures.
10. Production can’t be planned, because of unknown processes, too much fuzziness, improvisation and variability in their execution.
Typical Recurrent Issues

Systems

1. Software solutions lack features or are awkward to use. Software features restrain the end-users activities.
2. Software applications contain too much unused features and logic.
3. The processes, the systems’ architectures and the implemented concepts and the organisation of the data do not match with the real world.
4. Software systems are inflexible and not scalable.
5. Collaboration among software systems is limited and hard to achieve.
6. The software landscape and the technological landscape grew organically to an uncontrollable chaotic whole. Loss of insight in the implemented systems and in their interactions. A lot of corpses in the closets.
7. Adapting the systems became slow, hard and risky up to the point of paralysis.
8. Projects are late, cost more than estimated and, while matching the demand and requirements, they do not suit the needs and do not solve the problems. Often, they even create new problems.
9. Once operational, the systems underperform and do not meet the expectations.
10. Systems and projects do not create the intended business value.
Conventional Beliefs Driving Corporate IT

1. **Responding** to the business demands and delivering the software systems the business stakeholders wants or needs

2. The domain of IT is about technologies, applying these technologies and building software applications.

3. Building software applications which facilitate the execution of the business community’s activities

4. Delivering accordingly to requirements within time, scope and budget

5. Obtaining business stakeholders satisfaction by delivering the desired solution: a software applications matching the demand

6. Finding solutions in the products offered by the market and improving performance by implementing more powerful hardware

7. **Progressing** by following the market’s tendencies and **innovating** by acquiring the latest technologies
Core questions in this line of thinking

• What is the client’s demand?
• What are the client’s needs?
• What does the client want? Understanding what the client wants.
• What software do we need to build and to deliver to satisfy the client?
• What technologies can we use and what is the best way to use them to solve the client’s problem or need?
• Is the client satisfied with the presented or delivered software application?

Motivation: Obtaining end-users’ or stakeholders’ satisfaction by supporting business operations through the creation and delivery of software features and applications; accordingly to the demand, expectations, needs, wants and requirements; in order to facilitate the work performed by the business community.
Conventionally: What is an Analyst?

- Someone who helps the business community to implement the systems or solutions they think they need and want, and which they sometimes already conceived, at least to a certain degree.

- Someone who has to meet the business demand and expectations, and who has to satisfy the business stakeholders.

- An in-between, a bridge, an interface, a liaison between ‘business’ and ‘IT’

- A facilitator, coach, guide, mediator resolving conflicting goals and incoherence’s or other issues.

- Someone who looks inwards in a given subject and study its parts, the relations between these parts and the nature of all this. He or she refines (refiner) it and identify and eliminate incoherencies.

- A Requirements Collector/Recorder, Requirements Analyst, Requirements Manager

- A modeller (producing models)

- An UML-specialist, BPMN-specialist, ...

- A specification writer / translator (translating business demand into specifications for the developers)

- Someone who responds to the business demand by finding out what the business community and stakeholders need and want, as well as what their expectations are.
Conventional Beliefs about “Analysis”

1. About looking inwards of something, to the parts and to their relations. (≈ definition in dictionary) → the Analyst must be given something to analyse, to work with.

2. Looking at, verifying and working out the details.

3. The Analysis consists of looking at what the business demand, what they need and want in order to create value and satisfy the business stakeholders.

4. Translating the business demand into artefacts useable by developers.

5. The analysis is an obligatory step between business stakeholders and developers.

6. Analysis is about gathering, verifying, detailing and managing requirements. Requirements skills are crucial.

7. Analysis is about producing models (UML, BPMN, …) (representing the demanded solution) or specifications. UML or BPMN skills are crucial.

8. Everybody can do an Analysis or can become an Analyst.

“Don’t believe everything you think”
In many organisations, Analysis is considered as a discipline which has as **purpose** and **goal** to meet demands and satisfy people.

The **role** of the Analyst is often defined

- as “**bridge**” between business and IT” (communication),
- as **facilitator** (collaboration)
- or as **coach** to support the business community to identify issues and finding solutions.

This implies that **Analysis** is perceived and applied as a

- A **communication** discipline
- A **facilitation & coaching** discipline

These considerations contribute in shaping the view, expectations and approach of business stakeholders.

This has a fundamental influence in the way Analysis is commonly employed and practised in companies.

**Notes:**
This should be kept in mind when the discipline of Systems Analysis is defined and when it’s purpose is defined. Undeniably, communication, collaboration and facilitation are activities performed by the Analyst. This doesn’t mean they are part of the Systems Analysis discipline.
Room for Improvement

A lot has changed during the last half of a century. More powerful hardware and infrastructure, new technologies and tools. New software development approaches and new professions appeared.

What didn’t change much are the underlying widespread beliefs, the reasonings, the mind-set, the norms and criteria and the expectations.

These beliefs reflect activities performed during an Analysis and sub-roles of an analyst.

However, there is no other choice than to note that, in the earlier mentioned beliefs, even combined, the essence, the true purpose and the objective of Systems Analysis are completely absent. They are thus ignored.

Consequently, Systems Analysis is immensely reduced and limited. This result in a lot of problems, inefficiencies, waste, lost opportunities, and so on.

If Systems Analysis is not applied in accordance with its purpose, with its spirit, the benefits can’t be reaped.

Expecting an optimal use of “IT” and maximised exploitation of information is then unjustified.

But, there is room for improvement. This is the opportunity.
After more than half a century of "IT" …

• Project failures?

• Satisfied customers?

• Quality of implemented IT?

• Opportunity gap between “what is” and “what is possible”?

• Degree of Business-IT alignment?

• Degree of information exploitation?

• Quality of the Business-IT relation (trust, collaboration, …)?

• Still looking for better ways to develop software applications

• Confusion about terms, concepts, roles, …

• Waterfall? Agile? Business-IT fusion?

• …
Notes:

Progress is not achieved by improving or changing nothing. Every aspect needs to be in harmony with all the purpose, goal, level and with all other required aspects.
The goal of software development in corporate IT is not to develop software. It is to build and to improve the company. That’s a different worldview, mind-set and requires different priorities, skills, approach, ...

Exception: Some systems in organisations, like the intraweb, are for users.
SOFTWARE AS PRODUCT FOR END-USERS ≠ SOFTWARE AS PART OF A COMPANY

FUNDAMENTALLY DIFFERENT ENDEAVOURS !!!!!

Consumer IT & product development versus Engineering of “Systems of Systems”
Visualisation of Different Mind-sets

Focus in Conventional Software Development

Focus in AB6 Information Systems Analysis and Engineering
Two Ways of Thinking that dominate and driving software engineering approaches

Science
Analytical, Rational

Art
Creative

Some problems should be solved by the first type of approaches. The other approach will suit better for another type of problems.

It’s about a dominant idea defining the overall approach. However, each ways of thinking use the other when needed.

AB6 Framework is more oriented to the scientific, analytical, methodical way.

A person’s natural way of thinking might be more oriented to one of the two ways. The way the brain is wired determine whether one is stronger to science or in art.
**Different Perspectives**

**Creativity, art**

**Methodical, structured,**
**Analytical, rational**

*Example: Is software engineering an art or a science?*

- Mainly an independent endeavour
- Mainly a social endeavour, group activities, (overly collaborative)

**Different Views**

- Leads to different priorities, collaborations, types of approaches and methods

**TENSIONS in the IT/ informatics community**

- Hard to understand other perspectives
- Confusion, misunderstandings and disagreements
- Biased choice of approach or solution
- Not easy to collaborate or to function in a specific environment
## AB6 Document Organisation

### Introduction & Various Core Ideas

<table>
<thead>
<tr>
<th>Fundamental Knowledge Areas</th>
</tr>
</thead>
<tbody>
<tr>
<td>Systems</td>
</tr>
<tr>
<td>Company / Organisation</td>
</tr>
<tr>
<td>Process</td>
</tr>
<tr>
<td>Problem Solving</td>
</tr>
<tr>
<td>Information</td>
</tr>
</tbody>
</table>

### General Background Topics

<table>
<thead>
<tr>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Exploitation</td>
</tr>
<tr>
<td>Information System</td>
</tr>
</tbody>
</table>

### Overall Organisation of the Informatics Engineering

<table>
<thead>
<tr>
<th>Informatics Initiatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS Development Framework</td>
</tr>
<tr>
<td>SDLC Phases</td>
</tr>
<tr>
<td>Techniques</td>
</tr>
<tr>
<td>Organisation</td>
</tr>
<tr>
<td>Work Environment</td>
</tr>
<tr>
<td>Analyst</td>
</tr>
<tr>
<td>Roles &amp; Collaboration</td>
</tr>
<tr>
<td>Projects</td>
</tr>
</tbody>
</table>
INTRODUCTION
Introduction

Information is meaningful (valuable) to

• people
• companies and organisations

Information is a resources. It must be ...

• Available → captured and stored
• Fit for usage → processable / processed
• Accessible
• Stored
• Secured
• Organised
• Managed
• Exploited

COMPANY

• Collects
• Creates
• Stores
• Processes

INFORMATION

• Disseminates
• Uses
• Sells
Introduction

NEED for a solution able to deal with information consisting of:

- Capabilities
- Processes
- Mechanisms
- Systems
- Structures
- Rules and Principles
- Transfer channels
- Concepts
- ...

Information is captured, processed, stored and disseminated by

- People
- Machines
- Software/computer systems

HOW TO DEAL WITH INFORMATION?

HOW TO CONCEIVE SYSTEMS AND INFORMATION SOLUTIONS?
Evolving Nature of Endeavours

◆ **Small and simple endeavour**
   to solve a straightforward problem

   - Straightforward task
   - To achieve lower goals
   - Doesn’t require a lot of techniques
   - Doesn’t require advanced skills
   - Anyone can do it and achieve results
   - Not risky
   - **Easy success**
   - Lesser value creation
   - No or insignificant competitive advantage
   - A company can’t compete with small software initiatives

◆ **Larger and complex solutions**
   in complex environment
   to solve more complex problems
   and/or pursuing higher goals

   - Can not simply be executed promptly
   - Requires a lot of investigation, thinking and planning
   - Requires more and better approaches, methods and techniques
   - Requires higher norms
   - May require additional disciplines
   - True expertise: requires developed competencies and test these skills
   - Requires specific prerequisites and conditions
   - Requires more control, communication, organisation, collaboration, ...
   - Success is harder to achieve
   - ...

Understanding is easy, complete and reliable

Understanding is harder - much lesser insight
More aspects to take into account
More contradictions
More uncertainty
More assumptions

Notes:
Trying to achieve even higher goals, wanting to increase innovation and seeking to operate in a more heterogeneous environment is even more demanding.
Evolving Nature of Endeavours

Benefits of larger initiatives

- Greater efficiency and effectiveness
- More significant
- Create more value
- Realisation of a vision - Shaping the future
- More innovation
- Greater competitive advantage

Executing larger and more complex initiatives requires

- Higher norms
- Higher skills - skills way beyond average

An organisation that is able to deal with larger and complex initiatives is strong.

It is able to transform itself, to innovate and to beat the competition.

Notes:
- The development of larger, heterogeneous and more integrated system creates challenges which aren’t present (or not to that extent) in the development of small and simple systems. Hence, the development of larger systems requires a different work environment, approach, skills, and so on.
- These initiatives are more risky. However, risks can be limited and mitigated. Dealing with risks is also a competency based on analytical skills, flexibility, resilience and creativity.
Dealing with more challenging endeavours create some needs

1. The knowledge fields of information and of systems (in general) become critical.

2. Ability to learn a lot and quickly, and to have an organised insight.

3. Need to be able to get a grip on the complexity of problems and systems and their environment.

4. Supports for thinking activities.

5. Need to Communication and collaborate

6. Avoiding to do unnecessary or inappropriate work (rework) that could have been avoided by learning and thinking.

7. Limiting the risks

8. Increasing the certainty

9. Increasing the control

SYSTEMS ANALYSIS increases the chance of SUCCESS

Notes:
A fake analysis doesn’t all of this. It allows the project to carry on. Failure is then discovered late. Risks materialise. Or the product doesn’t solve the problem. The benefits it could have produced aren’t. and so on.
CORE of ANALYSIS

SYSTEMS ANALYSIS

Systems

“Study”

• Software system
• Engine
• Car
• Nuclear plant
• Recycling system
• Financial system
• Ecological system
• Human body
• Group
• Company
• ...

SYSTEMS ANALYSIS is

• the STUDY of SYSTEMS and their environment

+ 

• the CONCEPTION (DESIGN) of SYSTEMS

AB6 consider SA within the domain, context and perspective of Corporate IT / Informatics.
CORE of ANALYSIS

Definition:
Systems Analysis (and Design) is the **professional discipline** of studying environments and systems in order to devise new system solutions or to identify issues and opportunities for improvements and devising the required adaptations to achieve or to contribute to a certain goal.

- Professional discipline → hard to master, takes time, expert
- Study situations, processes, systems and environments
- Identify issues and opportunities for improvements
- Devise adaptations to processes, systems and their environment
- Devise new processes and systems
- Evaluate processes, systems and their results
- In order to contribute to goals

PRACTICALLY:

SYSTEMS ANALYSIS spans

from

the detection and identification of issues and opportunities
to

the definition of the solution to be built.
Evolving Nature of Endeavours

Software development exists for more than half a century. Despite during this time it evolved a lot, software development is still in its adolescence. The good news is that it can still evolve a lot. This evolution is not only a technological one.

**Software systems have evolved**

<table>
<thead>
<tr>
<th>TOOL</th>
<th>ENTIRE PART OF THE COMPANY</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALL</td>
<td>LARGE</td>
</tr>
<tr>
<td>SIMPLE</td>
<td>COMPLEX</td>
</tr>
<tr>
<td>LIMITED SCOPE</td>
<td>INTEGRATED</td>
</tr>
<tr>
<td>LIMITED FUNCTION</td>
<td>LARGE SCOPE</td>
</tr>
<tr>
<td>LOCAL</td>
<td>SCALE</td>
</tr>
</tbody>
</table>

- Single user
- Limited set of features
- Clear problem

- Heterogeneous
- Multi-user
- Integrated
- Tentacular and geographically spread
- Information systems
- Information sharing
- Different information usage
- Multiple stakeholders
- Multi-disciplinary
- Operating in a broader complex environment
- Connecting different organisations
- Shared information – multiple use

Both are completely different endeavours

Activities and initiatives (projects) evolved from developing software features or software systems as tools to developing an entire integrated part of the company.

Goals, drivers, mind-set, competencies, approaches and methods have to be adapted to the new nature of endeavour.
1. Identification of the Problem

The identification of the exact problem, as the root cause or real objective, is mandatory before further actions are taken.

2. Get understanding the existing systems, the broader context and the environment

It is impossible to conceive a submarine without a general understanding of submarines, of the systems the submarine is connected with, of warfare with submarines and without understanding of the ocean.

3. Conceive the solution

Notes:
A chapter is dedicated to problem solving.
The ANALYSIS Philosophy

The intent defining the Analysis philosophy:

“to conceive the right solution, often a system, to solve the right problem correctly from the first time”

This intent can’t always be fully realised. The intention is to converge as quickly as possible (and with the least iterations possible) to an appropriate, elegant and definitive solution.

This requires a real and genuine effort of a skilled Analyst. The Analysis and the solution should give the Analyst a true and justified confidence that the solution will correctly solve the right problem.

BASED ON THE IDEA

- Better results can be obtained,
- the building process can go smoother and faster
- and risks can be reduced

by acquiring knowledge and insight and by thinking before engaging in a building process.

LEARN & THINK FIRST, THEN ACT

Notes:
Doing work that doesn’t respect the Analysis philosophy can’t be called an Analysis.
There might be clarifications, refinements, (minor) corrections, adaptations, additions and changes after the first version of the solution design.
Learning and thinking never ends. Analysis and design never really ends.
Then, the solution will live its life and continue to evolve.
Understanding of each sphere thoroughly

- Role & Relations
- Theory
- Implementation
- Possibilities and limits

No knowledge sphere can be ignored.
CORE of ANALYSIS

Information perspective

Solving the right problem rightly
Implementing a strategy
Implementing & improving business activities
Improving & building the company

INVESTIGATING
DETECTING
DIAGNOSING
LEARNING
CONCEIVING

ENVIRONMENTS
ORGANISATIONS
SYSTEMS
STRUCTURES
PROCESSES
RULES, CONCEPTS

.. and more
Common Causes of Project Failure

- Unclear goals and objectives
- Loose scope definition
- Moving targets, changing objectives
- Conflicting interests of the stakeholders
- Changing decisions (until the right ones are taken)
- Lack of active executive support
- Lack of SME and end-user involvement
- Unrealistic expectations
- Poor planning or rigid usage of plans
- Unrealistic time and resource estimates
- Insufficient communication
- Tackling the wrong problem (consequences instead of causes)
- Bad quality of requirements
- Inappropriate skills
- Disappointing results
- Hidden wasted efforts, hidden costs, ...

"God laughs at those who deplore the effects of which causes they cherish."

Bossuet

Causes of the “Causes”

- People, skills, collaboration, thinking, dependencies, assumptions, belief system, abuses, ... Not technical/technological issues.
- Commonly, early failure warnings/symptoms are not identified or are simply ignored.

Notes:
To reflect: Why (root cause)? And how can we resolve this?
(Some answers are provided by AB6.)

Some issues can be solved through a collaboration of Project Manager, Analysts, Architects, project team and some stakeholders.
Causes: Part of the Analysis is performed by business people; no verification; no alignment of expectations with delivered results; ....
What is Systems Analysis?

Systems Analysis is the systematic study of systems and their environments and the conception of solutions, consisting of adapting or creating systems and adapting the environments. In informatics, the focus is on information used in and by systems and in their environments and on how information can best be exploited.

Common Usage:

Systems Analysis is used, as a bridge, to translate information solution described by the business community into software systems. It is used to implement what people want instead of what is necessary.

Reflection:

- What are the consequences of this difference?
- Is it an optimal way of applying Systems Analysis?
- Is this a problem?
- How should Systems Analysis be better applied?
Discipline versus Application

SCIENCE - DISCIPLINES
- Information Science
- Information Engineering
- Informatics
- Business Informatics
- Systems Analysis

What are their CONTENT / PURPOSE?

- Study of systems and organisations
- Study of information in organisations and systems...
  - ... to learn, evaluate, diagnose, conceive, solve, ...

= years of study

APPLICATION of the Science and Disciplines

Analysis = role of bridge
Business Stakeholders define the need, the mission, the outline of the solution, ...
Implementation of the Business stakeholders demand
  - Do what is expected
  - Stay within defined boundaries
  - Refining the demanded solution
  - Writing specifications or user stories

How much of these sciences / disciplines is really applied?

Is this the best way to apply these sciences/disciplines?

Is this problematic? Consequences? Risks? Success?
DEVELOPING SOFTWARE SYSTEMS

WITHOUT

(genuine) SYSTEMS ANALYSIS

• Solving the wrong problem
• Solving partial problems or solving problems partially
• Building “solutions” which solve nothing
• Symptoms and consequences are solved
• Conception of inefficient solutions
• Creation of new problems elsewhere in the organisation and/or in the longer term
• Waste of time and resources in figuring out and in building non-solutions, inappropriate solutions or in solutions that put a burden on future developments and on the future of the organisation
• Critical insight is acquired only after software applications have been built and/or implemented, leading to a lot of rework.
• Creation of little value, of no value or even systems detrimental for the company
• Little exploitation of the available resources, the possibilities and the capabilities
• Critical insight is not built
• …
You get what you created

Yesterday's solutions created today’s problems

AND

Today’s solutions will create tomorrow’s problems.

“Good” is “not good enough” and “not good enough” is appalling.

A system behaves, produces the effects and delivers accordingly to its capabilities, logic and characteristics.

Badly conceived systems delivers bad results and more adverse effects (limitations, difficulties, inefficiencies, other problems, …)

AND

Well-designed systems dealing with the right issues and fitting in their environment delivers good results.

Our solutions have to do much more than solving only today’s problem.
What goals drive our thinking, decisions, actions and initiatives?

**Notes:**

- Companies continue to computerise. More and more work is executed by software. Some systems are of strategic importance (example: supply chains). But the analyst can’t contribute at the strategic level if the mind still is driven by the idea of “developing demanded software features to satisfy the user”.

- Being driven by the higher level goals requires a more holistic view, different goals, different competencies, different relation with the business, different responsibilities, ...

- Working at a higher level is harder, but much greater value can be created.

- Trying to satisfy a lower goal, being driven by the lower motives, does NOT automatically imply a positive contribution to the higher level plans and goals. Even the opposite can be true (example: sub-optimisation)
Global Objective

GOAL: Company’s Prosperity

Develop and Strengthen the Enterprise for thriving (supra-system, body of the company)

Driving Business (Activities)

Maximise the Exploitation of Information

Information as Product and Service

Develop Information Capabilities

Develop and Improve Systems

Improve the Information Pool (collection of information; information = resource)

Strong Enterprise + Thriving Business

Durable Customer Satisfaction

Customer satisfaction about systems is an indirect goal, a desired consequence. (more about it later)

Notes:

How feasible is it to deploy profitable business activities with a weak enterprise? And over time?

How feasible is it to improve profitable business activities with a strong and well-designed enterprise?

Is a strong, well-designed, fit-for-purpose system or product more able to create customers satisfaction than badly designed and weak systems and products?

Customer satisfaction is discussed more in details in AB6.
Some Important Shifts

1. Not aligned

Systems Analysis Discipline

2. Information Component (IC)

Information System

Software System

3. Informatics

IT / ICT

4. “Business Analyst for IT”

Business Information Analyst (BIA)

or

Business Information Systems Analyst (BISA)

5. • Responding to a demand
   • Building Software

6. Design → Building → Deployment

Diagnose → Analyse → Design → Building → Deployment

- Contributing to business objectives
- Maximising the exploiting Information
- Building the company’s IC
- Improving the functioning of the company
Professionalisation

Perception of Systems Analysis

as “Activities” that anyone can do

as a “Professional Discipline” to be mastered by professionals

<table>
<thead>
<tr>
<th>ACTIVITY</th>
<th>ROLE</th>
<th>KNOWING (how-to-Knowledge)</th>
<th>FOLLOWING</th>
<th>APPLYING</th>
<th>RESPONDING</th>
<th>DOING IT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISCIPLINE</td>
<td>PROFESSION</td>
<td>UNDERSTANDING (WHY-Knowledge)</td>
<td>DECIDING</td>
<td>ADAPTING</td>
<td>LEADING</td>
<td>THINKING INFORMATION</td>
</tr>
</tbody>
</table>

Inferior Results
Higher Risks

PATH TO EXCELLENCE

Notes:

- Considering ‘Analysis’ as a set of activities, assuming anyone can be an analyst, assigning the role to an employee, learning and applying techniques and following standards, responding to demands, seeking to satisfy others, being lead, copying, perpetuating (bad) habits, ... → key to disasters

- Developing oneself to an expert, learning to master the discipline, thinking independently, determining and organising own work → professional
The Analyst

Anyone who performs Analysis activities

Anyone who **MASTERS** the Systems Analysis discipline, or any of its variants or sub-sets, and the capabilities to apply it

One does not become professional Analyst overnight.

"What got you here won't get you there“ (Marshall Goldsmith)

**Notes:**

- Every person can fight, but not everybody can fight like a black belt martial art practitioner.
- Assuming that performing analysis activities turns someone into an Analyst or that anyone can become Analyst in a short period of time is a immense underestimation of the discipline and of the true importance of the role.
PROFESSIONALISATION

Professionalisation is not a luxury. It’s a must.

Mastery of disciplines:

• vocabulary – Major terms
• positioning of the tool, technique, methodology, concept, ...
• knowledge of the process(es), method(s), tool(s)
• understanding the general idea behind the process(es), method(s), tool(s)
• Ability to use the tool or to apply the method/technique/process as prescribed
• Understanding the expected result
• Develop the own skills
• understanding the meaning, purpose and value
• Understanding preconditions, applicability, key aspects, importances, priorities, options, alternatives, strengths and weaknesses, constraints, the limits, risks, the evolution, forces, influencing factors and tendencies
• Understanding the why and effects of every step or aspect
• Able to adapt the technique, method or usage of the tool to the context and circumstances. Understanding what can be done and what can’t be done.
• Understanding variances; ability to tune
• Able to take a broader range of the aspects into account
• Understanding the underlying purposes, mechanisms, laws and principles
• ability to see and understand the true nature of things
• ability to put everything in broader perspective
• Limiting the unwanted effects
• understanding effects, consequences and implications of decisions and choices
• Identification of opportunities to apply the technique or to use the tool
• Optimal exploitation of the approach, method, technique, tool
• Able to further develop the discipline
• Attitude: open-minded, ability to learn, curiosity, critical thinking, ability to identifying and understanding what is imperfect or problematic, seeking to do right, methodical, sharing knowledge, ready to go the extra mile, pursuing higher goals or harder challenges, using higher norms, autonomous thinking, truth-seeker, ...
Pursuing Higher Goals and Professionalisation

• Better tools
• Better collaboration and social interaction
• Better methods and techniques

• More knowledge
• Better insight
• Understanding ineffective approaches and habits
• Clarify and eliminate assumptions
• Beliefs

Our overall view, our perception, our belief system and how we perceive our role, determine the results we can produce and thus the value we can create.

It can limit it or it can increase it remarkably!
“Who cannot remember the past are condemned to repeat it.”; George Santayana

Or, we are condemned to repeat our mistakes endlessly until we learn the lessons we have to.

If we keep run into the same kinds of struggles and problems over and over again, until we learn that lesson that allows us to progress.
“Insanity is doing the same thing over and over again, but expecting different results.” (*)

Rita Mae Brown

Likewise

It as is insane to do the same as everyone else and to think like everyone else while expecting superior results, results way above average.

Follow the example of the top-level analysts. Be careful whit thinking or doing like the majority.

Notes:

(*): quote is commonly misattributed to Einstein.

Set aside of intelligence or skills, what are the differences between lower than average analysts, average analysts and significantly better analysts?

It is easy to align the thoughts with that of the majority. But if the majority is wrong, it is not always the best thing to do.
KEY TO PROGRESS

“In times of turmoil, the danger lies not in the turmoil but in facing it with yesterday’s logic.”

- Peter Drucker

“We cannot solve our problems with the same thinking we used when we created them.”

- Albert Einstein

“There is no more neutrality in the world. You either have to be part of the solution, or you're going to be part of the problem.”

- Eldridge Cleaver.

“If you don’t understand it, don’t mess with it.”

- Louis Armstrong

“Nothing that results in human progress is achieved with unanimous consent. Those that are enlightened before the others are condemned to pursue that light in spite of the others.”

- Christopher Columbus

Since a paradigm shift is necessary and some beliefs and assumptions have to be reviewed, the next slides are about learning and changing beliefs.
Progressing

Creating More Problems and Preventing Progress (*)

• Partial or lacking understanding
• Ideas that do not correspond with the real nature of things and with the real situation
• Accepting seemingly obvious but unverified ‘truths’
• Hidden assumptions
• Valuing trying and exploring over learning and thinking
• Doing, activity, action is more important than thinking
• Knowing = Understanding
• Pre-cooked instant answers, relying on common and widely accepted answers at face value
• Aligning own opinion with the mind of the majority or group
• Inability to question one’s own view, ideas, beliefs, understanding, priorities, ...
• Rejecting contradicting information without prior careful investigation, without true understanding
• Preferences and dislikes to protect ourselves
• Single perspectives, local considerations, simplistic ideas, simple (but wrong) answers, oversimplifications, considering one step at a time, ...
• Rejecting problems, complexity and everything one doesn’t like to hear
• ...

Notes:
(*): These habits are contrary to the spirit of a true Analyst
Mental Harmony Puzzle

PROCESS

The mental picture is created like a jigsaw being solved. Each idea or information element is like a piece.

1. First, we have no mental image.
2. For each piece of information that is received, it is (or not) verified: Is it true? Believable? Does it fit with what I already know?
3. If the new piece of information doesn’t conflict with what I know, I integrate it. If not, it is rejected.
4. Gradually, as new pieces are added, a picture is formed. This picture is the reference for all new incoming information. To obtain a harmonious picture, the new pieces of information have to be coherent with the global picture.

= “Mental Harmony Puzzle” (Dr. Chaesan So)

3 Problems:

• New information that doesn’t match our mental picture, creates a discomfort and is likely to be rejected. It is filtered and ignored. Contradicting information is more likely to pass unnoticed.
• Information matching our picture is more likely to be noticed, to be searched for and to be integrated. Example: we are more inclined to select sources which are in line with what we already know and want to hear and are more likely to listen to them.
• It is more important that information is in line with what we know and with what we want to hear, than whether it is true or false.

It is hard to change our mind, to change our puzzle, to change what we already know.
Mental Harmony Puzzle

Known information forming a coherent mental picture

Incoherent?

new piece of information

Is this false?  Or  Is this false?

Ready to see information contradicting our puzzle. If there is an incoherence then we have to investigate what and why?

Need to be really ready, to be open-minded and to be willing to question our own knowledge, vision, understanding, belief system, ... as well (even if we don't like it)

Two ways:

Looking at real world, measuring, experimentation, ..

By thinking, by logic, by reasoning

or a combination of both

Important to keep questioning, correcting, adjusting and improving our mental picture, our mental map, our beliefs.

Having a coherent mental picture is important. But, keeping a vision of the world, a logic, a belief system that doesn't reflect the reality, that is distorted, that has unjustified priorities, doesn't make sense. You can't build something lasting upon. The mental image and thinking patterns have to reflect the real world as much as possible. It is better to be a truth seekers, to truly try to know, rather than a protector of the own knowledge.
"Unconscious knowledge": “we know, but we aren’t aware we know it”
Without loop 2 learning, progress is possible though limited.
Lacking of knowledge is a problem. But, there is worse: ignoring the very existence of all the knowledge we still can and should learn.

Dunning-Kruger effect: The one who knows little, thinks he knows a lot and is self-confident. And the opposite is true for the one who knows a lot.
Using AB6 Framework

Key Questions:

Why is this statement, principle, ... ?

Content of AB6

Statement, Principle, Idea, Concept, Definition, Model, ...  

What are the Implications?  
How (& when) to apply it?  
Is it applied? How?  
Differences with today?  
Consequences of the differences?  
How to solve this diff.?  

Towards origin, deeper understanding

Towards integration and practical application
A Few Words of Wisdom

... describing the mind-set of AB6

• Today’s problems come from yesterday’s solutions.
  Peter Senge

• If you only read the books that everyone else is reading, you can only think what everyone else is thinking.
  Haruki Murakami

• Mistakes are a great educator when one is honest enough to admit them and willing to learn from them.
  Alexander Solzhenitsyn

• You don't drown by falling in the water; you drown by staying there.
  Edwin Louis Cole

• Wrong does not cease to be wrong because the majority share in it.
  Leo Tolstoy, A Confession

• There is nothing wrong with change, if it is in the right direction.
  Winston Churchill

• Look deep into nature, and then you will understand everything better.
  Albert Einstein

• I’m trying to free your mind, Neo. But I can only show you the door. You’re the one that has to walk through it.
  The Matrix

• There are no traffic jams along the extra mile.
  Roger Staubach
Important Note

Please, take the following into account:


Most people understand the meaning of these common words. Their knowledge about the subject can be contained in a paragraph, in a few phrases. This creates only the illusion of ‘knowledge’ or understanding and adds confusion.

In the light of a professional discipline, such an explanation represents only an incredibly superficial understanding. Each of these terms is a subject that can fill books.

However, we are all on a learning path.
The term “Analysis”

**ANALYSIS** is:

- about **studying** the parts and their relations – looking **inwards** into something
- a **phase** in a project
- a branch, a group of tasks, in a **WBS**
- a **type** of activity
- a **process**
- a **method** or technique
- a **set** of activities (that anyone can perform)
- a set of **deliverables** (documents and models) resulting from analysis activities (“the analysis”)
- a **professional** discipline

All of these meanings are correct. The meaning depends of the perspective, the context.
The Term “Business”

What is “Business”?

• The business community
• A part of the company or organisation
• Business knowledge, business logic, business expertise, field experience, ...
• Something under the authority and responsibility of the business community
• Anything emanating from the business community
• Anything the business community deals with or should deal with
• Everything that is not IT
  – A way to indicate matters and issues IT don’t want to deal with (not wanting to take responsibility over it, non interested in it, time pressure, ...)
The Term “Business”

The label “Business” has two meanings

1. “Business” as activities deployed by the company directly related to a business domain.

2. “Business” related to the Business Domain

Business domain is a specific knowledge domain linked to the purpose or to the main functions of the company.

Examples: insurance, manufacturing, energy, accounting, marketing, production planning, ...

Beware! This label creates a lot of confusion!

It does NOT mean “related to”, “executed by”, “experienced by” or “emanating from” the business community (BC)

Not all problems are business problems
Not all processes are business processes, ...
...
**Notes:**

Old synonyms: Company, Organisation, Enterprise.

However, when we look at a company, it is constituted of parts of different nature. They require different fields of expertise to be engineered. These parts are modelled by different experts. This creates the need of a more precise differentiation of these parts.

(human) organisation *is part of* the enterprise *is part of* the company.

The company is the supra-system of the Enterprise.
Note Concerning Models

A model represents an idea, a concept, a process, a structure or a mechanism.

Models reflect a part and/or an aspect of reality or of a mental construct.

They represent an abstraction, generalisation, part or perspective of the reality or of a mental construction. They represent a simplification of them. There is always/often a difference between the model and the reality.

Models provide some insight. They are very useful, particularly to master complexity.

Models serve to guide the mind during a thinking process or to communicate an idea or mental construct.

Although, strictly spoken, they are wrong. They can not necessarily be applied “as is” (literally). They have to be interpreted correctly. It is the idea that they convey that matters.

This applies to the models the Analyst makes, but also to the models in this framework.
FOUNDATION

CORE IDEAS

------------------------
“Begin with the End in Mind”

The environment, the situation, the context and the goal we seek to achieve will determine everything else.

This is thus the starting point.
Overall Approach of the Elaboration of AB6

Environment
Supra-system
System
≈
Business Environment
Company
System of Information Systems

Environment, situation and context
Goal
Product or Outcome
Engineering and Building Process
Organisation
Work Environment
Equipment
Methods
Information, knowledge and understanding

What is the role of information in the company and the company’s information needs?
What is the goal of corporate IT?
What is the product of corporate IT?
Study of this product.
What is the nature of the engineering activities of the IT dept.?
What is the fundamental pattern within these processes?
Further detailing of this pattern and process and constituting building blocks for a process.
Study of these elements

Notes:
• Not all goals are expressed. Some are assumed. Or, we may not be aware of some goals and thus ignore them. Understanding the nature of goals (goal modelling techniques may help) is crucial.
• Many solution requirements will come from the study of environment, context, situation and goals.
A company is a man-made open dynamic purposeful evolving socio-technical managed living system

Man-made → understandable, manageable, ...
Open → interaction with environment
Dynamic → has a behaviour
Purposeful → has a specific role, objective, guided, ...
Evolving → changes, adapts
Socio → containing people
Technical → containing machines, computers, tools
Managed → controlled, managed and guided
Living → complex system seeking to exist and to prosper

These are goals, capabilities and characteristics that approaches, methodologies, methods, architecture and sub-systems have to respect, to implement, or to contribute to.
A Company as Heterogeneous System of Systems

System of Systems

- People organisation
- Industrial systems
- Accounting system
- Cost system
- Communication systems
- Transport system
- Software systems
- ...

Heterogeneous

Systems are of different nature
Example:
- people organisation ≠ software system
- financial system ≠ industrial system

Question:
- Have to fit, to collaborate and to function harmoniously.
- Requires different domain expertises and engineering skills and methods

System of systems ≠ collection of systems
Living systems are open self-organising living things that interact with their environment. (Wikipedia)

Living systems transport:
- Matter
- Energy
- INFORMATION

Living systems
- maintain themselves
- organise themselves
- develop themselves
- adapt themselves

Organisations (including the architecture, subsystems, capabilities, ...) must be designed with these abilities in mind

A matter of survival
“A COMPANY, AS LIVING SYSTEM, CAN’T EXIST WITHOUT INFORMATION”

Information is a CRITICAL RESOURCE

The company needs to

• Increase its amount of valuable information
• Preserve and maximise its value
• Maximise its exploitability
CONSIDER

• A company needs to capture, identify, connect, organise, control, store, process, manage, share, disseminate information (and much more)
• Information has multiple usages
• Information provides opportunities
• Information has Value. The value decrease and/or increase. It can be preserved and increased but also destroyed.
• Information can be captured or created \( \rightarrow \text{AMOUNT INCREASES} \)
• We tend to create and add, but omit to clean up.
• Tendency to ENTROPY (natural tendency to evolve towards chaos).

CHALLENGE

Information must be **EXPLOITED**
Information must be **MANAGED**

in an **OPTIMAL WAY**

Therefore, the company needs and must have

• an appropriate environment
• systems
• and the right capabilities.

To take up this challenge, the most skilled people are required
Evolution of Information in the Company

How is the company’s information evolving?

<table>
<thead>
<tr>
<th>Quantity</th>
<th>(probably)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality</td>
<td>Known ? Fluctuating</td>
</tr>
<tr>
<td>Value</td>
<td>Known ? Fluctuating</td>
</tr>
<tr>
<td>Organisation</td>
<td>.....?</td>
</tr>
<tr>
<td>Manageability</td>
<td>.....?</td>
</tr>
<tr>
<td>Exploitability</td>
<td>.....?</td>
</tr>
</tbody>
</table>

**CHALLENGE : PREPARE FOR THE FUTURE**

Today:

- Information explosion
- Communication explosion

**REFLECTIONS**

Endless growth of information? Good or bad? To be managed? To be controlled? How to deal with it? How to adapt systems to it?

Financial means are managed in a formal and strict way. Why should this not be necessary for information?
A PERSPECTIVE ON CORPORATE IT's MAIN QUESTION

The information component resolves the company’s information needs. Or, the information needs determine the information component. But this is the minimal viable solution for the short term. Continuously and actively seeking to maximise the exploitation of information, to innovate, keeping the internals of the information component right and a seamless integration are necessary for long term survival and prosperity.

Two other key-questions, regardless of needs or demands are:
• How to preserve and maximise the value of information?
• What information risks are or will be faced and how to avoid or to limit them?
Two Main Perspectives

How can we conceive, strengthen, improve and expand the systems, structures and processes dealing with information of the enterprise?

How can we maximise the capability of the company or enterprise to exploit information resource? How can we increase the value of the information and use it to the benefit of the company and its customers.
Company’s Needs

What does an organisation need to get the most out of information?

The Role of the IT department is to deliver all this

Explanation of the graph on next slide.

All these aspects form a single whole in which everything is related. The grey lines represent only the most important relations.
A company (organisation) needs to be able to exploit information and to innovate with information.

Therefore, it needs the resource ‘information’, the capabilities to deal with information and to deliver information as a product or service.

As much ‘information’ as possible has to be organised, managed and stored in a warehouse (the organised ensemble of all the means capable of storing information; not to be confused with a ‘data warehouse’ – BI concept).

The information capabilities require organised information. These capabilities are defined by the potential of “information usage”, which can be a real usage, a possible usage or an expected or intended one. This usage can be manual or it can be through defined information processes.

A correct use of information necessitates two competencies: subject matter knowledge and information competencies.

The execution of processes can be supported by systems or they can be executed, fully or partially, by them. Communication channels and information flows transfer information among different entities and systems.

Innovation with information is done through the information capabilities, more precisely, how the information is organised and used. And innovation can be realised through the offered products and services. Information innovation requires strong information competencies. Innovation in information is primarily achieved by filling in a need or by new concepts and new algorithms. Information innovation, information concepts and information competencies are actually strongly related to each other.

The concrete Information Component contains, among others, a technological architecture and infrastructure and information and communication tools and technologies.
The **Information Component** is the whole organised environment and means allowing the capture, storage, management and exploitation of Information.

- Information Capabilities
- Information Resource
- Information Value
- Information Exploitation
- Information Products
- Information Services
- Innovation with Information
- Organisation of Information
- Information Processes
- Information Usage
- Information Systems
- Information Management
- Information Risk Management
- Information Warehouse
- Information Concepts
- Information Competencies
- Technological Architecture and Infrastructure for Information and Communication
- Information and Communication Tools

**Note:** Since a person is able to deal with information, when (s)he does so, (s)he is (at that moment and from that perspective) part of the information component. So, there is an overlap between the components. Or, the border is blurry.
A company is a man-made socio-technical system

“A huge part of the company can be engineered.

• Some parts or sub-systems can be fully engineered.
• Other parts can be engineered up to a certain point.
• A company is a living system. This doesn’t imply that all its parts are living systems as well. Some of its parts are complex and will evolve, but are not complex adaptive systems or not living systems.
• Some qualities, aspects or behaviours may still be unexpected, either because of lack of knowledge and thinking, because users are more creative than builders or because they were (really) unpredictable (should be very rare).
A Company Can Be Engineered

Half (*) of the results determined by

- Engineered systems and processes;
- Appropriate, performant, reliable equipment
- Availability of resources; ...

Other half(*) of the results

- Culture
- Leadership
- Collaboration
- Communication
- Values
- Norms
- Attitude
- Interpersonal relations
- Social skills
- Competencies
- Motivation
- Intellectual skills
- ...

can inhibit each other

Notes:

(*) “half” is figuratively.
Psycho-social issues can completely prevent a good system to function properly.
An inappropriate organisation, system or process can block skilled and motivated people to deliver results.
• A system has a purpose and, through this purpose, it plays a role in an environment and in a situation.
  • The purpose of a car is to transport a limited amount of people and smaller goods over the road or over land. It should be able to drive on certain types of roads of different qualities, in different weather conditions, safely and for several years and for 150, 200 thousand kilometres or more.

• A system performs itself activities
  • Example: A car drives. Its engine runs.

• A system is also used to deploy activities
  • Example: A car (system) is used by the taxi driver to deploy taxi activities.
    • Note: a car is a system; but a car + driver is also a system

• Activities are performed with a certain intend, to get an outcome, to reach a goal.
  • The taxi-company should make profit.
**Systems - Activity**

<table>
<thead>
<tr>
<th>Systems</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Army</td>
<td>War</td>
</tr>
<tr>
<td>Body</td>
<td>Running a marathon</td>
</tr>
<tr>
<td>Car</td>
<td>Taxi activities</td>
</tr>
<tr>
<td>Company</td>
<td>Deploying business activities</td>
</tr>
</tbody>
</table>

Design based on purpose, system’s life, environment and operational conditions

Then, adapted to the specific activities

*Also true for company? For information systems?*

**Principles**

- Simple and cheap tools or features can be designed for purpose and usage.
- Complex, expensive and lasting systems must be designed primarily based on their purpose, their future life and their environment (operational conditions).

**Notes:**

- What if we declare war, without building or strengthening the army first?
- What if we run a marathon, but ignore the capabilities of the body and its signals?
- What if we are able to run one marathon, but finishing exhausted. What if we have to run continuously marathons one after the other? (sustainability)
- What if a car can transport people only for a too short period of time or if it can’t operate when it’s raining, when it’s freezing or when it’s too hot?
A system has a goal: "SELF-PRESERVATION"

**SURVIVAL PROSPERITY**
- Growth of company
- Continuing existence
- Strengthening
- Efficiency
- Effectiveness
- Increase of capabilities
- Increase of capacity
- Responsiveness
- Adaptability
- ...

Activities are executed with an intend, to get a result, to reach a goal

**Successfully executed activity**
- Results as outcome of the activity

Examples:
- Reaching a sales volume
- Dominating a market
- Making an amount of benefit

If the long term has a low priority or is ignored, then there is no solid foundation to build upon, let alone to build something sustainable.

Anything threatening the long term systemic goals, undermines the system’s chances for survival and prosperity.

Reflection:
- Can a person deploy activities with a system, if that system is weak or broken?
- What will happen if the focus is only on activity results while the system’s objectives are ignored?
- Who is interested in system’s objectives and who’s is much more focussed on the activity objectives?
- Which type of objectives is the most important? Why?
A system creates value by its \textit{functioning} and by its \textit{usage}. This value is \textit{for the final customer}.

Example: A car creates value by functioning and by the usage the driver does with the car.

How do Engineers (Systems Analysts) create value?

- Improvement of the System (Enterprise, Company)
- Creation of true value for the final customer
Value Creation

Higher outcomes are the trigger, the goal and the motivation of the Analyst

<table>
<thead>
<tr>
<th>Improve and renewing the supra-system (company)</th>
<th>Driving and renewing the Business (Final Customer)</th>
</tr>
</thead>
</table>

How?

<table>
<thead>
<tr>
<th>System</th>
<th>Activities - Processes</th>
<th>Products and Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduce required material</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reduce waste</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reduce risks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creating opportunities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Improve capabilities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Innovation</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Main constraints:

- Not degrading or endangering the supra-system (enterprise), its environment or the resources
- Aligned with strategy and plans
Man-made system is the outcome, the product, of an engineering activity.

Once conceived and built, the system is transferred to the user.

using a system ≠ engineering a system
Systemic Harmony

Achieving a well-functioning system

What is HARMONY?

- Components form together a larger unified whole
- Optimised contribution of every part to the whole’s interests
- Environmental fit
- Functional alignment
- Alignment of capabilities
- Optimal coordination
- Coherent logic
- Balance
- Efficient and effective collaboration
- Internal logical, ordered and balanced structure
- Systemic Integration
- Mutual accommodation of the components
- Optimal use of resources (including information)

This quality of systemic unity is a crucial aspect of any stably perduring whole.
Systemic harmony is achieved when different parts of a system form a harmonious whole and work seamlessly together. This unity is crucial for the long term stability and existence of the whole.

\[
\text{requires}
\]

**COLLABORATION among engineers**

The organisation must structurally facilitate the collaboration among the engineers & company must be structurally integrated.

**Note:**
No or difficult collaboration → no well-functioning system → problems
The collaboration is not restricted to a collaboration between the engineers only.
A company has to adapt. It has to **REINVENT itself**. It defines its own structure, systems and processes.

Harmony among components (sub-system)  
↓
Strong collaboration among engineering disciplines (and among engineers)  
↓
COMPANY = SYSTEM

COMPONENT “ENGINEERING”
- Engineering of the company
- Engineering of its products and services

SYSTEM’S USERS
- Business community are the users using the system (the company) to deploy business activities

**Engineering Roles & Disciplines**
- R & D
- Product Development
- Organisation Development
- Industrial Engineers
- All architect, analysis, engineering, modelling, developers roles of Informatics

**Note:**
Collaboration with business community remains crucial as well.
Levels of Thinking

Thinking globally, holistic, long term
More stable

Thinking locally, short term
Very variable

Each level has a different type of goals, scopes, environments and leads to different types of products and results. It requires different approaches and methodologies, different responsibilities and authorities, different methods and techniques and requires different competencies.
MAJOR SHIFT

IT department ≠ Informatics Department

“Business-IT”-model ≠ “Business – Informatics” Model

IT ≠ Informatics
Informatics is an interdisciplinary science
Notes:
It is not that Informatics is absent. It is watered down and undervalued. It is practised by business people and IT people. IT people have some more knowledge in informatics, but it is not really their focus or objective.
The flaw is somewhat mitigated by Analysts and Architects.
Strictly spoken, IT is a sub-discipline of Informatics.
FOCUS of INFORMATICS Department

1. INFORMATION AS RESOURCE
2. INFORMATION QUALITY & VALUE
3. INFORMATION CAPABILITIES
4. INFORMATION EXPLOITATION
5. INFORMATION COMPETENCIES
6. INNOVATION THROUGH INFORMATION
7. INFORMATION AS PRODUCT OR SERVICE
8. INFORMATION NEEDS
9. INFORMATION PRACTICES & USAGE
10. INFORMATION RISKS
11. INFORMATION SYSTEMS
12. INFORMATION PROCESSES
13. INFORMATION FLOWS, SHARING AND REUSE
14. INFORMATION CONCEPTS
15. ORGANISATION OF INFORMATION
16. INFORMATION WAREHOUSE
17. INFORMATION MANAGEMENT
18. AUTOMATION / AUTOMATED SYSTEMS
19. INFORMATION AND COMMUNICATION TECHNOLOGIES, TOOLS AND INFRASTRUCTURE
20. TRAINING / INFORMATION SKILLS DEVELOPMENT
Informatics Department is an alternative name indicating that the department plays a more important role and does way more than just “IT”. Another alternative name could be the “Information Department” (similar to the “Finance Department”).
Values

BALANCE

70% (+ or - 15%)

• Value-driven
• Plan-driven & plan-based
• Holistic
• Learning by analysing & thinking
• Top-down
• Collaborative
• Adaptive

30% (+ or - 15%)

• Opportunity-, demand-, problem, need-driven
• Local
• Do and Learn
• Bottom-up
• Exploration

Defining overall objectives, long term, main lines, overall architecture, ...

Adjust, fine tune, adapt, improve, seize local opportunities, ...
Consolidating the Role of the Analyst

Presently

Exploitation of the full potential of the discipline and increasing the added value requires the expansion and consolidation of the role of the Analyst

Analytical skills, Synthesis skills, Critical thinking skills, Abstract thinking, Systems thinking, Autonomous thinking, Structured thinking, Problem Solving, Curiosity, Open mind

Underestimated, misunderstood and missing in today’s “IT”

(Business) Informatics

Analytical skills, Synthesis skills, Critical thinking skills, Abstract thinking, Systems thinking, Autonomous thinking, Structured thinking, Problem Solving, Curiosity, Open mind

Specific Thinking Skills
Information Science, Systemology, Management, ...

Underestimated, misunderstood and missing in today’s “IT”
FUNDAMENTAL TOPICS

1. SYSTEM
2. COMPANY
3. INFORMATION
4. PROCESS
5. PROBLEM SOLVING
- SYSTEMS -
**Systems**

- **System**
  - has a purpose, a role

- **Supra-System**
  - System
  - can be part of a supra-system

- **System**
  - has capabilities

- **System**
  - has functions

- **System**
  - has a behaviour, mechanisms

- **System**
  - has internal structure, organisation, architecture

- **System**
  - has sub-systems and components + interactions among them
  - Note: Sub-systems are systems (co-systems)

- **System**
  - has limits, weaknesses, critical parts, ...

- **System**
  - has characteristics

- **System**
  - can be homogeneous – heterogeneous (components of different nature)
Systems

flows of matter, energy and information

exists in an environment

exchanges and uses information, energy and (often also) matter

Influences, forces, exchanges and interactions with environment. Even its existence, the fact of being present, has an influence on the environment.

is temporary, has a life (full of events), has a lifecycle (even if the end is not foreseen)

System

System

System

System

User

users

may contain users

can be used (by 1 or more users or users groups)

can be autonomous
System’s Characteristics

Going more in depth
System’s Characteristics

Different components are similar. Groups of characteristics may be repeated.
Identifying a System

The concept of “system” is a method to structure and organise the world to help the human mind to understand this world.

A system can be determined by selecting sub-systems and grouping the parts in many different ways.

The purpose, the goal, the collaboration, the cohesion, the logical domain, the natural ‘boundaries’ and how it is best to manage are good criteria to define a system and its boundaries.

What is true for a system, is also true for supra-systems, sub-systems and co-systems.

Overlapping Systems
The Core of a System

Everything that shapes systems, affecting the Structural Design and Core Mechanisms, usually strongly related to the system’s purpose

**CORE CONTENT**

- Core Processes
- Core Mechanisms
- Core logic of key processes
- Key persons
- Role, functions, capabilities of components
- Architectural Structure
- Architectural qualities (integration, fragmentation, separation, duplication, scalability, interoperability, ...)
- Organisation of key resources (energy, matter, information, money)
- Components and Mechanisms required for survival and thriving

Adapting the Core requires huge changes, possibly structural changes. It is a lot of rethinking and reengineering. Often, it influences other areas across the system. The core should be well-thought and as stable as possible. *(How to increase the stability of the core?)*. A system can be organised with inner-core, outer-core, foundation, ... These are just additional layers of the onion.

Periphery: (In software systems): features, user-interface, local logic, loosely coupled information, ... Changes in the periphery may affect other aspects of the periphery, but won’t affect the core and are often simpler, local and easier to implement. More layers are possible (like an onion).

Not an end-user or stakeholder’s issue: Something might be important or even vital for end-users and can be located in the periphery !! Or inversely, something can be of no importance to the end-users but be part of the core layer.
Criticality

Optional – Useful - Important - Critical

System’s Perspective
Structures, principles, mechanisms, components, capabilities, capacities or resources important for the survival, well-being and thriving of the system

User’s Perspective
Capabilities, features, capacities, outputs important for the usage of the system

Notes
Example: A system may be useable, but without internal controls and maintenance, over time, the system will degrade.
The more important the item or aspect, the more attention of engineers it deserves.
Hierarchy of Systems

Supra-Systems and sub-systems are all systems. The “System” indicates the level in the hierarchy of systems under consideration.

1. Sub-systems and components are within the system’s boundaries and contribute to the system’s goals.
2. A sub-system can be the system. A supra-system can be the system. It depends on the system under consideration.
3. The hierarchy can have as many levels of supra-systems and sub-systems as needed.
A co-system is a system at the same level as the system under consideration. Often there is an interaction, an exchange, a relation among them. They are often part of the same supra-system.

Components and sub-systems are connected forming the architecture. They interact and collaborate. They collaborate towards a common goal. Together, they implement the system’s capabilities and qualities.
A LIVING SYSTEM SEEKS TO EXIST INFINITELY AND TO PROSPER

Hierarchy of living systems

- Supranational system
- Society
- Community
- Organisation
- Group
- Organism
- Cell

Levels of company to individual

Body + Conscious Mind

Body

Activities

Formed by natural evolution (but equipped with conscious mind)

Execution is guided by conscious mind (implies decision making) (and unconscious mind)

Notes:

Decisions (intellect, mental, conscious mind) and activities may shape the body somewhat. But basically, it’s a product of natural evolution.

A company is a living system. However, this does not mean that all its sub-systems and parts are living systems as well.
Cynefin Framework ("KUN-iv-in")

**Complex**
- Unknowable
- Emergent Practice
- Pattern management, ‘The Approach’, Matriarchal/Patriarchal leadership
- Probe – Sense – Respond

**Complicated**
- Knowable
- Good Practices
- Analytical / Systems Thinking, Methodologies, Oligarchic leadership
- Sense – Analyse – Respond

**Chaos**
- Turbulent and unconnected
- Novel Practice
- Charismatic or Tyrannical leadership
- Act – Sense – Respond

**Obvious**
- Known
- Legitimate best practices
- Feudal leadership
- Sense – Categorise – Respond

*Inventor: Dave Snowden*

**TRAP:**
It is too easy to qualify a system as a complex adaptive system (CAS) based on the personal inability to learn, to analyse and to understand it or in order to avoid to do the effort of understanding it.

**Systems Analysis** has been developed in order to understand more complicated systems.
Scale of Systems

Principles and concepts are applicable to these systems

Some principles and concepts are only applicable to man-made systems
The definition and the understanding of the system’s environment is critical to the system’s survival.

The environment can be small (orange) or larger (green).

The boundaries can be solid, impenetrable, penetrable to different degrees, clear or fuzzy. It may vary in time.

The environment is defined by the elements and the space between these elements able to influence the system one way or another: input, output, constraints, obligations, forces, opportunities, exchanges and interactions, impacts, ... The more perspectives (holistic) are taken into account, the more meaningful the defined environment is. The environment may evolve over time.

The definition of the environment of a system, but also of a supra-system, a sub-system or a component, is dependent of the insight of the person defining the environment. The definition of the environment should be as stable as possible, but it is not static. Its definition can be adjusted.

The understanding of the environment and the relation and all interactions between the environment and the present or future system is crucial.
1. System
   • Purpose, structure, mechanisms, strengths, weaknesses, risks, prerequisites and limits. (This is valid for anything: process, methodology, system, …)
   • Every part of a (clean) system has a purpose, a sense, …

2. Supra-system
   • Including the supra-supra system, supra-supra-supra system, …
   • A system needs to be fully part of its supra-system (structural/systemic integration = integration by design).

3. Environment(s)
   • must be determined properly (must contain everything influencing, even indirectly, in a meaningful way, the system), including those of the supra-systems
   • the relations, interactions and influences between the environment(s) and the system(s).

4. Nature of the resources
   • Resources and materials inputted, outputted and/or changed by the system

5. Internals weakening or that may weaken the system
   • Internals: structures, mechanisms, dynamics and tendencies
   • Among others, mechanism creating disequilibria endanger the system !!

For studying, analysing, engineering and managing systems
(like software systems, IS, IC, company, …)

Analysis Approaches have to include this !!
A person, a group, a team, a project, a community, a society, a component, a (sub-)system, a process exist in an environment.

They are influenced by the environment.

They influence the environment.

They depend of the environment.

Their existence, their thriving and survival depends of their environment.

We can’t function, decide, solve, conceive, heal, restore, guide, build, grow, ... while ignoring the environment(s).

We can’t do without understanding the environment.

See, consider, focus also on the environment...

SEE the ENTIRE picture

We can build the nicest and most advanced submarine. But it will be completely useless if it is in the desert. Even building it with little knowledge of the ocean is a recipe for failure.
Key Design Principles for Man-Made Systems

Control and Protection
A system without control and protection mechanisms is at risk. Input, behaviour, performances, resource usage and outputs must be controlled.

Balance
Examples of balances are capabilities able to process approximately same amounts, win-win-situation, equity, and so on.

Modularity - Component-based
Modularity is achieved by having components that are loosely coupled (separation of concern) and high cohesion. Components can then easily be adapted and replaced. They create order. Components are also easier to support reuse.

Reuse
Developing for reuse can save a lot of time and effort. However, reuse requires a broader understanding and good anticipation skills. Components containing more general logic are usually potential candidates for reuse.

Internal Order
Internal order is, at glance of no concern of the business stakeholders. However, it reduces the time required to search for the concerned source code. It facilitates debugging, maintenance and adaptations. It decreases the learning curve. Order reduces complexity and complexity related risks.

Natural laws, Science, Rational Logic
Some laws, rules and principles may simply never been broken or the system won’t work or it will lead to its decline. It is important to be aware and to understand these laws, rules and principles and where and how they apply.

Foundation
A system has to be based on a solid, well-organised foundation. The foundation can be organised in layers.

Knowledge – Insight – Truth
In large and complex systems or environments, it might be a challenge to acquire the required knowledge and insight. Often the truth may not be what we expect, want to hear or are ready to hear. But designing a system or solution with insufficient knowledge or a distorted mental image leads to problematic results.
7 Main Survival Rules

A MAN-MADE SYSTEM has to

1) have a **worthwhile purpose**, to function accordingly and to use available resources for it

2) be **useful**, produce (real) **value**

3) fit in its **environment** and be accepted (acceptability: like moral, respectful, ...)

4) fit into its **supra-system** and serve it by contributing to its purpose

5) have internal **harmony**: maintaining balance, reflecting real world, respecting natural laws, absence of conflicting logic, no mechanisms undermining the system itself, sound internal structure, ...

6) be able to **cope with events** it will or is likely to meet during its lifetime

7) be better than competing systems or alternatives during its **whole lifetime** (thus being able to evolve).

And ...

• Each of these rules has many implications for the internals (the design).

• A broken rule can shorten the system’s lifespan.
System

A system that is neglected will deteriorate. As a result it won’t be able to perform activities.

Activities

If the system doesn’t fit with its intended usage or if it is not used (no activities are performed), then the system is useless. Some systems may ‘rust’ or die.

Environment

Each system exists in an environment. This environment is as important as the system itself. If the system is not aligned with the environment, if the system does not respect the rules of the environment, if the environment is not respected, then either the environment may kill, boycott or expel the system (like by stopping feeding the system) or the environment itself may deteriorate leading to the death of the system as well.

BALANCE AMONG THESE THREE ELEMENTS IS OF VITAL IMPORTANCE
Minimum Viable System

The **Minimum Viable System** (MVS) is the version of the system with the minimal set of features, qualities and capabilities which is still able to create value by performing a business activity or, for non-business systems, to fulfil a minimal meaningful role.

The MVS is equipped with ALL the **critical capabilities** (**critical features** and **essential qualities and capacity**) and is suitable for a basic usage.

An MVS can not work with partial logic, partial processes or incomplete set of features. Removing one single feature, capability or reduce a characteristic will render the system unable to function, unusable, unable to produce any business value and unable to play its role. It would make the system worthless. The MVS is thus the minimum that has to be delivered. All critical logic and features MUST be implemented and delivered. Anything less can not be operational.

**Examples:**

- A car with an engine but without steering wheel.
- An airplane with take-off and flying capabilities, but which isn’t equipped to land.
- A software system capable of accepting client data but which can’t accept the products data or orders.

The MVS leads to projects with (relatively) fixed-scope !!

**Possible Criteria for Determining an MVS**

- Ability to perform a business function, to perform its role.
- Ability to perform an end-to-end process
- Ability to provide a service or to produce and deliver a product
- Ability to serve the supra-system or external customer
Implementing Characteristics

The system has required capabilities and characteristics. This ripples down on the requirements of the components.

By implementing qualities and capabilities, the components create the behaviour, the capabilities and the capacity of the system.

Besides this model, connexions need to be included in this model.

Systems and companies are/have:
- Connected
- Permeable boundaries
- Influences, interactions and exchanges among systems and between systems and environments and between companies and systems
- Consider that each system has its own environment (sub-environment of the company’s environment or supra-environment). The environment not always / rarely a clear border. It is up to the practitioner to define its limits.
Systems have to be able to cope with events happening during their future life. They have to be engineered for it.
Controlling Systems

- Individuals are allowed to control a system within some boundaries. As long as the steering individual stays within these boundaries, he remains in control. If the individual crosses a boundary, (s)he some or all loses control.

- Control is a continuous dimension.

- A system has many boundaries (limits, thresholds, breaking points, ...).

- A boundary is not necessarily a precise line. It can be vague and broad.

- Crossing several boundaries may enlarge the effect. Crossing a boundary has not necessarily a linear effect on the system.

- Crossing the boundaries does not necessarily mean a loss of all control.

- These boundaries are not “fixed”. They are determined by the environment and by the system (thus defined by the designers and builders).

- Systems have to be respected. It is unwise for an individual to do whatever it wants with a system, unless it seeks to create havoc.

Notes:
Examples: human body, companies, information systems, climate system, financial systems, ..
System Failure

Weakest Link

Single Point of Failure

**Definition:** A single point of failure (SPOF) is a part of a system that, if it fails, will stop the entire system from working. (Wikipedia)

Applicable to any component, including users, interfaces, connections, services, external devices or tools, ... Components, tools and systems in other systems and organisations.
Domino Effect

Single System Failure

Multiple Connected Systems Failure (or SoS Failure)

A company is a set of inter-connected systems.
These systems might be connected to external systems in other organisations.

How to conceive systems resistant to the domino effect?

Note:
Domino-effect is commonly considered as repercussions on other systems leading to their collapse. This is an extreme form. One problematic issue on a system may influence the functioning of other systems. Or, they may contaminate them. This influence is not necessarily direct, immediate and obvious. Sometimes, it may even be immaterial, like a loss of reputation and trust.
Mechanisms

Stable and Unstable Systems - Tipping Point

Positive Feedback Loop – Reinforcing Loop

Negative Feedback Loop - Balancing Feedback

Positive and negative effects can be mixed in same dynamic model. The ensemble of these relations, their strength and the reaction of the components provides the final result.
Mechanisms

Dead Lock

Waiting for B to continue and to send something to B

A

Waiting for A to continue and to send something to A

B

Critical Mass

Critical mass is the minimal amount of something required to bring about an unstoppable dynamic, one that is hard to stop or one that creates an acceleration
Nature obeys only to its own laws.

A natural system always acts and reacts

- accordingly to the natural laws
- and accordingly to its own nature, its logic, its mechanisms, its capabilities, its characteristics, ...

Breaking natural laws and disrespecting natural systems beyond the point of resilience, inevitably leads to if not consequences, at least to increased risks.

Nature is an excellent, but intransigent, teacher. It lets you fail until you learn and respect its laws.

Systems kill us if these laws are violated
Equilibria in Systems (1)

**SYSTEM**

- Structure
- Behaviour, Dynamics, Mechanisms
- Weight, Importance
- Forces, Attractions, Tendencies, Inclinations, Gravity, Centrifugal and Centripetal Forces, ...
- Cause → Consequence

**Fundamental Rules**

- First Principles
- Natural Laws

The respect of these principles, laws and rules is critical for achieving equilibrium

Where do these principles, laws, rules come from?

1) Find the elements that makes up the future solution and its environment
2) Identify the disciplines related to these elements
3) Or study the nature of the elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Laws / Discipline</th>
</tr>
</thead>
<tbody>
<tr>
<td>People</td>
<td>Psychology, organisational behaviour, ...</td>
</tr>
<tr>
<td>Computers</td>
<td>Computer science</td>
</tr>
<tr>
<td>Money</td>
<td>Financial management, Economics</td>
</tr>
<tr>
<td>Electricity</td>
<td>Science of Electricity</td>
</tr>
<tr>
<td>Information</td>
<td>Information Science</td>
</tr>
<tr>
<td>Metal parts</td>
<td>Materials science</td>
</tr>
<tr>
<td>Water</td>
<td>Fluid Mechanics, Hydrology, ...</td>
</tr>
<tr>
<td></td>
<td>+ Systemology + Engineering + Problem solving + ...</td>
</tr>
</tbody>
</table>

Notes:
To be further investigated
Dynamic Systems, and their components, have a kind of balances. There are parameters of which the value must be within a specific range to achieve a state of equilibrium. These balances belong to and are formed by structures, behaviours, forces, weights and cause-consequences. Since these balances are part of the same system, they might be connected and influence each other.

**THIS IS ABOUT THE WELL-BEING OF THE SYSTEM**

A value may change, but as long as it remains in the acceptable range, the state of equilibrium is achieved. Some values must stay above or below a certain limit. Or, values have to exceed a certain threshold. There are trigger points or a critical mass setting the system, or a part of it, in motion.

Example:
- Speed must be kept in a range. It may not exceed a limit. Sometimes it must be zero.
- An energy supply must be kept within a range. And it must be continuous.
- The mass (weight, or volume) has to be spread across the system.
- A bearing structure has to be able to support a certain weight.
- A bearing structure has to be stable and to mitigate bumps, shakings and other lateral forces.

A disequilibrium may hinder a system from functioning properly, or even causing its collapse. A system can also become unstable or go in an uncontrolled or abnormal state of hyperactivity. It may react in many unforeseen ways.

---

Many states of equilibrium and states of well-being of the system have to be identified, understood, managed and the necessary protections need to be built in. It is the task of the engineer to detect them in existing systems as well as in the systems being designed. This is not an easy task for engineers and, a fortiori, for end-users (in existing systems).
A system

- responds to this input, usage, force, influence, ...
- accordingly to its own
  - Nature
  - Characteristics
  - Rules, Laws, Principles
  - Mechanism, dynamics
  - Capabilities
  - Limits
  - Constraints
  - ...

Notes:
- A system receives a trigger, a signal, a kind of input, forces an influence or its operational environment changes and it reacts to it accordingly to its own nature and to its inherent logic regardless of what people want.

  Example: A car behaves well and follows the orders of the driver as long as crucial parameters remain within a certain range. If one or more exceed a limit, the car will act in a way not meant and not desired by the driver.

- The person (or the people) has to create the conditions, provide the right input and activate the right behaviour of the system. This will produce the desired output or result.

- Using or adapting a system that is only partially understood is very risky. One never know what new problems it introduces and how the system will react.
Pieces and the Whole

The whole is greater than the pieces.

Why?

- Having a common goal
- The implementation of an overall concept
- Implementing the system’s capabilities and characteristics
- Forming the architecture (structure and order)
- Interdependency among the parts
- Alignment among the parts
- Interaction: communication, exchanges, transfers among the parts
- Collaboration and support among the parts
- Components may share same elements
- Shared environment (influencing the environment, reacting to same changes in the environment)
- Sometimes built on same foundation or container
- Sharing the same infrastructure
- Common principles, policies, rules, standards …
- A limit or risk of one component is a limit or risk for the whole
- Sharing resources

Implications:

Building pieces and assembling them (bottom-up) without overall design, is likely to lead to a lower quality of the whole.

Over time, by repeating this many times, it leads to a chaotic result.

Important for integration
Gradually the system will decline. Erosion and other similar tendencies may undermine the system. Probably, this will first affects its performance and its results by a decline. At a certain point, the cost/benefit (effort/result) ratio may become uninteresting. But an undermined system may also block or collapse. The decline may be sudden or gradually and predictable or unpredictable.

Factors like: erosion, misuse, overuse, abuse, technical debt, complexity, bad habits, wear and tear, lowering quality, increased entropy, ...

Volumes, result, benefits

Threshold: system reached a too bad quality

Time to reengineer, to simplify, to remove the ballast, to refocus, ...

Note:
The lines do not represent the course of real evolution. They are only drawn for the sake of clarity, to explain the concept.
Destructive Effects in Systems

Other effects:
- increased pressure, wear and tear, frustration, demotivation, learning the wrong lessons, irresponsibility, lower values and norms, various damages, wrong drivers (motivators, reasons for doing something), uncertainty, confusion, increased sensitivity, negative forces (influences, ...), aging, degradation of matter, SPOF, domino, destroying, slowing down, speeding up, erratic behaviour, unreliable, wear and tear, ...

Note:
- Some issues occur in software systems, other in physical systems and other in human systems.
- Not all of the issues (applicable to software systems) can be detected through the user interface.
- Idea: Make a list of all possible types of problems that may deteriorate a system. Then indicate on what type of systems/matter it may occur (natural system, human system, industrial systems, business system, software system, ...)
- Idea: Identify who can identify them and how to identify them (required competencies).
Decay of Systems

TENDENCY TO ENTROPY

OVER TIME

SYSTEMS

TEND TO DETERIORIATE GRADUALLY, HEADING TOWARDS THEIR COLLAPSE.

MAJOR AREAS

• System’s Functions
• Internal Organisation
• Usage / Input
• Resources

ISSUES

• Awkward to use
• Improvisation
• Misuse, abuse
• Too many
• Illogically organised
• Vague
• Unreliable
• Missing
• Duplication
• Different forms and shapes
• Incoherent, inconsistent
• Conflicting
• Confusing, unclear
• Overly complex
• Ballast
• Unsuitable
• Not standardised
• Chaotic
Decay of Systems

**TENDENCY TO ENTROPY**

**OVER TIME SYSTEMS TEND TO DETERIORATE GRADUALLY, HEADING TOWARDS ITS COLLAPSE.**

How to prevent this?

- Guiding the Usage
- Clarify Norms
- Barriers against Abuses
- Limiting the Freedom
- Appropriate Control Mechanisms
- Appropriate Monitoring Mechanisms
- Never lose control or give control away
- Foresee corrective mechanisms

**Notes:**
The more something is critical
- the more crucial it is to understand it fully
- to put the best people on it with the best competencies
- To acquire and strengthen the required competencies

Controlling and Managing a system requires advanced skills.
Intensifying Complexity

Imperfect System
- Doable with a “small brain”

Improved System
- More complex, still imperfect
  - requires a “larger brain”

Improved System
- Even more complex, but still imperfect
  - requires even an “larger brain”

Improved System
- Again more complex, but still imperfect
  - and so on

If a bigger brain is required, it may imply a maturation step (change of perspective, improved belief system, paradigm shift, ...)

How to combat this tendency?
- Don’t repetitively add layers, cases, exceptions, rules, criteria, differentiations, ... → this adds complexity
- Get back to the fundamentals, first principles
- Clean-up, re-engineer, re-simplify

Neglecting this will increase the effect of the tendency of complexification
Understanding Systems

A more profound understanding of systems is required to

• conceive systems
• use systems
• manage systems
• deal with systems in general

Notes:
A lack of understanding in how a system works, its constraints, prerequisites, right handlings, limits, risks and anything that may harm the system may damage the system and even destroy it.

Modifying a system, ignoring or not understanding the effects on other systems downstream, may create problems in these systems (domino effect).

In software development, this risk is increased if information is shared (several systems accessing the same databases).
Sustainability: Remain under the carrying capacity of the system.

Practical: Interesting to take decisions about capacity and scalability or for the implementation of mechanisms like buffers, triggers, alarms and controls.
Managing Systems

- **Engineer**
  - Understands and has access to the system’s internals

- **a person, a group, people**
  - Limited understanding
    - Understands the system through its inputs and outputs
    - Understands the usage
    - Some internal functioning can be understood or this insight can be assumed

Real insight in the system’s internals and having access to them, makes it possible to manage the system. It greatly reduces the risks.

Managing the usage, input, output, but also the environment can be achieved with black box insight. Managing may be very risky depending of the system.

We can only manage what we understand.
Some key notes about Systems for Engineers

1. Man-made systems are engineered or can be engineered (majorly, not necessarily for 100%).

2. Characteristics are engineered into man-made systems. They do not always materialise as intended and some appear spontaneously. Some characteristics are necessary or desired, while others are not. It is desirable that the creation of capabilities and characteristics happened in a methodical, intentional and controlled way. This belongs to the essence of engineering. Systems Engineers seek to determine what characteristics and capabilities are required and to engineer them into the system.

3. Systems have capabilities, capacity, prerequisites, minimal and ideal operating conditions, constraints, tipping points, breaking points, limits, weaknesses, bottle-necks, deadlocks, SPOF’s, …

4. A system reacts to input, to forces and to changes in the environment in which it operates.

5. A combination of specific factors or exceptional circumstances may trigger a complete change in the system’s behaviour. The change of behaviour can be (partially) unexpected due to lack of insight in the system, the circumstances and its environment.

6. A system reacts and behaves always accordingly to its own logic and characteristics. It follows always its own program. Piloting (or using) a system is in fact providing an input to the system. This input triggers a reaction and a behaviour of the system which will create an output or outcome. The pilot (user) loses control when the system exhibits a behaviour that was not intended or when the output is unexpected. Unsuitable circumstances in the environment or inside in the system or an incorrect input won’t lead to the intended result.

7. Misuses of systems is more important when the user doesn’t know how to use the system or when the freedom of the user is not under control.

8. When a system’s purpose is changed by design, or when a system is used for another purpose than the one intended by its design; the system is usually either lesser efficient and effective. This usage may even weaken or deteriorate the system.

9. A system may implement principles, concepts and mechanisms which may undermine itself from within. It may exhibit self-destructive behaviours and tendencies.

10. Flexibility and freedom increase the complexity and risks of misuse and abuse. Freedom should go together with responsibility, competencies and wisdom.

11. A system is equipped with internal controls to protect itself from misuse, abuses, other external influences, inappropriate circumstances and input.

12. Having as goal meeting the criteria of “a working system” leads to mediocre systems. This represents only a single perspective. Meeting more and higher norms is imperative.
13. Some consequences of a problem are visible. Some are undesirable effects that are experienced. The shortest and cheapest way to eliminate these annoyances is to mask or solve them, or sometimes by dealing with their direct causes. This is why consequences, symptoms, are often solved and not the real problem. We label them too easily as cause (cause of annoyance). We neglect to look to their cause, and then to consider this cause as a consequence and to look further upstream until we find the root cause.

14. Solving consequences allows the underlying problems to continue to exist.

15. Solving symptoms makes systems more complex. We don’t solve problems, we add solutions, layers, exceptions and by doing so we create chaos.

16. A system forms a whole. Changing something may impact the whole system or may impact it on a different place, area or aspect as the one where the change occurred. This is why it is advisable, and sometimes even imperative, to understand the whole system before changing it or seeking to change the system through influencing it.

17. In mechanical systems the interactions are much easier to understand. Therefore, the causes and consequences are easier to detect. In more abstract or complex systems; like organisations, software systems or natural systems; the logic and the interactions are much lesser obvious.

18. Understanding the environment and the interactions, tendencies, forces, opportunities, threats, events and other influences between the system and its environment is essential to engineer suitable systems.

19. It is wise to get first the foundation, the carrier structure and core part, right when engineering a system.

20. An approach based on systematically accumulating parts and changing logic is more prone to result in a overly complex, disorganised, confusing, incoherent, ineffective, inefficient system. A system is more than the sum of some elements. Structure, internal organisation, order, coherence, interactions, alignments of parts are key to systems.

With man-made systems

“Good” is “NOT Good Enough”
2 PERSPECTIVES

COMPANY as SYSTEM
COMPANY as ENVIRONMENT

Looking Outwards
Looking Inwards
Why discussing the company in Information Systems Engineering?

- The company forms the environment of the Information System. The IS has to fit into the company and must be able to operate in it.
- Information Systems are parts of the company. They have to be aligned with the company, its business model, its strategy, its culture, operating model, etc.
- All information systems have to contribute to the company’s objectives. The company is their supra-system.
- Information systems influence the company. They implement and influence its functions, capabilities and characteristics.
- Company and its environment influence aspects of the IS design and its functioning. They both play a critical role in the determination of objectives, role, purpose, functionalities, capabilities, characteristics, behaviour and constraints of the IS. They also determine or influence the logic and architecture.
- Information systems and information may drive the business and are a source of innovation.

The conception of the IS must respect the nature of the company.

The IS Engineer must understand what companies are, as well as the specific company and its environment, and take them into account.

Notes:

- If the company requires to be flexible, can the IS be rigid?
- If the company requires to be manageable, can an IS be an unmanageable black box?
- If a company is a guided system, can ISE then be a directionless process?
What is a Company?

Company’s Purpose

• Providing value to the society
  • Society: people, living beings, environment
  • How? Through its products and services
• Making society to progress

But, it is also

• a formalised entity consisting of people working together to a common goal
• a way for people to contribute to society and to make a living
• a way for personal growth

Achieving the Company’s Purpose
What is a Company

See also the few pages in section

“FOUNDATION – Core Ideas”
What is a Company?

5 Perspectives

1) A **Value Creator**
Solving needs of the society

2) **Implementation of a Vision**
An entity with a purpose, role, function in this vision

3) **A System**
(the “enterprise”)

4) **An economic entity**
in a dynamic, limited, constraining and competitive environment with scarce resources. An entity **seeking to survive and to thrive** in this environment.

5) **An Investment** of means

**Complex, man-made, dynamic, open, heterogeneous, managed, guided, economical, socio-technical system**

- A perspective is only a limited view on a company.
- The company’s (sub-)systems have to comply and to be aligned with the whole ensemble of these perspectives.
- Each perspective brings its own understanding, priorities and requirements. None of these perspectives can be ignored.
- Each has its own far reaching implications in aspects like the development approach, priorities, design or even organisational and collaborative aspects.

The Analyst has (to understand and) to reconcile these perspectives.
5 Perspectives

Implementation of a Vision

Some companies are truly implemented to realise a vision. Others simply evolved without. They simply deployed activities, took decision and seized opportunities which appeared as the most beneficial at that moment. A vision provides guidance and focus. It helps to concentrate the activities on some type of work, on products and services, on needs and/or markets. The entire company has to build on a coherent whole and direct its effort to a common goal. This is mainly a top-down guidance. It avoids inefficiency due to development of unrelated activities in different directions.

Value Creator

A company that doesn’t create value for the society is useless. It’s a parasite. Value is created through activities. Competition forces the company to be effective and efficient. The company has to stay ahead by having better products and services. Customer relation and innovation are crucial. Innovation requires a propitious environment and advanced skills.

An Investment

An investment is the allocation of money in the expectation of creating or getting a benefit. Typically, the investor takes the decision to invest based on ideas, plans or studies demonstrating the likelihood of a creation a benefit. The investor seeks to know more about the initiative in advance. These plans have to be implemented and executed.

Economic Entity

An economic entity is about value creation through the production of goods and services in an effective and efficient way. It seeks to maximise its return on investment. Money is a critical mean for a company to survive, to function and to adapt.

System

A company is a system. Some of its parts function in an undefined (free, ad hoc, erratic, variable, unpredictable) way. These parts don’t suit the concept of ‘system’. The system-like part of the company must be treated as a system. The company has to evolve, to renew itself. It has to engineer itself. Systems engineering (engineering of systems) is a key discipline.
What does the society need? How does the society evolve? How do we envision a future society?

What role will our company play in the creation or evolution of this society and by respecting what values?

What main products (or type of products) and (kind of) services will our company produce that will contribute or change the society in line with our vision, mission and values?

What high-level system has to be created in order to produce these products and services and this while being inline with the vision, mission and values?

Notes
A company operates in an environment. This environment provides opportunities and threats. The company’s internals are influenced by this broader environment. The company has to fit in its environment. Understand the environment and take it into account when conceiving solutions.

See also
- Pesteli-Framework
- Michael Porter’s Five Forces Model
Company’s Environment

- Society
- Community
- Country organisation
- Governments
- Political system
- Politics
- Law makers
- Legal bodies
- Economical system
- Healthcare system
- Financial system
- Recreation
- Ecology
- Technology
- Natural environment
- Climate
- Partners
- Suppliers
- Competitors
- Suppliers and partners of competitors
- Clients
- Prospects
- Work force
- Trade unions
- Academic Institutions
- Research centres
- Media
- Transport infrastructure
- Communication infrastructure
- Energy infrastructure – Energy supply
- Water infrastructure
- Service providers
- Raw materials
- Resources
- Waste disposal infrastructure
- Social Security
Company’s Supra-System

VALUE NETWORK

Notes:

• Represents
  • What party creates what value
  • Exchanges: Flows set up between different parties involved in the creation of value (final product) and for the deployment of business activities.

• Flows can be enriched with descriptions and other data (value, capacity, delays, ...)

• Suppliers, transporters and distributors are part of this system.

• Model can be enriched with plenty of information: amounts, problems, critical/optional flow indicator, positive/negative exchange, and so on... Own symbols can be used.

• We don’t just set up a company. We create a much broader system beyond the company’s borders with clients, suppliers, partners, ... An analyst has to look even beyond the company’s borders.

• Example: Supply Chains, Distribution Networks
The Business Model describes the global business idea. It answers the question: “How do you (plan to) make money?”

It provides an overall framework for developing a coherent and streamlined and focussed set of activities to create value.

More detailed questions:

- What value is created?
- Who are your target customers?
- What customer’s problems or needs will be solved?
- What are the products and/or services?
- Who are the customers ready to pay for them?
- How do you reach, acquire and keep customers?
- How do you differ from the competitors?
- How do you produce the products and services (from a business perspective, not from a technical perspective)? And how do you get them to the clients?
- What is the turnover?
- What is the cost structure?
- What is the profit margin?
- And so on..

Note:
The “Business Model Canvas” is a template for developing a new or documenting an existing business model.
Business Model ≠ UML model
Business Modelling ≠ Business Process Modelling (!!!)
A business model is a response to an opportunity and has to suit the company, be appropriate for the product and market and fit into the company’s environment.
## Business Model Canvas

<table>
<thead>
<tr>
<th>Key Partners</th>
<th>Key Activities</th>
<th>Value Proposition</th>
<th>Customer Relationship</th>
<th>Customer Segments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Key Resources</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channels</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cost Structure</th>
<th>Revenue Streams</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
LIST OF TYPES OF BUSINESS MODELS

<table>
<thead>
<tr>
<th>Add-ons/In-app purchases</th>
<th>Low-Cost model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advertisement (Ad-based) model</td>
<td>Make More Of It</td>
</tr>
<tr>
<td>Affiliate/Referral fee</td>
<td>Manufacturing model</td>
</tr>
<tr>
<td>Aikido</td>
<td>Mass customisation</td>
</tr>
<tr>
<td>Auction model</td>
<td>Merchant model</td>
</tr>
<tr>
<td>Bait and Hook model</td>
<td>No frills model (discount or budget model)</td>
</tr>
<tr>
<td>Barter or Swapping for products</td>
<td>Open Business model</td>
</tr>
<tr>
<td>Barter or Swapping for services</td>
<td>Open Source model</td>
</tr>
<tr>
<td>Brokerage model</td>
<td>Orchestrator</td>
</tr>
<tr>
<td>Cash Machine</td>
<td>Pay-as-you-go model (PAYG)</td>
</tr>
<tr>
<td>Commission-based model</td>
<td>Pay-per-use model</td>
</tr>
<tr>
<td>Cross Selling</td>
<td>Pay-what-you-want (PWYW)</td>
</tr>
<tr>
<td>Crowdfunding</td>
<td>Peer-to-Peer</td>
</tr>
<tr>
<td>Crowdsourcing</td>
<td>Performance Based Contracting</td>
</tr>
<tr>
<td>Customer Loyalty</td>
<td>Premium model</td>
</tr>
<tr>
<td>Data model</td>
<td>Razor and Blade model</td>
</tr>
<tr>
<td>Digitalisation</td>
<td>Recurring Revenue model (subscription model)</td>
</tr>
<tr>
<td>Direct Sales model / Direct Selling</td>
<td>Rent Instead of Buy model</td>
</tr>
<tr>
<td>E-Commerce</td>
<td>Revenue Sharing</td>
</tr>
<tr>
<td>Early Exit strategy</td>
<td>Reverse Engineering</td>
</tr>
<tr>
<td>Experience Selling</td>
<td>Reverse Innovation</td>
</tr>
<tr>
<td>Flat-rate</td>
<td>Robin Hood</td>
</tr>
<tr>
<td>Fractionalised Ownership</td>
<td>Self-Service</td>
</tr>
<tr>
<td>Franchise model</td>
<td>Shop-in-Shop</td>
</tr>
<tr>
<td>Freemium model</td>
<td>Single purchase model</td>
</tr>
<tr>
<td>From-Push-to-Pull</td>
<td>Solution Provider</td>
</tr>
<tr>
<td>Get-one-give-one model (G1G1)</td>
<td>Subscription model</td>
</tr>
<tr>
<td>Guaranteed Availability</td>
<td>Supermarket</td>
</tr>
<tr>
<td>Hidden Revenue</td>
<td>Target the Poor</td>
</tr>
<tr>
<td>Ingredient Branding</td>
<td>Tip jar/donation</td>
</tr>
<tr>
<td>Integrator</td>
<td>Trash-to-Can</td>
</tr>
<tr>
<td>Internet Bubble model</td>
<td>Two-Sided Market</td>
</tr>
<tr>
<td>Layer Player</td>
<td>Ultimate Luxury</td>
</tr>
<tr>
<td>Leverage Customer Data</td>
<td>User Designed</td>
</tr>
<tr>
<td>License fees</td>
<td>White Label</td>
</tr>
<tr>
<td>Lock-In</td>
<td></td>
</tr>
<tr>
<td>Long Tail</td>
<td></td>
</tr>
</tbody>
</table>

**Note:**

Not an exhaustive list

These business models are like templates. The chosen model is a starting point and has to be adapted and further developed to suit the company’s specific situation.
A **STRATEGY** is a long term plan, 5 years of longer, for a company (organisation) to realise a vision or to achieve a goal.

A strategy is (often) meant to change the position of the company in the market.

A **STRATEGY** is

- a PLANNED PROCESS of TRANSFORMATION of the company
  - the Business Model
  - the company’s Value Network and partnerships (partners, collaboration, suppliers, sources, ...)
  - its activities
  - the company’s location
  - the company’s capabilities
  - the company as system
  - its products or services
  - its policies, rules, principles, ...

A strategy **“predicts” the internal changes.**

To **increase the chance of a successful execution** of a strategy

- The strategy has to be communicate to all the company’s employees
- The enterprise (the system) has to be changed in time
- Information is required to understand the company and its activities and to understand how the strategy works out.

**Notes:**
This is the main idea of a strategy. A strategy may also be related to priorities, timing, policies, resources, leadership, company culture, competencies, internal organisation, and so on.

The timespan of a strategy concerns, by nature, years (like 3 to 10 years).

“strategic” is often confused with terms like ‘important’, ‘critical’, ‘key’, ‘fundamental’, ...
Basic Business Models

SIPOC – Model

**COMPANY**

Supplier → **Process (Activities)** → Customer

*Limited model: Linear model. Not holistic.*

Cyclic Model

Recycling Company → **Company** → Client

**Company**

products → **Client**

Old products (for recycling)

waste

*Remark: This is an example. Different cyclic models are possible.*
Basic Business Models

Essential Cycle

1. Society → Investment → Production System → Products/Services → Clients
   - Money

2. Supplier → Investment → COMPANY → Client
   - Products/services/work
   - Money
   - Work
   - Salary
   - Employees

3. Suppliers → Process/Activities → Product/Services
   - System
   - Process/Activities
   - Money

4. World → Suppliers → System → Final Client
   - Matter
   - Energy
   - Information
   - Knowledge
   - Workforce
   - Money

5. Society → Labour Market → Suppliers → System
   - Money
Putting some things together

VISION = GOAL

Entity contributing to the society; “FOR PURPOSE”-organisation; meaningfulness

INVESTMENT

ENVIRONMENT

COMPANY

ORGANISATION
SYSTEM of SYSTEMS

ACTIVITIES
performed by the system

ACTIVITIES
deployed using the system

Products & Services

to deliver

for

CLIENTS

Notes:
Everything has or should have a purpose and a goal (target, intention, reason, ...). Knowing it is important to understand what and why we do things, why they are necessary (or not), ...
Company’s Life Cycle

A more optimistic model ...

Notes:
Model may represent the maturity of a set of competing products on a market. Cost of development, sales, sales revenue, profit, ...may be added to the graphic.

Why do companies end? To avoid at all cost. Sub-systems have to help to prevent this. Simple linear model. Why? Can it be different? Why?
Company’s product portfolio

Impacts projects and systems.

A company strives to have always one or a set of growing to mature products to ensure permanent stream of income.
Product: BCG - Matrix

Notes:
- Question marks: Uncertain products, services, activities, opportunities
- Rising stars: Selected good opportunities
- Cash cow: provide business volume, benefits, ...
- Underdog: old products and services in decline. Revival (innovation) or disinvest?

The BCG-matrix must be considered in the light of the strategy and business priorities.
A company needs to have cash cows and rising starts that, later, will become cash cows. Therefore it needs to invent new products that will evolve into new rising stars.
BUSINESS KEY DRIVERS

Notes:
Identify the few key drivers generating most of the business results.
Identify their main enablers allowing the drivers to function
Understand the driving forces (motivation, reasons, intentions, ...) driving the drivers?
Results are expressed as measurements of the KPI’s (there are also PI’s (Performance Indicators))
Know the strengths and weaknesses (+ risks) of the drivers and enablers.
The key drivers must be considered in the light of the BCG-matrix, the strategy and business priorities

Still need to understand the environment and the relation between this core aspect of the business and the environment.
Company’s Evolution Drivers

<table>
<thead>
<tr>
<th>External drivers for change</th>
<th>Top-Down (mainly)</th>
<th>Bottom-Up (mainly)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Markets</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Legal</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Clients</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Competitors</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Internal drivers for change</th>
<th>Top-Down (mainly)</th>
<th>Bottom-Up (mainly)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy, tactics, plans, priorities</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>New activities</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Improvements</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Innovation</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

Analysis like SWOT, PESTELI (or PESTLE), Michael Porter’s Five Forces Model and others may identify external changes.

Implication: The question is not whether a top-down or bottom-up should be followed (dualistic thinking). Main directions are mainly coming from the top. Top-down is clearly the major approach. However, bottom-up approach should be use as well and should be integrated in the approach.
Events in the Company’s Life

Events the company may have to deal with during its lifetime (not exhaustive):

- Adapting the company culture
- Changing the business model
- Changes in strategies, policies, tactics or priorities
- Organic growth
- Maturing organisation (optimisation, formalisation, ...)
- Restructuring, rightsizing, downsizing
- Acquisition
- Merging
- Relocation of activities (like offshoring)
- Externalisation of activities (like outsourcing, privatisation, ...)
- Disinvestment
- Diversification
- Consolidation
- Takeover
- Bankruptcy
- Relocation of HQ or plants
- Changes in product portfolio or in service portfolio
- Changes in customer base
- Change of geographical market location
- Changes in investment and sources of revenue and profit
- Changes in distribution and transportation network
- Changes in partnerships and inter-organisational collaborations
- Changes in work philosophy, approach, methods ...
- Technological evolution
- Evolution in automation and computerisation
- Digitization
- ...

Notes:
How well can the architecture and systems cope with such events? Does the implemented system hinder these events? How to prepare for them? How well can the engineers deal with such an event?
Notes

This reasoning can be further detailed.

Since a company is also an investment → goal, guidance, predictability, control.

Control, predictability and planning are required, as well as flexibility and adaptability. A balance must be achieved. Sometimes this will lead to a trade-off. Sometimes these are not conflicting and both can be achieved.
The sub-systems and components have to respect these requirements.

These required qualities have to be engineered in the enterprise. They can be found in the principles, capabilities, structures, concepts, features, mechanisms and characteristics.

They influence the systems design, the engineering philosophy and methods, the development processes, the transformation initiatives, ... but also roles and competencies of people (authority, responsibilities, activities, interpersonal relations & collaboration).

Notes:

- The listed characteristics are critical for the company “as a system” (for the engineers). From a broader perspective, other characteristics like leadership and culture are critical as well.
- These characteristics don’t appear by themselves in the enterprise and its systems. They have to be engineered in it.
- Each characteristic can be further detailed:
  - Sustainable: effective, efficient, legal, ethical, planet-friendly, socially respectful, ...
  - Flexible: elegant architecture, modular design
- None of these characteristics or deduced characteristics can be ignored or neglected, at any level of engineering.
The geographical situation often plays a role in the organisation of the company and in the architecture of processes and systems.
Inside a company a system can be more cohesive, more tight. Outside the company the systems tend to be more tentacular and lesser bound to a location (web applications, Apps, ...).

Inter-organisational information systems, like supply chains, are somewhere in between.

Unfortunately, often the term Interorganisational Systems (IOS) is used. This term is too vague and should specify that the system has to be an information system. A energy distribution network, a transport network between organisations, a close collaboration between two or more companies are all systems between organisations.
The Company – “Intra Muros”

COMPANY
- environments
- systems
- people
- transportation means
- machinery
- sites and plants
- flows
- processes
- networks
- resources

Area outside the company’s wall in which the company deploys its activities. (example: sales man visiting prospects) Environment is not under the company’s control (no ownership)

Interfaces with outside world

Engineered, controlled, managed and guided environment / system (because of ownership)

harmonious and managed system, seamless collaboration
Internal Operations Model

Business Model & Value Network
(value creation and value exchange models)

Value Stream: *level above the Business Processes*

Methods: Value Chain Model, Value Stream Model, Rich Pictures, ...
Business Functions

SIPOC

Suppliers → PROCESS / SYSTEM / COMPANY → Clients

Value Chain Model / Value Stream Model

Business Functions Model

Model organising the company’s main functions

Company’s Main Functions

Knowledge Area  Role

Competencies  Responsibility

Organisational Units

Business Functions ≠ Organisational Units
Notes:

- Organisation of activities.
- Organised per business knowledge area and main expertise, per purpose, per goal, per responsibility and authority,
- Types (production, selling, support; internal, external) can be indicated.
- Can be enriched with main relations or flows among business functions.
- Business functions often coincide with divisions and/or departments. However, business functions ≠ department or other organisational units.
A **Business Capability** is the organisation’s ability to perform a unique function or unit of business activity. This ability is the result of the intentional implementation of a defined set of necessary means.

In **management**: allows to manage functional strengths.

In **enterprise architecture**: a building block to define the business architecture and to translate it into the enterprise architecture.

**Business Function - Business Capability**

Business Capabilities can be grouped under a Business Function. A Business Capability is a sub-function of a Business Function. A Business Function is rather an abstract grouping used in models. A Business Capability is, when implemented, more concrete since it links between means and ability.
Related:

- Business Capability Map
- Capability Management
As example, an approach can be deduced:

- Definition of goal structure (for example, deduced from the strategy)
- Definition of the required business capabilities necessary to reach the goals
- Define the hierarchy of capabilities (key indicators and targets)
- Link the business capabilities to the business functions. Are they present in the business functions? To what is their strength? Is there a gap?
- Determine the required/involved processes
- Determine whether the processes are aligned with the required capability.
- Determine the required/involved systems.
- Determine whether each system is aligned / implement its part of the capability.
- Same for organisation (people), competencies and resources.

Notes:
- The process can be adapted for listing the available capabilities, for verifying if a plan or forecast is feasible, to identify a gap, weaknesses, ...
- This model can be enriched with risks, limitations, links to on-going projects, ...
BUSINESS CAPABILITIES

Core Capabilities
Essential capabilities required to fulfil the company’s purpose. These capabilities should be strong, solidly embedded and optimised. They should be better than those of competitors. Innovating here is certainly useful.

Strategic Capabilities
Critical capabilities especially developed to enabling the execution the strategy and which are developed. They are developed to execute the strategy. Innovation in these capabilities is very important. Over time, they may become a core capabilities.

Common (Business) Capabilities
These are usual capabilities.

Supporting Capabilities
These capabilities provide support to all the three other capabilities. Examples are HR and warehouse management capabilities.

ENGINEERING CAPABILITIES
To be developed: like Engineering Capabilities (R&D, Industrial Engineering, Product Engineering, Service Engineering, ISE, ...), Common Capabilities, Supporting Capabilities
Activities

Managing
  • Guiding
  • Planning
  • Organising
  • Controlling

Executing
  • Counselling
  • Coaching
  • Negotiating
  • Learning
  • Researching
  • Communicating
  • Social processes
  • Conducting
  • ….

Intra-muros
  • both
  • Extra-muros

Pre-production
  • Production
  • Post-production

Core processes
  • Supporting processes

Critical
  • Other degrees of criticality

Industrial process
  • People process
  • Computer process

Intellectual process
  • Manual process
  • Mixed process

Predictable,
Structured,
Repetitive

Partially predictable,
Partially structured,
Repetitive to some degree
(cases)

Spontaneous,
Ad hoc,
Unstructured

Main discipline applied in a process
  • Business discipline (ex. insurance)
  • Other discipline(s) (ex. statistics)
  • No specific discipline
Some information processes may support other processes. And, all process types contain some information processes.

Business Process vs Information Process

Business process structures the process. It may process matters, money, information, .. It may include transport or social processes. Business processes implement always some information logic.

Processes only information. Information processes implement business logic.

Not all processes executed by the business community are Business Processes. A Business Process can be decomposed into more detailed processes. These more detailed processes are not Business Processes. Some other detailed processes exist which aren’t Business Processes either. We can’t simply label every software feature or
Business Departments, Domains and Business Processes

Business Processes & Organisational Units

One business process may implement logic of different business domains

Business Processes & Organisational Units

Each organisational unit has its own goals, priorities, plans, needs, intentions, rules, ...

Notes:
Business processes are (often) a collaboration between business domains and between organisational units (like departments, services, ...). Analysing or designing a business process may require consulting different experts and taking into account the variety of goals, contexts, priorities, ...
Processes

Processes inside the company:

- Knowledge (information)
- Money (mostly information)
- Matter
- Energy
- Information

Processed resources:

- Matter
- Information

Executors:

- Machines
- People
- Computer / software systems
1. People are able to deal with higher level logic. They have the capability to interpret, to take ad hoc decisions and to adapt. They can deal with high-level more vague instructions, with non-formalised logic, variable process execution and with intuitive knowledge. Not everything is or need to be pre-defined, formalised, ...

2. The Industrial and Information component contain pre-defined, hardwired (even if configurable) logic. Faster and more reliable process execution.

People organisation and automated components are very different.
May lead to misunderstandings, conflicts, ...

Notes
These are the three most common main “executing” components in a company.
We can also have an “energy component”, “transport component” (doesn’t matter).
Automation is done either by moving work to the industrial component, or by moving work to the information component (reduction of the people organisation?)
People can learn. Able to acquire new skills.
People can deal with fuzzy situations.
The mind activities and products of the mind are partial, distorted, subjective and fuzzy. These are lines of thoughts, ideas, visions, beliefs, motivations, intentions and expectations.
Difference between the talk, thinking and actions.
Intellectual activities are not linear. Their pace is highly variable.
People are creative.
People can inform themselves, investigate, research, ask, interpret, evaluate, take decisions and adapt their actions accordingly.
People’s mind is variable. The understanding changes over time. Ideas may pop up. Preferences may change. Different decisions can be taken.
People tend to create a set of beliefs and an understanding that makes sense to them and which is coherent.
People have social relations which are dynamic, temporary and of different intensity and quality. This impacts collaboration.
People influence each other for the best or for the worst.
People tend to copy each other.
People like simplicity and dislike complexity and problems.
People are bad problem solvers.
A person can get tired or sick.
Many factors influence the speed, efficiency and quality of activities of people.
People need to relax, take a break and have some rest, days off and vacation.
People are different, have different qualities and require different work environments.
People may use freedom well or abuse of it. With freedom comes responsibility.
People react on fear, uncertainty, lack of information, threats, and so on.
And so on... !!

The strengths, weaknesses, limitations and other characteristics of people have to be taken into account when designing IS’s, but also when working in projects or elaborating methodologies.
SOFTWARE SYSTEMS

• Designed to deal with information
• Implements information capabilities
• Implements information processes
• Solves (or help solving) **information problems and information needs**
• Man-made
• Have purpose
• Process information **incredibly faster** and way more precise than people and can deal with huge amounts of information
• Information is stored more reliably (lesser volatile than people’s memory)
• Work defined by predetermined logic (with AI, things can change)
• Fully formalised systems
• User interface is only a very limited and superficial view on the software
• Expresses high-level logic as well as extremely detailed logic
• (Pure) Business logic is only a part of the implemented logic
• Weaknesses: interpretation, improvisation, ad hoc decision making and thus ad hoc adaptation (because of prior detailed definition of the logic (slow and tedious))

MACHINES

• Man-made
• Have specific purpose
• Transform or transport materials, usually faster, more precise and at lower cost than people
• May reduces risks for people
• Machines produce repetitive work with similar results (good or bad)
• Work is predefined
• May need information; may contain software (robots)
The company consists mainly of people, processes, industrial systems (not in administrative companies), information systems and information. They are organised in architectures. These architectures need to be aligned. They are inter-dependent.

**People Organisation**

Best known. More on next pages.

**Industrial Component**

Only applicable for industrial companies. Out of scope in this paper.

**Process Organisation**

- Structure of the execution of the processes
- Structure of the logic of the processes

Information processes are executed by people, information systems
Processes dealing with matters are executed by industrial systems

**Information Architecture**

The information in a company is not a collection of databases. It should be organised around a global high-level information model.

**Information and Software Systems Architecture**

Organising information systems and software systems around an architecture is likely to be more effective and more cost-efficient than having simply a collection of systems.
These models show, as example, a common basic idea of dividing a company into organisational units. Other organisational structures and names are possible.
The company can be seen as a pyramid of pyramids. Flat organisations contain lesser layers of pyramids, while larger more hierarchical organisations are based upon more superposing layers of pyramids.

A “pyramid” can be temporary (like for a project initiative).

Organisations executing a lot of project may chose for a matrix organisation.
Notes:
Organisational units often receive the names of business functions. The relation between both is often very strong.

Traditional and common way to represent a company
Represents:
• units, departments, services
• or management hierarchy (names and roles)
• or both together

Reflection:
Is it beneficial for the company as a whole to allow the departments to define themselves how they work? Why or why not?
A Logical Company Organisation

Management

ENGINEERING
- Research and Development
- Product development
- Service Development
- Organisation Development
- Informatics
- ....

BUSINESS COMMUNITY
- Finance
- Marketing
- Sales
- Production
- HR
- ...

Systems builders

Systems users
All organisations are partly ...

- organised like a **machine** (formal structure, procedures, processes, controls, specific skills, cost structure, ...)

- a **human** and **social** aspect

These aspects are present to different degrees depending on their economical activity but also depending of the part of the organisation (product development, customer service department vs industrial production line)

**Notes:**
Engineering IS, or larger parts of the organisations, has to take both perspectives into account. The weight and priorities must match the specific nature of the organisation.
Dynamic Environments

Essence:
Some industries, some parts of a company are rather stable environments. Other industries or departments are very dynamic.

Equipping dynamic environments with a solid Information Component is much more challenging

Remarks:
• Only rough estimations (can be inaccurate, subjective, ...)
• Depends on the department
• Depends of the company culture
• Every company is different
• Know your environment
Process Architecture

Process Architecture (Execution Perspective)

Groups of Business Processes

Business Processes

BP Activities

Special Services, Features, Functions

Process Logic Architecture

Organised in systems, libraries, components, modules, service software agents, ...

Decomposition of the processes as it appears when executed. Function and services are called and executed as needed.

This architecture organises the process logic, like the source code. They are usually grouped by purpose or by information domain to be accessed.

Invoice management processes

Client management processes

Access management processes

Unit conversion processes
Information Systems

Notes

• IS’s share information
• IS’s communicate with each other. IS spheres overlap.
VERTICAL FLOWS

These flows concern mainly budgets, information for guidance and management, like plans, guidelines, policies, authorisations, performance data, results, ideas, … These flows move up and down through the organisation. It often concerns selected, filtered, aggregated, global data, or in the other direction, a refinement occurs as the flows goes downwards.

HORIZONTAL FLOWS

Horizontal flows are often embedded in business processes or concern sharing and collaboration. Information, material, ‘work’ and knowledge tend to flow horizontal. Horizontal flows happen at different levels (in layers). Detail of information across the flow remain the same.

Flows happen not only inside systems, but also between systems. A flow represents a dependency between systems. Flows help to achieve final outcomes. Today, no system can work in isolation anymore.

Flows are critical and should be engineered

We tend to engineer architecture and systems, but we tend to forget the flows.

Electricity, water, gas, computer networks, telecom networks reach to all corners of the company.
Systems

Very important internal global systems

Goal Hierarchy System
System for analysing, decomposing, deducing, aligning and spreading goals across the company

Planning System
Elaboration and alignment of the various strategies, tactics and plans across the company

Communication System – Communication Channels
Established communication channels or processes across the company

Decision Making System
How decisions are being made and spread

Incentive System
Gathering and aligning all the incentive policies or mechanisms across the company

Creativity System
System to capture ideas across the company, evaluate them, and so on.

Other interesting/potential systems: knowledge system, collaboration system, network system (people), investment-profit system, priorities.

Notes:
• The key here is to establish first about the mechanism, the logic of the system. Thinking about software comes later as a way to implement the solution. (‘system’ ≠ ‘software system’)
• Each or most system should include evaluation, control and feedback mechanisms.
• These systems are not independent: goal, decision making, planning and incentive systems.
Three Executors
- have to fit into each other
- are intermingled
- have to collaborate seamlessly to achieve a same goals

AUTOMATION

<table>
<thead>
<tr>
<th>AUTOMATION</th>
<th>Work performed by the company</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prescribed, Structured, Predictable</td>
<td>Semi-structured</td>
</tr>
<tr>
<td>Ad hoc</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Matter</th>
<th>Automation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semi-structured</td>
<td>Partly automated</td>
</tr>
<tr>
<td>Human activity</td>
<td></td>
</tr>
<tr>
<td>Supported with OAS, AI?</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Matter &amp; Information</th>
<th>Robotisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semi-structured</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Information</th>
<th>Computerisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semi-structured</td>
<td>Supported by software systems</td>
</tr>
</tbody>
</table>

Ideally, forming a coherent single set of activities working towards some common goals.
Person ↔ Company (1)

PERSON

Identity

Goal

Body  Activities

• Seeks to remain alive, well and happy;
• Personal development and self-realisation;
• to achieve the full potential and dreams;

COMPANY

Identity

Goal

Company’s body

System  Activities

Seeks to remain in existence; to flourish and to prosper (too often: to be profitable now)
## Person ↔ Company (2)

<table>
<thead>
<tr>
<th>PERSON</th>
<th>COMPANY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name, Identity</td>
<td>Name, identity, brand names</td>
</tr>
<tr>
<td>Personality, personal values, culture, belief system, thinking patterns, mental strength, norms, habits, autopilot, maturity</td>
<td>Company culture, values, leadership, CSR (Corporate Social Responsibility), world view, belief system, way of thinking, habits and common practices, automatic responses, maturity, market position, ...</td>
</tr>
<tr>
<td>Relationships, social network</td>
<td>Relationships, collaborations, partnerships, network</td>
</tr>
<tr>
<td>Reputation</td>
<td>Reputation</td>
</tr>
<tr>
<td>Activities</td>
<td>Business activities, production</td>
</tr>
<tr>
<td>Skills, capabilities, limits</td>
<td>Skills, capabilities, limits</td>
</tr>
<tr>
<td>Body</td>
<td>Company as a system</td>
</tr>
<tr>
<td>Skin, guts</td>
<td>Boundaries: walls, gates, entries, firewalls, ...</td>
</tr>
<tr>
<td>Skeleton</td>
<td>Structural elements, like human organisations, organisation in business units, business functions, ...</td>
</tr>
<tr>
<td>Brain, guts</td>
<td>Management, guidance of operations</td>
</tr>
<tr>
<td>Nervous system</td>
<td>Information network</td>
</tr>
<tr>
<td>Blood vessels system</td>
<td>Matter and energy transportation network</td>
</tr>
<tr>
<td>Digestive system</td>
<td>Pre-processing of matter</td>
</tr>
<tr>
<td>Senses: vision, audition, taste, smell, touch (environment, input, ...)</td>
<td>Input, monitoring</td>
</tr>
<tr>
<td>Liver, kidneys, lungs</td>
<td>QC, Filters, cleaning mechanisms</td>
</tr>
<tr>
<td>Lungs, lymphatic system, sweat glands, excreta</td>
<td>Waste disposal system</td>
</tr>
<tr>
<td>Liver, Fat</td>
<td>Storage</td>
</tr>
<tr>
<td>Immune system</td>
<td>Recruitment (selection), Security</td>
</tr>
<tr>
<td>Healing mechanism</td>
<td>R&amp;D, Engineering, Change Management</td>
</tr>
<tr>
<td>Pain, symptoms</td>
<td>Alarm systems, control agents, ...</td>
</tr>
</tbody>
</table>
• A person has a private and public life.
• Organs perform each a function for the body (specialisation).
• Organs are placed inside a structure.
• Vital organs are better protected.
• Organs collaborate with each other.
• Organs are semi-autonomous. The conscious mind can focus on what matters.
• The body is hyper-efficient and has survival strategies built-in.
• The body and brain learn. We learn (also) from mistakes.
• The body and brain can be trained to acquire skills and capabilities. Not training and learning keeps the body weak.
• The body has the ability to heal itself.
• The body needs rest and sleep to restore.
• Prolonged period of heavy activity or stress is bad.
• If one organ fails, another organ may take over its role, be it partly or temporarily.
• Not treating a body and mind properly and abusing them leads to problems (over time).

Notes
This comparison provides lessons useful for designing systems. Probably, other similarities can be found.
Information

Information is the meaning (or knowledge) attached to something produced by humans.

Example: A letter: Nor the letter, nor the words or letters (symbols) are the information. Information is the meaning attached to the letters and words.

Information is the expression of a knowledge.

Message producing a change of a knowledge.

- Information can be “produced” by man-made systems.
- The information doesn’t need to be picked up to be present (books, traffic signs, bottle with a message thrown in the sea, sign in the desert, ...)

**communication**

Captured information produces a change (can be very small) in the knowledge
Information IS the meaning. However, information can not exist without its expression.

Metaphor: A person is not simply the human body. But a person can not exist without the body.
An “Information Element” is a piece of information with an atomic meaning.

Examples: 1984, grey, Roger, cloud, ...

An information element is NOT a field. A field is a Software concept designed to contain an information element. An information element can be stored in a field.

Remark: For now, the term “information” is used for somewhat broader concepts also like an idea, a concept, a situation, ...

Fields, records, files, documents, messages, data files, databases are all concepts related to logical data storage.

Bits, nibble, bytes, words, blocks and pages are concepts representing units in the coding and the electromagnetic storage of this coded information.
Levels

Data
Symbols describing characteristics of objects or events. (raw data, facts)

Information

Knowledge
The memorised information. Knowledge also often answers “How-to”-questions.

Understanding
Explains “Why”, the reason, the causes, the mechanisms, the forces, the dynamics, ...

Wisdom
Ability to consider an issue holistically and objectively, to understand the various degrees of importance and the ability to relativize allowing to see the consequences of (potential) decisions and actions and based on this taking sound decisions.
Information

Information is:
- the meaning assigned to and conveyed by symbols or sets of symbols.
- anything that provides an answer to a question.
Information

1. Brown fox
2. The colour of the fox is brown.
3. La couleur du renard est brun.
4. Fox \[\rightarrow\] Brown

5. 5x same “information”, but expressed in different forms

Capture expressed information \[\rightarrow\] Interpretation of what has been captured \[\rightarrow\] Extracted “Information”

Information is about ‘meaning’. It is immaterial. It has no specific form. Information is an abstract resource describing from the real world to the most abstract and the most intangible, from truth to fiction.

Information is not a tangible, palpable, concrete, physical matter made of atoms. It is not even visible. Only an expression of the information is visible or perceptible.
1. **OMNIPRESENT**

   Information is everywhere in the company.

2. **CRITICAL**

   What would happen if suddenly ALL information of the company disappeared or was inaccessible?

3. **INTANGIBLE / AMORPHOUS / INVISIBLE**

   Information is an abstract concept. Different expressions of information may reveal the same information, a very slightly difference or even the opposite. Information itself is invisible. Only its expression is visible. We can see books and servers or hard disks, but we may not see the letters and the bits and bytes. Even they are nothing more than the expression of information. Even what is NOT said or a silence can be information. Information is similar to knowledge.

4. **MULTIMORPHIC**

   A same information can appear, can be expressed in many different forms and formats

5. **MULTIPLE OCCURRENCES/INSTANCES**

6. **MULTILOCATION**

   Since multiple occurrences (of the expression) is possible, the information has to be present at different places at the same time.

7. **INTERPRETATION**

   Information has to be interpreted by the brain. People may interpret or turn a same piece of information (meaning) into different ‘knowledges’. The value will be different.
**Information**

**Decreasing information value**

**Ambiguity**

Example: term ‘key’: 1) piece of metal used to open or close a lock, 2) piece of keyboard 3) Code 4) Crucial

**Interpretation**

**Vagueness**
Uncertain information is useable, but it is important to take the probability of being either true or false into account and to keep it in mind as long as the information is true.

An information element is (in)validated by other (ex. new) information elements. The “confirmation” itself is an information element.
Human’s Preferences

0101

Yellow

“B4x3C”

Brown

Beach of an island in a purple sea. An orange sunset and in the upper right corner 5 birds flying.

Picture

Video

Visual, richer and easier to process by our brain
Exploitation of Information

Value

Importance is subjective

- Personal interest
- Usefulness (practical usage)
- Already known or not

• Connecting information increases the value, insofar the information is true and not conflicting.
• Links are information in themselves.
• Linked information provide a richer picture.
• Linking information allows to verify information ant to detect conflicting information.

Data + Context → Information

What is context? Context is the set if data linked to considered data element. Context increases the meaning and the value to data.
Information Processing

5  Number (integer)
5,0  Floating number
“5”  Text string
“Five”  Text

5  Graphic

Graphics to be interpreted

EASY to process

HARD to process
An “Information Entity” is a set of naturally related information elements describing a same concept, object, aspect, ...

An “Information Chunk” is a piece of information.
- PROCESS -
What is a Process

Examples of Processes

• Producing a car
• Selling a car
• Driving a car
• Executing a strategy
• Conducting a meeting
• Applying a methodology
• Correcting a client’s address in a database
• Making coffee
• Learning
• Taking a decision

Artefacts representing processes

• Plans
• Methodologies
• Procedures
• Process models
Process Meta-Model

Notes
Not all elements are not always present.

SEE “PROCESS ANALYSIS” for additional key-elements(!) for the Meta--Model
Process Elements

1. Goal
2. Value created
3. Trigger
4. Pre-requisites
5. Input
6. Output / Result
7. Steps
8. Test / Condition
9. Decision
10. Repetition / Loop
11. Discrete, Continuous
12. Linear, Divergent, Convergent
13. Transformation, Assembly
14. Size
15. Complexity
16. Degree of formalisation
17. Degree of structure
18. Phase
19. Gate, stage
20. Used resources
21. Domain of application
22. Required Competencies and Skills
23. Waste
24. Cost
25. Risks
26. Actors
27. Executor, performer
28. Monitors
29. Inspection / Control point
30. Measure point
31. Metrics, Performance, Indicators
32. Capacity
33. Process start
34. Process end
35. Duration of activity
36. Workload per resource
37. Throughput
38. Cycle time
39. Touch time
40. Takt time
41. Work in Progress
42. Work in Queue
43. Critical Path
44. Bottleneck
45. Variability
46. Criticality
47. Supplier
48. Source
49. User(s)
Types of Processes

- Industrial process
- Chemical process
- Physical process
- Transport process
- Social process
- Psychological process
- Intellectual process
- Decision making process
- Engineering process
- Management process
- Business process
- Operational process
- Information process
- Financial process
- Communication process

- ...

Matter being processed

Type of Product / Outcome

Type of Process
Impact of a Process

What does a process change?

Products have themselves effects on the environment (their simple present already is a change)

Side effects
Positive (constructive), neutral or negative (destructive)
Noticed - awareness / unnoticed - ignored

Executor(s)

Environment

Input

PROCESS

Tools Material Energy Information Side effects

Product

Changes / produce

Changes

Changes

Changes

Changes
From Structured to Ad Hoc

**Note:**

Not advisable to try to design a case-based process as if it was a structured process. This is ignoring the freedom of decision making of the executor and the freedom to adapt inherent to that type of process.
Process Views

THREE DISTINCT PERSPECTIVES

1. THEORETICAL EXECUTION MODEL

Presents all the steps and decisions and all the possible executions paths.

2. INTERNAL ORGANISATION

Organisation of the process steps and of the logic

- Layers
- Hierarchy
- Libraries
- ...

3. EXECUTION

- How the process can, is or has been executed
- Specific instance of the theoretical model

ENGINEERS MUST DETERMINE (MODEL) THESE PERSPECTIVES
Hierarchical Decomposition

VERTICAL VIEW

MAIN IDEA

• Dividing the process into steps (top-down)
• Defining the grouping of the steps

A sub-process is a process which is being a part of a higher process.

A process can have 0, 1 or more sub-processes.

Notes:
Conditions, repetition can be indicated at the nodes
In software applications, a process can be 5, 10, 20, ... 50 or more layers deep.
Hierarchical Decomposition

- In how many ways can you divide the rope (by marking the segments, not by cutting it)?
- Work may differ → different steps. Where exactly stops one step and begins the next one?
- Many ways of grouping steps (defining “phases”)

Different ways to divide a same process into steps

<table>
<thead>
<tr>
<th>Process</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 Steps</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td>4 Steps</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td>8 Steps</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
</tr>
</tbody>
</table>

Different organisation of a process in layers

- 2 Layers

- 3 Layers
Hierarchical Decomposition

Many different possibilities are possible

Or tasks are grouped differently in the phases.

Or in three phases.

Can be the exact same process

Notes:
Many ways to represent and organise a same process without changing the execution affects clarity, maintenance, reuse and flexibility
Need to chose the most logical and practical definition of steps, the grouping of steps and number of layers.
Hierarchical Decomposition

IMPORTANT

This represents a **specific existing or designed** (*) process. All the logic is located in the atomic steps (lowest step of each branch). Atomic steps can be located at different levels (in different layers), and not necessarily all in the lowest layer. The “hierarchy” above is mental construction, normally based on a certain logic defined by a functional grouping combined with a level of detail.

**Notes:**

(*) A different case the analysts face is that of process that is still unknown, inexistent and to be engineered.

The “layers” of bubbles tell the degree of decomposition. They don’t represent a level of detail.
Notes
The “goal” is often not just the obvious. It is often not simply the product, problem or business goal. It is actually so much more than what is expressed. Therefore the goal must be identified with the greatest care.

Example:
A marathon: Goal: Winning the marathon
Is the goal reached when winning the marathon and dying after the finish line?
MAIN IDEA
Representing the successive steps of (approximately) same level of detail

Ideal for representing Business Processes

Examples
- Business Model
- SIPOC
- Value Stream Network
- Value Chain
- Data Flows
- Work Flows
- Activity Diagrams
- Swimlanes
Linking horizontal views with more detailed horizontal views
Detailed Process Diagrams

DETAILED VIEW – Flow Chart

- Shows conditions, decisions, repetitions, branches, ...
- Shows main stream and exceptions
- Very easy to read and to understand
- Can represent any level of a process
- Ideal for more detailed levels
- Can be combined with swim-lanes to illustrate who executes steps.
Process & Data

**Trigger**
Event in real world, transmitted as data

**Input** → **PROCESS** → **Output**

- Execution Parameters
- Consulted Data
- Intermediary Results
- Tracing and logging

**Performance Data**

**Meta Data** *(Process Description)*
Identifying existing and required processes

- What **outputs** need to be created?
- To what **events** does the company need to react?
- What may happen in the **real world**?
- What are the **existing** processes?
- What are the **inputs**, **resources** and **prerequisites** for these processes? Other processes are required to get these inputs, to create the prerequisites and to obtain the resources. What are these processes?
- What processes are needed or possible to keep the information up-to-date?
- State transitions in information entities also points to events and processes to trigger and perform these transitions.
Process Analysis

WAKING-UP process

**Action-Level:** alarm clock rings, waking up, brush teeth, put clothes on, prepare breakfast, have breakfast, ...

**Mental level:** Why did you or didn’t you push the snooze button? Did you take the decision to brush your teeth? How did you choose your clothes? To make the choice, did you think on the activities of today? Why? Did you check the weather outside or the weather forecast? What else were you thinking? How did you choose your breakfast? Did you decide that morning about the order of actions? And so on.

**Meaning / Value level:** It is important for you to feel good, to look good, to impress and not to be hungry later in the day. And so on.
## Process Analysis

### Goal - Result

<table>
<thead>
<tr>
<th>Goal / Result</th>
<th>Goal / Result</th>
<th>Goal / Result</th>
<th>Goal / Result</th>
</tr>
</thead>
</table>

### Concrete Process Dynamics

- Step  
- Step  
- Step  
- Step  

or activity, loops, the inputs, outputs, timing, messages, ... (level of UML, BPMN, Flow charts, ...)

### Meaning & role of outcome

- **What is the result** of the step or of the whole process? What is it worth?
- **Role** of this result / outcome?
- **Value** of this result / outcome?

### Meaning of the step

- **Meaning**, sense of the step or of the whole process?
- **Why** is the step, action, ..., process executed? Why executed/designed like this?

### Intellectual perspective of the process step

- What **decisions** are made defining the process?
- What is the **thinking** driving the process?
- What is the **intention** during each step (or whole process)?
- What are the **expectations** of the steps (or whole process)?

### Beliefs

- What **perspective(s)** is/are at play?
- What is the **belief system**?
- What **assumptions** are made?

### Fully understanding of the process

---

**Notes:**

‘Step’ can be a phase, an action, an activity, a process, a function, a feature, ... a group or a mix of these.

Models of processes can be drawn by looking only to the actions, the inputs and outputs. But it won’t get further than the existing processes or “order taking”. A process can’t be designed or properly evaluated without a consideration of the deeper levels. The deeper levels bring true and essential and even critical understanding of the deeper sense of the process. A **methodology** is, in essence, also a process which without deeper understanding can’t be applied correctly, let alone adapted to suit the project and to create excellent results.
**Scope of Analysed Process**

**Limited Scope**

```
Process → Input → Part of a Process → Output → Process
```

"our part of the process"
(like the ‘whole’ process within a department, within a system, within our scope)

**Whole Scope**

```
Input → Process → Output
```

whole process within the supra-system
(within the enterprise, the organisation, the company)

**Extended Scope**

```
Source → Input → Process → Output → Customer
```

- The sources, the sources of the sources and their sources
- The customers, their customers and their customers
- The waste
- Triggers, (hidden) consequences, side effects, ...
- And the behaviours, cycles, relations and environments of all these elements

*Example: SIPOC-model in management*
Process Levels

Process Phases

Main steps (activities, functions, ...) within a phase

An activity changing multiple composite information entities of different types

Activity changing one or more composite information entities of a same type or changing a set of different atomic information entities (a list)

Changes multiple atomic information entities of same type (technically: set of records in same table)

Changes on an atomic information entity (in technical terms: a database table)
Notes:
High-level horizontal cross-border processes shows the process in its entirety, often/mainly regardless of the underlying data architecture.

At a deeper layer, we may find process steps dealing with closely related data entities. Even deeper we may find process steps dealing with lists of same data entities and finally steps dealing only with a single data entity.

The lower levels of process steps can be better organised accordingly to the data architecture.

Representation: It presents one view: the decomposition of a process in its sub-processes, steps, sub-steps, .... These steps can be implemented as services, performed by agents or organised in libraries. They can be executed iteratively or form a network structure. These are just other possible views of a process, its execution or its organisation.
GIGO is not just a phenomenon that happens or caused by bad input. It is a **process design issue**.

It is caused by processes that don’t check their inputs and outputs and are not able to deal with wrong input.

**Input Control**

![Input Control Diagram]

Question: Is the input fine?

**Output Control**

![Output Control Diagram]

Question: Is the output, the work performed by the process, fine?

**I/O Control**

![I/O Control Diagram]

Question: Are the input and output of the process fine?

**Continuous integrated testing**

![Continuous integrated testing Diagram]
Processes Automation

Process executed by people (manual process)

Process executed by people with support of software systems

Computers do not execute the process. They are used to increase effectivity and efficiency, like for gathering information to take better decisions.

Partly automated process

Partly automated process with support of people on automated parts

Fully automated process
Some Design Tips

• Assigning an owner
  • Processes without owner tend to deteriorate.
  • Owner must understand process engineering, monitor the application of the process and keep the process efficient and effective.
  • Not an honorific title

• Organise processes in a logical and practical manner

• Define processes for reuse (modularity)

• Allow processes to be triggered, put on hold, interrupted, ...

• Allow processes to be traced and monitored

• Configurability of processes

• Record process metrics (key performance indicators, and other performance indicators)
More about problem solving:
“Problem Solving – Concepts and Approach for Systems and Strategies”
https://goo.gl/IgxPw0
A PROBLEM is a difficulty to be resolved, an intricate question to be answered, a difficulty to understand or accept. It indicates the existence of a cause, an obstacle or a difficulty.

Why solving problems?

Some problems may:

- Go away by themselves
- Be solved by someone else
- Remain the same
- Amplify
- Change
- Create one or more negative consequences

Solving problems is a way to progress.

Advanced Problem Solving skills help to avoid problems, solve them early and more easily. This limits their negative impact.

The Analyst is first of all a “Problem Solver”
Problem Solving

?  Problem  (In-)Ability

No Problem  Anything that is OK or that can be solved easily

A Problem  Anything we are confident we can solve. But it will require some dedicated effort.

Big Problem  Anything we don’t know yet how to solve. But with a big effort, we believe, we might still be able to solve it.

Very Big Problem  We are unable to solve it.

Each problem tests and trains our ABILITIES

“Problem” = personal & subjective concept

A lesser competent person is more likely to create more problems and to have more problems.

A smart and competent person will have more challenging problems, which may require more effort and time or a higher intelligence to solve them.

Notes:
• A *problem* is a *subjective perception* linked to our own ability to understand, conceive, build, adapt, change, improve, perform and action, get a result or reach a goal.
• It is also linked to our (im)possibilities, to various barriers and limitations existing in the situation and to the importance and criticality of the problem.
A **SOLUTION** is only a solution if it solves the problem (the cause, the gap, the difficulty) without creating (too many) new (significant) problems. Anything else is a **figment**.

- A solution can only exist if a problem exists and has been identified.
- A solution to an unidentified problem is not a solution.
- A ‘solution’ is not a solution if
  - it doesn’t solve the right problem
  - it doesn’t solve a problem in a beneficial way.
- “**SOLUTION**” is NOT synonym for “**PRODUCT**”
  - As a product to be bought
  - As the delivered system of a project

Peter Senge’s 1st Law:

“Today’s problems come from yesterday’s solutions.”
Problem Solving Attitude

- Something doesn’t work out
  - Try harder, more of the same
  - Abandoning (or externalisation)
    - Try something else (idea, trial, intuition) (sometimes: throw away the present method)
  - Think
    - Adapt / Improve the method
  - Think deeper and differently
    - Change the belief system, method, improve competencies, ...

Notes:

Examples:
- Delegating, subcontracting, divestiture, closing down a company branch, privatisation
- Quitting instead of improving
  - Plans don’t work. Let’s stop planning or plan lesser.
  - Requirements are frozen but aren’t good. Let’s allow them to change freely.
  - Documentation isn’t used. Let’s document lesser.
  - The ‘waterfall’ doesn’t work. Let’s abandon the waterfall model.
  - Systems Analysis doesn’t lead to good solutions. Let’s try hope emergence will bring good solutions.

Reality: It’s not the tool. It’s an incorrect application or lack of skill. Improve and Learn
Unfortunately, these models \textbf{don’t help} in problem solving. The whole difficulty of problem solving is concentrated in the step “conceive a solution”. The models don’t explain how to solve a problem.

There are better approaches than cycling until a solution is obtained.
Creative Problem Solving

Define Problem → Analyse Problem → Generate Ideas, Explore, Develop Alternatives → Evaluate Solutions → Select Solution → Plan Action → Implement Solution

Design Thinking

Empathise → Ideate → Define → Prototype → Test

Creative phase
Problem: How to augment a restaurants profits?

1) **Rational, Analytical, Structured Approach**

1. Identify the problem
   - Study of the location, visibility, accessibility of the restaurant.
   - Study of the clients (who, do they come back, complaints, …)
   - Analysis of the welcoming process, the restaurant service, waiting times, …)
   - Analysis of the processes in the kitchen
   - Analysis of the restaurant and kitchen collaboration
   - Analysis of the offered plates
   - Analysis of the cooking processes
2. Solve the identified issues
   - Identify the causes
   - What can be changed and what can’t
   - …
3. Implement the solutions
4. Evaluate the results

2) **Creative Approach**

1. Generate ideas by posing the question: “What can be done to make the restaurant profitable?”
   - Introduce a new theme every month (Mexican food, Italian food, sea, hunting, …)
   - Get some more healthy child dishes
   - Offer a little gift to children
   - Install a playground for children
   - Organise now and then an activity or performance for children
   - Offer rare dishes to customers
   - Disguise the waiters and waitresses and decorate the restaurant
   - Install a bar
   - Get more adapted music
   - Get waiters and waitresses better trained
   - …
2. Think of the feasibility, investment, cost, implementability, risks, consequences, …
3. Select and plan the order of the ideas
4. Implement the ideas
**Notes:**

Does it make sense to discuss the colour of internal parts of a pump in an industrial plant? Or discussing the technical characteristics and production process of a romantic dinner you are about to enjoy in a nice restaurant?

Do information systems in corporate environments solve mainly practical problems or rather choice problems?

How do you see the balance (practical vs choice) in consumer IT products compared to the balance of information systems? What is the impact on the development approach?

What happens if we try to solve a practical problem with an approach that suits much more choice problems?
Two Types of Problems

1. Obtaining a **tangible** result? → Solution: **SYSTEM**

- Production of products or services
- A dysfunctional system, a system that doesn’t function as expected or which doesn’t produce the expected results
- A system missing abilities or capabilities

2. Obtaining **intangible** result? (Reaching a GOAL) → Solution: **STRATEGY**

Goal seems to be unreachable - Path to a goal is unknown.

Obstacles, uncertainties, unknowns, risks, dangers, traps, dead-ends and uncharted territories...

Examples: Winning a battle; winning a medal in a sport; increase the market share, learning strategy, ...

---

**Notes:**

A system executes processes and a system may be part of a solution to reach a goal. Some problems may require both, a system and a strategy (process) to be solved. The conception process of systems and strategies differ from each other. We need a process (strategy, approach, methodology) to build the right system rightly. An inappropriate strategy will hinder the conception.
The main problem in problem solving is (often & mainly)

THE UNKNOWN

• Not knowing what is happening
• Why it is happening
• What the causes are
• What (all) the consequences are
• And not having (not knowing) the solution
• ...

The problem solving initiative is (often/mainly) about

searching and solving the unknowns.

It is about acquiring understanding.

Understanding the anatomy of problems, systems, processes, ... helps

(and the dislike of effort that will be required)
Anatomy of Problem

Focus is on ‘System’-Problem (not on ‘Strategy’-problem)

- A problem is solved by a solution. (problem – solution relation)
- One problem may have several solutions.
- Several problems may have a single solution.
- Not all candidate solutions are equal.
- Different problems may be linked. (common deeper cause? Solving together?)
- Different solutions may be linked.
- Several causes (…) may exist, or several conditions may form a cause.
- A cause can be inside and/or outside the system.
- A problem creates one or more consequences (missing or bad results).
Anatomy of Problem

Consequences can be

- Missing
- Inappropriate
- Visible / invisible; detected / hidden; known / unknown
- Positive/ negative
- Spread or concentrated
- Specific, generalised
- Irregular, regular or continuous
- Annoying, embarrassing, painful, causing discomfort, irritating, ...
- Temporary, permanent
- Reversible, irreversible
- ...

A symptom is a noticeable and/or experienced consequence or a side-effect; a sign of an underlying problem
Anatomy of Problem

Behaviour of system can be influenced, hindered, obstructed, ...

Notes:
The identification of indirect consequences, contamination, impact area, etc. help to correctly diagnose problems, understanding problems, solving them and cleaning up their consequences.
Anatomy of Problem

Problems are (often) dynamic or exist in a dynamic environment.

**Problem Area**

**Solution Area**

**Impacted Area**

**Influencing Factors**

Contamination: *Evolution, evolving, changing, ...*

**Problem lifecycle**

**Solution Element**

Anything that can be part of a future (candidate) solution

**Candidate solution**

Any potential solution; the alternative solutions.
Problem Concepts

Understanding the following concepts helps in analysing and solving the problem

1. Complaint
2. Symptom
3. Impact
4. Consequence
5. Risk
6. Damage, Loss
7. Problem
8. Nature of the problem
9. Perceived (subjective) problem
10. Problem Domain
11. Indirect Effects / Ripple Effect
12. Trigger, Event
13. Input
14. Condition, Circumstance
15. Cause-effect-Relation
17. Problem’s Evolution / Life
18. Forces, Tendencies
19. Problem Area
20. Impact Area(s)
21. Root-Cause Analysis

Note: Concepts related to the solving-part of the process are presented later.
Problem Solving – Cause - Effect Relation

- Implies presence of a mechanism
- Not always just a single cause-effect relation. It can be much more complex.
Sometimes, no root cause can be identified. It might be better to start solving one or a few causes. This may activate a dynamism that will solve many problems.

Note that a cause alone may not materialise on its own. Circumstances also play a role. Are these circumstances normal? What caused these circumstances?

Example: finding a job may solve many other of someone’s problems.
Root Cause

Effect = Cause

Effect = Cause

Effect = Cause

Symptom

Impacts

Disturbing, annoyances, ...

Complaints

Triggers to find the true problem(s)
Basic Problem Solving Approach (1)

Clear and easy solution

Solution is unknown or complex

Problem is not clearly identified and understood

Assumption: The solution is simple and known.

Assumption: The problem is known.

Assumption: The environment is rather simple and very well known.

Communicating the diagnosis is important. The diagnostic has to be established and shared. Only then individuals can work together on a same solution.
In general, the main part of the solution can often be built rightly from the first time. The backward loops are necessary for rectifications, adjustments, improvements and subsequent additions.
Basic Problem Solving Approach (3)

Diagnosis has to be performed by an expert

The person detecting the problem, isn’t an expert. An expert is needed for rightly diagnosing the issue.

Each task (phase) isn’t necessarily performed flawlessly

Although it is not intended, work of earlier activities may have to be clarified, checked, corrected, refined or expanded.

Subjective diagnosis vs Diagnosis: subjective diagnosis is diagnosis of non-specialist. Diagnosis is posed by expert.
Solving the Unknowns

The Symptoms

Problem

Solution

Diagnosis

Solution Conception

Subjective Diagnosis

Study Problem Environment

Objective Diagnosis

Study Solution Environment

Conceive & Build Solution

Apply Solution

Known

Unknown

Unknown

clarifies

clarifies
A problem can’t be solved unless the diagnosis is right.

A solution can’t be devised before the problem and its environment are properly understood.

Engineering

= Methodical Approach
= Based on Problem Solving Approaches

 Doesn’t fit the true spirit of engineering:
• Taking actions with little understanding
• guessing
• repetitively trying until success
• counting on luck and accept increased risks.
A Simple Problem Solving Process

1. Define the Problem
2. Gather data
3. Cause Analysis
4. Conceive the Solution
5. Build the Solution
6. Evaluate the Effects of the Solution
7. Solution Candidate for Standardisation
8. Evaluate the Process
   (what could have been done better?)
Problem Solving Approach

**SOLUTION ORIENTED**
- Investigation
- Learning
- Requirements
- Options
- Conception

**PROBLEM ORIENTED**
- Detection
- Investigation
- Identification
- Localisation
- Diagnosis
- Learning the problem, context, impact, ...

**Building and Implementation**
- Evaluation

Flowchart:
- Start
- Problem Detection & Identification
- Subjective Diagnosis
- Objective Diagnosis
- Conception of the solution
- Implement the solution
- Post-implementation Evaluation
- Stop
1. **Awareness** of the existence of a Problem
   
   We are aware that “something” is wrong but we don’t know what. The problem itself can not be pinpointed. This happen often through symptoms, consequences and results.

2. **Identification** of a Problem
   
   The problem is recognised. It is vaguely located and vaguely known to “get a name”.

   Trying to solve the problem at this state is likely to result in symptomatic solution.

3. **Localisation** of a Problem
   
   The problem has been precisely localised. The circumstances and cause(s) are known.

4. **Understanding** of the Problem
   
   The true root cause(s) is/are identified. The whole mechanism leading to the consequences and symptoms are understood. The consequences throughout the company are identified. The behaviour, evolution and importance of the problem are precisely understood.

5. **Deeper Understanding** of the Problem
   
   These understanding is about the underlying principles, beliefs, the assumptions, the designer’s reasonings and reasons, the WHY’s, …
Do we need to spend more attention on the problem or on the solution?

Note that if the diagnostic is wrong, it will be hard to solve the problem (guess, luck, ‘intuition’, many trials, ...).

No relation between complexity (or size) of problem and complexity (or size) of solution.

A complex problem may be solved with a simple, small and/or cheap solution. A simple problem may require a complex, huge and/or expensive solution.
Decision Making in Problem Solving

Art, Leisure, Clothing & **Candy Shops**

- Likes & dislikes
- Preferences
- Aversions
- Choices
- Wants

- Picking
- Guessing
- Assuming
- Trying
- Exploring

I want ...
And I would like ...
I prefer ...
This is nicer.

**Not a big deal**

"Corporate IT is not a candy shop"

**REAL PROBLEMS**

- Real life problems
- Practical problems
- Problems in systems and environment

Real problems require real solutions

- Understanding of the situation, real world, system
- Right diagnostic, real solutions
- Decisions, reasoning, arguments
- Tastes, preferences and the like are of secondary importance or optional.

**Secondary importance**

**Primary importance**
(approach, methodology, decision making, techniques, ...)

30/10/2018
Some Considerations

• These model depict a general, ordered and ideal process. They show a sequence of aspects to be considered.
• They don’t describe a strict sequence.
• Overlaps are possible.
• If something is wrong, unclear, ambiguous, unverified, ... perform the activity that solves it and go on.
• If necessary or useful, the problem solver can jump backwards or forwards in the process.
• A step can be as short as the mind considering an aspect and may happen in a blink of an eye. But it may take hours or days as well.
• The problem/solution environment might already be known. However, often a problem occurred because we didn’t understand something of this environment. A deeper understanding might be required. Problems teach us something.
• The study of the problem environment may provide the required insight and therefore may make the study of the solution environment useless.
• Such a process model is a basic template that has to be adapted and enriched to the specific problem. For example:
  • maybe different smaller solutions may together solve a single problem
  • maybe some preparatory work needs to be done
  • maybe a solution has to be applied in phases
1. Describe the problem
2. Determine nature of problem
3. Categorise the problem
4. Describe the objective
5. Information gathering
6. Symptoms analysis
7. Stakeholders analysis
8. Do background research
9. Analyse the context and situation
10. Understand the problem
11. Identify assumptions
12. Determine the causes
13. Determine the factors contributing to the problem
14. Determine the root cause(s)
15. Determine the impacted areas
16. Determine and evaluate the impact (damages)
17. Conceive a solution to clean up the damages and apply it
18. Determine solutions to contain the problem
19. Determine solutions to solve the impacts
20. Establish what is fixed, unchangeable, beyond your span of control and what can be adapted, changed, replaced, ...
21. Establish requirements
22. Verify requirements
23. Find elements of solutions
24. Generate ideas (brainstorming, finding analogies, ...)
25. Think of alternative solutions
26. Identify feasibility, implications, obstacles, risks, limits, cost, resource requirements per alternative solution
27. Elaborate a proof of concept, mock-up, prototype
28. Investigate collateral effects
29. Establish criteria to evaluate the alternative solutions
30. Evaluate the alternatives
31. Select the best alternative solution
32. Implement solutions to resolve the impacts
33. Implement the solution
34. Evaluate the solution
35. Improve the solution
36. ...
A More Elaborated Approach

Start

Problem Detection & Identification

Objective Diagnosis

Subjective Diagnosis

Structure the Problem

Analysis

Problem & Solution Area

Objective Diagnosis

Problem & root causes

Impact Analysis

Consequences

Evaluation & Selection

Establish the criteria

Start

Contain the Problem

Conception of the solution

Plan

Correct Impact

Execute (Build & Implement)

Correct and Improve (optional)

Post-implementation Evaluation

Stop
The mind doesn’t think or learn perfectly from the first time. The first analysis or answer may be imperfect. It doesn’t work in a strict linear or sequential way. It jumps back to a previous step to correct or to complete a question already dealt with, and seconds later it may leap forward. There is a clear convergence towards a definitive insight and solution.

The global process won’t be perfectly or strictly sequential. But a pattern of different phases can be distinguished.

The mind can somewhat be guided. By establishing what question has to be resolved in what order, our attention, our focus can be better controlled. This reduces the erratic behaviour and increases the tendency towards a structured approach.
Notes:
If the symptoms or other disturbing effects (impacts) are labelled as ‘problem’, no effort will be made to question their occurrence and no investigation to find the true problems, the root cause will be initiated. Result: symptoms will be solved and true causes will remain.
Diagnosing

Rules of Tacks (Medical)

1. If you are sitting on a tack, it takes a lot of aspirin to make the pain go away.

2. If you are sitting on 2 tacks, removing one does not lead to a 50% improvement.

Syd Baker, M.D.

Accurate diagnosis is important !!

• Do not rush to build solutions solving the symptoms

• Symptoms (complains, obstacles, flawed design, late delivery, ...) are messages about underlying problems. Don’t ignore these messages.

• Remove tacks where possible. Treat underlying causes.

Notes:
Diagnosing Process

Note:
The subjective diagnosis can be right. We don’t know if it is right. Cause and Goal dimension allows to get a confirmation and a valuable broader insight, even if the subjective diagnosis is right.
Diagnosing Process
ASKING QUESTIONS

Two directions to pose questions

- Objectives
- Causes

Never enough questions can be posed about the demand, the assertions, assumptions, context, situation, evolutions, ...

GOAL dimension
Identified by subjective / superficial diagnosis (= unreliable !)

CAUSE dimension
Finding the real problem

Repeat as many times as needed

Why? What for?

Environment, context, situation,
Efficient Problem Solving

During problem solving, the mind runs continuously and iteratively (without specific order) through a myriad of questions like:

- What is the essence?
- What is at stake?
- What are the boundaries?
- What do we know?
- What is the root-problem? Diagnose it!
- What is the main objective? Identify it!
- What is important? What do really matters?
- What doesn’t matter?
- What is impacted?
- Where can we get information?
- What do we assume?
- How reliable is an information? Can it change?
- What can we deduce?
- What don’t we know? Maximise the knowledge and insight?
- Is what we “know” true? (not an evidence at all)
- What can we change?
- What can’t we change?
- What will mandatory have to change?
- What are the mandatory steps?
- What is likely to be changed?
- What are the solution elements which may be part of the solution?
- What are the options?
- Are their different ways to the goal?
- If this or that is changed, what is the impact? What are the consequences? What does it imply?
- Can something be more detailed?
- Can something be clarified? Remove ambiguity.
- What is still unclear? What do we assume? What did we forget?
- ...
A person can be a part of the system. It is harder for a person who belongs to the system, who is inside the system to contemplate that system from a higher perspective with a higher level of thinking. Similarly, it is also harder for the system builder to contemplate the system (s)he builds to consider it with a different way of thinking used to build and or to adapt it.

It is easier for a person outside the system and for a person who didn’t build or adapt the system to consider that system with a higher level of thinking and a fresh view. Issues and opportunities may more quickly be noticed. But knowing the whole system may take some more time to know the entire system/situation.
Problem Solving Traps

1. SOLVING CONSEQUENCES

Problematic situation creates Consequence

Visible, experienced, concrete → known, awareness

Complex, fuzzy, unclear, not in plain sight, demands an effort

Unpleasant, disturbing experience
- Limitation
- Destroy
- Waste
- Conflict
- Nuisance
- Annoyance
- Embarrassment
- Discomfort
- Pain
- Irritation
- ...

Real causes don’t disturb us. We rarely experience them. Why should we solve them?

Desire solve or eradicate the consequences

Focus is on Consequence

Easily labelled as ‘problem’

Strengthen the focus

Closes the mind for investigation for deeper understanding of the problem and searching for real causes

Result:

1) Bad solutions

2) Problematic situation (real causes) and invisible consequences remain unresolved.

3) Symptom → “solution” → Chaos

Systematically solving symptoms creates chaos

Any ‘problem’ should be considered by default as a consequence, until no problem can be found that created that consequence.
Problem Solving Traps

2. PROBLEM = THE MISSING SOLUTION

“The problem is that I have no umbrella”

The actual problem is:

Since it is raining, you will be wet when going outside

Once we have the solution, the problem disappear.
But, the problem is that the solution is missing.

**Solutions:** use a raincoat, find/buy an umbrella, go by car or by train, find a shelter, leave when rain stops, ...

**In practice:**

“We don’t have a database that...”

“I need such a software feature...”

“I want a tablet to ...”

“We need a website.”

etc.

*Missing solutions*

*Solutions to what problem or need?*

*Talking about desired solutions, not about problems. They can be built and delivered, but will they solve the real problem? If not, what is the created value? Obtained satisfaction?*

The business community needs **information**. It has a **information problems** and **information needs**.

Information is needed for business activities, for decisions, ... to achieve business objectives.
3. RUNNING AWAY FROM A PROBLEM

- Minimising the issue
- Postponing taking measures
  - “Not now because...”
  - “First we have to...”
  - “Too busy now...”
- Take some initiatives, doing some attempts
  Not with the genuine intention to solve it (you never know), but with the idea that we did something. This brings a peace of mind. But it doesn’t solve the problem.
- Avoidance:
  - Wait and do nothing. Look the other way. Ignore it.
  - Pass the problem to someone else:
    - Informing someone about the problem.
    - Making sure someone else notices the problem
    - Letting the problem grow until it becomes too big for you to handle.
    - Externalisation: privatisation, outsourcing, consulting, disinvestment, ... (getting it solved by someone else)
  - Quitting. Stop the plan, the practice, the activity.
Problem Solving Traps

4. RUSHING TOWARDS A SOLUTION

- without knowing or assuming knowing the problem
- without knowing or assuming to know the problem’s context
- Attraction to solution: Solutions ease our fears.
- Once the solution is known, the building can start. This makes progress more tangible and visible. Many prefer ‘doing’ over ‘thinking’.
- Common mistake: wanting a particular “solution” instead of wanting to have a problem solved. Very different in approach, talk, attitude and result!! We may end-up with a “solution” but the problem is badly or not solved.
- Assuming too quickly the understanding is sufficient
  - No proper diagnosis
  - No Proper understanding of the context, situation

“YESS! We have a Solution”, “We know how to solve it”, “We know what to do”

Overly focus and to early focussing on the Solution

Early engaging in searching for a solution, in solution thinking, in conception of the solution

Result: Non-solutions and more problems
1. Decisions and actions are (sometimes) motivated by the desire to get a specific result.

2. Attention goes to the decision, action and outcome (the intended and the real outcome).

3. Decisions and actions have effects on the system, environment, resources. These are consequences. Something that changed. (red arrows)

4. Even the simple presence of a new system or object in an environment changes this environment.

5. The obtained outcome (result) also has effects on the system, environment and resources. (blue arrows)

6. These effects (changes, consequences) may alter the capacity, the capabilities, the behaviour or other characteristics of the system. The effects may also have no further (significant, noticeable) impact on it.

7. These effects may create circumstances for an adverse event, from small to catastrophic, to happen.

8. Or, they may increase or decrease the impact of future changes and events.

9. Other effects, particularly the indirect effects are not observed. We may remain unaware of the effect. Or, we may not be aware of the link between a decision or action and some of its effect (light grey ovals).

**Examples:**

- Motivation or demotivation of people is the result of other factors.
- An awkward design may make maintenance or future changes on the system very costly. The effect of this additional cost is unlikely to be recognised.
- A change of colour of a car won’t affect the car’s characteristics. But in some circumstances the colour may, as a camouflage, make the car lesser visible ‘causing’ an accident.
1. Limited DIAGNOSIS – Solving Consequences

1) Disturbing, undesired, visible effect experienced by individuals or by the company

2) This must disappear. ("be solved")

3) What causes these issues?

4) This part must be changed?

5) This is the problem. Diagnosis is done.

'A' is seen as the cause. It produced the unwanted effects. It is not considered as a consequence. B, and a fortiori C, remain unresolved.

→ Not much fundamental matter is learned.
2. LOCAL SOLVING

1) **Locate the area** in the system producing ‘the problem’
2) Locate ‘the problem’ (from broader area to a more precise area)
3) **Analyse that part** of the system
4) Look how it can be adapted – come up with a few solutions
5) Decide how to adapt or add something to the system
6) Adapt the system
7) Test if the adaptation offers the desired behaviour

**Points of intervention**

- Adapting A to avoid producing the negative effect
- Control, Blocking, Filter
- Clean up

Adding a layer/component to manage the consequences

**SYSTEM**
Limited Diagnosis – Solving Consequences

A process (chain of steps) or a set of connected systems, or other chains of similar or physical components can be investigated to find the root cause. This makes sense to correct mistakes, but not to prevent further similar mistakes. It prevents from learning at a deeper level and to improve in a more fundamental way. Often a true crisis or a repetition of similar flaws forces us to leave this investigation path and to search for a new one.

We consider only the question “where did what go wrong”. The question how we evolved to that situation, what is missing or wrong in the design process that created the flaw is not raised.

Finding the True Causes

All kinds of flaws come in a design because of bad choices and decisions, incorrect methods and unsuitable work environment. Work environment include here things like collaboration, communication, mutual support as well. The cause of unsuitable decisions, methods and environment can be traced back to the lack of skills, too little thinking, insufficient awareness and a lack of understanding. The latter one is the most important. Learning is key.

Backwards investigation (analysis) towards the root cause

If cause is here, what implications? How to solve the cause?

Understanding why things evolved to the flaw (from scratch to flawed solution)
Risks in Local Intervention (Local Adaptation)

Knowledge after local investigation:
- Global understanding of the system and environment
- Investigated local area
- Still many vague or unknown areas and interactions
- Requires (also) knowledge of the internal organisation of the system

Risks:

Adapting or influencing the functioning of a system or its environment, may have undesired effects (not always immediate, not always visible) due to pre-conditions, interactions, limits, relations, behavioural aspects, .... not taken into account (ignored)

Intervening on a complex system which is not fully understood is taking a huge risk.

The more we know the system, the fewer problems will occur due to ignored interactions.

Analysis, controls and simulations may help to reduce these effects.
Notes:
- Useful for matters which are difficult to predict like response of people (preference, interests, ...)
- Useful for concrete, local, practical aspects like User Interfaces improvements
- Trying is good if no other alternative does exist.
- Not very useful when a broader, deeper more holistic insight is required.
- Adaption: Redo some work; some work has to be thrown away and replaced by new \(\rightarrow\) includes a waste of time and resources
- What is the guarantee that what we learn is valuable and that after learning it, no other iterations will be required?
- How much do we learn compared to the time spend on building a non-solution? Is there a faster way to learn that? Can we learning by trying and by getting feedback in a system (the whole organisation) where everything is connected, but where knowledge and insight is more specific and local?
Problem Solving Traps

Rushing into problem solving

1. I understand the problem and what we have to do. Let’s design the solution or adapt it.
2. Let’s build the solution.
3. Let’s implement and apply the solution.
4. Let’s use it.
5. It didn’t solve the problem. There are new problems. Let’s look what we can learn from it and so we can fix it.

30/10/2018
Problem Solving Traps

Mind is driven by

- “What solution is required?”
- “What should the solution look like?”
- “What solution will solve the problem”

Rushing into problem solving with too little insight is not without consequences.

1st cycle:
Conceiving ‘solution’

Subsequent cycles:
Conceiving corrections

Building ‘solution’

Implementing ‘solution’

Feedback & Learning

PDCA: Plan - Do - Check - Act
PDSA: Plan – Do – Study - Act

Size of produced work

Attempts 1 2 3 .... n

Problem solved after n attempts
BUT
- Will it ever be solved (certainty)? Predictability of required cycles? Predictability of timing?
- How well solved? How much rework will it take?
- If solved, then cycling until we learned what is necessary... or not even that.
- After n cycles, possible conclusion that the wrong problem is being solved
- What is learned is certainly useful. But no guarantee that we learned what we need to learn.

Learning comes late in the process.
Building solutions with superficial insight is inefficient and risky.
Investigating and getting understanding

Building solution

Implementing solution

Feedback & Learning

Conceiving / adapting the solution

Cause of number of attempts:

Lack of understanding when building

How to limit the number of attempts?

• Put learning, acquiring understanding early in the process
• and learn from the result

Attempts:

1 2

Size of produced work

Ensuring to tackle the right problem
Problem solved (preferably/more likely) after 1st iteration.
Ensuring problem is correctly solved.
Doesn’t exclude (minor) corrections or future adaptations.

Two prerequisites:

Identify the right problem

Understanding the problem, the supra-system, the problem area, the context, ...

Conceiving the right solution to solve the right problem

KEY: HUMMILITY – CURIOSITY – QUESTIONING
Problem Solving Traps

- Confusing the “understanding of documents, of demands, of requirements” with “the understanding of the situation”
- Confusing ‘knowing’ with ‘understanding’
- Not identifying the unknowns & the assumptions; not verifying the assumptions; filling in the blanks with own answers; guessing

- Choosing instead of deciding (based on unimportant aspects; based on an interpretation, assumptions and preferences; taking uninformed / ill-informed decisions)
  - The choice based on preference (like for taste, smell, colour, fashion, meals, deserts, …) is fundamental different from decisions about systems, solutions, and so on.
- Oversimplification (= ignoring some existing complexity)
- The brain doesn’t like complexity and effort. This is why a simple but wrong solution (or idea, concept, …) has far more chances to be accepted and to be spread than a complex but right one. Anything, like abstract thinking, requiring an effort of the brain is unpopular and everything requiring no effort is popular (neuroscience).
- Avoidance by externalisation (= avoiding to have to deal with existing complexity).
  - Avoidance is a missing chance of strengthening the own skills.
- Going along with the first answer, accepting the first solution (fear disappear as soon as we have a solution. Proposed alternatives creates a new problem (which one to chose, which one is better, new discussion, … a new effort for the brain)
- Fake alternatives: Often a set of alternatives are proposed. Only one of them is / looks like a real solution (which is the one someone wants to sell), the other “solutions” are obviously worst. This is a fake choice among fake alternatives.
- People often are not aware of what they need. They confuse what they want and what they need.
- People may define the missing solution as being the need, instead of what it is supposed to solve. “We need a new database. So, the missing database is the need.”
- Believing the labels: “solution”, “business”, … Accepting anything that has been labelled as “solution” or which just looks like bringing progress and is related to the problem.
- In a choice between the truth or conforming with the choice or idea of the group, people will often chose for the latter. Choosing for an answer (the truth) different of that of the group may lead to exclusion, rejection (survival strategy). Result: weakest ‘solution’, lesser innovation, …
- “We have acquired a solution. Let’s now look how we can solve the problem with the solution. ( no understanding of the problem, no cause identified, …)
INFORMATION COMPONENT
The information Component is that part of the organisation that organises, regulate and formalises information and the way information is processed and managed.
The IC: a heterogeneous multidimensional system

Notes:
Indicating all parts, motives, aspects, dynamics in a single model would make it too complex. However, it gives a general idea, although it is still simplified. This forms a whole in which harmony has to reign.
Today, the focus is essentially on engineering software systems. Engineering the whole Information System is rarely considered.
‘Entropy’ is here defined as “the gradual decline into disorder of systems or societies”. 
(Remark: different meanings and definitions do exist for the term ‘entropy’)

Information systems (and software systems) are subjected to the tendency of entropy:

1. Degradation of Design
   Wrong abstraction have been used, changes of purpose, inelegant design, wrong concepts, ...

2. “Degradation” of source code
   Bad programming: elegance of code; organisation of code; containing inefficient code; unused remains of old code cluttering the source code; inappropriate object, variable, function and method names; ...

3. Information contained in the system
   Unreliable information, not updated, incomplete, encoded differently, not respecting standards and formats, old information, abuses of the system, ...

4. Technological fragmentation and chaos

5. Increased misalignment with co-systems, supra-system(s) and environments

Reflection:

“How do you combat entropy in Information Systems?”

Find as many principles, approaches, techniques, and other ways as possible.
Types of Software Systems

- Mainframe System
- Client-Server System
- Single/multi-User System
- 2, 3, Multi-tier System
- (Near) Real-time System
- Batch systems
- Distributed System
- Networked System
- Agent-based System
- Web-based System
- Intelligent System
- Embedded System
- Back-End / Front-End System
- Enterprise-Wide System
- Communication System
- Centralised / Decentralised System
- End-User Software
- Operating System
- Cloud System
- Workflow System, Routing Systems
- Messaging Systems
- Peer-to-peer System
- Pilot system, Prototype System, Test System
- ...
BUSINESS KNOWLEDGE & EXPERTISE
Business Domain Knowledge (BDK), Business Knowledge and Business Expertise (BKE) is critical. It is the INPUT of the project.

Need for understanding “Business Knowledge and Expertise”

- What is BDK, BKE, ...? And what doesn’t belong to these fields?
- What level and quality of this knowledge and expertise is required by the project?
- What is the quality of the available business knowledge?
- What is the quality of the source?
- What are the limits of competencies of Business People in relation to systems development?
Assumptions about Business Knowledge

In general, nobody knows the business better than the business experts.

Assumption 1
Business experts know the business very well / well enough

Assumption 2
Business experts are the only source of business knowledge and expertise

Assumption 3
The business knowledge is sufficient and readily usable for information systems development

Assumption 4
Business experts are able to estimate what business knowledge the Analysts and other IS engineers need to know, when they need to know it and, more importantly, what they don’t need to know (filter). They are rightly controlling the flow of knowledge to the IS Engineers.

Assumption 5
Business experts are able to communicate all the required knowledge (completely, unambiguously, degree of certainty, guesses in a methodical way, ...)

Assumption 6
Business logic has to be determined first. This creates a first global picture of the solution. Non-business logic can be determined later and added to the solution.

Organisations, approaches, collaboration and decisions are based upon these assumptions. We can’t simply continue with assumptions. They must be investigated, clarified and checked to know whether they are true, to what extent, and so on.
For every statement (or decision) it is important to understand to which discipline it belongs.

Who captures information, where and how it is captured, where it is stored, how it is processed (partly), what information is kept, what information is measured, how it is shared, ... all this doesn’t change how business is conducted. It is not business domain. It’s informatics.

**Note:**

**EXERCISE** (useful!): For some industries and for every business function think of statements which do belong to the industry or business function and also to which they don’t. This helps in identifying the limits of these areas. It will help to identify when an SME in one area talks outside his/her area of expertise.

Beware of labelling any logic other than user interface management or data writing or retrieval logic as “business logic” !!
Examples of Business Knowledge and Expertise

• Everywhere we do business we use as much as possible local products and raw materials and do business and collaborate with local companies.

• No ship shall enter or leave the port without the permission of harbour master's office. This regulation does not apply to tug boats and pilot vessels while conducting ships entering or leaving the port.

• New orders above the 10k € have to be validated by a manager if the client has still overdue invoices.

• Our company sells standardised items and customised items. Customised items require meeting the client in person and discussing the customisation before any order is placed.

• In dealing with immigration, the first step in the administrative process is to identify the person.

Examples of Real World knowledge

• A family is a group consisting of at least two persons and can be described as one or two parents and their children living together as a unit.

• Every city is located in a single country.

Examples of Informatics

• The client information has to be sent to the account manager.

• The clerk has to capture the information.

• This information will be stored, but we won’t keep that information.

• We need to be able to search for these orders with these criteria in the database.

• This feature needs to do this and that.

• The resulting features have to appear in this screen.
Required Business Knowledge

WHAT BUSINESS KNOWLEDGE PROJECTS NEED

- 100% of the Business Knowledge and Expertise
- Multiple business domains knowledge
- Enterprise-wide knowledge
- Multi-level: from operational to strategic
- Relevant
- Complete
- Coherency between predefined logic and applied logic
- Clear, Unambiguous
- Detailed
- 100% certainty
- Structured, organised
- Mainstream and exceptions
- Realistic, applicable, achievable
- Settled in time: difference between what is valid now, what is valid later.
- About “As-Is” and “To Be”
- Coherence among the stakeholders
- Immediate (right) definitive decisions (no guesses, trials, partial, conflicting, ... )
- Accessible
- On time
- Stable
- ...

REQUIRED? YES
REALISTIC? NO

PROBLEM!
(= job of the Analyst !!)
In practice, the intended business knowledge to be applied is adapted and expanded:

- Unforeseen events, situations and exceptions happen
- Problems have to be solved
- People can improvise to cope with lack of details
- Learning path
- Interpretation of the intended business
- Memory (knowledge is variable)
- Not all individuals do the job the same way
- Some degree of freedom in the application. (Example: determined price isn’t used, but is negotiated with the client)

**Proposed definitions:**

**Business Domain Knowledge:** Academic field, business theory, generic knowledge that can be used by different companies as a foundation.

**Business Knowledge:** The specific BDK and business logic implemented in the company.

**Business Expertise:** Knowledge and experience in how the company conducts business and how it turns out in practice.
**Managerial level:**
The knowledge at this level is broader, global and may concern the past, present and the future. Knowledge of objectives, intentions, plans, policies, ... Knowledge of the details and practical experience, as needed for practical execution, can be fuzzy, unreliable or missing.

**Operational level:** The knowledge is detailed and practical. Operational objectives, habits, exceptions, concrete gaps, hindrances, local opportunities for improvements, ...

An employee or SME knows his role fairly well. The own service or department is also known. However, this knowledge may be incomplete. Knowledge of other roles, services and departments becomes much more superficial and uncertain. Only the information usage and the parts of the business processes in the own department or service is known. A more global picture can be lacking.

A business subject matter expert has expertise in one or in several domains. Business domains often overlap. The business subject matter expert may have some knowledge of other areas as well. His/her expertise in other domains may be lesser. Being an expert is not a generic qualification. Someone can be an expert in an area, but not in other areas.

Usually, each organisational unit has (0?,) 1 or more main business domains. There is not necessarily a 1-to-1 relation between a department or service and the applied business domains. A department may use this knowledge for its own purpose, for a global goal or to suit the needs of other departments.
Business Knowledge and Expertise

An SME has knowledge of his business domain. Some of this knowledge is well-mastered. “Knowledge” (and expertise) is not collection clear, nicely delimited, concrete, factual amount of information. An SME knows some areas of his domains. There might be blind spots, vague areas, unknown areas, distorted knowledge and so on. Knowledge is not black or white, true or false. Many qualities of knowledge is a matter of gradation. Additionally, there is a difference between knowledge and understanding. It is also a matter of perception, perspective and belief. Some knowledge is not always consciously present. Some knowledge can be inferred. An SME can always correct, improve and expand his expertise in his domain. Knowledge is not a static fact. It may increase or decrease. Memory is volatile.

A company has usually different persons with business domain knowledge. So, there is more business knowledge present in the company than there is in the head of a single SME of that company. The business knowledge of each SME of the company varies. They have a lot of knowledge in common. But there might be nuances, gaps and contradictions. SME’s may have a different knowledge, a different understanding, a different perspective on a domain or issue. SME’s consult each other to exchange knowledge. They don’t always agree with each other. The application of business knowledge is often a combination of business knowledge and knowledge of the daily world (the reality).

There is more business knowledge or knowledge useful for a particular domain in the world (universities, libraries, web, companies, organisations, other experts) than present in the company. The business knowledge outside the company, although useful, is not necessarily the knowledge the company uses.
The function or term of “Expert” is not a measure of solid business knowledge. Some SME’s do a genuine educational effort to improve their expertise. They are keen to select the sources they learn from. However, a common learning tactic is to learn on the field. The SME learns from colleagues. These colleagues learned themselves from other colleagues. Activities, work approaches and methods are transmitted. Belief systems and habits are perpetuated in the company. Norms and expectations aren’t raised. They are used as objectives. So, some tend to be satisfied with somewhat better than average, being average or not so bad. The knowledge of the expert may not go beyond the functioning of the own company and is not backed up by a solid knowledge of the theories or broader industry knowledge. Many people don’t deepen their insight and further develop their skills. Some don’t like reading or are demotivated for one or another reason. Some persons are considered as expert already after a few years. Some experts may acquire 1 year of new experience and then repeat this experience for the next years without fundamentally learning something new. These tendencies are not favourable for the development of real expertise. The Analyst has to take such situations into account.

Business expertise is developed and learned to be understood and used by people. Its purpose is to allow people perform their job. People understand high level instructions. They can interpret them, evaluate and decide to act in the best way accordingly to the issue and to the circumstances. They can deal with uncertainty. They can adapt and improvise. This knowledge and expertise is meant for people. It has NOT been developed for software applications and computers.

The quality of information, particularly knowledge and instructions, meant for people is very different from the quality required as input for engineering information systems.

Notes:
The Analyst has to verify the information received from the business community and to deal with these types of difficulties. If the Analyst doesn’t, the system design may be conceived based on weak information.
Business knowledge is a narrow view of all the knowledge required to diagnose and to conceive suitable information solutions, let alone to innovate. Users satisfaction is, as criteria, by far insufficient to evaluate a system properly. It is based on this same perspective.

The users/business perspective, although essential, is inappropriate to conceive and to evaluate a system.

**Note:**
Can a driver (user) decide whether a car (system) is safe and roadworthy? Customer and Business perspectives and their knowledge are important. They are important to be taken into account in the design. But they are insufficient to diagnose, to design (conceive) and to fully evaluate information issues and information systems. Is it possible that a solution satisfies the business stakeholders, yet hinders, limits or harms the company as a whole?
### Engineering vs Usage

<table>
<thead>
<tr>
<th>Engineer</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Focus</strong></td>
<td></td>
</tr>
<tr>
<td>System oriented</td>
<td>Usage &amp; Result oriented</td>
</tr>
<tr>
<td><strong>Driver</strong></td>
<td></td>
</tr>
<tr>
<td>How can I make the system better?</td>
<td>How can I use the system in a better way?</td>
</tr>
<tr>
<td>How can I make better systems?</td>
<td></td>
</tr>
<tr>
<td><strong>Thinking</strong></td>
<td></td>
</tr>
<tr>
<td>Systemic Thinking, Engineers Thinking, Holistic, Methodical</td>
<td>Actionable Thinking</td>
</tr>
</tbody>
</table>

#### Iceberg

- Ice above and below the waterline

#### In IT

The part above waterline is different from part of below waterline. GUI is by no means comparable with internals of systems.

Both parts are different

- in nature
- in possibilities
- in size
- in complexity
- in conceptual approach
- in required knowledge and competencies
- ....

#### Reflection:

A User is NOT an Engineer!!

- Does a car driver understand the car engine?
- Can he/she engineer a car or provide car requirements?
- Can he/she lead a team of car engineers?
- Can he/she says when the car is roadworthy?

Car: The dashboard, steering wheel and pedals (users view) is very different from chassis and engine (engineers view). \(\Rightarrow\) Ability to drive a car ≠ Ability to engineer a car.
A System is more than Features

What the users know of software applications

What a software applications is

“User Interface” and “features”

• Does a user understand the car (system)?
• Can a user judge and value a whole system through its interface?
• Is it the role of the driver to engineer the car?
• Can a user engineer a car?
• How would the car look like if it was engineered by a user? Would it be roadworthy?
Business Activities vs Enterprise

- Business defines the activities deployed by using the Enterprise.

- Business defines some(!) of the capabilities of the Enterprise

**Business, as activity and as field of expertise, impacts the design of the enterprise as a system, but it does NOT define it.**

Why? Because the business perspective is only one perspective; a partial, on-sided and limited view. It does not provide a holistic view combining all the perspectives.

**Notes:**
The physical activities don’t define our body. They defines the capabilities required by the body. Training creates these capabilities. The need for survival impacts the design of the human body much more.

A same business (same business activities) can be implemented by very different systems of systems. These systems implement different concepts, different logics, different architectures, but they implement the same business logic. (like “one problem can be solved by different solutions“)
Business Knowledge and Expertise in Software Systems

Besides the business logic, software applications contain a lot of other logic that matter very much as well.

Implementing business logic with just enough of other logic (like screen and menu management and logic to store and retrieve in interactive software application) is unlikely to lead to robust, well-functioning and lasting systems.
Can be true for simple problems, concrete and local logic, small(er) solutions

Larger, sharing, multi-business-domain, integrated more complex software systems require more of THIS model

CAUSE OF FAILURES OF LARGER PROJECTS

is the **underestimation** of the required degree of Project Management expertise, Systemology and Informatics (non-IT part) expertise required by the project.

This leads to

- ineffective project organisation and forms of collaborations
- and to the non-respect of principles in the above-mentioned disciplines.
The implemented (pure) business knowledge does rarely/not necessarily form a uniform upper layer in software.

Implemented business logic appears more as tests, computations, connections, calls, and so on. It is spread throughout the software application’s logic and databases. It is intermingled with other types of logic.

Not all logic used to guide activities executed by business people is business logic.

If business logic would fundamentally define the solution then once the business logic is settled, only one possible solution can be created. This isn’t the case. Business knowledge and expertise influence the solution, but doesn’t define it. It may determine parts of it.

Non-IT people are not used to deal with such abstract, detailed and formalised logic. All the program logic behind the user interface is completely different from the logic end-users are accustomed to deal with. It is the formalisation of a very detailed mathematical precise formalised fixed(*) logic. (*:people can interpret instructions, computers execute the code like it is written). End-users also know pieces of logic implemented in a software application and the result of some functions.
We know (common complain):

“The Business doesn’t know what they want”

• Given the nature, the quality, the fragmentation, the spread of business knowledge,
• Given that the role of the business community is to make the business running
• Given that most people don’t know what computers can do or can’t do
• Given that most people have no clue about IS engineering, Systems Science, ...
• Given that most people’s brain is not wired like that of an engineer

and

• Given the business community is not qualified to engineer information systems

any approach expecting the business community to define its information needs, to specify what information solution and its outline they need, taking informatics decisions and even “guiding” IS projects is a way of working to avoid.

Notes
Business community should take business decisions. Informaticians should take decisions related to information, information organisation, information processes, information systems, ...
• Information products & Information services change business models
• Web, RFID, Smart devices, BI, Big Data, AI, ... change the organisational structures, the collaborations, how business is conducted, and even business models. ( )
• The power of Informatics is at the level of information concepts, information organisation and information processing (example: AI). Although this is “abstract”, this shouldn’t be underestimated. Therefore, one need to know what software and computers can do and can’t do.
Business – IT Knowledge

OLD Paradigm

Business Knowledge and expertise

Business knowledge governs the IT expertise.
“IT serves the business.”

Consequently:

Habitual Project Methodology or Approach

Business Knowledge and expertise

Consequently:

Business stakeholders guides the IT projects
(determines, defines, controls, manages, ...)

2. “Throw-it-over-the-wall” - approach

First, determine the Business knowledge and expertise to be implemented.
Pass it on to IT. IT implements it and also adds its own logic.

This belief does not match the reality, yet it is commonly implemented in approaches and applied. It is a source of problems and serious limitations (among others, inhibiting innovation).
Notes:

The business logic remains the same in the 3 implementations. But the actions executed by the business community are changed.
Business & Information Logic

Business Logic may be dispersed in an information process and can be more or less concentrated in one or another area.
(example: business rules)

Business logic may influence or determine the structure of an information process

Information logic may influence or determine the structure of a business process

Notes:
This impacts the design approach, the priorities, the focus, the leading competencies, ...
2\textsuperscript{nd} drawing: shows only the amount of both types of logic. It doesn’t show ‘layers’ of knowledge.
Notes:

• Some business knowledge has no influence on the concerned information solution.

• Or, its influence is local and limited to the execution of it (like a specific formula).

• Some business knowledge influence the information solution. For example, some parts of the business may be very strict and formalised and use procedures, while other area’s are more free and function based on cases.

• Some business knowledge determines how information is processed (like the computation of price). However, not every information processing is determined by business knowledge.

  • Example:
    • Selection algorithms
    • Sorting algorithms
    • Matching algorithms
    • Identification of data elements
    • Data quality verification
    • Prioritisation algorithms
    • ...
1. The user may assume the demand, requirements, blueprints and specifications are implemented in the software application.

2. The user may deduce the internal logic and behaviour of the software application through its user interface and other observable outputs.

   **AND**

3. The user may imagine how the logic expressed in the transmitted documents have been implemented.

**THE CONTROL IS VERY WEAK, NOT TO SAY ILLUSIONARY**
Software Applications Under Control of Non-IT People

What is in the black box?

“It must have 4 legs and it must be possible to sit on it.”

Mission, requirements, ...

Project Team A  Project Team B
build  build
Software system  Software system
≠

Same requirements for same problem, same environment, same ... going to 2 different team lead to 2 different solutions implementing the same business logic.

Why?

• A same business logic can be executed by different mechanisms.
• A software application is also guided by non-business principles and contains a lot of non-business logic and even complete non-business mechanisms.
• Business logic often influence, but does not necessarily determine the architecture.

Although useful, requirements are insufficient to conceive and to build the right solution. And it is still a control, but it is a control which is still too weak.

Notes

If requirements are given to the teams, the differences may be important. On the other hand, if specifications are provided to the teams, the differences will be smaller.
Without information, business activities cannot be applied. The application of business knowledge and expertise creates information needs. Solving these information needs can only happen with information capture, processing, transport and storage and retrieval.

While the information aspect has to respect the business logic, one should not confuse both. They are NOT the same.

But information, through its intrinsic value, provides also opportunity for business activities. Available information and potential of informatics create opportunities to improve business activities.
Knowing Software Applications?

The user interface (together with some concrete outputs) is the only part of the software application visible to end-users. The end-users know and can use the user interface.

The user-interface forms a barrier between the end-user and the rest of the software system.

It prevents the end-user to see the internals of the software application. It is like a concrete layer on top of the software application. They have no access to anything beneath the user-interface.

The user-interface is like the dashboard of a car, a drink distributor or a movie. A dashboard is very different from the car engine. It tells very little about it. Knowing how a drinks distributor looks like and knowing how to use it, does not imply knowledge of the internals of a system. Or a movie tells little about the movie set.

Knowing and being able to use software through the user-interface is very different from knowing the internals of the software application.

What prevents developers to convert invoice data into chess moves and to send them to a chess game? The end-user wouldn’t notice anything.
Sources of Business Domain Knowledge

Three Types of Sources

Documents
- Policies, documentation, procedures, ...(present); Mission, vision, strategies, tactics, objectives, plans, novelties, opportunities, ...(future)
- Formalised, but can be outdated

Implemented and Execution
- Existing systems, existing procedures, communications, actions, work, organisational structure, decision making processes, structures, concepts, principles, ...
- It is the present, the “as-is” ... and it is an important source for diagnosis

People’s Knowledge and Experience
- Knowledge, understanding, insight, experience
- Thinking patterns, beliefs, norms, values, priorities, expectations, fears, tendencies, habits, decision forming, unspoken objectives, personal drivers, cultural aspects, limiting beliefs, ...
- Through communication
- Subjective, interpretative
- Relies on memory → variable, different perspectives, different understandings
- Different persons may have contradictory ‘knowledge’

Notes:

We may find information in one source that we don’t find in another. It is up to the analyst to put it all to sort it out (and to get confirmation and consensus).
1. Business knowledge is
   • intended to people and human activities, not to software and computers.
   • high level, partial, ethereal, uncertain, fuzzy, variable, not always complete and coherent, ambiguous, interpretable, ...
   • hidden in many forms, places and of different natures
   • not readily available for usage
   • constitutes only a part of all the required knowledge

2. Information solutions don’t solve business problems (unless the business is about information). They solve information problems experienced by the business community, allow information needs to be solved and it is a way to seize opportunities offered through information (improving the exploitation of information). This is why business knowledge is only a part of the knowledge required to diagnose issues, understand environments and design solutions.

3. The required competencies to diagnose, fully understand and conceive information solutions and software systems are: information, processes, Systemology, engineering, Systemic Thinking, Information Science, understanding of possibilities and limits of software and computers, ...

4. Software applications are, basically, black boxes to the business community. The mastery and understanding of software application is based on assumptions and confused with the usage. The control of their internals, which is what matters, is rather an illusion. IT people are also users of software applications, but they know the software applications they developed themselves.

There is a huge work for the Systems Analyst (or BA) to search, to gather, to study, to consolidate, to stabilise and to verify and to select the necessary business knowledge, as one of the inputs of the required knowledge.
HISTORY
Evolution

Problem Scope & Functional Breadth

- Size
- Complexity
- End-users
- Number of systems
- Data volume

Timeline:
- '60
- '70
- '80
- '90
- '00
- '10
- '20
Evolution

Relation between automated work and organisation

- System supporting a specific task
- System supporting a business function (like a service or dept.)
- Systems implementing processes running across the company

Information Usage

- ‘personal’ data
- Data shared for similar usage
- Data shared among different people, for different usages and purposes, at different places in the organisation at different moments in time.
- Data shared with other organisations and with individuals outside the company

Geographical spread

Architecture

- Monolithic
  - Modular
  - Layered
  - Agent-based
- Connected
  - Tentacular
Evolution

Inter-systems Interactions

Integration

Focus of Engineering

Development Languages and Techniques

Technological Fragmentation
Role of Software Systems in Business Operations

Supporting business operations

BEING the BUSINESS
Executing the main business operations or executing the major share

Role and size of Soft. Syst. in the Company

Tool

Integrated Part System of Systems

Timing

Working hours
Start the computer
Launch the Application

24/24 – 7/7 – 365/365

Contribution to the Business

Solving a Need or Problem
Improve operations
Contribute to higher objectives
Innovate; Shape the business; Drive the business
Evolution of IT Systems in the Company (1)

1) A new system supports administrative tasks or executes some administrative tasks of an organisational unit.

2) More systems are built independently from each other. Issues: different technologies; duplicated data; no consideration of reuse, enterprise-wide processes, sharing data, …

3) Communication between the systems happens by printing data on paper and re-encode the data in the other system.

4) Communication between the systems is automated: file transfer (E.g. FTP)

5) Middleware

6) Enterprise Application Integration

7) Enterprise Systems Bus
7) The **Extended Enterprise**: systems goes beyond the company’s borders, connecting systems of different organisations with each other.
The early software systems consisted of a software application running locally on one computer using data stored locally and was operated by a single end-user. Likely, this software has been written in Assembler. It is very demanding to develop software in Assembler. The computer hardware had very limited computing power. The first software applications were rather small.

Better programming languages appeared. Among them the so-called structured languages and later the 4GL languages. They facilitated the development of larger software systems. At the same time, hardware became more powerful. The number of users for a single software application increased. First, multiple users using each an occurrence of the same software application and later the multi-user systems appeared.

The Analysts and software engineers faced challenges that couldn’t be solved with the regular approaches and techniques used by then to conceive software applications. New analysis methods were invented to surmount those challenges. Now, even larger software systems could be developed. The object-oriented programming languages and middleware appeared. New architectures could be developed. Again, the habitual methods were not adapted to the OO-philosophy. New analysis and design methods were needed. Case-tools followed the development of new analysis and design methods.

The size of developed software system continued to increase. Project management and methodologies facilitated the execution of such larger projects. Their importance increased. They became a necessity to respond to the size and complexity of the systems.

Needs drove innovation. These inventions increased the scale and complexity of the systems. The inventions and the larger systems brought new issues in matters of engineering approach and methods to the engineers. New methodological solutions were found to cope with them.
Evolution of competencies? Transfer of competencies?

Building a small piece of software application is relatively easy. Larger software systems are a challenge. They have their own specific issues and needs.

What triggered these evolutions? Is this evolution logical? (we need to think way further than the conventional answers)

Does the evolution of systems and the challenge of developing them match the evolution of software development?

The evolution doesn’t stop here
From Tools to Integrated Core

- Small
- Lesser critical
- Limited complexity
- Specific role/function (limited local scope)
- Supporting end-user’s activities
- Serving 1 user, later a few users
- Without inter-system information exchange

- Large
- Critical
- Complex
- Broader goals
- Performing the core activities
- Performing a huge share of the activities performed by the company
- Interconnected
- Crossing boundaries
- Inter-organisational information exchange

From building set of tools to

Conceiving the ‘core engine’ of the company

+ Explosion of information, communication, systems and connections

Fundamental Change

The Importance of the Role of Analyst INCREASED and became CRITICAL

This is absolutely not about more of the same or the same at larger scale. It is a much deeper change requiring a different way of thinking, different competencies, ...
Methods, methodologies, frameworks, project management developed in response to issues and needs created by the increasing size, complexity, risks and cost of software development projects and to increase the efficiency. Each step in the evolution was a learned lesson and it enabled them to cope with the encountered issues.
GENERAL
SUBJECTS
AND VIEWS
It is important to use these terms correctly and to classify elements correctly (to avoid confusion).

**Examples:**

* KPI is a concept, tool, metrics; not a method.
* UML and BPMN are languages, not SE methods or methodologies. It can be a method of representing something.
* Agile is a philosophy, not an SDLC or a methodology.
* SDLC is not a project management methodology.
* Waterfall is an SDLC, and as SDLC it is a methodological concept. But it is not a methodology. Some methodologies are based on the Waterfall SDLC.
* Process Analysis is a discipline, not a method.
The item or its usage can be adapted to suit a specific situation or to solve a specific issue. The decision can be made that “the best thing to do” for a specific issue or a particular situation is more important than to stay between the lines and follow the prescribed rules.

Always questioning: Wrong application/usage or unsuitable item? It is useless to blame the item for an inappropriate usage.
Quality of Application and Usage

Application of Theories, Principles, Concepts, Methodology, Methods, Systems, Tools, ...

- **Inventors**: Excellent, Deeper understanding
- **Early Adopters**: Genuine Application, Faithful to the original way, Often (real) experts, Trying to grasp the underlying ideas, reasons, ...
- **Majority**: Superficial understanding, Literal application, Either strict or very loose application (cargo cult), Watered-down versions, Oversimplifications, More assumptions, Personal interpretations, Preferences and cherry-picking, Individual adaptations, Irresponsible usage, Politics, Abuses, Copying (w/o understanding), Used by unqualified individuals, ...

30/10/2018
Some major risks can often be eliminated fairly quickly. But a lot of smaller risks (smaller impact) may remain for a while.

At a certain point, most new information will confirm the present understanding and little new insight is acquired or the new insight concerns details. This is the point where we can go on.

The arrival of new information invalidating major insights or having a major impact on the design can never be ruled out. But the likelihood that this happens becomes gradually smaller and smaller.
### Reality of Software Development

**MYTH, ILLUSION or HYPE**
- Easy
- Fast
- Cheap
- Highest Quality
- Linear

**REALITY**
- Complex
- Arduous job
- Slow
- Difficult and Risky
- Non-Linear

Small and simple software applications are easy, fast and cheap to develop. This is not the nature of most present-day systems.

Accepting reality and finding wise ways to deal with it

**Danger:** oversimplification, simple but wrong answers, short-cuts, ...

### Notes:
Novice may not understand the purpose, value and right use of decisions, techniques or tools. They may unwittingly take wrong decisions and short-cuts creating issues, increasing risks, limiting the solution, technical debts, ... This is even more true for laypersons.
If a customer can articulate what he wants, he is more likely to get it. What the customer wants is articulated in the demand, as well as his desires and preferences. This demand describes the needs and what is needed. Therefore, it expresses what is required. The customer’s problem or need is solved by building and implementing a solution which complies with his demand. Value has been created. It is by responding to the customer’s demand that his needs and expectations are met and customer’s satisfaction is achieved. The product will contribute positively to the overall organisation.

!! MANY ASSUMPTIONS !!
These are totally different concepts!!

AND the correspondence between these elements may never simply be assumed.

What is wanted is not necessarily what is needed. What is demanded doesn’t always reflect what is required to solve the problem or need. It doesn’t always allow to meet the expectations. Building accordingly to the demand doesn’t always create satisfaction. The envisioned solution doesn’t always increase the value for the company. And so on.
Customer Satisfaction

Customer satisfaction is important for a few reasons:

- Indication products and services are good
- Increased earnings
- Increased volume of business
- Increased reputation
- Expansion of customer base
- Facilitation of business development

Nature of Customer Satisfaction is

- Intangible
- Personal
- Variable - Temporary
- Vague
- Unexpressed or partially expressed

Delivering what the customer wants leads to Customer Satisfaction

Not necessarily !!

What a customer wants and what will satisfy a customer are two different questions.

Satisfying a customer does not necessarily lead to future decisions or actions from which the supplier can benefit.

If knowledge or insight is missing → the desire is incorrectly identified, still immature, unstable or not well expressed → no customer satisfaction

“The client doesn’t know what he wants”
Or “he knows what he wants, but it is not what he needs”
Customer Satisfaction

Freedom

Do and/or ask what we want
Aims to create satisfaction
May lead to unsolved and new problems
Creates dissatisfaction

Do what is necessary
Aims to solve problems
Not always what we want or expect
Solved problems create satisfaction

Example 1: Do-It-Yourself Store

Vendor 1: Sells the product the customer asked for, choses or wants. If the client buys the wrong product, it’s his/her problem. The client might be satisfied. If it doesn’t solve the problem, the purchase of the product was unnecessary and dissatisfaction will follow.

Vendor 2: Asks the customer what problem (s)he wants to solve with it and what the situation is. The vendor uses his/her knowledge to advice more appropriate products, better solutions and tips. The vendor may sell a different product than asked. Once the problem is well-solved, the client is satisfied about the solved problem AND about the delivered service.

Example 2: Diet and Lifestyle

Many people prefer to eat unhealthy food. Over time, they may start feeling in a bad shape and get sick. They will feel better or heal once they eat the right healthy food and switch to a healthy lifestyle (doing what is necessary, doing the right thing).

Example 3: Planet

We may like the modern life standards and try to maintain it. However, it is unhealthy and unsustainable. It is destructive for the planet. As long as we ignore to do what is necessary, reasonable and responsible, the effects will backfire and we may not like that.

More than the desired “solution”, people want the problem being solved without new problems or annoyances being created.

Doing what is necessary is important, doing what is wanted is supplemental.

Both are not necessarily mutually exclusive. Sometimes they can be combined.
Expresses Needs and Demands, Wants, Desires, Expectations

- Subjective: Based on an interpretation of the situation and ‘diagnosis’
- Often symptomatic (Consequences of problems are noticeable. The experience of the consequences are upsetting. They have to disappear.)
- Can be influenced by emotions; like impatience and frustration
- Different visions may exist (different persons may have different wants, expectations, desires, ...)
- Can be the result of a compromise (ends up loosely related to the real problem)
- Lack of alternatives (have they been investigated?)
- Hard to capture
- Can be misaligned or conflicting
- Based on uncertain facts (partial insight)
- Can easily be changed, even in a blink of an eye (while a system’s lifespan is several years)
- Can be unrealistic
- Demand may not be inline with intentions, wants, expectations, ...
- Vague and partly unexpressed (not all wants, intentions, reasons, desires, expectations, ...)
- Based on personal insight of the situation, limited perspective and scope
- Based on understanding of the possibilities and limitations of software systems, which may be a flawed
- Based on understanding of how to deal with information, which may be based on limited insight in this matter, false beliefs and assumptions
- No guarantee that it constitutes a solution
- No guarantee that it leads to added value for the company
- Can be desired, wanted and asked, and, once built, not used.
- May undermine the company (truly damaging)
Customer Satisfaction

Responsibility of the Customer in this line of thinking

- Posing the diagnosis: detection of problem, limitation, opportunity
- Determining the objective, intention, goal and scope
- Articulating the “needs” and/or “wants”:
  - Requirements
  - Conception of nature of the solution, the role and main functions, solution description in business terms
- Evaluating the solution to accept it or refuse it

= ANALYSIS!!

Misalignment between responsibility and competencies

Brief overview of required skills to pose a diagnosis and to conceive a solution (in information issues in corporate environments)

- Business knowledge and situational expertise
- Overall business and company knowledge
- Insight in the company’s systems and processes
- Profound understanding of “Information”
- Profound understanding of “Systems”
- Real Problem Solving skills
- Thinking like an Engineer
Some stakeholders may be satisfied, while others are not.

Stakeholders may be satisfied, while the ‘solution’ doesn’t solve the problem, create new ones, bring in risks and limits, is inefficient, is awkwardly designed or does not fully exploit possibilities.

The competencies of stakeholders is only one perspective to judge a solution. It is limited.

The satisfaction at the delivery (1) does not guarantee or imply a satisfaction during its operational lifetime (2)

Even if customer satisfaction is obtained at the delivery, the day after the customer may change his mind and not be satisfied anymore. Systems exists beyond customer’s satisfaction.

A system, a good system, is tangible. Customer satisfaction is intangible. Value?

Successes are too sporadic and won’t last. Impossible to meet on regular basis to create permanent satisfaction.

Focussing on satisfying a “delivery according to a demand” is way more easy than seeking to get value and satisfaction over the long term.

Seeking to have appropriate systems and solutions requires a different way of thinking, organisational setting and approach from what is required to respond to a pre-established demand.
CREATION OF CUSTOMER’S SATISFACTION

Customer’s wants → Customer’s demand → Building & Delivery → Customer is satisfied

Success

Successes are too sporadic and won’t last. Impossible to meet on regular basis to create permanent satisfaction.

Even if customer satisfaction is obtained at the delivery, the day after the customer may change his mind and not be satisfied anymore.

PROBLEM:

Vague, intangible, changeable, linked to emotions, different visions (no unity), unreliable, interpretation, alternatives, hard to capture, can be unrealistic, can be misaligned, uncertain, can be conflicting, local, short term, thinking small, personal, subjective, based on interpretation of reality...

Customer’s wants → Customer’s demand → Building & Delivery → NO Customer’s Satisfaction

UNRELIABLE MOVING TARGET

Slow process

Creates pressure in project

Often a huge waste of time

"If I had asked people what they wanted, they would have said faster horses."

- Henry Ford
Who is the Client?

WHO TO SERVE?

Is the User the client?

No. Software applications may facilitate the work of software users. They benefit from software application to do their job. However, they are part of the Information System. Their work needs to create value for someone else.

Is the Sponsor (sponsoring business stakeholder) the client?

Yes and no. Projects can be funded with the budget of a business unit, department or service. Would it be different if the money came from a budget from the top of the company or from the IT department itself? The money, wherever it comes from, belongs to the company and the company can fund projects in a different ways. The product of the project should benefit to his/her organisational unit. Information system should belong to the company, particularly if they or their data is shared across the organisation. A sponsor is rather a key stakeholder than a client.

WHO ARE THE REAL CLIENTS?

1. EXTERNAL CLIENT: The client outside the company to whom the company delivers products and services. The real client can be a person, or organisational unit, member of the organisation.

1. THE COMPANY: The company, as the overall entity, is the second real client. Its primary role, its main mission, is to serve society and therefore to thrive.

2. INTERNAL CLIENT: Higher organisational units defining the purpose, values, directions, strategy and higher goals for the business are also clients.

The client is not simply the one who asks or fund.
Main Customers =
• ENTERPRISE (company’s body)
• COMPANY
• COMPANY’s CUSTOMERS
• SOCIETY

If the system doesn’t function properly, it will hinder or limit the activities at best, preventing from reaching the objectives. It may lead to bad products and services for the external clients.

Notes:
A taxi company can’t thrive if the car (taxi = system) is old, dirty, worn out, regularly broken, ...
Value Creation (1)

Value is defined by the final net positive contribution

The value for final (external) clients can be estimated by interest, sales, price ready to be paid, ...

Evaluating the “value” for the company requires a deep understanding of all the effects an item or solution has on the company.

Major Areas of Value Creation

Enterprise (System)  Business (Activities)  Services and products (Results)

- Competencies
- Information Exploitation
Value Creation (2)

Ideas with little solid ground
FALSELY affecting the estimation of value of a system

The product/solution/feature has value because:

• it is the goal and mission of the project
• it is the delivery of what is agreed on
• the product correspond with the demand
• the software application works as expected, looks nice, is attractive, ...
• it is our product, the project’s product, ...
• a lot of effort spent building the product
• building it or acquiring it costed a lot
• the company has now new features, new processes, new software
• the system is adapted or expanded. Features have been added.
• It can do more.
• the new is better than the old, the old was outdated
• the product is desired
• the product or new solution facilitates the job of the end-user
• the end-users, stakeholders, sponsor or business community are satisfied
• the product is useful to the company
• our people, the project team members, are competent
• everybody implements it these days
• it has been advised by solution providers
• the builder is proud of his work
• ...

30/10/2018
Value Creation (3)

Possible Obvious Effects

- New functionalities
- Information needs being solved
- Additional Capabilities
- Increased Capacity
- Improving the products and services to the final clients
- Improved information exploitation
- Innovation
- Better KPI results (like reduced throughput time)
- Better alignment with the environment
- Lowered cost
- Reduced waste
- New competencies
- ...

More Possible Hidden Effects

- Increased maintenance and/or operational cost
- Increased fragmentation
- Inefficiencies have been introduced or shifted to elsewhere
- Incoherencies, conflicting logic and bugs,
- New weaknesses, bottlenecks, SPOF’s, risks have been introduced
- New hindrances, difficulties and obstacles in usage
- New hindrances and limitations (impossibilities, closing doors) in evolvability
- Increased waste
- Solving consequences, masking the causes (real problem)
- Increased complexity
- Loss of knowledge and comprehension
- Decreased manageability
- Hidden mechanisms undermining the system from within
- Risks of misuse and abuse
- ...
We can’t judge a gift based on its wrapping paper.

A solution or system can’t be judged on superficial facts, impressions or unfounded criteria.

**Main Aspects to Value of System** (non-exhaustive)

- Purpose is ethical, useful and accepted
- Fulfilling its role swiftly and with sufficient capacity
- Solves root causes or satisfies true needs
- Producing benefits and minimizing costs
- Design intent in line with purpose
- Elegantly designed
- Contributes positively to the supra-system
- Respects supra-system
- Respects the environments
- Respecting harmony in the system and in the supra-system
- Respects system’s qualities like manageability, evolvability, ...
- Acceptance by people
- No degradation mechanisms or effects
Reflections

What are the consequences of these differences?
Can we simply copy (w/o adaptation) work done in the human world to the computer world?
Can we design computerised solutions by not taking the characteristics (strengths, weaknesses, possibilities, limits, risks, ...) of computers and software into account? Without a thorough understanding of the specificities of this world?
Do we need a different way of thinking when designing computerised solutions?

Notes

Computers may be equipped with logic simulating human qualities, like rules system, fuzzy logic, back tracking, Artificial Intelligence (AI). But most systems aren’t that far yet.

An IS has a “human part” and a “computer/software” part. Both have to collaborate. Both have to be engineered to form a single system and to work in harmony.
Over time, what human activities and business community will be left over after large scale mechanisation and informatisation?

Note: Industrial component is optional. Not all organisations are industrial.
Evolution of Automation

Automating the “As Is”

“Copy-paste” of work (or work is slightly adapted) from people environment to computer environments. Basically, a same way of thinking is applied. Mainly repetitive work is transferred. It values the work of the people. With this transfer people can do more and other work. Productivity increases.

Exploitation of the ability of computers and software

Maximally exploiting computers and software is achieved by designing systems, not based on human abilities, but based on the specific abilities of computers and software. Work of computers is valued. People do work computers can’t (yet). Requires a way of thinking different than thinking in terms of people and work performed by people.

Notes
Aiming for higher goals leads to greater achievements

More difficult to achieve – more demanding
- Requires different roles, focus, approach, methods, techniques and tools
- Requires higher norms
- Requires stronger competencies
- Requires different beliefs
- Requires a more global and holistic view
- Much more thinking

Focusing on lower goal only

Assumption:
If all sub-goals are met, then the higher goal is achieved.

More local thinking
Lower norms may be used
Little pressure to improve our skills

- Activities contributing to the lower goal (sub-goal) may unwittingly work against the higher goal(s).
- Risk: Activities that don’t contribute to the sub-goal may unintentionally hinder reaching the higher goal even more.
INNOVATION

1. Innovative software features and interface
2. Innovative and more powerful algorithms
3. Innovative concepts
4. Innovative system architecture
5. Innovative processes
6. Innovative information usages
7. Innovative information products & services
8. Innovation supporting new forms of organisation
9. Innovation in learning and capability development
10. Innovation contributing to business models
11. Innovation allowing new or better strategies

Reflection:
Why is innovation in 1 and 2 easier?
How often does the IT department come up with innovation on points 3 or higher? If the answer is “rarely” of “never”, then why doesn’t it happen frequently?
Should this happen frequently? How can this be changed? What conditions are missing?
INNOVATION

Innovation doesn’t start with a vision or good idea. It starts with an attitude, curiosity and learning, an open mind and an appropriate environment.

Innovation needs …

• Goal, direction, a problem
• Context
• Constraints
• The right mind-set (curious, investigative, what-if, ...)

Innovative Environment

• Structure, guidelines (but not strict structures, unbreakable laws, ...)
• No pressure or stress, peaceful environment
• No fear
• Failure is allowed. Failure is learning.
• Some freedom
• Some autonomy
• Time, Resources
• No strict plan, deadline (pressure)
• Innovation often happens at the crossroads of two or more domains (or concepts) (overlap of domains)

Revolutionary Idea – Three Phases

1. Ridicule
2. Dangerous
3. Obvious
Innovation Inhibitors

- Authoritarian hierarchical structure (**STRICT** rules and structures)
- Lack of support
- Fearing risks and failure
  Control (or “transparency”) linked to judgment, evaluation and potential punishments, fear of job loss (restructuring, rightsizing, outsourcing, ...)
- Pressure
- Internal politics, personal agenda’s, ego
- Obligation, procedures to follow, rules
- Chaos, firefighting
- Frequent interruptions, one-hour time sliced day
- Divide and conquer as strategy to manage the company
- Organisational silo’s
- Job silo’s
  the job description, the function, the authority and responsibility define what activities one may execute, what the individual may think about or to what activities (s)he may participate.
- Knowledge silo’s, job specialisation
- Superficial knowledge, lack of insight
- Short-term thinking, cost-thinking
- Low norms
- Attitude of maintaining and protecting; a defensive attitude
- Dependent thinking, group thinking, single thought, mainstream thinking
- Rejecting and banning disagreements, different thoughts, ...

"If I had asked people what they wanted, they would have said faster horses."
- Henry Ford
Change as Investment

A Change is an Investment

First the effort, cost, issues, ... the benefits follows.

Some drawbacks (like bugs) may appear much later, when the change is implemented and operational.

### Downsides

- Effort
- While experts implement one change, they don’t work on other developments/changes
- Risks of introducing incoherencies, conflicts, inefficiencies, ...
- Risk of confusions
- Additional stress
- Requires training
- Cost
- ...

### Benefits

- Learning, Improved skills, new experience
- Improved system, situation, ...
- Create additional value
- Retain or bring customers
- ...

Investment + cost < income

Development

Release

Payback Period

Break-even Point (BEP)

profitable

time

376
Change

How successive changes destroys the profitability

First build  1st change  2nd change

Every piece of source code has passed through a development process

Dev. Process

Development  Release  Payback Period  Break-even Point (BEP)  time

System

Σ Cost(Change) < Σ Benefit(Change, time period) (simplified !!)

Some Principle

- Every change is work, an effort.
- Every change implies waste (old work thrown away).
- Every change uses resources (analysts, engineers, developers) which can’t be allocated to other work.
- Every change is a cost.
- Every change has a payback time.
- Every change influences the profitability of the entire system, first negatively (increased cost), later maybe positively (increased profit). A change is not necessarily beneficial simply because it is a change.
Real Changes

SOMETHING changed in the “Company’s Environment” or in the Business part of the company.

“Fake Changes” – Corrections – ANALYSIS BUGS

NOTHING changed in the “Company’s Environment” or in the Business part of the company, yet the system has to be adapted. They result from bad decisions and choices. Some main causes are a lack of insight, thinking or skills.

SYSTEMS ANALYSIS PREVENTS FAKE CHANGES

Some fake changes are acceptable (adaptation of strategy, new idea, investigation led to new insight). Even a decent Analysis can not prevent all fake changes.

TIP:
If too many changes are due to lack of information, lack of insight, lack of thinking, lack of skills or other similar causes, then solve the causes.
To foresee:
- Understand systems (general systems theory)
- Understand evolution of company’s systems
- Understand evolution of business
- Consult plans
- Look at objectives, usage of systems, complaints, intentions

Some internal changes are easier to foresee or are under control of the company itself.
If the management functions in an ad hoc and fire fighting mode, then they are taking a lot of decisions that truly couldn’t be foreseen.
Change

Evolution of the Environment

Superficial Environmental Changes
- Times scale: a day, a week, a few months, a year
- appear and disappear quickly
- lack of direction
- easily noticeably (newspapers, magazines, advertisements, ...)
- superficial, minor, small opportunities
- responds doesn't take a great effort
- pick the battles

Medium Environmental Changes
- Times scale: several months to several years
- Harder to notice (visible tendencies in society)
- Some tendency and direction can be observed
- Can be worth setting a project or programme for it
- Harder to escape from it

Profound Environmental Changes
- Time scale: many years to several decades or more
- Hard to notice, requires to consider the long term, knowledge of the past (history)
- Clear direction
- Obligatory to follow it
Resistance of change is bad because change is inevitable. The resistance must be broken. Those who resists fear the future. They don’t understand or they can’t adapt. They are nostalgic and can’t let go the past. We live in modern times and need to progress. Such unfortunate clichés are well-known.

Resistance is natural and is inherent to change. It is healthy to have resistance. Resistance is there for a reason. It is a signal that must be captured. Its message has to be understood. First, resistance prevents us from accepting and responding to every opportunity of change. It tells us something about what and how we do things. If we would do things correctly, then why would there be a resistance?

Resistance exists

• Because of a loss for a concerned party
  Is the loss justified? Can this loss be compensated?
• Because of fear
  Fear is or should be abnormal. What is that fear? What created it? How do we deal with people? Can we reassure people? Or take measures to resolve the fear?
• Because of uncertainty
  Change means uncertainty. And uncertainty is close to fear. But it is important to listen to what these uncertainties are. Maybe we created them and maybe they can be resolved by listening, by providing more and clearer information and by taking specific measures.
• Because there is disagreement with the decision
  A disagreement can be justified. If there is a disagreement, then maybe something has been overlooked.
• Because a lack of confidence and belief
  This should also be investigated and measures can be taken.
• and indeed because of change requires an effort and not all persons are ready to do the effort.

In some cases, resistance is created by lack of communication, by a lack of information: what are the intentions? What do we know and what don’t we know yet? How much do we care about the people? How will we deal with people during the transition?

Key concepts: right intentions - information and communication – empathy and caring – respect – support
The design of what layer of logic should be based on what evolutionary wave? (superficial, medium or profound)

<table>
<thead>
<tr>
<th>Type of Environmental Change / Evolution / Opportunity</th>
<th>Profound, Long Term</th>
<th>Medium</th>
<th>Small, Superficial</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peripheral logic and features</td>
<td>☑️ ☑️</td>
<td>☑️</td>
<td>☑️ ?</td>
</tr>
<tr>
<td>Outer-core Layer of Logic</td>
<td>☑️ ☑️</td>
<td>☑️</td>
<td>☘️</td>
</tr>
<tr>
<td>Core Logic &amp; Architecture</td>
<td>☑️ ☑️</td>
<td>☑️ ?</td>
<td>☘️</td>
</tr>
</tbody>
</table>

The design of the core, shouldn’t be based on superficial changes. It should be based on the more stable deeper, long term evolution. It is important to understand if a piece of logic is fundamental (long term) or if it is likely to change every few months, so to speak.

Size of change, its criticality and its profitability have also to be considered in the light of the type of environmental change (evolution).
1) Evolution of speed

**Line A**: Increase of speed is not limitless. Likely companies may meet a plateau, a maximal speed (blue area 1). The increase of speed of change creates pressure.

**Line C**: Fast-Cheap-Local-Tangible (short-cuts, quick wins, technical debts, reactive initiatives under pressure). Wanting to go even faster may lead to an overshoot. Pressure and speed may lead lack of quality and various kinds of problems. It’s a kind of “speeding ticket”. Understanding and awareness of the consequences of speed may lead to a downwards regulation of the speed and pressure and an increment of cost and risks in the medium or long term. We may not be aware of these cost and risk increment or fail to link them to the speed and pressure. They are considered as normal, while they are and should be avoided.

2) Different pace of change

A change in a company often requires many components and aspects to be changed. Not all components in a company can change at the same speed (**Line A** versus **Line B**). Often, this difference in pace of change puts a pressure on the components with a slower pace of change (**green area 2**). They slow down the whole change. This may also put too excessive pace of change and pressure in some areas which lead to lack of quality and other kinds of problems.
1) **Top-Down**: Guidance based on plans. Long Term. Defines the overall structure and main functions, pro-active

“THINK BIG, ACT SMALL”

2) **Bottom-up**: Solving local problems, filling gaps, fine-tuning, removing local obstacles, short term, local needs, reactive

3) **Bottom-up tackled top-down**: e.g. emerging strategies becoming overall strategy; groups of local issues requiring top-down approach in order to ensure alignment and integration with top-down (avoiding opposite evolutions)

4) **Engineering-driven and Technology-driven changes**: Engineering-driven: through continuous Analysis, new concepts, ideas, optimisations, and other improvements and innovations

**Notes**

Environmental scans may detect changes, obligations and opportunities in the environment. These may trigger one of these flows.

“Think small, act small” = short quick wins; no fundamental changes; creates chaos.
Dealing with Unpredictability

Unpredictability is inherent to software development projects ... but ...

Reducing the Unpredictability

- **Analysis aims to reduce** the unpredictability and limit the issues
  - Learning and understanding
    - Business domain knowledge, Business expertise
    - Business environment
  - Right diagnosis done early
- Study of plans, objectives, intentions and expectations
- Verify, don’t assume
- Understanding of people
- Understanding the medium and long term evolutions (history, deeper underlying currents)
- Think, imagine, what-if, ...
- Leading, creating / shaping the future, having a vision and goals

Ability to deal with unexpected changes

- Vigilance to detect conflicts, issues and opportunities early
- Be flexible and reasonable
- Increase flexibility through
  - A well-designed Architecture
  - Component Based Development
- Adapt plans

Before accepting the ‘Change’

- Investigate the cause and nature of the change
- Could the change have been detected through Analysis or through some more thinking?
- Is it worth (cost / benefit) (effort/benefit) ?
- Refused or accepted? Alternative solutions?
- Can different changes be regrouped?
- Do the different changes have something in common?
- Should the change be implemented now or later?
- Can the present solution be prepared for a future implementation? Or is a partial and phased implementation possible?
Chaos vs Structure (1)

We live in a Chaotic World

The world is full of chaos. **However ...**

There are countries, cities, road infrastructure, energy networks, companies, traffic rules, law systems, laws of physics, laws of biology and chemistry, thinking patterns, languages, planetary systems, ecological systems, economical system, software systems, and so on...

We live in a Structured World

Allows us to
- Understand the world
- Deal with the world
- To anticipate
- Shape the world
- To function more easily
- To collaborate
- To communicate
- To evolve
- To invent and innovate
- ...

**Notes:**
Note that the product of innovation is always a system, method or technology (something structured), something to get a predictable result.
Chaos vs Structure (2)

Purposefulness
   Intention to shape the future
   Limited resources

Competition
   Investments
   ROI

create needs of

Guidance
   Control
   Effectiveness
   Efficiency
   Speed
   Focus

create needs of

Manageability
   Measurability
   Controllability
   Stability
   Predictability
   Clarity
   Evolvability
   Harmony
   Anticipation

create needs of

Rules
   Method
   Structure
   Models
Chaos vs Structure (3)

**Objectives**
- Planning
- Predictability
- Investment in future

**Management**
- Top Down
- Bottom-Up

**Visionary**
- Shaping reality
- Leading
- Purposely
- Planning
- Faster
- Top Down

**Predictability**
- Control
- Management
- Ability to build foundation
- Anticipation

**Uncertainty**
- Change
- Reactive
- Concrete

**Quicker result and benefits on small issues**
- More in touch with real needs

**WEAKNESSES**
- Lack of higher purpose
- Lack of direction
- Short term-driven
- Foresightless
- Moving target
- Reactive; lagging; no leading
- Following
- Constantly changing
- Lots of unsuccessful attempts and failures
- No predictability
- Inefficient
- Slow
- Costly
- Evaluating real progress in long term evolution
- Responsibility?
- Manageability?
- Security? Other required system’s characteristics?
- No long term expectations
- How to get investments?

**RISKS**
- Assumed predictability
- Ignoring uncertainty
- Overly rigidity
- Losing touch with real needs, issues, ..
- Bureaucracy over progress and results
- Overly control
- Sticking to plans and to models

**WEAKNESSES**
- Harder to stay in touch with reality
- Harder to guide and plan in tune with reality
- Slow to detect opportunities
- Difficult to detect changes and to adapt
- Difficult to get balance between productive activities and control and managerial activities

**RISKS**
- Are the emerging system, emerging architecture and emerging characteristics good or bad?
- May lead to inefficient systems
- May lead to overly complex systems

**DANGER ZONE**
- No delivery
- No ROI
- Chaos
- Standstill

**DANGER ZONE**
- Death March
- Unsuitable product or system

**WEAKNESSES**
- Quicker result and benefits on small issues
- More in touch with real needs

**WEAKNESSES**
- Harder to stay in touch with reality
- Harder to guide and plan in tune with reality
- Slow to detect opportunities
- Difficult to detect changes and to adapt
- Difficult to get balance between productive activities and control and managerial activities

**RISKS**
- Assumed predictability
- Ignoring uncertainty
- Overly rigidity
- Losing touch with real needs, issues, ..
- Bureaucracy over progress and results
- Overly control
- Sticking to plans and to models
Chaos vs Structure (4)

<table>
<thead>
<tr>
<th>Chaos</th>
<th>Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Difficult to grasp</td>
<td>Easier to understand, Clearer</td>
</tr>
<tr>
<td>Hard to maintain systems</td>
<td>Easier to maintain</td>
</tr>
<tr>
<td>Hard to manage systems</td>
<td>Easier to manage</td>
</tr>
<tr>
<td>Hard to measure (except I/O)</td>
<td>Easier to measure</td>
</tr>
<tr>
<td>Hard to optimise systems or their environment</td>
<td>Easier to optimise</td>
</tr>
<tr>
<td>Difficult to change</td>
<td>Easier to change</td>
</tr>
<tr>
<td>Risk of side-effects</td>
<td>Smaller risk for side-effects</td>
</tr>
<tr>
<td>Hard to react in case of problems</td>
<td>Easier to react</td>
</tr>
<tr>
<td>Lesser likely to be scalable</td>
<td>More likely to be scalable</td>
</tr>
<tr>
<td>Confuses the mind</td>
<td>Guides the mind (thinking)</td>
</tr>
<tr>
<td>Steep learning path</td>
<td>Smoother learning path</td>
</tr>
</tbody>
</table>

**STRUCTURE IS IMPORTANT**

A structure means something organised and defined. It doesn’t mean “fixed”, “unalterable”, “set”.

A structured and organised environment supports the normal activities. We don’t need to think too much about it. The focus can then be directed to creative activities.

Example: We can rely on public transport system (structure) to commute daily to work. If we had to spend two times a day to find out how to get to the job or to home, we would waste a lot of time and energy. This time and energy can now be spent to more creative activities. Similarly, some habits allow us to save time and energy.

We can play follow the structure or (temporarily) stepping out of the structure. A structure can be adapted when needed.

A structure (rules or any other formalised element) doesn’t allow to deal with a specific situation. The structure doesn’t allow this. We stop following it and do what is necessary, while staying as close to the structure. Once the job is done, once the situation allows it, the structure is followed again.
Chaos vs Structure (5)

The world is partly structured.

The world is partly chaotic.

A company needs to be able to deal with both parts

What we don’t want: Chaos

→ Confusion, steep learning path, unpredictability, how to produce desired results in a chaotic environment?, difficult to understand and to control, inefficiencies, risks, costly, difficult to adapt, paralysis, ...

How To Solve this? Structure and method

Structure and method organise, bring order, provide clarity, allow efficiency, effectiveness and provide structural flexibility, allows to deal with a higher degree of complexity

Need for manageability, efficiency, effectiveness, control, clarity  
Need for ability to change, flexibility, adaptability, reactivity

Conflicting?
Chaos vs Structure (6)

Imprisonment in structure and method by
• faithfully respecting structure
• following methodologies and standards
• applying principles systematically and strictly

Chaos through lack of structure and method

A more flexible way to deal with structures, procedures, rules, ...

Systems, structures, procedures, processes, rules are put in place to increase the efficiency and effectiveness. That’s their purpose, their goal.

They should not hinder work !!

1) Implement structures (processes, ...) containing alternatives and “free area’s”
   The executioners can decide how they work within predefined free area’s, boundaries, .... (example: case based processes, methodologies, ...)

2) Following the structure, processes, .. in a more flexible way
   A situation which can’t be solved by following the predetermined pattern (system, structure, ...). Either the pattern is adapted, be it temporarily and locally, or the particular situation is dealt with outside the predefined pattern. The executioner stays as close as possible to the predefined pattern and follows it again as soon as possible.

3) Implement structures suitable for adaptation
   The more modular the structures, systems, processes are, the easier it is to adapt them.

4) Capability of Vigilance
   Mechanisms offering Vigilance can be implemented in systems, processes, ... This is the implementation of monitoring mechanisms, controls, functions of environmental scanning, mechanisms to collect complaints, ideas, issues, opportunities, innovation, ...

5) Adapt systems, structures, processes, procedures, plans
   They should improve and evolve over time.

Notes:
Systems, structures, processes and other guiding structures make sense, are valuable and are meant to help. It makes sense to follow them as long as they are suitable and to handle in a different way when necessary. Getting a result is more important than blindly follow predefined patterns.
Predefined structures are often considered as

- static
- unalterable
- dictating
- rigid

This assumption is often a matter of perception, interpretation, preconceived ideas and maybe a few other reasons.

They are very useful and offer a great flexibility, freedom and facilitate the functioning of the mind. Our society is based on systems, structures, processes, and so on.

A lot depends on:

- How they are designed
- What higher authorities impose as mandatory
- How the structures are used (or misused)

We shouldn’t blame the idea of “structure” because of a wrong design, application or use.
A right structure is needed. It organises and clarifies a way of working. We don’t need to worry about solutions already provided by the structure. The structure forms a foundation.

But the structure shouldn’t be strict. We have to be able to decide that, to resolve a specific situation, the best way is to temporarily and to a minimal degree ignore the structure. Once, this necessity is over, we fall back again on the structure.

A structure may have periods of freedom and ‘unstructured’ areas, domains, zones, … of built in.
IS development is a slow and arduous job. It requires a maximal stability. But business and the business environment is not static. Both needs have to be taken into account.

**Changes vs Stability**

**Fixed Target**
- Stability required for IS development
- Easy for “IT”, but utopic

**Erratically Moving Target**
- Adapting to changes in business
- Impossible for IT

**Solution:**
- Choosing target that is more/most stable
- Reduce movement of target
- Vigilance – shortening the path between target, movement, changes, issues detection and reaction
- Anticipate movements, changes, needs, …
- Offering flexibility
Variability of the Target

Project lifespan: Several months or even years
Duration of Customer Satisfaction: can be extremely short (individual & subjective)
Lifespan of a software system: 10? 15? 20 years?

Primary Target?
Secondary Target?

Moving Target

More Variable
Lower-level Business Goals
Local Needs
End-users needs
Stakeholder’s wishes
Stakeholder’s satisfaction
Stakeholder’s requests

Global Business Goals
Global Business Needs

More Stable
Company’s needs
Systems characteristics
Long term goals, mission, vision,
...
Situational requirements

Value of Customer Satisfaction vs Value of a good system
(Appropriate, well-designed, ...)
Control

Control is a VERIFICATION

Purpose (Why do we control?)

- **Guidance**, readiness for further progress, course correction, adaptation
- Evaluation, validation and learning
- Checking for issues and preventing issues.

Control is necessary to increase certainty, to ensure, to learn, to guide, to correct or adapt, to align, to take new decisions and to prevent problems, misuses, aberrations, discrepancies, deviations and regressions. It is necessary for decision making and to keep things on track.

**Notes**

Control and control mechanisms require effort and time. They cost.

Problematic uses (misuse)

- overly guidance (usually at the cost of other important things); to pilot something that doesn’t need or want to be piloted
- Control as verification if we stick to plans, respect rules, as a way to impose rigidity
- tool to exercise and increase power over something
- freedom limiter (where freedom is required)
- creativity killer (imposing own ideas, decisions, ...)

This might be the consequence of the attitude of the controller and the decisions and actions following the control. Control is then misused in an inappropriately exercised role or in a unhealthy relationship.
Need for Flexibility

Flexibility is a commonly valued quality.

“Flexibility” doesn’t mean anything

- Flexibility in the process allows course corrections. It doesn’t guarantee a flexible product.
- Flexibility offered by internal organisation of the product facilitate the evolution of the product.
- Flexibility in use of the product allow end-users to approach a task from different angles and allow them to do more in different ways.

We need to know WHAT FLEXIBILITY and WHAT FOR is desirable, and then take the advantages and drawbacks into account.

Warning: If used to correct hasty decisions due to lack of understanding and/or sloppy thinking or bad compromises, then it is misused to solve consequences, not causes. The cause will remain and will continue to create troubles.

With flexibility comes freedom and freedom requires responsibility and competencies. The mistakes made by abusing of flexibility and freedom leads to chaos.

In general, we are interested in flexibility offered by the product’s internals. This allows

- to adapt to unforeseen circumstances
- to adapt the course of a strategy when needed
- to implement new ideas
- to more quickly seize opportunities

Notes:
If a lot of flexibility of software development process is demanded, it is a symptom of lack of mastery in the discipline; lack of understanding in the environment, system and mission; bad guiding and so on. Flexibility should, in essence allow to respond to unforeseen issues, not to alleviate lack of competencies, a bad approach or any other similar type of issue.
Validation

Confirmation that something is right, that the project is still on the right track. Validated elements has been investigated, analysed, verified and found correct. They are ready for further use. They should be stable. They are not meant to change anymore. Only, in exceptional circumstances like unforeseen events, they still can change if beneficial to the project. Change means also a delay, a cost and a risk.

Freezing

Putting yourself or the project into a straightjacket until the end of the project.
Proactivity (1)

Proactivity is NOT about building features which are not asked for, which are unnecessary, just in case they might be needed.

**Reactive approach** is acting in response to an issue that has already become a problem, responding to a demand.

Process execution to conceive, build and implement a solution

Problem happens, need is detected
Problem or need exists during this period. This means a damage, risk, cost
Problem or need is solved

**How to limit these costs and risks?**

**Proactive approach** is to respond to potential problems and to future needs before they take place.

**GOAL:**

Benefits of profitable measures

> Loss caused by unfortunate measures

Result of right proactivity

Result of failed proactivity
Proactivity (2)

1) Investigate, analyse

2) Identify

3) Forecast

4) Evaluate

5) Think of options

6) Evaluate the options

7) Decide

Impact

Probability

Criticality

Effort, time, cost of development and of dealing with impact (if there is an impact)

- Resolving / implementing
- Work-arounds
- Preparation work (already conceiving a solution but not implementing; adapting the design w/o for a future solution; implementing controls or barriers to avoid impact; ...)
- Attention (vigilance, monitoring)
- Doing nothing

Proactivity is a conscious decision and an intentional action taken for each individual issue or opportunity.

Proactivity can not be successful without understanding the systems, the environments, the involved domains, the evolution of the company and without empathy. It requires mastery of the systems analysis discipline as well.

Notes:
This formal flowchart is often reduced to a few questions like: “Will this be useful?”, “Does it cost a lot of effort?”, “Is it good to implement it now?”, “What is the best way to deal with it now and later?”
Proactivity (3)

**REACTIVE ATTITUDE / APPROACH**

- Allowing the environment and events to control the course of actions; following the events as they happen
- Focus is on problems
- Being overwhelmed by problems, large back log.
- Under permanent pressure
- Not feeling in control
- Lacking of foresight
- Being unprepared
- Increased uncertainty
- Forced by circumstances to take decisions and to act
- Dissatisfied customers. Frustration. (experiencing many problems, waiting before they got solved, working under pressure \(\rightarrow\) lower quality, ...)

**PROACTIVE ATTITUDE / APPROACH**

- Avoids or diminishes future problems and firefighting
- Requires being truly competent
- Focus is on what can be changed and improved.
- Driven by higher values (get the most out of the environment)
- Define the course of actions
- Not just seeking to solve problems presenting themselves, but actively study the situation and look for opportunities.
- Much more planned progress.
- Increased grasp on future and evolution.
From Problematic to Innovating IT

1. Problematic
   Enterprise Information Solution

2. Misalignment

3. Alignment

4. Driving existing business

5. Innovation

- Services
- Working methods
- Organisational structure
- Capability acquisition
- Business Model
- Partnerships

Increase business volume, market share, profit margin, quality of service, reliability, reputation, ...

Simply responding to present needs. Can be a normal and limited delay. Business is fairly satisfied.

Not responding to the present needs. The business community can work. Existing IT solutions support some business activities.

IT solutions are more detrimental to the business than it delivers benefits.

About sizeable innovation in business (not about small innovations with little impact)

Strong & independent IT department

Pulling Upwards

Weak & dependent IT department

Dragging Downwards

30/10/2018

402
TCO – Total Cost of Ownership

SET-UP
- Purchase research & purchase process
- Hardware acquisition
- Off-the-shelf software acquisition
- Licenses, warranties
- Customisation
- Software development project
- Data migration
- Testing
- Set-up and deployment
- Training & documentation
- Development or configuration of interfaces to other systems

OPERATIONS
- IT Personnel
- Energy
- Audit & Testing
- Administration & Maintenance
- Support
- Security
- Insurance
- Upgrades and patches
- Training
- Change management cost
- Changes and enhancements
- Downtime, outages and failures
- Disaster recovery
- Depreciation
- Data centre and Infrastructure (acquisition, installation, configuration, management, maintenance, upgrades, replacement, ..)

LONG TERM
- Scalability expenses
- Replacements

DECOMMISSIONING
- Data export / migration
- Dealing with the archived data

- A 'solution' may not solve the problems or may solve the symptoms and create more chaos.
- A system may create problems. Cost for resolving these new problems.
- A solution may not be optimal. A better solution could have lowered the cost or bring more benefits.
- A sub-optimal solution may put barriers, obstacles, limitations.
MODELS

A MODEL

• is an abstractions
• is perspective
• represents a part of a world, real or imagined
• represents a concept, and idea

All models are incomplete (thus wrong).

However models are or have to be useful

• to master complexity
• to support thinking
• to facilitate communication

A MODEL is Worth a Thousand Words
SA/SD Artefacts vs Documentation

SA/SD Artefacts
Supporting the **thinking**
Supporting the **conception**
Supporting the **communication**
among team members and stakeholders
Meant to facilitate the building process
“what has to be built”
For Analysts, Architects, Engineers, Developers, ...

Documentation of what has been built.
Produced often late in the development
“MANUALS” meant for users: installation, deployment, maintenance, administration, usage, limits, prerequisites, capabilities, ...

System Documentation

Serves as “input” in next engineering round.
(else you need to re-describe the “as-is” every time again. Former to-be becomes new as-is. No need to re-describe it.)
Documenting Differences

- Documents grouping a set of changes: Easier, smaller documents
- To understand the present system, one need to read the artefacts of the original systems, then chronologically all the artefacts describing the successive set of changes. → reading a lot of unnecessary things which aren’t true today anymore = waste & confusing
- No clear view of the entire system! → Hindering understanding → RISKS!

GOOD PRACTICE:
Document changes in the current system documentation. This changed document becomes the new current system’s documents.
Limited View

The world we consider is often constrained within a defined zone. It often stops at some abstract or concrete borders (example SIPOC). We may consider a process until the perimeter of our own department, until the product is delivered to the client, until the information is delivered to the other organisation. Our mind thinks within the perimeter defined by a scope, a mission, a problem statement, a demand, a deadline, a month, a year or other limited period, a concept of action-reaction, our house, our street, our city, ... our live. Our mind often limits our thoughts to a zone or timespan. Our thinking is not integrated into the larger world.

We consider our job and assume what we do integrates well into the larger picture, or, someone else will take care of it. Failing to consider and think outside our perimeter causes a huge amount of issues.
Natural Design - Intelligent Design

Evolution by Natural Design

- Purposeless
- Foresightless
- Randomness, directionless
- Slow
- Extremely costly (Waste on the many failed trials)
- Produces organism suitable to the environment
- but in the end ... it is brilliant
- Happens by many clueless minds in a very huge amount cycles (iterations)

Intelligent Design

- Purposely (goal oriented)
- Top-down
- More foresighted
- Directed search
- Relatively fast
- Governed by cost considerations
- Beautiful designs is done by great minds

Notes:

Today, we can simulate evolution in computers (huge amounts of iterations are possible; genetic algorithms, deep learning)

Intelligent design matches better the world of the company. In specific cases, natural design can be applied, or as a combination of both.

Additionally, we can use natural evolution by getting inspiration from nature (biomimetics or biomimicry). It all still requires insight and intelligence.
INFORMATION
EXPLOITATION
Information is power. Information has value.

Information is key in decision making, choices, inventions, ...

A company without information stops functioning immediately.

Information

- supports the functioning of the company
- supports the management of the company
- as product or service
- is knowledge / understanding
- is key to innovation

3 Ways to exploit information

30/10/2018

How to maximally exploit Information?

Information must be taken very seriously
Information Need

- Information needs can **only be solved by** having the information when needed and where needed.
  - Information will only fulfil needs **if it has certain qualities** (identifiable, linked (relations), reliable, complete, precise, consistent, up-to-date, accuracy, accessible, process-able, timely, ...)

- **User dependency** (needs depend of **user and user group** (function, service, department, ...))

- **Situation dependence** (needs may vary according to the situation)

- **Time** dependent (needs may vary over time)

- **Stability**
  - Information needs of the company → more permanent → more stable
  - Information needs of people → more temporary → more changes

- Some needs remain **undetected** or **badly understood** (not always easy to detect and understand real needs – symptoms of information issues)

**ANALYST REQUIRED !!**

**Notes**

Important slide! Should actually be one of the first in the course. It’s all about “information needs”, not about “technology needs”!! Technology are only means to be better able to cope with information needs.
Why Keeping Information

A) Need’s Perspective

• **Company’s Information Need**
  - Required for the company to function
  - Independent of what people wants, desire or decide
  - Ability to respond to usual information needs and to more unpredictable information needs
  - If considered broadly, rather stable, may change over longer period or a change is announced (decisions, plans, ...)

• **People Information Need**
  - Ad hoc needs:
    - required by people to reach one-off goals
    - to perform, unique tasks
    - to increase their insight
  - Lesser predictable

B) Value Perspective

• Some information, through its meaning, has intrinsic value
  - Potential usage
  - Curiosity
  - Interest
• To be estimated: criticality, likelihood, potential purpose, opportunity, value of decisions or actions made possible by the information
• Possibility to learn:
  about the organisation, its environment, its functioning, its results necessary for management, improvement, innovation, ...
• Possibility to sell as product or service
Information Needs

<table>
<thead>
<tr>
<th>Management Information Needs</th>
<th>Hard to satisfy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predictable</td>
<td>&quot;Easy&quot; to satisfy</td>
</tr>
<tr>
<td>Known</td>
<td></td>
</tr>
<tr>
<td>Unpredictable (ad hoc, variable)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operational Information Needs</th>
</tr>
</thead>
</table>

- If the focus is only on responding to operational needs, it will be hard to respond to management information needs.
- If the focus is expanded to the known, evident or very predictable management information needs, then only these questions can be answered.
- If the information is considered more broadly, more of the unpredictable information needs can be answered. This allows to increase the insight of the management in the company and thus its control over the company.
This is a basic model that may help when investigating operational issues. It may help to come up with some questions. There might be lesser or more entities involved.

A right classification of these elements is crucial for later use by management.
Value of information is subjective. It’s an estimation. It depends on the individual, circumstances, time, ...

**Meaning**
- What does it mean?
- Meaning is increased when the information is **connected** to other information.

**Quality**
- Reliability, clarity, unambiguity, precision, accuracy, completeness, up-to-date, recent, format, ...

**Time**
- Information has to be available on time or must pertain to meaningful period

**Accessibility & Processability**
- Inaccessible information is worthless
- Difficulty to process decreases the benefits of information (processing costs time and resources)
- In time, on the right place, in the right form increases the value (or doesn’t decrease it).

**Possible Usage**
- What is it used for and what **can** it be used for
- Is its usage essential, critical or optional?

Consider the maximal value or the maximal potential of the information
**Practical value:** Value by practical usage of information. Responding to an existing information need or to a predictable need that will occur in the short term.

Example: The address of the client where the product will be delivered within a week.

**Estimated potential value:** Subjective value assigned to information based on possible usages in the future.

Example: Names and addresses of former customers. They are still useful. Today, they have no practical value (no ongoing order). But these clients can put a new order in the future and this information can be useful for future mailings or marketing initiatives.

**Operational value of an information element**

Usage is known. Relatively predictable.

Example: A customer order has operational value from the moment the order is placed until it is completed. Once completed the operational value falls back to nearly zero.
Information Value in Time

Management value of an information element

<table>
<thead>
<tr>
<th>How did the company in the past?</th>
<th>How does the company perform now?</th>
<th>Plans for the future?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Present</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

How does the company evolve?

Predictable
Habitual information about the functioning of the company (example: KPI’s)

Unpredictable as a response to ad hoc investigations
- Detection of issues or opportunities
- Investigations
- Simulations

Management needs information, among others, to plan future operations, to monitor performances, to look for tendencies.

Operational data is often used as a selection. Aggregated data or lesser detailed data are used for management purposes. The value of historical data tend to decrease over time.

At some moments in time an information element may regain in value. The value curve is not as smooth, but rather irregular.
Information must be

- Present, captured, recorded
- True
- Meaningful, relevant
- Complete
- Up-to-date
- Coherent
- Variability (time or period of validity, usefulness, value)
- Reliable, certain
- Clean (not polluted with other information)
- Clear, Unambiguous
- On time
- Valuable
- Precise, detailed
- Processable
- Accessible
- At the right place
- ORGANISED, structured
- MANAGED
- UNDER CONTROL
1. **Availability**

Information must be recorded. Information that is not present cannot be used. It is worthwhile to look at the clients and to search in the company’s environment for interesting information.

2. **Accessibility**

Then, it has also to be accessible to the business community.

3. **Timing**

The information has to be available and accessible and presented to the employees in time. Information coming too late (too soon) is of no or lesser use.

4. **Accuracy**

Information has to be accurate enough for its intended usage. Obtaining information that is 100% accurate for might be too costly. Information that is not accurate enough may decrease its usability and its value.

5. **Reliability and Objectivity**

Information has to reflect reality. It has to be as objective and as true as possible. Information that is not reliable will introduce even more unreliability. It undermines the proper functioning of the system as well as the trust.

6. **Completeness**

Information needs to be complete. Some operations, processes or decisions are more difficult, more risky or can not take place if some information is missing.

7. **Relevance - Appropriateness for Usage**

Information has to be relevant for the activities to be performed or decisions to be taken.

8. **Processability – Appropriateness for Processing**

Information has to have the right structure, format, level of detail, encoding, support.
Degrees of Information Quality

Continuous spectrum:
- High-Level ↔ Detailed
- Coarse ↔ Fine
- Clarity, unequivocally ↔ Ambiguity
- Precision ↔ Vagueness
- Coherence ↔ Incoherent (conflicting, opposite)
- Up-to-date ↔ Outdated
- Past ↔ Present ↔ Future
- Theoretical ↔ Practical, applicable
- True ↔ False
- Certain ↔ Unsure
- Complete ↔ Incomplete, partial
- Right ↔ Distorted
2 Ways to increase information

Capturing information is crucial

Missing information is a cost, a lost opportunity.
It may have far reaching consequences.

REFLECTIONS
(1) Endless growth of information? Good or bad? To be controlled? To be managed? How to deal with it? How to adapt systems to it?
Information Operations

Information tends to increase over time + Information Entropy

IF

- the nature of information is not understood

AND

- the principles of information engineering and information management (including proper information usage) are not respected

THEN

it’s easy to create INFORMATION OVERLOAD and INFORMATION CHAOS

Losing control over information

= 

Losing control over the company

Over time, quantity tend to increase whilst quality tend to decrease
Information Pool

INFORMATION POOL

- Respect essential information qualities (below)
- Only valuable information
- Organise information
- Avoid information duplication
- Maximise information and information quality
- Maximise information value
- No useless information (remove, clean up, archive)

Information architecture brings stability. It is better to develop processes upon good information architecture than to develop an information architecture that suits the present processes. Why better? The information structure is better which results in better exploitability, expandability and changes will have smaller impact.
Information Organisation

"information"
when emphasising the purpose, the usage and the value of data

"data"
when considering information as a matter that has to be captured, processed, stored, ...

“information”
when emphasising the purpose, the usage and the value of data

“data”
when considering information as a matter that has to be captured, processed, stored, ...
Information Organisation

Planned, being executed or implemented & what has been done and results.

Describing the organisation, meaning and format of the data
Methods to Organise information

- Per Concentric layers
  - Core: reference data
  - Operational: transactions, process
  - Peripheral: Managerial info, BI
- Per Business Domain
- Per activity, activity type or purpose
- Per nature
- Per availability

Usually, a mix of the perspectives is used
Information

Structured data

Semi-structured data

Unstructured data

Graphical

Related data elements (connected)

Aggregated data

Deduced data

Calculated data

Meta data

Relation description

Meta-meta-data

For systems engineering purposes

For methodology engineering
Information Exploitation

How to maximise the value and exploitability of information?

1. Relevant, meaningful
2. Correct
3. Up-to-date
4. Precise, Clear, Unambiguous
5. Clean
6. Reliable (verified)
7. Standardised
8. Connected
9. Atomic (elementary, indivisible)
10. Logically organised in an architecture (reflecting real world, criticality, purpose, processing or usage)
11. Structured (avoiding aggregations, documents, ...)
12. At the right place
13. On time
14. Available, being present → need to be captured, gathered, ...
15. Accessible
16. Findable and retrievable
17. Quickly accessible
18. Easy to process
19. Meta-data (described with)
20. Captured, stored
21. Unique occurrences
   avoid having to update of multiple occurrences of same information
22. Shared and Re-used
23. Protected
24. Inventoried, Maintained and managed
25. Under control, under own control

Goal of the Analyst
striving to maximise these qualities in corporate information solutions.

These aspects have to be taken into account when conceiving information architecture, information systems and information treatment and usage.
How to improve Information Exploitability

1. Capture as much relevant data, data with a true potential value, as possible
2. Use an Information Inventory
3. Keep control over the information
4. Keep information as atomic and structured as possible
5. Connect information elements as much as possible. It increases their value. A connection is also information.
6. Share information (avoid multiple instances. Though, in some cases multiple instances is advisable)
7. Make information accessible (but secure it as well)
8. Understand the meaning of the information (what precisely does it describe)
9. Use information formats which are easy to process
10. Avoid cryptic and confusing codes
11. Standardise information as much as possible. Limit the number of formats
12. Implement agents controlling data quality (respect of formats, missing information, incoherent information, ...)
13. Clean up (delete, archive, ...) old data.
14. Have data policies and data governance

15. Train People

Notes:
Some items are on the previous list as well.
Information Operations

**Easy operations**

- Capturing
- Transferring
- Duplicating
- Unconsidered dissemination
- Changing
- Aggregating
- Unstructured information
- Disconnecting
- Usage of freedom and flexibility to manipulate data

*(End-users can do this)*

**Challenging operations**

- Identifying
- Meta-data (creating, describing)
- Decomposing
- Analysing
- Selecting
- Standardising
- Cleaning
- Modelling, structuring, organising, formalising
- Connecting
- Keeping up-to-date & reliable
- Maintaining consistency
- Managing information status
- Managing information throughout its life-cycle

- Preserving and maximising the value of information
- Maximising its processability

**Better exploitation**

- Decrease of value of information

**of information as a resource**
UNREGULATABLE INFORMATION AREA
End-users decide freely about what information they capture, organise, process and use and how they do it; information is not formally controlled; not standardised; not inventoried; not quality checked; hard to process; maybe copied but not shared; ... on the edge of chaos.

ANGLEED INFORMATION AREA
organised, formalised, computerised information and information processing

CANDIDATE INFORMATION FOR ENGINEERING
Information and information processing that can be engineered in the future

Notes:
Expand the Engineered Information Area (A) by engineering area B.
Expand the Information Candidate for Engineering Area (B) by detecting information and processes in area C that might be engineered. Overtime, area C will shrink.

Area C is difficult to formalise, to clean up, to maintain, to manage, to exploit, to engineer, etc., at least, for now. (OAS supports the exploitation of ‘B’ and ‘C’-type information).
Prepare information for multiple usages

Design information solution (information architecture/model) that allow multiple usages of information
Information Domain and Information Sphere

Information Spheres

*Usage oriented*

Information Domains

*Logic oriented*

**Information Sphere:**

Information a user or user’s group might need to perform his job

**Information Domain:**

Information relating to a same subject, object, topic, issue, aspect, item (reflects the real world)

**Notes**

Business people may need to use information from different domains (like products, orders, financial,..) to perform their job.

These two concepts help to manage accesses and usage of information.
Main Causes of Information Problems

• Lack of information

• Bad quality of information

• Unorganised or badly structured information

• Inappropriate “solutions”

• Bad information practices

Some wrong, imprecise or unreliable information in a database can destroy the trust in all the information in that database and thus destroying the value of the database.
Trust & Information Contamination

A few unreliable data, other data is OK.

Since we don’t know what data is reliable and what isn’t ...

we suspect every data element to be unreliable.

The whole database is considered as unreliable & not trustable and not trusted.

• Contaminated data is transferred to other databases
• Data is connected to other data
• Data is used for different purposes, by different persons

Notes:
How to limit this? Automated data input; appropriate and facilitated data input; appropriate checks on input; Data Quality Checking software running through the DB; get employees verify data when using it (if possible)
V.I.P.: Vital Information Practices

- What information has value?
- How to capture information (easily, efficiently, early, reliably, ...)?
- How to identify information?
- How to verify information?
- How to structure information?
- How to link information (relations)?
- How to keep information process-able?
- How to keep information up-to-date?
- How to keep information consistent throughout the company?
- How to keep the information useable?
- How to evaluate information?
- How to share and reuse information?
- How to control the usage of information?
- What is the lifecycle of information?
- What meta-data should describe information? How to determine and manage meta-data?
- How to preserve or increase the value of information?
- How to secure information?
- How to innovate with information?
- How to clean up information?
- What information can be erased?
- ...
Conclusion

Information is a

1) Critical resource
2) Extremely valuable
3) Hard to manage

Think “INFORMATION”

- Be vigilant about any “information destroyers” and “information risks”
- Use your best information experts
- Sharpen information skills
- Avoid giving away control over (critical) information
ENTERPRISE ARCHITECTURE
Architectures

Some Major Structures / Architectures

- Goal Structure (Architecture, Model)
- System of Plan (organised and managed set of all/major plans)
- Decisions Hierarchy (?)
- Capability Structure
- Geographical organisation
- Organisation
- Process Architecture
- Information Architecture
- Production / Assembly Line Architecture
- Information Flows
- Communication Flows
- Transportation Infrastructure
- System of Information Systems
- System of Software Systems
- Functional Software Architecture (FSA)
- IT Architecture
- Security Architecture
- Network Architecture
- Energy Network
- Water Network
- Waste Network

Notes:
- This set of organisations, structures, architectures and systems gives an idea how things are entangled. It emphasises a holistic approach.
- Each architecture requires specific knowledge area specialisation and skills. Yet, they need to be aligned. All these engineers have to collaborate very closely.
- Some architectures may be not applicable to an organisation.
- There is more required to achieve alignment, balance and sustainability than aligning architectures.
# Enterprise Engineering Framework

<table>
<thead>
<tr>
<th>Identity level</th>
<th>Values</th>
<th>Mission</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Company Culture</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Business level</th>
<th>Key Products and Core Services</th>
<th>Business Model</th>
<th>Geographic Organisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy</td>
<td>Value Network</td>
<td>Value Stream</td>
<td>Business Functions</td>
</tr>
<tr>
<td>Policies</td>
<td>Supporting, derived or additional products &amp; services</td>
<td>HL Business Processes</td>
<td>Business Capabilities</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Enterprise level</th>
<th>Environments</th>
<th>Business Processes</th>
<th>Enterprise Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>System of Systems</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System of Flows</th>
<th>Human Organisation</th>
<th>Industrial Component</th>
<th>Process Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>System of Systems</td>
<td>Software Syst. Architecture</td>
<td>Information Architecture</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Technical level</th>
<th>Energy Infrastructure</th>
<th>IT Infrastructure</th>
<th>Security Architecture</th>
<th>Waste removal; Transport network; ...</th>
</tr>
</thead>
</table>
Notes

These diagrams present some important elements of a company and from where these elements can be derived. A model is always a reduced picture of the reality. And since every company and situation are different, the relations between these elements are specific to each case. They vary from case to case.

For example, the human organisation can be existing and be a given for an initiative.

It is the duty of practitioners to analyse and to understand the company and situation they deal with. These diagrams can serve as a basic template helping to understand and which then can be adapted to the particular situation.
The **Business Architecture** is a business description of the system by which business activities are deployed, the way of executing the business activities and the way it will achieve the realisation of its vision, mission, purpose and objectives. The Business Architecture is a business perspective.

**Principle:** The business architecture has to be defined by

- Management
- Business Experts
- Business Consultants

Consequently, anything requiring non-business expertise to be established, doesn’t belong in the Business Architecture. It is a part of another Architecture.

### Content of the Business Architecture

- Actors, Entities, Organisations
- Partners
- Suppliers
- Main Resources
- Sources of resources
- Key Products
- Core Services
- Key Business Capabilities
- Geographic Locations
- Geographic areas
- Customer Segments
- Interorganisational Flows
- Value Streams
- Business Functions
- Business Policies, Rules, Principles
- High Level Business Processes
- Financial Structure

### Complementary to the Business Architecture

- Strategy
- Metrics and Measures
- Initiatives and Projects
- Decisions

**Notes:**
This definition of business architecture differ slightly from the mainstream definition.
Notes:
Why is this important?
1) All these aspects are expressed in information
2) They have to be supported with information
3) Information systems have to be aligned with this.

It is the Analyst’s and architect’s job to contribute to this alignment by identifying misalignments and conceiving aligned solutions.
Alignment
Henderson – Venkatraman

Strategic Alignment Model (SAM)

FUNCTIONAL INTEGRATION

Business Strategy

External
- Business Scope
- Distinctive Competencies
- Business Governance

Internal
- Administrative Infrastructure
- Processes
- Skills

IT Strategy

External
- Technology Scope
- Systematic Competencies
- IT Governance

Internal
- Architectures
- Processes
- Skills

Organisational Infrastructure and Processes

I/S Infrastructure and Processes
Alignment of the Company

System of Plans

Enterprise
Outside World

Alignment of Plans

Plan Architecture - System of Plans – Organised and managed Set of Plans – Hierarchical Network of Plans

Strategic Plan
Financial Plan
Marketing Plan
Human Resource Plan
IT Plan

Alignment of information and Information Needs

Information
Information Needs
Information Systems
IT Implementation

Notes:
Alignment is a momentary state. Because of variability, everything has to be kept in sync (aligned). They have to evolve together. (different speeds of adaptation!).
Alignment is a reactive process. It’s a minimal state Companies need to drive, to lead, to create, to innovate.
Enterprise Architecture Framework

MODELLING LANGUAGES

TOOLS

DELIVERABLES

DELIVERABLES DEFINITION (ex. templates)

ORGANISATION OF DELIVERABLES

MODEL DEFINITION

ORGANISATION MODELS

REFERENCE REPOSITORY & LIBRARY

ENTERPRISE ARCHITECTURE METHODOLOGY

Meta-EA-Entity Relation Diagram (influence, dependence, deduction)

References, definitions, generic concepts, theories, principles, generic business domain concepts, ...

BASIC BUILDING BLOCKS (elementary versions or ready to use)

ENTERPRISE MODULES & COMPONENTS

ENTERPRISE ARCHITECTURE

IMPLEMENTED ENTERPRISE ARCHITECTURE

EA MANAGEMENT STANDARDS, POLICIES & PROCEDURES

EA CONTROL AND MANAGEMENT

MODELLING METHODS

supports

supports

supports

supports

organises

organises

organises

organises

forms

forms

forms

forms

is stored in

manages, allows to adapt and keeps under control

(\*\*: guides how to represent it, levels of details, how to adapt it, validations, ...)

is input for

delivers

delivers

located in

is implemented (by means of projects)

is used by

is used by

provides feedback to

trigger major changes

supported by

supports

supports

supports

supports

used in

used in
INFORMATION SYSTEMS
IS’s are part of the company. They partly form the company. The company, its activities and the overall environment have to be taken into account when designing an IS. Therefore they should be understood.
Supra-system determines or strongly influence the design of its parts

**Reflection:**
When designing a solution / IS, can we ignore the supra-system?
What is the relation between the company and the IS?
What is a company / organisation?
1. Transformation mechanism
2. Queuing mechanism
3. Filters
4. Buffering mechanism
5. Finding mechanism
6. Selection mechanism
7. Sorting mechanism
8. Prioritisation mechanism
9. Decision making mechanism
10. Measuring mechanism
11. Conversion mechanism
12. (Re-)Formatting mechanism
13. Decomposing mechanism
14. Grouping or assembling mechanism
15. Matching mechanism
16. Identification mechanism
17. Connecting mechanism
18. Integrity checking
19. Balancing / Assignment mechanism
20. Evaluation mechanism
21. Messaging system
22. Routing mechanism
23. Push / Pull Mechanism
24. Transaction / rollback mechanism
25. Feedback mechanism
26. Pattern detection/recognition
27. AI mechanism
28. I/O management system
29. Encoding / decoding mechanism
30. Log-in system
31. Access Control mechanism
32. Alarm mechanism
33. Notification system
34. Control unit
35. Monitoring mechanism
36. Debugging mechanism
37. Tracing mechanism
38. Logging mechanism
39. Archiving/ Restoring mechanism
40. Deployment mechanism
41. Patching mechanism
42. Configuration mechanism
43. Version control mechanism
44. Deployment and activating mechanism
45. Start, Pause, Resume, Stop mechanism
46. Clean-up mechanism
47. Back-up/Restore mechanism
48. ...
Software Systems Qualities

Functional Suitability
- Helpfulness: deals with most important and valuable issues, create most value
- Appropriateness: (solution is appropriate)
- Process / Feature Accuracy: (solution is accurate)
- Degree of automation
- Degree of control over the processes

Operability
- Technical accessibility
- Understandability / Recognisability
- Ease of use
- Attractiveness
- Learnability

Reliability
- Availability
- Fault Tolerance
- Recoverability

Performance Efficiency
- Time Behaviour
- Resource Utilisation

Security
- Authenticity
- Accountability
- Confidentiality
- Integrity
- Non-repudiation

Compatibility
- Standard compliancy
- Co-existence
- Interoperability

Data Suitability
- Data importance
- Data organisation
- Process-ability
- Reliability
  - Completeness
  - Accuracy
  - Consistency

Integrate-ability
- -

Maintainability
- Maturity
- Degree of control over the system
- Supportability
- Configurability

Adaptability / Evolve-ability
- Expandability / Extendibility
- Scalability
- Replaceability
- Documentation
- Source code organisation
- Analysability
- Changeability
- Modification Stability
- Modularity
- Reusability
- Testability

Transferability
- Installability
- Portability
- Adaptability

Other qualities:
- Geographical spread
- Cohesion, coherence, fragmentation of responsibilities
- Cohesion, coherence, fragmentation of technologies
- Degree of structured information
- Matching between implemented concepts and reality
- Degree of alignment
- Degree of neatness and elegance of the source code

Sources:
ISO9126
ISO25010
Needs of the Informatics Department

The responsibility for the administration and management of information systems creates needs:

- Deployment system (updating, activation, upgrading, ...)
- Release & Version Management
- Configuration tools
- User and Access Management
- Start-up, On-Hold, Stop (shut down) procedures
- Monitoring
- Logging Management (tracing bugs, errors, exceptions, ...)
- Alarms
- Data Quality Control (E.g. consistency checks)
- Data Synchronisation
- Queue Management
- Performance measurements
- Environment Management
- Configuration Management
- Resource Management
- Capacity Management
- Hardware Management
- Archiving & unarchiving
- Problem Analysis
- Back-up & Restore
- Data Recovery - Disaster Recovery
- Clean-up
- Security
- ... ?

The Informatics Department is a MAIN USER of software application systems.
Man-Made Systems

A man-made system is the creation of our brain

Therefore

• our understanding
• our vision
• our thinking patterns and habits
• our norms
• our mental flexibility
• the openness of our mind
• our creativity
• our knowledge
• our competences
• our experience
• our priorities
• our collaboration
• our emotions

but also our interpretations, assumptions, misunderstandings, distortions, unknowns and other weaknesses

are reflected in the design of the systems and solutions we create.
Why we need to learn and understand systems?

Each automated system has a purpose. It implements a mechanism based on principles. It has characteristics, limits and constraints.

A person, as a user, a manager or as a part of the system, feeds, uses and influences the system. Dealing in a wrong way with the system will create damage. The system will inevitably respond accordingly to its own logic, irrespectively of that person’s knowledge, opinions or intentions.

Similarly, adapting such a system without understanding it fully is a very risky action. Again, the system will react accordingly to its nature and logic.

Systems obey their nature, natural laws, principles, built-in logic characteristics and limits. They have to be respected.
The design of systems must respect the laws and principles governing systems. Some of these laws and principles are essential to the thriving of the system.

Similarly, information systems have to respect laws and principles governing a proper organisation, processing and use of information.

A system which doesn’t respect them risks to die from within. A system dies or run into trouble, not so much because of events, but much more because of bad design or trying to solve an inappropriate problem. A good system should have a alerting mechanism warning that it is going to the end of its life.

The understanding of systems and information, both as a knowledge domain, are critical in the conception of information systems.

This knowledge is neither to be found in the business domain knowledge nor in technological knowledge.
Tale of the 3 Little Pigs

Good Enough?  Seemingly strong  Appropriate

Not (only) about materials (technology). It’s also about

- Vision
- Mind set
- Norms
- Approach
- Skills
- Seriousness & Professionalism
Levels of Views

A wall isn’t simply bricks piled up orderly with joints filled with mortar.

A room is not simply a few walls put together.

A house is more than an assembly of walls or rooms.

A city is more than an arrangement of houses (or buildings).

Notes:
Why is the idea of piling up bricks with mortar in between not useable to conceive cities? (similarity with software/IS engineering?)
Winchester Mystery House

**Owner: Sarah Winchester** (widow of William Wirt Winchester (firearm magnate))

**Approach**
- From the ground-up
- Owner and Builders meet daily
- No architect (no analyst)
- No global plan
- Based on sketches

**A few numbers about the building:**

161 rooms, 40 bedrooms, 40 staircases, 47 fireplaces, 17 chimneys, 6 kitchens, 13 bathrooms, 2 ballrooms, 2 basements, 3 elevators, 10.000 window panes, one shower

**Building Process**

16 carpenters worked in shifts 24 hours a day, 7 days a week during 36 years (1886-1922). Sarah met the carpenters daily. She hastily sketched designs on paper to explain what the carpenters had to build. There was no architect. Sarah built, demolished and then rebuilt. Sarah ordered errors to be torn out, to be sealed, to be built over or around it, or sometimes simply to ignore it.

**Oddities**

The house is a labyrinth. Windows are overlooking other rooms. Cabinet and doors open onto walls. Small rooms are built within larger rooms. Some chimneys don’t reach the ceiling. Stairs have odd-sized risers or leading to nowhere. A door opens out into nothing. Balconies are inside. Floors have skylights. This approach led to a chaotic architectural jumble. Rooms were hidden and lost behind new construction. Just to name a few.

Today, the mansion is a tourist attraction
The castle is a system that is vital, well-thought, functional, practical, lasting, operating as one system, ... and critical for thriving. (It’s not about being outdated or monolithic). It is about the focus and mind-set to build great systems.

**Notes:**
Metaphor: Restaurant

This metaphor helps thinking about what corporate IT is required.

Customers may find it long to wait for their meal. They want it quick, cheap and tasty. They want a great experience. The restaurant may deliver that.

To deliver a decently cooked meal, it takes some time. The customer has to have some patience. If he can’t wait, either he will have a badly cooked meal.

The restaurant may also try to look for simpler meals taking lesser time to cook. Possibly, it may end up transforming itself into a fast food. Fast food restaurants are “restaurants” without chefs.

The restaurant may seek to offer a great experience or to offer them nice looking and very tasty meals.

The restaurant may seek to solve the hunger of their customers. And customers may be glad not to have hunger anymore. However, since hunger is just a symptom, it ends up solving symptoms.

Resolving the desires of experience, taste and hunger can be fine. However, in the long run, it may have devastating effects. The human body needs a healthy diet to be healthy and to live long. It needs specific nutritive elements on regular basis.

Patience and decent meal is maybe not what the customer may ask for, but it is what he or she needs, particularly when we consider the broader picture.

Decisions made based on pleasing or on short term may undermine the system, the foundation and the long term, particularly if they are made frequently.

A chef needs to understand cooking tools, cooking techniques and taste. He or she needs to understand what (s)he creates and the materials (s)he deals with. (S)He has to understand food and nutritive elements, their value, their nature, what happens when they are processed, and so on.

A good restaurant can not be satisfied with a chef that knows only his tools and techniques or one who can only cook by using recipes. Recipes are fine. But they are only the basic process. It has always need to be adapted to the specific situation. Being able to follow a recipe doesn’t make a great chef. Recipes don’t replace the knowledge of the chef or the cooking skills.
Demand and Want-Based Development Cycle

Notes:
As all models, this is an approximation of how it can look like.
Or, software development w/o analysis and without methodology.
Does this approach match the needs of efficient and effective software development?
Demand and Want Analysis

What is the QUALITY of the Intentions, Wants, Demands and Expectations?

MUST be:

- Justified
- Possible
- Coherent
- Complete
- Stable
- Reliable
- Good / Optimal
- ...

CUSTOMER SATISFACTION

leads to

If no more problems, obstacles, risks, ... a are created

Intention(s) ➔ DEMAND ➔ Wants ➔ Expectations

Highly volatile

Intention, wants, expectations and demands of business stakeholder, customer, end-users, sponsor
The average analyst has no (or little) clue on what this is based.

He/She doesn’t know the reasoning behind it.

How much is this in line with reality? (cfr practical versus choice problems)

What changes these expectations and wants?

- New information
- New insights, deeper insight
- Detection of contradictions
- Uncertainty, hesitation

We don’t know ...

- whether the demand, the wants, the expectations fit with the intentions
- whether it solves the problem,
- we don’t know if responding to the demand, expectations, wants that customer satisfaction is created. We simply assume this. Projects and actions based on assumption ... ???
Demand and Want Analysis

Intention(s) → DEMAND

Expectations ← Wants

Business Knowledge
Perception of IT
Belief system about IT

Procedures, Practices, Habits, ...

REALITY, CONTEXT, SITUATION,
COMPANY ENVIRONMENT,
IMPLEMENTED SYSTEMS, ...

Cfr chapter dedicated
to business knowledge

Distorted view: Consumer vision,
End-user’s perspective, people versus
computer, historical evolution of IT in
the company, experience (with lack
of understanding what happened), ...

Not always (rarely) the right ones

surface

Superficial knowledge

Gaps in knowledge are filled up with interpretations,
assumptions, guesses, opinions, ...

Accurate knowledge, deeper insight

Factual, reality, truth, logic, ...

depth

Much Work for the Analyst
Art vs Engineering

- Mainly an engineering discipline, partly also an art.
- Building the IC of the company, when possible also please end-users

Software Development is an **ART**

Software Development is an **ENGINEERING** discipline

Please business community (end-users and stakeholders)

Build the IC of the company

Not sure this will work

Not every thing can be fully engineered

Common widespread way of (ab)using software engineering

Agile

Two System Building Processes

**Process 1**
Simply doing anything that seems necessary

**Process 2**
Engineering Process
(well above 50% is engineering)

More likely to be a system that is badly tied together

More likely to be a better system

Not every system development process can be called an “engineering process”
Two Types of Design Approaches

LOGIC - driven

- Rational, based on logical thinking, reasoning
- Product, result is more important than taste
- System-oriented
- Architecture-centric, component based
- Requires understanding, holistic, knowledge-based
- Top-down, integration
- Plans and models
- Some more control
- Rigorous, methodical, structured
- Longer iterations (short iterations is possible)
- Examples: Systems Design, Systems Thinking

CREATIVITY - driven

- Experimental, trial and error, experience
- Usage-centric, people-oriented
- Can be matter of taste
- Creativity, idea generation
- Ad hoc, unpredictable
- Short iterations, fast feedback
- Lesser structured approaches
- Requires lesser understanding
- Lesser (supra-)architectural
- Bottom / Bottom-up
- Integration is lesser important
- Examples: prototyping, design thinking, user-centred design, …

Creativity-driven gets more attention these days increasing the uncertainty, unpredictability and amount of changes.

Gradation

Logical → Creative

Each type of design approach uses the other, but to a lesser degree (more locally, sporadic, for a specific usage, as subordinated approach, …)

Notes:
It is the nature of the challenge; the nature of the system, its purpose and usage that determines which type of engineering is required.
Different Types of Approaches in Man-made Larger Complex Systems Conception

For this type of systems, the approaches should be in this area.

It is mainly top-down, analytical and architecture-based. It make also possible to have creativity, some bottom-up and some organic growth.

This combines the strengths of all tendencies.
<table>
<thead>
<tr>
<th>MEDIocre Driving Ideas</th>
<th>Superior Driving Ideas</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fill the gap</td>
<td>Higher goals</td>
</tr>
<tr>
<td>Get the problem solved</td>
<td>Purpose-based</td>
</tr>
<tr>
<td>Get the job done</td>
<td>Root causes-based</td>
</tr>
<tr>
<td>Follow the flow</td>
<td>Holistic</td>
</tr>
<tr>
<td>How does anyone else do it</td>
<td>First Principles</td>
</tr>
<tr>
<td>What do you want me to do?</td>
<td>Natural laws</td>
</tr>
<tr>
<td>Analogy, Copy, Buying, “this is similar to”</td>
<td>Multi-dimensional</td>
</tr>
<tr>
<td>“Having it working is just fine”</td>
<td>Sustainability</td>
</tr>
<tr>
<td>Local, short term</td>
<td>Long-term</td>
</tr>
<tr>
<td></td>
<td>Evolution</td>
</tr>
</tbody>
</table>
Engineering Systems

Information is critical for the company. Information systems, and their automation, constitute a huge benefit especially for larger companies.

Building Software Systems is ...

- very Complex
- very Slow
- very Demanding

Techniques to speed up development:

- Tackling the right problem rightly from the first time
- Planning
- Creating an appropriate environment: relational environment, appropriate work atmosphere and physical environment
- Availability of people and competencies
- Smooth decision making processes
- Project team support
- Organisation of artefacts
- Sound architecture and concepts
- Internal organisation of source code
- Parallel development
- Good choice of standards, tools, technologies
- Modular software components and Reuse

Other techniques:

- Increasing pressure (shorter deadlines)
- Increasing the number of people
- Taking all kinds of shortcuts
- Leaving out intermediate steps and work

- Higher risks
- Lowers the quality
- Defers costs to the future, which may turn out to be much higher and hidden

Notes:

Shortcuts: ignoring complexity; ignoring the purpose, the value and the importance; pretexting “it has no value”, “it’s faster”, “we can already start” (leaving unsettled questions for later), ...

These decisions require understanding of projects and of ISE/SE.
Small & Large Projects

**Small Projects**

Usually:
- Simple, easy and short
- Can be done with limited understanding of the discipline, limited skills
- Can be done with little organisation, simple environment, little management
- No or little incentive to learn about the discipline and to improve
- Everybody can do it
- Easy to be successful
- Small, unimportant, superficial things can be achieved

**Large Projects**

Usually:
- Complex, difficult, long
- They really test our capabilities, understanding, ...
  
  *one can’t pretend to be skilled or expert and do only basic, small, easy, basic work*
- Requires great insight in the discipline, advanced skills
- Requires a more developed, formal and managed organisation and environment
- Obliges people to learn and to improve their skills
- Not everybody can do it
- Is NOT synonym of / does NOT imply “big bang” development/implementation !!!
- Serious stuff, things that do matter, sizeable challenges, ...
- Not necessarily harder to be successful (it just take more time)
- People built great things to show their skills. Societies built great things to show how advanced they are.

**Reflection:** Should large project be avoided because

- We can’t do it?
- We want to be successful? Fear of Failure?
- Impatience & quick wins

Or, on the contrary, are there good reasons to learn doing sizeable and larger projects?
THINK HIGH LEVEL

The mind has to be focussed on, to be driven, inspired and guided by the higher systems and their environment.

- **Goals** are deduced from the goals of the higher systems (supra, supra-supra-, ...)
- **Requirements, Design** and **Functional Integration** are deduced from these higher systems, their goals and purposes and from their environments.
- **Technological Integration** takes the supra-system into account.

The supra-systems may be of a different nature than the system and can be heterogeneous. Same is true for the sub-systems.

A mind driven by and focussed only on the change, the features, the system or the solution to be delivered is likely to create inefficiencies and, through repetition, over time, chaos.

*Also important for methodology development*
PRINCIPLE: SEPARATION OF CONCEPTUAL SOLUTION and CONCRETE SOLUTION

KEY PRINCIPLE: Conceptual thinking precedes and must be separated from concrete thinking.

The brain has first to think about how to solve a problem conceptually. Once this is done, it has to think how to realise it with technologies, with matter, with concrete pieces.

Process:
1) Define the conceptual solution
2) Define the technological solution, the technologies, the materials

Implication:
It is way better to separate the conceptual/functional/non-technological roles from technical/technological roles of architects, analysts, engineers.

Note:
It is favourable for the conceptual thinker to know (or to have a good idea) of what technology can do and what it can’t do.

The way business is ran and how information is used in the business activities and how it is organised, stored and processed in the company should be separated as well.
The objective, the scope, the intended solution (if present), the stakeholders, the global context and overall situation and the ISE experts and/or project team determine the approach required to achieve the objective or required to produce the intended product. They also define the required discipline, competencies and skills.

The approach defines the required work environment (room, space, tools, processes, schedules, ...) and the organisational setting (roles, responsibilities, collaboration, ...).

Performing jobs like sculpting a statue, writing a book, repairing cars, conceiving an advertisement or managing accounts or business information, trading, require a specific environment and work atmosphere.

It is up to the project team to define what work environment is required. This is not an easy task. For now, a lot assumptions are made about this topic.
**Systemology** is the disciplinary field studying systems.

Building ≠ Solution Conception; Using / Executing ≠ Solution Conception
Driving Goal of Initiatives

<table>
<thead>
<tr>
<th>Company / System Design</th>
<th>Dissatisfied</th>
<th>Satisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excellent</td>
<td>Dissatisfaction is possible, probably fixable</td>
<td>![thumb_up]</td>
</tr>
<tr>
<td>Inappropriate</td>
<td>Undermines the Org./System Hard to fix</td>
<td>Undermines the Org./System Hard to fix</td>
</tr>
</tbody>
</table>

Excellent System Design

Everything that shapes systems, affecting the Structural Design and Core Mechanisms

- Core logic of processes
- Core information concepts
- Organisation of the core of information,
- Landscape of systems (roles of systems, areas, interactions and collaboration between systems, (business) functions per (sub-)system)
- Systems solving wrong problems
- Integration, fragmentation, separation, duplication, interoperability, scalability, ...
- Missing sub-system and mechanisms necessary for the control, management, evolvability and survival of the system
- Architecture

It does not include: issues with features, adjustments of processes, missing information, end-user interface issues, ... they can create dissatisfaction, but does not require rethinking and reengineering core aspects of the system to be solved.

Inappropriate design

The core of the system is inelegant, weak, creating imbalances, internally chaotic, ...
Some False Beliefs

Beliefs based on the ASSUMPTION:
The demand is correct, complete, reliable and stable.

- The demand describes the needs.
- The demand describes what is required.
- Responding to a demand = solving the problem
- Responding to a demand creates satisfaction.
- Understanding the demands = understanding the problem
- Understanding the demands = understanding the situation
- Understanding the demands = sufficient to create a solution
- The demand takes the situation and all aspects and parameters fully (sufficiently, reliably) into account.
- If the demanded solution is built, value is created.
- Projects (always) produce solutions.
- Business community can diagnose its information problems and conceive information solutions.
- Business community knows what software and computer systems can do and can’t do. They understand their power, constraints and impossibilities.
Reactive Approaches

All approaches based on existing needs, existing problems or on demands

- Problem exists and creates already damage.
- Need exists and is already a missed opportunity (cost)
- No control over when what problem, need or opportunity will pop-up
- No control over the direction
- Following, not leading or shaping the future
- Always lagging, always gap between problem, issue, need or opportunity and solution
- Existing problem and opportunity creates frustration and impatience
- Creates pressure
- Assuming received information is reliable and true. No verifications.
- Pressure, assumptions, lack of verifications lead to mistakes, bad solutions, misunderstandings, conflicts, stress, exhaustion, …
- Focused on solution design, instead of on (understanding) the problem (understanding the demand or symptoms ≠ understanding the problem)
- The less time is spent on verifying, learning, thinking and testing, the more time can be spent on building and the quicker something can be delivered.
- Focus on building without understanding. Building based on assumptions.
- Limited to translating the demand into design of ‘something’ to be built.
- Less competencies are used.
- Not proper for innovation

Is a reactive approach appropriate as main stream type of approach?
NO SINGLE BEST?

SOLUTION - SYSTEM

- Not always a single best solution
  
  ( ≠ No (/never a) single best solution)
  
  Sometimes, but not always, there is a single best solution. We may neither assume there is a single best solution or that there is no single best solution. Often we don’t know and can’t know.

- If there are several solutions then
  
  • Those solutions differ from each other
  • Have their own specific strengths and weaknesses
  • Some are better than other

- Try to find better solutions and select the best of the known solutions

- Continuing to find alternatives and improvements

SIMILARLY

PROCESS – APPROACH – METHODOLOGY - FRAMEWORK

- Not always a single best APPROACH
  
  ( ≠ Never a single best way)
  
  Sometimes, but not always, there is a single best way. We may neither assume there is a single best way or that there is no single best way. Often we don’t know and can’t know.

- If there are several approaches then
  
  • Those approaches differ from each other
  • Have their own specific strengths and weaknesses
  • Some are better than other

- Try to find better approach and select the best of the known approaches

- Continuing to find alternatives and improvements
Life Cycle

Describes the evolution, the phases and major events that may happen during the lifetime of an entity.

Project Life Cycle (example of PMI’s PMBoK)
Development (Life) Cycle

Describes the phases, stages of a development of an entity.
The concept of feedback loop to adjust the behaviour of a dynamic system.

Any dynamic system must have one or more feedback loops

- Company
- Project
- Development Methodology
- Information Systems
- Individual
- ...

This theory answers the question why we need control in dynamic systems.
GOOD SYSTEMS ENGINEERING

Foundation

Based on solid foundation. Systems built upon weak foundations are weak. Solid foundation: TERMS (based on right definitions; already a huge problem in IT), natural laws, first principles, based on understanding of nature of things, atomic elements, building blocks, ... Business demands are useful, but commonly, they are not a solid foundation.

Reality

Reflects reality. Critical to understand nature of things, relations between things, nature of these relations, abstractions of reality, ...

Fitness for Use (Quality)

Obvious

Environment

Respecting the rules of the environment and respecting the environment

Events

Able to respond and to cope with events

Evolution

Able to evolve. Adaptability doesn’t come by its own. It is engineered in the system. The development environment has continuously to capture the needs and opportunities to evolve.

These are areas/sources for requirements
Engineering Flaws

Force aimed at reducing the probability (and the severity of impact) of flaws:
Approach, Norms, Techniques, Skills, Resources and Time used to reduce the probability of flaws and impact. These ‘forces’ are not equal. Their ‘size’ and nature depends of what quality (or flaws) is acceptable for the project. This depends of the project, its environment, its stakeholders, ...
Example: a technique aimed at reducing design flaws will not eliminate a faulty diagnosis

Requires different perspectives, different measures, different methods and skills, different knowledge (thus often by different persons) ... at different levels, at different moments in the development lifecycle

Notes:
These types of mistakes form actually a continuum. These categories are not strictly defined.
Major Design flaws implicate keeping the objective and (mainly) the scope, but need to review the architecture and overall design.
Faulty diagnosis, objective implicate the halt of the project. A brand new project has to be started.
A methodology and project process has to take this into account and implement a maximum of ways to prevent these fault to happen.
1. Implementing qualities by design vs adding it
   • Not about features. Some can be added easily to a system
   • Adding qualities like integration, quality, control, security, flexibility, connectivity, scalability, as a post-design (or late-design) activity is very hard. Rarely good results.
   • Much better results (more powerful and lesser complex) when added from the beginning.

2. Change of purpose, function, role
   • “Designing a screwdriver and trying to make a hammer from it.” Never as powerful as when it was designed from the ground up for that purpose, function and role.
   • Like using a spreadsheet as DB
Notes:
There is always some pre-implementation analysis (learning), continuous learning and post implementation learning.
Integration

When talking about Integration – 3 Questions

1) **What** is integrated?
2) **In what** is it integrated?
3) Which **aspect** is integrated? From which perspective?

**Integrated Information Flow**

The most basic form of integration, through automation, is to add some technologies to ensure exports and imports of data. It limits the integration between the systems it communicates with. A more advanced form consists of a deliberate and purposely engineering of the entire set of information flows.

**Technological Integration**

Technological integration is realised by using same technologies as those already present in the company or by using technologies allowing interoperability among the existing systems and technologies.

**Architectural Integration**

Architectural Integration integrates the new system, components, interfaces and changes in the landscape of systems or in the system of systems. The thinking about integration happened deliberately during architecture activities.

**Functional Integration**

Functional integration is based on domain knowledge, functions and purpose. Functionalities are logically organised. There is no duplication of features, functions, processes, ... All functions are well-thought.

**Holistic Integration**

Holistic integration combine all previous integrations. It also includes the integration in supra-systems and in their environments. All perspectives are taken into account and everything is aligned.
No integration

System is in isolation. Information is exchanged manually.

1) Post-implementation (basic integration)

Once the system is operational: “How can we automate information exchange?” This may bring in new technologies (middleware or other technology allowing information exchange).

2) Post-design integration

After the solution itself has been designed, the concern is raised of the exchange of information.

3) Integration as part of design

During the conception of the solution, the question of sharing or exchanging information is raised. This may influence the choice of technologies.

4) Architectural integration

5) Functional integration

Architectural and functional integration happen essentially during the analysis phase.

6) Holistic Integration

Holistic integration happens during the whole engineering process. However, it starts with the definition of the objective, the mission and the scope. It has to start as early as possible.
Teething problems are problems due to a lack of insight, additional information surfaced, imperfect solution logic, unexpected system’s behaviour or unexpected ways of using the system, missing features, and so on.

It is better to make a difference between teething problems and programming bugs. Teething problems require additional analysis or design activities.
Changes Eroding Systems

**Eroding System**
- Oversimplification
- Overly complex
- Unnecessary elements
- Inelegant internal organisation
- Lack of clarity

Per iteration:
- Increased size
- Increased complexity
- Decreased Manageability
- Decreased Adaptability

For future changes:
- Increased risks
- Increased time and effort (➔ paralysis)
- Increased cost

Benefit/cost (ROI) tend to decreases
Tendency to entropy

Common Causes:
- Time pressure
- Lower norms ("if it works, it is good enough" 😞)
- Unawareness of decreasing quality of the system
- Adaptions are made with the same perception, same assumptions, same priorities, same skills, same level of thinking, same beliefs and assumptions.
Evolving Systems and Crisis

Birth of System

Repetitive adaptations and expansions (by adding elements) of System

- A company has usually many systems in different stages of their lifecycle.
- Over decades the nature of developed systems evolved as well.

Manageability

Mastering the internals

Evolvability

Manageable situation

Problematic situation (REAL CRISIS)

Engineering by applying the same perspective, same thinking patterns, same intelligence, same insight, same belief system

Time for reengineering, paradigm shift or other radical measures and shift

Solving this requires a new paradigm, a new belief system. (Hard to achieve)
Trial and Error

Synonyms

- trial and failure
- trial and practice
- generate and test
- guess and check

Characteristics

- Every trial teaches something
- No upfront knowledge or insight can be acquired
- Plenty of time and resources
- Attempts can be done safely
- Limited number of cases to test or some options have better chances of succeeding
- Only ‘a solution’ is required, and no deeper understanding, ...

Trade-off between 1) number of possibilities; 2) effort, duration and cost of each attempt and 3) importance of the problem being solved.

Weaknesses and risks

- No alternatives or optimal solution. Aim to find “a solution”, not all or “the best solution”.
- Delivers single solution to single problem. No attempt is made to generalise the solution.
- Does not do a diagnosis. No cause/problem identification. Assumes the diagnosis is right.
- A solution may appear after a few attempts, only after many attempts, or not at all.
- May take much more time
- May use much more resources
- May turn out to be costly
- Making an error may be very detrimental. Only one or very few attempts are possible (think of surgery)
- Doesn’t search to explain “why” it works and no attempt is made to get this understanding. Maintaining superficial insight or ignorance. (Learning lessons?; Control and management of the broader whole?; Evolvability?)
- Found solution can be hard to optimise
Frameworks

- BizBoK
- OMBoK
- EABoK
- TOGAF, Zachman, TEAF, FEAF, PERA, GERAM, ...
- BABoK
- DAMA
- CobIT
- ITIL
- PMBoK
- Prince2
- SE BoK
- DA BoK
Bad Engineering Practices

Knowing bad practices help to recognise and to avoid them

- Throw it over the wall (limited collaboration)
- Accept anything at face value.
- Don’t question earlier findings, decisions, ...
- Don’t explain the higher goals, intentions
- Retain information (filtering)
- Freeze everything made or validated by the business stakeholders
- Apply principles, rules, methods because they are ‘prescribed’ or because of habits
- Build upon understanding of business demand and/or requirements as only insight.
- Be positive. Be a “YES”-(wo-)man. Everything is fine. Everything is possible.
- Do only the work you prefer. Work you don’t like is not necessary anyway.
- Perform unnecessary work
- Ignore the purpose, reason, value, right usage of concepts, tools, ...
- Do not collaborate or collaborate as little as possible
- Decisions can be taken without knowledge and understanding. Either the decisions is pretended to be right regardless of the cost and consequences, or it doesn’t matter because any decision can be changed anyway later on.
- Don’t do what is necessary. Instead, do whatever is asked hoping to meet the expectations
- Do not disagree. Avoid disagreeing with the chief, with the group or with the majority.
- Let others take decisions and responsibilities, that should be yours.
- Don’t review made decisions. Don’t ever backtrack.
- Be reactive and follow.
- Don’t doubt, even if the world is complex, dynamic and unpredictable.
- Pretend to know everything.

**Note:** These practices are neither prescribed nor advised in waterfall projects or in any project approach.
MANAGEMENT
Leadership

**LEADING**

Leader above the team

- This is our mission.
- This is the direction.
- This is the objective.
- This is the context.

*Guiding by setting the objective, Organising, Guiding by defining the process, ...*

**SERVING**

Leader under the team

- Team, what do you need to be successful?
- What can I do for you?

*Providing, resolving, ...*

**PROTECTING (*)**

*from above the team

*from below the team

Project Team

TEAM

*: interferences, conflicts, pressure, risks, disturbances, politics, ...

Management ≠ Leadership
WORK ENVIRONMENT
ORGANISATION
A Manager is not an Engineer

- A manager manages
- An engineer engineers

Managing and engineering

- have to different goals
- require two completely different mind-sets
- and require different ways of thinking

Exceptions: Some engineers may become managers and some managers have engineering degrees.
Conceiving a solution requires understanding

• how to maximally exploit information
• what is possible, difficult and impossible in IT
• what is necessary for an information system to survive and to strive
• what undermines an information solution
• what architecture fits the endeavour and environment
• understanding how to preserve or increase the value of information
• understanding information risks
• etc.

Who possesses this insight? Or, who is the most likely to possess this insight?
Hierarchical Structure

Creative with hierarchy – a few ideas

Top-Down – Bottom-Up Feedback

1. Top-Down
2. Bottom-up
3. Bottom to high top-down
4. Bottom to low top-down

Cross-border collaboration

- Horizontal and/or vertical
- Persistent, temporary or ad hoc

Area’s, Zones

Notes:
Can be vertical, horizontal or both. Permanent, temporary or ad hoc, guided or free, measured or not, controlled or not, ...
ROLES & COLLABORATION
Notes:
Collaboration (1) and (2) are extensive. Collaboration between (3) is useful, though somewhat lesser (when business-informatics issues are settled: reviews, wrap-ups, feedbacks, ...).
These factors have to be aligned in order to successfully perform a work. They are operational prerequisites for a job.

More alignments:
- Cultural alignment
- Alignment with “the boss”
- Alignment with the group
- Alignment of the evaluation criteria, evaluation system, reward system, …
KEY PRINCIPLE

Conceptual, functional, logical roles and the technical and technological roles should be assigned and performed by different persons.

The brain has to be able to focus freely on the global solution, on the big picture, on the higher goals, on the purpose. The attention of the brain is easily directed on what technologies to use and how to use them, on how to build the product, on what to do now, on the short term. These are very practically matters. When the brain is busy with these matters, the attention is diverted from the overall conceptual and global thinking, higher goals and long term. Great opportunity to get real or much better solutions can be missed. The risk of building non-solutions increases.

Also important for innovation!!

Positions, functions, job titles, job definitions should either concern conceptual work or technical work, but not both.
Collaboration is the Key to Project Success

1. Clear Common Goals
2. Motivation: Wanting the project to succeed
3. Mutual Understanding (communication, listening)
4. Availability / Time
5. Agreements
6. Knowing and respecting the roles (responsibility, authority, action)
7. Respecting the principles of the different disciplines

The project’s success and the created business value depend of every stakeholder and of every project team member!!

Note:
Not all definitions of ‘stakeholder’ include the project manager and project team members. Frequently the terms is used in statements to indicate business stakeholders, clients, ... but not the project manager and project team. This definition is more practical.
## Hindrances in Collaboration (1)

<table>
<thead>
<tr>
<th><strong>Consumer IT</strong></th>
<th><strong>Corporate IT</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Oriented towards consumer (selling and end-user experience)</td>
<td>Oriented to groups of end-users and serves the company’s interests</td>
</tr>
<tr>
<td>Solves personal needs, responds to personal tastes, personal choice</td>
<td>Solves various needs, primarily company’s needs, within a broader heterogeneous system of systems</td>
</tr>
<tr>
<td>Single product (except cloud, social media,...)</td>
<td>Set of complex interconnected systems part of the enterprise around an architecture, using an infrastructure</td>
</tr>
<tr>
<td>Focus on end-user features, appearance, ease of use, preferences and taste, experience, fun</td>
<td>Focus on value, processes, services, ROI, ...</td>
</tr>
<tr>
<td>Same product developed for many consumers</td>
<td>Developed for limited number of end-users</td>
</tr>
<tr>
<td>Usage: individually, accordingly to own taste</td>
<td>Usage: used by community of people, with different roles, but as part of a same organisation with a same objective. Shared, regulated, standardised</td>
</tr>
<tr>
<td>Easy, fast, cheap, flexible &amp; powerful</td>
<td>Slow to build, expensive, more rigid, powerful in a different way</td>
</tr>
<tr>
<td>Fuzzy, approximation, guessing end-user’s wishes</td>
<td>Abstract, detailed, consistent, complete, clear</td>
</tr>
<tr>
<td>A problem usually impacts individuals</td>
<td>Problem has impacts on the company</td>
</tr>
<tr>
<td>Easy to acquire and replace (buy and dispose)</td>
<td>Difficult to acquire. Requires customisation, tests, integration, courses, maintenance, security, recovery, ...</td>
</tr>
<tr>
<td>lifespan: 1 to 5 years</td>
<td>Has to evolve – lifespan: sometimes decades</td>
</tr>
<tr>
<td>More use of unstructured data and multimedia</td>
<td>Uses preferably structured data</td>
</tr>
</tbody>
</table>

Consumer IT are the IT products and services developed for personal consumption (usage). These products and services can be bought, downloaded or online accessed by consumers.

One can not confuse consumer IT and corporate IT (enterprise IT). The users and goals are different. There are different expectations, priorities and selection criteria.
Hindrances in Collaboration (2)

IT Literacy

What do end-users/consumers capture of IT?

- **Philosophy**: Intuitive, explore and discover, guessing, trial & error, experiencing, ad hoc
- **Ideas**: Quick and easy acquisition of technologies, “The sky is the limit”, “One click”
- **Plenty of software**: office automation, multimedia, games, tools, ...
- **Software characteristics**: attractive, intuitive, fancy graphical user interface, flexible software, rich features
- **Installation**: easy, automatic configuration, Plug-n-play, immediately usable
- **Usage**: Drag-n-drop, Cut-n-paste, Copy-Paste-Delete, Do-redo/ undo, wizards, templates, WYSIWYG, install-uninstall, problem? Ctrl-Alt-Del-solutions, ...
- **Data dissemination**: extensive use of unstructured data and individual data packets (documents), ad hoc copying and data transfers, seamless connections, invisible data exchanges
- **Development**: presentations, social media, blog, macros, websites, small software
- **Other**: IT magazines; unrealistic IT in Movies;
- **Assumptions**: Consumer IT = similar to Corporate IT; User’s Knowledge = IT Literacy; ...

**THIS PICTURE REFLECTS A VERY DISTORTED PICTURE OF CORPORATE IT**

*Not a good base for expectations, decision making, collaboration, …*
Hindrances in Collaboration (3)

Difference Between Human World and IT World

- Specific Capabilities in Information Processing
- Capabilities of doing what humans can’t do and vice versa
- Ability to perform a huge amount of work very quickly
- Omnipresence in the entire company
- Entanglement with the human organisation
- Speed of execution (huge numbers, small differences may have a huge impact)
- Connectedness, Interoperability
- Largest part is invisible to human’s world (only computers, GUI, appliances, cables, printers, …)
- Abstraction of virtual concepts and daily life elements
- Degree of formalism, precision of language and of instructions (process description) versus interpretation
- Range of details from very high-level to extremely detailed and precise
- Complexity (real complexity is hidden from the end-user by the GUI)
- Required consistency, repeatability
- Need to foresee ‘everything’ \(\rightarrow\) number of individual cases
- Higher norms are required
- Required level of thinking/planning ahead
- Effort required and speed of change – estimations !!
- Effects of changes
- Possible impacts (hidden parts of logic, connectedness, level of details, entanglement, …)
- Internal tendencies creating risks or leading to chaos
- ….

Hard to imagine, particularly for non-IT people

Gaps of ignorance is closed with assumptions
Lack of ability to detect the opportunities, to define better solutions, to define the right way of working, right norms, limitations, implications, to anticipate risks, …
Role of the Business Community in Information Initiatives

**INPUT** of the business experts, business stakeholders

1. The existing situation (how does the business work now)
2. Objectives, complaints, limitations, expectations, intentions, plans, ... related to business results, work and information
3. Business Expertise, Business domain Knowledge
4. Answers to Analyst’s requests
5. Disagreement or confirmation of the proposed solution

**What they must not provide as a given:**

- Identification, definition and description of the (information) problem
- The (information) solution they want

But business specialists may suggest it as a possibility to the analyst.

Why not? This requires “Systems Analysis” skills.

It is related to, applied on the business domain, but it is not business domain.
The mission creates needs on the project side: resources, time, information, budget, tools, work environment, collaboration, ...

So, to start the project ...

IT experts need to identify their needs: time, budget, resource, knowledge, collaboration, work environment, tools, time schedules, level of autonomy, ... to be able to work in ideal conditions. They become a ‘client’ as well.

If this is ignored, they can’t do their job properly and the final client can’t be served well

Note:
The Business Stakeholders are sources of expertise, “guiders” or parts of the system. The main client is (from a systems perspective) the company (organisation) and, from an activity-perspective, the final external client is the real client.
If (1) fails, then (2) and (3) will fail as well. If (2) fails, then (3) will also fail.

• (2) Gets often not the right attention. Frequently, the Actor B decides of what Actor A will need and what not, and when (s)he will need it. This is a recipe for failure.
Separation and specialisations of functions (roles) emerged as from the necessity to cope with complexity, scope, span, diversity and scale in the evolving software development initiatives.

This is an approximate evolution of the appearance of different functions. Each organisation has its own functions and function titles.

Functions like process analyst, information analyst, information architect, solution architect, application architect, IT architect, enterprise architect and many others are not represented.
Collaboration

1) Throw it over the wall

![Diagram showing the separation between roles in a project team]

Some collaboration exists between two actors adjacent in the process.

2) Compartmentalisation of the project team and their work

![Diagram showing walls between different roles]

Walls exist between the different roles in the project. The work is subdivided by the project manager and assigned to the individuals or sub-teams. This is all managed from above. Team has no or a vague global overview of the project and there is little collaboration among them.

3) Full & free collaboration

![Diagram showing a free flow of collaboration between roles]

This form of collaboration has drawbacks and poses risks. Which ones? And why?
To perform a work or to obtain a result, some parties are mandatory.

In some occasions (meetings discussing settled (global) issues, not unsettled details), the presence of other upstream or downstream parties is useful

- to familiarise with the broader subject
- to learn from wrap ups and conclusions
- to solve specific issues
- or as a matter of training

Key question: Who can contribute significantly to a meeting/workshop or who can learn a lot of it?

- Avoid absences of people that can contribute or who could learn from it.
- Avoid the presence of people who can’t contribute or who has little benefit from knowing what is/will be discussed. (Possibly, they may be informed by a conclusion or overview that will soon be given).
THE ANALYST
The role of “Systems Analyst” emerged from a need. It evolved as needs were better understood and the awareness grew of better ways to use this discipline in different areas. Different areas lead to specialisations of the function.
Goals and Guiding Principles

Goals, beliefs and principles guiding the analyst:

• to “satisfy the business demand”
• to work within the scope
• to have the requirements to be implemented
• to deliver a product that “meets the requirements”
• to “build and deliver the product, solution, software features as defined in the project”
• to “implement a set of needed software features”
• to build a software solution
• to create customer/stakeholder satisfaction
• to meet or exceed the customer/stakeholder expectations

These goals, beliefs and principles LIMIT the Analyst. They LIMIT the thinking and CONTRIBUTION.

Business objectives, enterprise objectives (survival of the system) and broader outcomes are the drivers and objectives of initiatives.

Notes:
The analyst is often reduced to order taking. Then: The potential of systems analysis not exploited. And the analyst has little responsibility. The role of Order Taking Analyst makes real and good analysts to run away.
Professional vs Layperson

The WHY something is necessary and WHY a certain approach, techniques, activities, steps and techniques are rarely known.

A professional differ from a layperson in that the professional knows the right answer to the WHAT, HOW and WHY’s.
Avoid blindly rely on WHAT PEOPLE ASK, WANT OR EXPECT

INSTEAD

DO WHAT IS RIGHT, WHAT IS NECESSARY, WHAT MATTERS AND HAS VALUE, WHAT IS REQUIRED TO REACH GOALS and what contributes to the company’s LONGEVITY and THRIVING

**FIND OUT WHAT IS NECESSARY !!!**

What the client says he wants is not always what is necessary.

“The client doesn’t know what (s)he wants.”

We may not base approaches on “what people want”
Examples

- In consumer: Delivering what the consumer wants may endanger the consumer (the consumer is not always aware of dangers of some products, like unsafe cars, fast food, chemical products, ...)
- In ecology: Doing what we wants, destroys “the planet” (nature, our life environment)
- In politics: Even doing what the majority wants, may destroy the system.
Motto: “What the Customer Wants” 
“Order Taking” – Solving Wants

- Customer doesn’t know what he wants
- What he wants doesn’t solve the problem or doesn’t allow him to reach the objective
- What he wants has changed
- Although he received what he wants, he is still not satisfied.
- Unaware of the missed opportunity of creating so much better solutions
- No innovation possible (or unlikely, only minor innovations, ...)
- Wrong problem or wrong solution and find out that much time and resources have been wasted.

People ASSUME they know what they want

Customer Satisfaction ???

The key to failure is trying to please everybody

The Real Analyst

Solving Problems, Achieving Goals

Tell me why you want it?
What is your goal?
What are the plans or intentions?

Let me investigate this!

I found out that the real problem is .... What do you think about it?

To solve the problem, I suggest this solution. How do you feel about it?

Building the right solution solving the right problem; well-designed; using best possible concepts of software; w/o creating new problems; possibly innovative;

Customer Satisfaction
Perspectives and Filters

Business Stakeholders
Perspective
Filter &/or Distorting lens

Incomplete (partial) and very subjective

The way to a MORE objective understanding

Knowledge
Insight
Vision
Ideas
Drivers
Skills
Experience
Intentions

Desires
Preferences
Needs
Priorities
Expectations
Norms
Success criteria
...

Role of Analyst
Gather, merge, align, remove incoherence, share these things

More complete, coherent, objective understanding \(\rightarrow\) likely to obtain good decisions

Incomplete (partial) and very subjective

Fragmented, distorted, partial, incoherent & superficial understanding \(\rightarrow\) More prone to bad decisions

15/03/2019
Often the analyst needs to get through the subjective world to perceive the objective world, the reality, the real situation, as it is. The analyst has to be able to make the difference between the objective world and subjective aspects. (S)He has to take the subjective aspects into account and deal with it (manage it).
Emotional Intelligence
- Relationship Building
- Empathy
- Diplomacy
- Listening skills
- Leading skills
- Group facilitation skills
- Collaboration skills
- Negotiations skills
- Interviewing skills
- Communication skills
- Writing skills
- Presentation skills
- Persuasion skills

... Social Skills

Attitude / Character

... Intellectual Skills

Professional Skills

Information science
- Information Engineering
- Systems science
- Computer Science
- Software / Computer systems
- Software projects
- Companies, organisations, business
- Project management
- Stakeholder Management
- Information Research & Management

Techniques
- Use Cases

... Standards
- UML
- BPMN
- BABOK

... “Thinking like an Engineer”
- Problem Solving
- Critical Thinking
- Systemic Thinking
- Synthesising
- Abstract thinking
- Methodical thinking
- Analytical thinking
- Creativity
- Sound judgement / Common sense
- Engineering skills
- Strategic thinking
- Thinking at different levels of detail, from different perspectives, dynamical and evolutive thinking
- Organisational skills
- Autonomous thinking

Ability to find out what really matters and ability to make the difference between what doesn’t matter and what does in any domain and the ability to keep focussing on what matters.

Ability to learn what matters.

Ability to envision the future.

Ability to lay out paths

Critical for Recruitment and Professional Development
Single Most Critical Skill

“Problem Solving”

Including all sub-skills like

- Critical thinking
- Abstract thinking,
- Systems thinking
- Holistic & Systemic thinking
- Sound judgment
- …

Must be super-developed
>1000 x stronger than average problem solving skills

Critical !!

One can’t be a (competent) analyst, engineer, architect, modeller, designer without this skill. (S)He may do more harm than good.

KEY TECHNIQUE: QUESTIONING
Analysts to Avoid

- Seeking to please people
- Expecting to be told what to do
- Liking to be told what to do
- Seeking to comply
- Doing what is expected
- Accepting everything at face value
- Thinking in terms of black or white, all or nothing, 0 or 100%, yes or no
- Having very quickly make up their mind
- Making a lot of assumptions and being unaware of them (it’s fine to make assumptions deliberately)
- Believing the majority is always right
- Confusing knowledge with understanding
- Seeing what is happening, not why things happen
- Thinking locally, short-term
- Interpreting and confusing terms and statements
- Following the rules, procedures, methodology (That’s safe. Avoiding being blamed)
- Copying colleagues and habits (spread of ignorance and bad habits)
- 9-to-5 mentality
- Not caring about the product and result. Low norms: “It’s fine if it works”.
- Not seeking to understand the discipline
- ‘Doing’ is more important than ‘Thinking’
- Can’t work autonomously, need continuous help
- Fearing problems and complexity
- Fearing too big change
- Lacking of focus and direction
- Unable to take decisions (some hesitation is certainly fine!)
- Systematically consulting others for their opinions about problems. Fully relying on them for solutions.
- Spending a greater importance on opinions rather than on facts
- Going along with the simple but wrong answers
- Being glad with the first answer or solution that comes to the mind
- Systematically buying or copying solutions as approach to solve problems
- Calling everything that is related a ‘solution’ even if it doesn’t solve anything or creates more other problems
- Solving symptoms
- Preferring simple or quick ‘solution’ (design debt)
- Doing more of the same, give up or throw away the ‘old’ and replace it (don’t fix it, don’t change the thinking, let alone the beliefs)
- Not having all the necessary social AND thinking skills
Most of the standards, frameworks, … majorly are how-to-guides. “Follow the guide / the majority / the present tendency! ” – mentality. Knowledge level only doesn’t suffice. It leads to “cargo-cult”.

### Competency Development

<table>
<thead>
<tr>
<th>KNOWLEDGE – level</th>
<th>Theoretical</th>
<th>Practical</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;How-to&quot;</td>
<td>Frameworks</td>
<td>Experience, habits</td>
</tr>
<tr>
<td></td>
<td>Standards</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BoK’s</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Methodologies</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Techniques</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Languages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tools manuals</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Courses</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>INSIGHT - level</th>
<th>Ability to perform activities rightly</th>
</tr>
</thead>
</table>

- Meta-thinking
- Critical thinking
- Questioning and investigating the assumptions, beliefs
- Questioning
  - Why ...?
  - Is it true that .... ?
  - What if ... ?
  - What is the nature of...?
  - What is the relation ...?
  - What is the relative importance ...?
  - What is the true purpose of ...?
  - ...

| Ability to adapt, to do the right things right and get higher results |

The development of the “insight”-level is the most difficult but also the most critical
Projects are a way to build and to transform the company. Projects are a way to execute strategies.

Projects skills and/or engineering skills are critical

A project is not a social gathering of nice people. It is a collaboration of experts ... at least if experts have been recruited and/or assigned to the project.
A Project?

PROJECT MANAGEMENT’s PERSPECTIVE

General definition

• Endeavour
• A process
• Producing a unique product
• Temporary (has begin and end)

• Progressive elaboration
• Executed by people
• Risks are inherent

A project is more than a process. A project is:

• a temporary organisation
• functioning in an environment
• a process to produce a unique agreed outcome
• using resources
• as efficient and effective as possible and limiting risks
• an initiative that must be set up, planned, guided and managed.

COMPANY’s PERSPECTIVE

(IT) Projects are

• A mean to execute (parts of) a strategy
• A way to develop new products and services
• A way to create assets
• A way to innovate
• A way to build (a part of) the company / organisation
  • to strengthen, improve and transform / adapt / change the company/organisation.
• An INVESTMENT!!

Conclusion:
The capability of managing and executing projects is VITAL, a matter of life and death; a matter of disappearing, surviving or flourishing for the company.
Project Overall Process

1. Project Proposal
2. Project Evaluation
3. Project Selection
4. Project Prioritisation
5. Project Planning
6. Project Authorisation
7. Resource assignment
8. Project Guidance,
Management and Control
9. Project Assessment
10. Post-Implementation
Product Assessment
ASSIGN COMPANY’s GOALS to ANALYSTS’s jobs, ARCHITECT’s jobs and ISE/IT PROJECTS

Then you get creativity, innovation and much better solutions and benefits.

Give target and context and let people do their job.
Executing a Project?

Objective – Goals
Goal structure/model

What must we do?
During execution: Provides focus

Cost / Benefits

Feasibility

Risks

Is it worth to do it?
Execution: Should we continue?

Resources

Work Environment

Process

Let’s do it?
Execution: plan, organise, execute, control

Notes:
Other aspects to be considered: strategic fit, criticality, priority related to other projects
### Project & Product Success

<table>
<thead>
<tr>
<th>PROJECT success</th>
<th>PRODUCT success</th>
</tr>
</thead>
<tbody>
<tr>
<td>According to scope, time, budget and quality</td>
<td>Maximum Practical value</td>
</tr>
<tr>
<td>According to plan</td>
<td>Maximum benefit</td>
</tr>
<tr>
<td>Project executed without (big / unacceptable) problems</td>
<td>Cost &lt; Benefit</td>
</tr>
<tr>
<td>Improves competencies (learning)</td>
<td>Strengthens the company</td>
</tr>
<tr>
<td>Measured in duration: several months, few years</td>
<td>Opens up new opportunities</td>
</tr>
<tr>
<td>Measured at the end of the project, sometimes based on estimates made early in the project</td>
<td>Improves competencies (learning)</td>
</tr>
<tr>
<td></td>
<td>Measured in duration: over several years</td>
</tr>
<tr>
<td></td>
<td>Measured over a long period of time (once operational)</td>
</tr>
</tbody>
</table>

**Notes:**
Defining the right success criteria is important. Measuring correctly. Learn the right lessons. Put it in perspective: project success = more important than product success? On what type of success does the company/stakeholder focus most?
## Project Objectives

### OBJECTIVE
“mission”

<table>
<thead>
<tr>
<th>Objective</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Develop the demanded software application</td>
<td>demanded software application</td>
</tr>
<tr>
<td>Solve a stated information problem/need</td>
<td>Solves the stated information problem/need</td>
</tr>
<tr>
<td>Contribute to a specific department’s goal</td>
<td>A contribution to the department’s goal</td>
</tr>
<tr>
<td>Consolidate the enterprise’s systems</td>
<td>Consolidation of the enterprise’s systems</td>
</tr>
<tr>
<td>Innovate and Drive Business</td>
<td>Innovation and improved business</td>
</tr>
</tbody>
</table>

### Low objectives leads to low results
### High objectives leads to high results

The objective assigned to the project determines the result produced by the project.

The stated **objective** is habitually the **maximum** that the project will deliver.

---

**Notes:**

It is better to turn Software Development projects at least into Information Systems Development Projects.
Project Objectives

1. Produce the product described in the customer’s demand
2. Meet the requirements
3. Meet the expectations
4. Produce a specific product
5. Improve company’s KPI’s or other measures
6. Contribute to a pre-determined business objective and allow to reach it
7. Drive business through optimisation, improvement, innovation, new ways of working and new products and services

Allowing only a specific or a limited way to contribute

Allowing to contribute in a broader range of ways; Offering more freedom; better use of possibilities and skills; More can be learned

ASSIGN/LINK HIGHER OBJECTIVES TO PROJECTS

RESULT DELIVERY rather than PRODUCT DELIVERY

(organisation, human resources and collaboration have to be adapted)
1. What is a successful project?
2. What leads to a project’s success?
3. What means and techniques do exist that can be used?
4. What endangers a project?
Criteria

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>On Scope</td>
<td>According to requirements</td>
</tr>
<tr>
<td>On Time</td>
<td>According to specifications</td>
</tr>
<tr>
<td>On Budget</td>
<td>Passed the Acceptance Test</td>
</tr>
<tr>
<td>According to plan</td>
<td></td>
</tr>
</tbody>
</table>

Taking into account: Taken risks, leaving a deteriorated situation behind (e.g. chaos, burn-out, people leaving, ...), ...

Does a project fails when it misses one criteria? Can the same criteria be used to define failure? (not success = failure?)

**Duration of Satisfaction:** Project closure, acceptance of product, longer period?

But....

**QUESTIONS**

Is a project executed to deliver on time; within budget; respecting the scope, demand and specifications and meeting expectations?

**Or**

Is it executed because the product, solution or change will be beneficial to the company and to its clients? (investment to build <value produced by the product, solution or change)

Is delivering within the iron triangle more important than having a product creating a maximum of value and limiting risks?

TIP: Criteria to be agreed upon and to be recorded in the Project Charter.
Product Success

- Project
- System in Operations

**Cost (Investment)**

*Producing benefits; takes time; income is variable; cost of operations*

Product is Profitable? **Total Cost < Produced Benefits**

**Simplified System’s Lifecycle (financial perspective)**

- Project
- Introduction
- Growth
- Maturity
- Decline

- Income
- Investment
- Operational Cost
- Payback period

Remarks:

1) During the life changes may be implemented. Changes have a cost, an increase of benefits (or lowering of cost) and a payback period.

2) Each innovation may bring in an increase of income in diagram of the system, or, for example postpone its decline.
Product Success

CRITERIA

A PRODUCT / SOLUTION is SUCCESFULL if ...

• Working software

• Meeting the business demand

• Meeting the business stakeholders expectations
  Which ones? Have the expectations (not requirements, but real expectations) been recorded and validated before the project started?

• Improving the activities of the business/organisation
  smoothness, flexibility, efficiency, effectiveness, safety, respect of rules and laws, ... and a decent quality.

• Allowing the department to reach its objectives

• Allowing the company to reach its objectives

• Improving the value and exploitability of information

• Creating value for the company

• Creating value for the company’s client

• Fully exploiting the opportunity

• Strengthening is the company

Meeting the different criteria require a different level of thinking, mind-set, norms and values, skills, collaboration, approach, methodology, set of objectives, ...

Notes:
Can the same criteria be used to define product failure?
Does it solve a problem or need? Does it contribute to the sustainability of the company?
When evaluating the contribution of the solution/system, other factors that influenced the business results have to be taken into account.
Project Scope

PURPOSE

**Delineation.** Establishes the project’s content and boundaries.

It tells what’s inside the project and what’s outside, what should be done and what not.

• ≠ of a simply high-level description of the project
  
  If the ‘scope’ doesn’t mark the boundaries, then it’s not a scope.

• **Solidly linked to objectives!!**

• Scope is **multi-dimensional**

  • Problem domain
  • Stakeholders
  • Organisational boundaries
  • Systems and interfaces
  • Technological boundaries
  • ...

• Processes
• Set of features
• Project activities
• Project deliverables
• Degree of automation

**SCOPE TEST** (examples):

• Is this feature part of the project or not?

• Does this part of the interface to be developed within the project?

• Is this person/function/role to be interviewed (stakeholder)?

• Is this degree of automation to be reached?

• Is this project activity to be done?

• Is this deliverable to be produced?

• Do we need to involve this business expert as a stakeholder?

*Crucial for the Analyst who needs to work within the scope*
<table>
<thead>
<tr>
<th>STAKEHOLDER</th>
<th>POSSIBLE OBJECTIVES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company</td>
<td>To survive, to grow, to evolve, to live accordingly in accordance with its purpose and mission, to serve the society, ...</td>
</tr>
<tr>
<td>Management</td>
<td>Reaching its objectives, increasing benefits, lower the cost, executing a plan, ...</td>
</tr>
<tr>
<td>Business community</td>
<td>Conducting business, reaching objectives, ...</td>
</tr>
<tr>
<td>Informatics Department</td>
<td>Providing information solutions to the business, creating value for the business, responsibility and capability to manage and maintain the implemented Informatics solutions and to make them to evolve, implementing a strategy, innovating, ...</td>
</tr>
<tr>
<td>End-Users</td>
<td>Reaching his/hers objectives, having a good evaluation, avoid annoying work, offering a better service to the clients, release pressure on his/hers own work, ...</td>
</tr>
<tr>
<td>Sponsor</td>
<td>Financial benefit, business objectives, ...</td>
</tr>
</tbody>
</table>
Main aspects Managed by the Project Manager

The Project Manager

1) sets up this **temporary organisation** and **work environment**
2) ensures the project fulfils the **mission** by reaching the **objectives** (within certain conditions).
3) obtains, creates, organises, plans and manages the initiative
4) manages Risks, Stakeholders, Communication, Budget and Resources.
5) protects the initiative (shield, filter, ...) and remove all negativity in it
1. Project Management **Integration**
2. Project **Scope** Management
3. Project **Risk** Management
4. Project **Time** Management
5. Project **Human Resource** Management
6. Project **Procurement** Management
7. Project **Cost** Management
8. Project **Quality** Management
9. Project **Communication** Management

Source: Project Management Institute : Guide to the Project Management Body of Knowledge
| Initiation | Check mission, objectives  
Check management support  
Check feasibility and risks  
Identify stakeholders (role, alignment of objectives, ...)  
Obtain authority and resources  
Develop the Project Charter |
|----------------|--------------------------------------------------------------------------------|
| Planning      | Structure the project  
Define the WBS  
Develop a schedule  
Estimate resources  
Estimate costs and plan budget  
Plan quality  
Plan human resources  
Plan communication  
Plan risk management  
Plan procurement  
Review and adapt plans  
Align plans |
| Execution     | Acquire, develop and manage project team  
Direct the set up of the work environment  
Conduct procurements  
Manage the work environment  
Manage stakeholders and their expectations  
Perform quality assurance activities  
Manage changes (eg inspect and negotiate changes)  
Execute the project (analysis, design, software development, testing, migration, implementation) |
| Monitoring    | Control scope and progress  
Monitor and control resource usage  
Monitor the work environment and the project team  
Identify, monitor and control risks  
Identify need for change  
Perform quality control |
| Closing       | Learn lessons  
Close the project |
Project Management Process

- **Initial Planning**: Planning and adaptations of planning
- **Set Up**: Installation of project team, work environment, procedures
- **Planning**: Authorisation by sponsor and acceptance by PM

**Acting upon the project execution**
(= directing, facilitating and supporting the Software Development Process)

- **Launch** → **Initial Planning** → **Set Up** → **Planning** → **Handling**
- **Controlling**
- **Learn Lessons** → **Project Closure** → **Post-Implementation Product Assessment**
- **Product Delivery and Transfer**

30/10/2018
"Six Factor Model+1"
Contributing to Project's Success

1. Business Community & Input
   - Objectives, intentions
   - Realistic expectations
   - Issues
   - Constraints
   - Business knowledge & Info
   - Accountability & Commitment

2. Project Management
   - Clear vision & scope
   - Hierarchy of objectives: clear, same for everybody
   - Manage stakeholders (expectations)
   - Management against success criteria linked to business value
   - Real authority of PM and specialists
   - Structure the initiative
   - Plan & review plans
   - Organise project, team, work environment
   - Resolve problems
   - Risk Management
   - Communication

3. Process
   - Approach, methodology
   - Methods
   - Reviews & Validation
   - Feedback

4. People
   - Skills
   - Motivation
   - Understanding
   - Collaboration
   - Communication
   - Common sense
   - Analytical thinking
   - Critical thinking
   - Creativity
   - Flexibility
   - ... Hard work

5. Resources
   - People
   - Time
   - Budget
   - Tools
   - Hardware and software

6. Work Environment
   - Office space, meeting rooms, creativity room, ...
   - Software tools
   - IT environments
   - Hardware and software

The "+1"
The project MUST respect
FUNDAMENTAL PRINCIPLES

of the involved disciplines like
- information systems engineering,
- Information management,
- project management,
- change management, ...
Iron Triangle

TIME
(delivery date, milestone, deadline, schedule)

SCOPE

COST - BUDGET

QUALITY & RISK AVOIDANCE

Priorities in Triangle is product/situation–dependent (not approach dependent)

Regardless of the approach ...

ALWAYS Prioritise the Triangle

Always ask to the main stakeholder (sponsor?):
• What are the priorities? What is the most important?
• If one had to change, which one would it be?

<table>
<thead>
<tr>
<th>Highest Priority</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value-driven Goal, Quality</td>
<td>Minimal value to be delivered (at least this or more) Creating value has the highest priority. Cost and time are adjustable to create the maximum of value. If critical applications, long term, sustainable, has to evolve, to be built on top, to be integrated, ...</td>
</tr>
<tr>
<td>Time-driven</td>
<td>Delivery at deadline or earlier. For events, one shot</td>
</tr>
<tr>
<td>Cost-driven</td>
<td>Deliver accordingly to budget (or cheaper). One shot, (really) temporary, “cheap” is expensive.</td>
</tr>
</tbody>
</table>

Main stakeholder may need advice about the priorities.
Not all three (or four) can be fix.
The stakeholder’s perspective is only one perspective. He/She may not understand all the factors, aspects, risks and implications of a choice.
But if an opportunity or issue occur, decisions will always be taken against these four aspects.

Fix all Three (Four) is a Recipe for Failure !!!
If a project runs late and the scope is fixed (has to be delivered), then one can increase the time, increase cost, lower quality (sometimes in a hidden way. Creates debt) or a combination of this.
QUESTION 1: Heart surgery is necessary. The patient is offered for the **SAME** operation two options:

a) operation of 30’ and Cost 500 €

b) operation of 1h30’ and Cost 1000 €

Difference: in (a) the surgeon works very fast. In (b) (s)he has the time to verify, to do things right, ...

What would you chose if you were the patient (provided you can afford the 1000€)? Why? What is the lesson?

QUESTION 2: A team gets a software development mission.

1. It delivered a valuable, high quality, innovative, software system after 6 months.

2. It delivered an awkward system with reduced scope on time and on budget

3. The team could, together, overcome unexpected challenges, had the opportunity to do things they didn’t think they could do, to learn and to conceive and create a valuable good system without having to worry too much on scope, time and budget.

In which case would the team have the most satisfaction and pride? Why? In what order? Why? What is the lesson?
Scope can be determined ... provided the focus is on what is necessary, rather than on wants, demands, preferences, ... and provided a decent SA-SD has been done.

Required Quality is relatively easy to be determined.

Time = mainly \( f(\text{Scope, Quality}) \) + non-linearity + uncertainties

Cost = mainly \( f(\text{Scope, Quality, Time}) \)

Often the scope and quality are determined first. Then the required time is estimated and then the cost.

What element of the Iron Triangle should we best take as fixed and which as variable?
Iron Triangle

We can negotiate on 5 aspects.

So, there are SIX questions

1. Can we adapt the scope? For example, by postponing some parts.
2. Can we simplify the solution while keeping the scope?
3. Can we reduce the project duration?
4. Can the cost be reduced?
5. Can we be satisfied with reduced quality and where can the quality be lowered and until what point?
6. Or can any of the answers to the question above be combined to have a satisfactory result?

- Better is to try to do a good analysis, better estimations in the first place and reduce everything that may poses delays.

- A project runs out of scope, time or budget for a reason: inefficient analysis, bad estimations, bad leadership, bad planning, work environment, insufficient collaboration, received information, slow or bad decision making, ...

Maybe first find and deal with the causes rather than trying to solve the consequences. Then we can learn valuable lessons !!

- What if we could assign lesser importance to the iron triangle and more focussing on the product?
Why Do We Have a Very Rigid Way of Dealing with Deadlines?

Why Do We Think so Obsessively over Deadlines?

• Estimated Deadlines
  • Not Delivering within a Deadline (on time) is a FAILURE

• Sprints have a fixed length

• Time boxing

Deadlines are used to create pressure.

Pressure is a bad substitution for motivation, for drive, for being interested, for attitude.

Pressure inhibits learning, curiosity, innovation and may destroy quality.
It may create conflicts and burn-outs.

Does a company want/need a decent system or solution or does it want something that can be built in a period of time and at a certain cost?

Tips

• Define mission, product, release + time required

• Principle: “It’s ready when it is ready”

• Start Early - Remove all pressure – Remove all fear
Deadly Spiral of Debt, Pressure and Lack of Quality

Increased pressure
- Decreased quality
- Increasing risks
- Loss of control, manageability, ...

Increased urge to direct
- I will tell you what to do, how to do.
- I will create your work environment.

Analyst (Expert)

Dissatisfaction
- This is what I want/need.
- This is what I expect.

Lack of clarity, organisation, order, structure, ...
- Unfinished jobs, corpses in the closet;
- ... → Harder to progress

Minimising the effort
- Limits verifications
- Limits time to think
- Increases risk of conflict
- Reduces collaboration
- Reduces Innovation
- Reduces learning

(Some) PRESSURE

TIME ISSUE !!
DON’T USE STRICT DEADLINES AND MILESTONES

Progressing correctly and steadily is most important

Deadlines and milestones may serve as a guide, as an ideal situation.

Deadlines and milestones are (often) based on wishes or on estimates.

It is (often) better to miss a deadline and

• to do a sloppy job .. consequences and cost will come later
• to be able to implement a better solution creating much more value
• to be able to innovate.

Missing a deadline or milestone can be more beneficial for the company than meeting it.

Notes:
Although, there are circumstances when deadlines are critical.
PM – Analyst Collaboration

Notes:
Some tasks, like Scope definition, may, originally been defined prior to the project start.
Examples: The Analyst notices that a scope change might be advantageous. This affects the agreements, plans, required resources, ... The Analyst has to discuss this with the PM. The stakeholders will also be involved in a discussion.

The Analyst detects a risk. This has to be discussed with the PM to see what measures can be taken.

*: The issues the Analyst discusses with the PM are essentially those which impacts project management. The Analyst may inform the PM about some problems related to the logic and product, yet they have to be solved by the Analyst. A PM doesn’t solve an issue in process modelling or information modelling.
Goals provide either an objective, the idea of the pursued final desired outcome, or a direction. It brings focus. It allows to align all the means and efforts in one direction. It brings focus, stability, effectiveness and allows efficiency.

A goal avoids wasting resources and efforts in different and/or opposite directions or on opportunities of lower interest. Frequently changing the goal (and the direction) is the same as having no goal.

However, fixing the goal, cutting it in stone, is also a misuse of the concept. It makes no sense to continue to pursue a goal that has been found being useless. Small goal changes or goal refinements are normal. Frequent important goal changes are a symptom on an underlying problem. 180° change of objective can sometimes be necessary, but should happen very rarely.

Goal and strategy (or other plan) shouldn’t be confused.
Traditional Representation in Methodologies

Analysis:

In Software Development ...

- Design activities can be performed during an Analysis phase
- Analysis activities can be performed during a Design phase
- Programming can be done during a testing phase (for example, to correct bugs)
- Analysis can be done during a Testing phase to know how to correct a bug in the higher logic.
- And so on ...
Project Phases

Project Phases do / are

- Indication of the main **FOCUS** and **LEVEL of THINKING** now required
- Indication of the **MAIN TYPE of ACTIVITIES** currently being executed
- Indication of **TYPE of OUTPUT** to be expected soon
- Indication of the **PROGRESS: WORK** (and **TYPE of WORK**) that has been done and what is being done
- Facilitate the **MANAGEMENT** of projects
- Result from a Project Management Decisions

Project phases do NOT/ are NOT

- Do not limit the type of activities being done
- Not meant to provide a precise measure of progress (in time, cost or effort). Other measures do.

A **PHASE** is also a **BUILDING BLOCK** that helps to structure the software development approach. Each phase has its **purpose** and **goal** and can be applied on a subject, component, issue. A phase can also be executed at a **certain level of detail** or from a **certain perspective**. The phases can be defined and assembled in many ways.

**Phases offer flexibility and scalability to projects**
Project Phase & Activities

**PHASE 1**
- Transition by Intermittent Execution of Other Activities
- Fuzzy Transition
- Delayed Sequential Execution
- Length of phases?
- Single and double phase gate
- Gradual Transition (activities are clearly of different nature)
- Fuzzy Transition
- Main Activities & Background Activities
- Transition by Intermittent Execution of Other Activities
- Transition by Intermittent Execution of Activities of various Types
- (Semi-) Parallel Phases

**PHASE 2**

= PM decision

Boundary between 2 phases

Sequential Execution
- Activity type 1
- Activity type 2

**Phase 3**

Or a combination of some of these transitions
Project Phases?

Project Management Methodology

(\textit{simplified framework})

“\textit{How to manage a project?”}

\begin{itemize}
  \item Project Initiation
  \item Planning
  \item Monitoring and Controlling
  \item Executing
  \item Organising
  \item Closing
\end{itemize}

Project Management Vision on the Project’s Process

\textbf{How PM sees, organises and plan the project. (example: Deciding to move the project to next project phase)}

\begin{itemize}
  \item \textbf{PM Perspective: PM Purpose, PM decisions, PM Needs}
  \item \textbf{IS / SW Dev. Perspective} = some correspondence
\end{itemize}

Software Development Perspective on the development process

\textbf{What project members do. Nature of activities, focus,}

\begin{itemize}
  \item \textbf{Phases defined as Activities Types}
\end{itemize}

\begin{itemize}
  \item \textbf{Analysis}
  \item \textbf{Design}
  \item \textbf{Building}
  \item \textbf{Testing}
  \item \textbf{Deployment}
\end{itemize}

\textbf{Project team’s perspective:}
\textit{What is necessary to be done, what types of activities are executed now.}

Standard Methodology

\begin{itemize}
  \item \textbf{Methodology}
\end{itemize}

\begin{itemize}
  \item \textbf{Analysis}
  \item \textbf{Design}
  \item \textbf{Building}
  \item \textbf{Testing}
  \item \textbf{Deployment}
\end{itemize}

\textit{The standard, the template, the theory, the whole set of artefacts proposed as a basis for IS/software development endeavours}
Project Phases?

1) Selected Methodology

2) Adapt the selected methodology to the project

3) Use the Adapted Methodology to organise Phases and Activities in the Project Plan

4) Use the Plan to Organise, Manage and Guide the Project

5) Execute Plan

**Goal:** to have a methodology/approach/process useable to develop the product

**Goal:** to have a plan useable to manage the project

**Project Management’s Perspective:**
*PM Purpose, PM needs, PM decisions*

**IS / SW Dev. Perspective**

Essentially done/driven/determined/decided by the project team members (Architects, Analysts, Developers Leaders, ...)

**Goal:** to have a methodology/approach/process useable to develop the product
Structuring the Project

**Project**
- **Releases**
  - Subsequent projects on same product.
- **Sub-projects**
  - Small projects within larger project. Each has independent project documentation, planning, ...
- **Phases**
  - Subsequent deliveries within a same project.
- **Tracks**
  - Tasks are grouped by nature. A track can have an (internal or external) delivery (E.g. a component).
- **Work Packages**
  - Smaller set of tasks planned together assigned to a specific person or group.

**STRUCTURING A PROJECT MAKES A PROJECT SCALABLE**

**Notes:**
Structure is the key to scalability.
Without clear structure, no scalability is possible.
The **WBS** can be organised in different ways:

- per sub-project, track, ...
- **component**-based
- by development **phase**
- by **nature of work** – output / deliverable-based

Example: All test activities are gathered together in one branch of the WBS. All Analysis activities are gathered in another branch. All this, regardless of the phase in which the work is executed.

- by **executor**
- ...
- Or a mix

Remark:
The WBS may or may not reflect the work executed in a project phase.

**Choice to be discussed:**
- What suits the project the best?
- What is the easiest to manage the project?
- Impact reporting!
Traditional Planning

Why this doesn’t work?

Assumes:

- All information necessary for planning can be gathered and taken into account
- The planner excels in planning
- Plan is always right
- The execution of the project (the reality) can and has to be adapted to the plan.
- People’s mind (thinking, creativity, …) works at a linear pace
- People work and collaborate at a linear pace
- No skeleton in the closet or other difficulties, obstacles, setbacks
- Executing the plan = Achieving the goal
- Executing “as planned” is more important than reaching the objective.

Ignores:

- Uncertainties and risks are greatest at the beginning of the project (exactly when the project plan is defined)
- Limited understanding of the initiative in the beginning of the project
- Limited skills of the planner
- Unforeseen events may happen

Cause of Death March

How to prevent this?
Traditional Planning

Why do Planning Fail?

- We underestimate the work to be done (size, complexity, ..)
- We overestimate ourselves (competencies, knowledge, insight, abilities, ...)

How to execute a project successfully?

Multiply the estimates by 10 (or by 2)

Then ..

- Why to make a plan?
- Will it solve the estimation abilities, the abilities to manage buffers and the project management skills?
- = symptomatic problem solving

Wrong ways of using a plan:

- Not using a plan. Leave it on the disk or in the drawer.
- Freeze the plan and stick to the plan.
Traditional Planning

Why NOT FREEZING a plan made early in the project?

A Plan is PREDICTIVE AND ADAPTIVE

Any plan made in a situation with reduced information; a lot of uncertainties, unknowns, risks, unsettled issues and/or for an endeavour and environment that isn’t under full control, is very likely to change. It will have to be adapted as more is known, as more is being settled, as the situation unfolds. It has to evolve to continue to be as realistic, feasible as possible and suitable to the situation.

Notes:
Re-emphasising: “Validation” is different from “Freezing” and does not imply it. A plan can be validated. However, a plan is a dynamic, adaptive, evolving tool.
Why this doesn’t work?

1. **RIGID Planning** of hard to estimate activities in dynamic environment

2. Intellectual activities and collaboration are absolutely **non-linear activities**.

3. Requires information about the product, outcome

Analysis and design provide information required for planning. But these activities happen **after the planning phase**.

**How to solve this?**

A part of the solution is to do a HL pre-analysis & HL pre-design (before planning). Uncertainty is decreased, but a lot of it still remains.

A plan is not static. It has to evolve, to be corrected, to be refined, to be adapted in order to converge to the goal. **Plan Dynamically**.
Planning

A project is a one-time (unique), uncertain and risky initiatives producing a unique product usually using scarce resources. There is no established approach for them.

Need:

• To achieve the goal of the project
• To be effective and efficient
• To reduce risks
• To organise and control the project

How? Establishing a plan

The purpose of planning is to execute the progress effectively and efficiently while controlling and/or reducing the risks and to move swiftly and steadily towards the goal. It increases the chance for success to achieve the goal.

The plan is right but the project (the reality) is wrong. The reality has to proof the plan was right. The goal of planning is NOT to have a plan.

It’s all about the activity of planning. Establishing the plan forces the planner to investigate, to think, to foresee, to structure and to organise.

Therefore we need to know things like:

• What process is required to execute the project
• What and how much resources are needed
• When are these resources needed
• We need estimates (we need estimate skills)

Warning: Planning aims to save time and resources. But planning takes time. The value of the plan and planning should be greater than the effort spent in planning. Spending too much effort in planning may kill the benefits.
Dynamical Planning

1. Determine goal
2. Investigate and gather information
3. Plan
4. Execute Plan
5. Check project vs plan
   - Difference between project (reality) and plan
     - Y
       - Can the project/reality be adapted to the plan
         - Y
           - Adapt the PLAN
         - N
           - Adapt the PROJECT (reality)
     - N

End of Project or Plan

Plan versus execution

Adaptation of project execution

Adaptation of the plan
Factors influencing the progress and success of a project: (some are external to the project):

- Availability of stakeholders
- Conflicting goals and/or visions among stakeholders
- Time to align visions and get agreements among the stakeholders
- Pace of decision making and quality of decisions
- Frequently changing priorities and changes
- Hidden agenda’s
- Collaboration with stakeholders
- Pace of receiving information & quality of received information
- Size of the problem area & solution
- Skeletons in the closets
- Degree of knowledge of systems and organisation/company
- Quality of existing source code
- Collaboration within the project team
- Work environment
- Skills of the team
- ...
### Distribution of Effort

<table>
<thead>
<tr>
<th>Phase / Types of Activities</th>
<th>Effort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-Analysis (Requirements)</td>
<td>5-10%</td>
</tr>
<tr>
<td>Analysis</td>
<td>15%</td>
</tr>
<tr>
<td>Design</td>
<td>10%</td>
</tr>
<tr>
<td>Construction + Unit Testing</td>
<td>25-30%</td>
</tr>
<tr>
<td>Integration + Integration Tests</td>
<td>10%</td>
</tr>
<tr>
<td>Systems Tests</td>
<td>15-20%</td>
</tr>
<tr>
<td>Acceptance Tests</td>
<td>5%</td>
</tr>
<tr>
<td>Implementation</td>
<td>5%</td>
</tr>
<tr>
<td>Documentation</td>
<td>5%</td>
</tr>
<tr>
<td>Training</td>
<td></td>
</tr>
</tbody>
</table>

- Limiting the early phases will reduce the insight. A part of the learning shifts then downstream to later phases causing late investigation additional rework due to changes (corrections due to bad decisions).
- Project estimations are often too optimistic or too many impediments will appeared. Therefore the global estimate is often multiplied by a factor of 1.5 to 2.5.
- Seasoned project managers know not everything is foreseeable. They use time and money buffers to deal with it. These buffers can be used to regulate pressure. They, and no one else, manage them.
- Bad practice: Reducing Testing to meet the delivery milestone.
- Bad practice: Having a “Requirements” phase. Requirements is a technique or an outcome of Analysis activities.
**Planning: Estimates**

**Project Schedule** *(not “project execution”)*

\[
\begin{align*}
\text{Project start} & \quad \text{Estimated project ends} \\
& \quad \text{t1} \quad \text{t0} \quad \text{t2}
\end{align*}
\]

**t0:**
Ideal project duration estimated by Laplace’s Demon

**t2: overestimation** *(team has more time)*
- More verification
- More aspects can be taken into account
- More learning
- Idea maturation
- More tests
- Experimentation, exploration, optimisation
- Time for alternatives
- Deployed later
- Risk for wasting resources

**t1: underestimation**
- Faster delivery
- Insufficient resources
- Higher pressure. Working faster
- More assumptions are made
- Stress, conflicts, burn out
- Lesser investigation
- Lesser verification
- More superficial thinking
- Fewer tests *(risk of releasing solutions with more bugs)*
- Higher risk for inappropriate, awkward solution, or even a non-solution creating more damage.

Even if \( n < m \), the drawbacks and risks of t1 are more important than for t2.

Better to overestimate, than to underestimate.

**Notes:**
Laplace’s Demon is a superintelligence that knows everything of the whole universe at all times. It can deduce everything that is deductible from this knowledge.
Project Estimations

Final duration and real expenses of the project (represented linearly for the sake of simplicity)

Too short estimation of time or budget. Often done by business stakeholders instead of by the IT department and/or project team. It can be negotiated, which is also bad. Creates pressure, insufficient resources, ... Too short estimates are a reason of project failures, particularly when they are frozen. A cause of “Death March”.

Additional budget and/or time are allocated (by negotiation?) until it meets the final duration and cost. Not only the short estimates have negative effects, allocating budget and time requires also a lot of additional time and effort.

Time and budget estimates tend towards the final real time and budget.

Rarely time and budget are overestimated.

Reasons of project failure (related to estimates):

- Underestimation
- Estimations done at the beginning of the project when greatest uncertainties and most unknowns.
- Both in combination with frozen estimates.

Dealing in a rigid way with estimates is very problematic
A slight underestimation may quickly pose a high pressure. If a project’s duration and budget are underestimated, the risk is likely to remain high. Right estimations may become underestimations when hindrances occur.

Risks will quickly decrease with right estimations. Even on delivery some risks remain. It is impossible to exclude all risks.

With overestimations, the risks may continue to be lowered (unsure). The question is whether the effort and time are worth this reduction.

The assumption is made that the team works well and don’t waste time. Free time can be used for experimentation (innovation), improvement, ..
t0: expected cost and value is created

t1: likely to have much higher cost, and much lesser value and increased risk. Risks of getting lesser net value if seeking to save too much on project.

t2: Cost can be lower or somewhat higher, but value is more likely to be significant higher (or risks much reduced)

Probably ...

Estimating too short is more risky than to estimating too generously.
## Project: Estimating

### UNDERESTIMATION

<table>
<thead>
<tr>
<th>Duration</th>
<th>Incorrectly estimating duration can lead to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Too little</td>
<td>Unchecked info, superficial insight in problem and impact, weak foundation, inelegant design, lesser features, short cuts, more errors, work not done, ...</td>
</tr>
<tr>
<td>Realistic / Acceptable</td>
<td></td>
</tr>
</tbody>
</table>

### OVERESTIMATION

<table>
<thead>
<tr>
<th>Duration</th>
<th>Incorrectly estimating duration can lead to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Too much</td>
<td>Waste of time (likely)</td>
</tr>
</tbody>
</table>

### Cost

<table>
<thead>
<tr>
<th>Insufficient experience and competencies</th>
<th>Realistic / Acceptable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wasted money</td>
<td></td>
</tr>
</tbody>
</table>

### Difficulty

<table>
<thead>
<tr>
<th>Oversimplification, real complexity is postponed</th>
<th>Realistic / Acceptable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overly complex (unlikely)</td>
<td></td>
</tr>
</tbody>
</table>

### Quality

<table>
<thead>
<tr>
<th>Lack of quality, inappropriate design</th>
<th>Realistic / Acceptable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Luxury (likely)</td>
<td></td>
</tr>
</tbody>
</table>

May render the product completely useless and/or lead to serious post-project problems

Waste, but unlikely to menace the project

Devastating effects are on this side.

“Faster is Slower”
“Hasty speed is seldom good”
“Qui va piano va sano”
“Cheaper is more expensive”
There are lower limits:
• In time: 9 women can not make a baby in one month
• In cost: if you pay peanuts, you get monkeys
• In quality: the price to pay will come later.

Fast becomes Slow
Cheap becomes Expensive
Many Features becomes a lack of quality
Planning: Estimations

• Estimations are **ALL WRONG by definition**, but they should be useful.

• **Unrealistic estimations are worthless.** They do more harm than good.

• Estimation must help to manage the project. If they don’t, something is fundamentally wrong. *The same is true for all kinds of plans, methodologies, methods, tools, ... Often the usage is wrong.*

• The point is to **reduce the error margin** to get an estimation that is as optimal as possible.

• The fewer and **uncertain** information, the greater the uncertainties (beginning of the project), the more **unreliable the estimations** are.

• The **more knowledge and insight and the more things are settled** we have about the objective, the product, the situation, and so on, the **more precise** the estimation can be.

• Improve the estimation techniques, improve estimation skills and decrease uncertainty and risks.

• Don’t stick with first estimation. Get the estimations reviewed and confirmed. Or, **re-estimate regularly**.

• **Estimations are, by their nature, never negotiable.** Maybe one can negotiate the scope, resources, features and other characteristics. Estimations are the deductions of these.

**Notes:**
Wrong estimations is not about “being wrong”.
Reviewing and re-estimate a project may lead to adapting the project plan. Yes, this is more work for the project manager. But this is part of the job.
Missed Deadline

Which case is the worst? A or B?
Missed Deadline

Missing a smaller deadlines is not as bad as missing a big deadline. But this is misleading.

Case A

By cutting the process or time in slices (like short iterations), the deadlines can then be spread. The missed delays are smaller, but they are more frequent as well. It hides the true effect of the delays.

Small differences are not always documented (recorded, written, ...) They aren’t (always) visible.

Case B

This principle exists in other situations: energy, effort, waste, costs, ...
Course corrections may imply scope changes, adaptations of project plans, budgets, schedule, delivery date, ...

Analysis reduces the likelihood and the amplitude of deviations from the ideal course.

Course corrections shouldn’t be caused by “changes of mind” or preferences.

After Analysis the likelihood on changes should be severely reduced. It should be even more reduced after the design.
Software Development Projects should be more than implementing and delivering a software application.

OCM to ensure a better collaboration between “IT” and the business community.

OCM to ensure the acceptance of the solution/product (by the end-users) to contribute to a better integration in the human organisation and to ensure feedback of the product/solution/system.

Training might also be useful (about the product and about information usage) to ensure a good integration.

Notes:
This view should influence the way the project mission and scope are defined. It may also influence the role of the Project Manager and of the Analysts, Architects, ...
Risks are inherent to projects.

Major risks are often easily and quickly identified. This doesn’t mean that all major risks can be identified immediately. A lot of smaller risks (smaller impact) may remain hidden for a while. Risks may surface at any time in the project.

It’s a good habit to deal with major risks early in the project. Risks with minor impact can be dealt with later in the project.

The acceptance of risks depends of the company, of the stakeholders and of the product.

The severity of the impact, the probability and the cost to mitigate the risk are key factors in the evaluation. Other factors, like professionalism, responsibility, moral values and reputation, may also play a role.

Risk identification, which has preferably has to happen as early as possible, require an understanding. This is provided by a decent upfront correct Analysis.

"Risk comes from not knowing what you're doing." - Warren Buffet
Learned Lessons

ONE OF THE **KEY ACTIVITIES** IN THE PROJECT

It increases the chance of future successes

**How?**

- Personal Improvement
- Development of company’s capabilities
- Lessons have to be shared (learn from someone's else mistakes)
- Allows to avoid to make same mistakes in all the future projects (saves cost, resources, time and problems in the future)
- Allows to tackle more important challenges in the future

The project is the way companies use to develop and to transform themselves. If project and engineering skills lacks, the company can’t adapt and evolve. If it can’t adapt and evolve, it is doomed to die.

- What went wrong? What caused frustrations, dissatisfactions, ..? What could have been done better?
- Why did it went like that?
- How can it be improved? What had to be done? What was necessary? What would had been a better way?
Two Disastrous Triangles

Reduction

Reduction of cost, effort, time is a receipt for disasters. In a short time, it may seem to save money or time. Soon this may turn out to be much more expensive.

Short Cuts

Looking for shortcuts to speed up is a losing strategy

Diluted version

Faking, imitating, cargo cult, pretending: Often artefacts, products, documents are produced without doing the real work, without understanding the true purpose of it, its meaning and value, without the real substance.

Low Quality

The required level of quality is often underestimated. High quality is not wanted because it is expensive and because it is confused with luxury.

Notes:

There are true techniques that facilitate or speed up work. Some ways, like CBD, speed up work in the future. Real professionals can make a difference between short cuts and true techniques.
INFORMATION SYSTEMS DEVELOPMENT FRAMEWORK
### Approach, Framework, Methodology

<table>
<thead>
<tr>
<th>Philosophy</th>
<th>Set of values, ideas and principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framework</td>
<td>A basic global description of an organisation or structure. It may include patterns useable to conceive a global organisation for a specific initiative.</td>
</tr>
<tr>
<td>Approach</td>
<td>Vague description of a global process</td>
</tr>
<tr>
<td>Methodology</td>
<td>A precise description of a global process, consisting of activities and methods. It can be enriched, for example, with artefacts, principles, standards or rules and techniques.</td>
</tr>
<tr>
<td>Method</td>
<td>A particular process with a well-determined purpose to accomplish something specific and consisting of several ordered steps.</td>
</tr>
</tbody>
</table>

### Goal
- To Facilitate the project, the work
- To help the team
Meeting the customer, end-user or SME to discuss the overall project

Meeting the customer, end-user or SME

Get feedback from the customer, end-user or SME

Present work done to the customer, end-user or SME

Develop a discussed / agreed part

Design models, mock-up screens or programming

This works for small teams, simple problems, simple and local solutions.

For larger, tentacular, multi-disciplinary solutions crossing organisational borders which share components or resources and interacts with other systems a more holistic, supportive and organised approach is required.
Some operators (users) can operate the machine, but have no clue of the internals of the systems, thus of the required implemented logic.

A spacecraft can’t be sent into space with only the most important features required for taking off installed. A spacecraft’s software system must be complete before it is launched.

DIFFERENT PROJECTS → DIFFERENT METHODOLOGIES

- Different philosophies, vision, values, priorities, constraints
- Different levels of integration
- Different spread of knowledge
- Different types of stakeholders
- Different ways of deployment, deployments constraints
- Different risks and risk acceptance (criticality, quality requirements)
- Minimal required software to be complete, functional and valuable
In Software Engineering doing things by the book and thinking dogmatically is a recipe for FAILURE regardless of the approach.

Be PRACTICAL and ADAPT so that your processes and tools fit your situation.
Software System Lifecycle (1)

First Build Project

Installation

Decommissioning Old Release

Systems Management

Operations

Support

Maintenance

Post-Implementation Assessment

Evaluation

New Release

System Decommissioning

Subsequent Project
- Optimise
- Adapt
- Expand
- Connect
- Replace (parts)
- Upgrade
- Merge
- Split
- Migrate
- Port
- Re-engineer

Lifecycle ≠ Development Cycle !!
Software System’s Life (2)

BUSINESS COMMUNITY
Business models, products, services, alliances and partnerships, objectives, plans, policies, decisions, guidelines, ideas, choices, complaints, ...

BUSINESS ENVIRONMENT, MARKET, SOCIETY
Laws, Ecology, Technologies, Infrastructure, Economy, Competitors, Trade Unions, Media, ...

Operations / Usage
Changes triggering projects

1st Build Project → Project → Project → Decommissioning

- Monitoring, Control, Maintenance, Configuration, Support, Training, System/Application Management, Administration, ...

Evolution of a system through projects

ENSURING security, continuity, reliability, performance, responsiveness, consistency, accessibility, efficiency, interoperability, adaptability, up-to-date, fitness, ...
Software System’s Lifecycle (3)

First Build Project

- Systems Management

- Operations (usage by the business community)

- Support

- Systems Maintenance

- Subsequent Projects
  - Optimise
  - Adapt
  - Expand
  - Connect
  - Replace (parts)
  - Upgrade
  - Merge
  - Split
  - Migrate
  - Port
  - Re-engineer

System Decommissioning

Responsibility of Business Community

Responsibility of “IT”
System / Software Development Lifecycle

Evaluation → Planning

Maintenance → Analysis

Implementation → Design

Testing → Building

Note: Some SDLC have lesser phases or the names may differ. However, the concept, the idea, remains the same.

Development Cycle ≠ Life Cycle !!

“Requirements” or “Requirements Analysis” are techniques, not phases !!
**Notes:**

- Systems Analysis is exercised throughout the whole project. However, it is essentially exercised during the Analysis and Design phases (whatever their names). The “Groups of activities” represent a part of the “Analysis” phase. They can be defined as a phase on their own, but the practitioner must make sure not to forget critical activities in the process.

- “Requirements” is rather a technique requiring several activities like (requirements elicitation, req. analysis, req. management, req. prioritisation, req. validation, ...)

---

30/10/2018
Philosophies in Approaches

Thinking ➔ Conceive solution
Solution is for 100% known ➔ Build ➔ Ideal case. Not possible in most cases.

Thinking ➔ Problem: fairly well known.
Solution: 80-95% of certainty ➔ Build ➔ Adapt: Minor changes & extensions
Normally, no fundamental changes

Little understanding
Little thinking ➔ Build
Prototype ➔ Experience
And Evaluate ➔ Design Final
Solution ➔ Build Final
Solution

Learn & Adapt
Prototype

Little understanding
Little thinking ➔ Build First
Version ➔ Experience
And Evaluate ➔ Final Solution

Adapt (Rework)

➡ Slow, Costly, Time and budget depending on # cycles and amount of rework, late final delivery, ...
Methodology

METHODOLOGY is a global methodical (structured) process, which is commonly decomposed in phases, sub-processes, gates, activities and milestones, and which can propose or be enriched with values, principles, rules, roles, skills, products, methods, techniques, standards, artefacts, tools, technologies, ...

It’s an organised and ordered collection of elements proposed to projects (not just one) and which can be useful to them.

Advantages

1. Facilitates transfer of Project Experience
   • The methodology accumulates this experience, insight, ...

2. Facilitates the set-up and planning of projects
   • The project can start with a ‘template’ instead of from a blank slate. Not everything need to be redefined from scratch. It’s “just a matter of” adapting the standard methodology to the specific case of the project and its needs.
   • It supports the knowledge and experience (lesser chance to forget taking something into account or to include something.)

As a way to introduce some standardisation

2. Facilitates communication
   • Usage of common terms across projects
   • Common concepts facilitates reporting to management

3. Facilitates collaboration
   • Easier to step in into a project.

4. Facilitates the management of programs and project portfolio
   • Obligatory steps and artefacts common to all company’s projects can be defined.

Risks

1. Overly trusting and relying on the methodology.
2. Assumption that a methodology replaces understanding of Software Development Processes and projects.
3. Misuses and Abuses of the methodology. Confusion between a methodology and a procedure.
Methodology Usage

- A methodology ≠ a procedure, a recipe.
  - A methodology describes (and doesn’t prescribe – unless the methodology is also defined as a standard)
  - It shouldn’t and can’t be followed blindly. Never do something “because it is prescribed in the methodology”. A methodology does NOT dictate!

- A methodology must be adapted to suit the specific situation of the project
  - Is an activity, model, document or other product needed?
  - Adding or removing products – selection of methods
  - Adding, reordering, removing activities

- Richer methodologies are usually better / offer more possibilities
  - The practitioners have more suggestions, more choice
  - It’s easier to ignore (leave out) an unnecessary activity, artefact or technique than to search, create and add one.

- A company can decide to have some parts of the methodology to be mandatory to every or to certain classes of projects (milestones, artefacts, KPI’s, ...).

- A methodology is based on knowledge. It contains (some of the) knowledge the practitioner should possess. However, it doesn’t replace this knowledge!! Without knowledge, deeper insight, critical thinking and sound judgement ... follow the methodology is a receipt for disaster.

- No methodology, approach or philosophy can cope with the consequences of lack of insight, a restricting vision or with a rigid mind-set. The vision must be clear and large. Mind must be open and flexible.

- The application of a methodology, on its own, is NEVER a guarantee for success. Let alone a strict and faithfully application.

- A methodology is never responsible for a project failure. People who chose it and executed it are !!
Methodology Usage

Method Engineers and the IS/Software Engineers (including all types of analysts and architects), require a profound and understanding of projects and their products in general. The IS/Software Engineers need also to have a good understanding of the particular situation of the initiative and its context.
The execution of a plan and the plan are different things. And both, from a process perspective, always differ to some degree.

Despite the divergence between these 3 concepts, a great similarity should be distinguishable. A lot of elements and patterns should still be present and recognisable.
It is ALWAYS the responsibility of the project manager and the project team to decide how the project is executed and what is needed, what is necessary to succeed with the project.

A methodology is a basic template suggesting activities, artefacts, techniques, an order of activities, roles, ...

The main role of a methodology is to support the thinking of the project manager and the team, it does not replace this thinking.
Mapping Methodological SD Approach on Problem Solving Approach

Problem solving process matching software development activities.
Fundamental Steps in Software Development Approaches

Evolution of the Software Development Process:

**Small and Simple**

1. **DESIGN** → **PROGRAMMING**
   - Reason: Avoiding spending time into re-organisation and rewriting of source code.

2. **DESIGN** → **PROGRAMMING** → **TESTING**
   - Reason: Avoiding to deliver a bugged application to the end-users.

**Larger and complex systems**

1. **ANALYSIS** → **DESIGN** → **PROGRAMMING** → **TESTING**
   - Reason: Avoiding to solve the wrong problem or to design solutions without understanding the problem and its context.
Challenges as Initiative Scales Up

1) Issue: **Development of a very simple and small software application**
   - Programming
   - Can be started right away

2) Issue: **Development of larger software application**
   - Software Design
   - Design and organisation of the software elements and components.

3) Issue: **What if the solution is unknown**
   - "Solution Engineering"
   - Assumption: Existing systems and environment are simple

4) Issue: **What if systems and environments are complex**
   - Systems Analysis and Design
   - Problem-oriented analysis: Study of existing systems and their environments.

5) Issue: **What if the problem is unknown**
   - Complaints and symptoms surface. The problems are not understood or not even correctly identified (root cause). The existing systems and environment is insufficiently unknown.
   - Systems Analysis and Design
   - Diagnosis !!!

Continuing on next page
Challenges as Initiative Scales Up

6) New issues emerge ...

- Sharing information
- Reuse of information, allowing multiple usage of information
- Coherence in the business logic across the company
- Ensuring real information problems (root causes) are solved
- Dealing with multiple users groups, different business domains and different objectives (sometimes conflicting)
- Conceive solutions that maximally exploit the possibilities of software and computers (no simple-minded solutions or impossible solutions)
- Integration in the environment
- Keep the system of systems coherent and clean
- Information security
- Ensuring evolvability of software systems (easy evolvability)
- Reuse software components, component based systems
- Ensuring the manageability of the entire collection of stored information
- Preserving the value of the information
- Information entropy
- Possibility to maximally exploit information
- ...

609
The concept of this simple SDLC is a pretty simple and straight forward processes. It represents a logical approach that puts understanding the problem and the solution requirements before starting to build it. A Waterfall may take as little as a day to multiple years. Thanks to its modularity, it allows to deliver intermediate solutions, increments, systems and sub-systems, releases, ...
What is the “Waterfall”? 

- A methodology
- A methodological idea, concept, principle, pattern
  - based on earlier problems and general problem solving
- An SDLC

Remarks:
- A drawing, a model, that may slightly vary
- No established number of phases or defined names
- Not a set of practices
- No established rules and principles
- No formal standard exists
- No guideline on how to apply it
- But a lot of interpretations, assumptions, ...
- A lot of principles and bad practices added to it → confusion

How do we know how to apply the waterfall?
How do we know how to apply it correctly?
How not to Waterfall

Practices Assigned to the Waterfall (which are not):

• The sequence of phases is strict and unidirectional
• Complete a work for 100% before moving to a next phase
  • 100%? Is this possible?
• Phases may not overlap
  • what with projects with sub-projects running in parallel?
• Only activities of one certain type may be executed during a phase
• Plan in advance and in great details and don’t do course corrections
• Freeze work done earlier before moving to a next phase
  • verifying and validating ≠ freezing
• Perform work imposed by the methodology (even if it is unnecessary)
• Perform a lot of bureaucracy
• Write extensive documentation
• Death march. Do not correct a situation by performing activities of a type performed earlier or don’t jump back to an earlier phase. Continue until delivery.
• If a type of activities that have been performed earlier have to be executed, you have to move the whole project to the corresponding earlier phase.
• Testing happens only at the end of the project during the Test phase.
• ...

Groundless interpretations of the diagram leading to nonsensical practices of the Waterfall SDLC
How Not to Waterfall

Unidisciplinary, local, symptom-driven, partial, ...

Detection of problem, scope definition, problem definition, preliminary solution, various decisions about the future solution, solution outline, some limitations and constraints

Ersatz of analysis:
Limited to refining the demand;
Eliminating some incoherencies;
Transposing into UML, BPMN-models;
Translating to specifications, ..

Here everything is defined. If this is not well-done ....

... then a lot of ‘learning’ will happen here.
How Not to Waterfall

1. The business stakeholders, SME’s, end-users determine the problem, what they want, the scope and the outline of the desired solution.

2. The analyst receives (gathers) requirements, refines, checks, translate, models, turn into specifications.

3. The IT analyst, designer, SE or developer(s) elaborate the technical solution.

4. The developers build the requested software system.

**Plan**
- Scope
- Delivery Date
- Budget
- Solution Outline

**Importance of Decisions**

**Business**

**IT**

**Course of the Project**
How to Speed Up Software Development

Discovery, identification, unreliable localisation and diagnosis of the problem by the business community

Business engages in various attempts to solve with the problem or its consequences. Lots of discussions, ...

Problem exists

Business Demand & Project Selection & Planning

Project

HIDDEN WASTE OF TIME

Responsible for frustration, impatience, pressure, additional work and cost.

Can we eliminate this?

Answer:

1) CONTINUOUS ANALYSIS to hunt for problems and improvement opportunities directly in implemented systems

2) Don’t design flawed solutions

The solutions designed today cause the problems of tomorrow. Information solutions must be designed by top professional analysts, architects and designers (particularly if system or solution is critical).
WISE WATERFALL

- Preferably linked to goal, outcome rather than to a defined solution
- Focuses on and maximisation of value creation
- Innovation
- Doing more the right and necessary thing and less responding to demands, wants, preferences
- Servant Leadership
- Structure and organisation
- Scalable
- Multi-disciplinary
- Collaborative
- Freedom not to follow rules, principles, structure, ...
- Iterative
- Incremental
- Modularity in waterfall - phases or stages, releases, sub-projects, sub-tracks, ...)
- Continuous investigation-driven over demand-driven
- Preference for proactivity over reactivity
- Righteousness, quality and steadiness over Speed and Deadlines
- Sustainable
- Take decisions as early as possible, but not before understanding
- Put risky, structuring and critical elements first, then most valuable, and then other aspects.
- Continuous Learning & Root Cause Analysis
- Planning and Analysis over testing and rework
- Predictive and Adaptive
- Holistic - including in time
- Top-down, bottom-up and bottom-top-down
- Understanding precedes Action - Know what you do before you do it
- Analysis-driven
- Model-based
- Integrated
- Feedback
- Flexible product over flexible process
- Component based and solution scalability when possible and useful
- Build stable systems – foundation, layers, agents, ...
- Reuse
- Avoid solving symptomatic solutions
- Conscious execution - Know why you are doing something, why not, why this way and not the other way
- Situational Process Decisions - Adapt the process
- Testing during the whole project
- Remove waste - Don’t do unnecessary work.
Waterfall Variations

Waterfall 1

Waterfall 2

Waterfall 3

No standard about waterfall!!
Overlapping and Parallel Development

Sashimi–model – Waterfall with Overlaps

Waterfall with Iterations

Parallel Building
Wise Waterfall

Phases used as Building Blocks

Parallel Development
Wise Waterfall

Phased Development

Overall Planning

Analysis

Planning → Analysis → Design → Build → Testing → Deployment

Sub-project 1
Version 1

Sub-project 2
Version 2

Sub-project 3
Version 3

First big release

Small release

Small release

Fully workable solution: platform & complete set of useable features

Additional features of lower priority and changes
Wise Waterfall

Feedback and Testing during the whole process
## Wise Waterfall ‘Execution’

*A possible execution of a Waterfall methodology*

### PROJECT PHASES

<table>
<thead>
<tr>
<th>Analyse</th>
<th>Design</th>
<th>Construction</th>
<th>Integration</th>
<th>Testing</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Clarifying aspects</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td>Adapting design</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Construction</td>
<td>Construction can start earlier</td>
<td>Corrections ensuring integration</td>
<td>Bug correction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integration</td>
<td>Considering integration from the design on</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing</td>
<td>General test activities</td>
<td>Preparation of final Test phase can start as soon as design starts</td>
<td></td>
<td>Work related to the Test phase</td>
<td></td>
</tr>
<tr>
<td>Deployment</td>
<td></td>
<td></td>
<td>Deployment mechanism can be taken into consideration during design, development, integration and testing (deployment must be tested as well)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

ACTIVITIES

Conception is a very gradual process.
Overlap of Activities

<table>
<thead>
<tr>
<th>Analyse</th>
<th>Design</th>
<th>Construction</th>
<th>Integration</th>
<th>Testing</th>
<th>Deployment</th>
</tr>
</thead>
</table>

Why tasks are normally executed with overlaps?

- The **mind thinks often ahead** on certain aspects or parts of the future product and on subsequent work. For example:
  - During the Analysis phase, the Analyst may already think on possible solutions (Design). (S)He can already prepare some sketches and models of the future solution.
  - Integration and testing have to be taken into account well before the actual project phase.

- Despite the effort, it happens that a task has **not be completed for 100%** or that the result is **not for 100% correct**.

- **Some parts may already be settled and ready for future work** to be started, while other parts still need additional work.
  - For example: Some parts of the design may be ready for construction while other parts still have to be further defined.

- **There is not always a clear burden between two types of different activities.**
  - For example, integration and testing is also a matter of design.

- **An activity requires often specific activities like planning and preparation.** These activities may happen before the actual phase. Example: Writing test scenarios for the test phase can start once the Analysis phase is finished.
Note:
This represents one single track in a project showing a more flexible interpretation of the waterfall SDLC.
AB6 is favourable to a continuous monitoring and analysis of the system. This leads to packages of changes which are then implemented in a project of variable duration (from a week or month to a year, longer?) depending on the size and priority of changes.)
Prototyping

Types:

• Forms:
  Proof of Concept, mock-up screens, simulation, paper prototypes, computer animations, video, scripting language, ...

• Categories:
  discovery prototype, rapid prototypes, iterative prototype, developmental/evolutionary prototype (to be part of the final system), ...

• Strategies:
  horizontal prototyping, vertical, task-oriented, scenario-based
Prototyping in Waterfall

**PROJECT**
- Analysis
- Design

**PROTOTYPING**
- Elicit Requirements
- Build Prototype
- Adapt Prototype
- Evaluate Prototype
- Refine Requirements
- Brainstorming

**Programming**
- Testing
- Deployment
Rapid Application Development (RAD)

Analysis & Quick Design

Design

Record Adaptations

JAD

Develop

Test

User Review

Implement

Deploy
Other Types of Approaches

- Joint Application Design (JAD)
- Exploratory Programming
- Lean

- Developed in-house
- Turnkey
- Off-the shelf
- Contracted out - Offshore
Methodology

- Project Management Methodology
- Software Development Methodology
  - Analysis and Design Methodology
  - Software Integration Methodology
  - Software Testing Methodology
  - Data Migration Methodology
  - Software Deployment Methodology
  - ...

METHODOLOGY?

- Based on principles
- Based on a set of common/best(?)-practices, methods, ...
- Adaptable process
  - One-size-fits-all methodologies don’t exist
  - Ready-to-use methodologies don’t exist !!
  - A Methodology ≠ Procedure !!

(can’t be executed like a procedure)
Select Types of Architectures → Design Target Architecture → Analyse Impact and gap → Review the Architectures → Validate the Architectures

Types of architectures:
- Information Architecture
- Process Architecture
- IT Architecture, ...

ARCHITECTURE

SOFTWARE DEVELOPMENT

Functional Analysis → Design → Programming → Unit Testing → ...
Different Types of Environments

- Development Environment
- Test Environment (a few ex. load tests, stress tests, .. Each requiring a specific environment)
- Training Environment
- Sandbox Environment
- UAT Environment
- Stage Environment
- Production Environment
- Disaster Recovery Platform

Some environments may be duplicated when dealing with different releases at a time.
Process: Testing
Software Testing LifeCycle (STLC)

Select Testing Techniques → Determine Test Requirements, Test Tools, Test Environments, ... → HL Test Planning

Develop Test Scenario → Review and Refine Test Plan → Test Environment Setup → Test Case Development

Test Case Preparation → Testing the Tests → Test Execution → Test Evaluation & Review

Re-Test Selection & Planning → Correction → Corrections Planning → Prioritisation of corrections

Programming
Notes:

Information Path: Defining what information is required and stored. Modelling this information.

Process Path: Identifying the required processes and engineering them.

Synchronisation: Processes need information. The synchronisation makes sure this information is foreseen in the information models, databases and other information artefacts.

Information has value. Interesting information has to be captured and processed. Each identified information entity will thus have to be captured and transformed by processes.
Flawed Work in Previous Phase

Phase A: Mainly activities of type ‘a’, producing artefacts ‘a’
Phase B: Mainly activities of type ‘b’ producing artefacts ‘b’

**Problem:** Artefacts ‘a’ resulting from phase A are unclear. It didn’t take some aspects into account. There might be something wrong or incomplete.

1) Keep the project in phase C
2) Recheck the concerned problem area, rethink it
3) Adapt the artefacts ‘a’
4) Check the broader impact of the adaptation
5) Validate artefacts ‘a’, with focus on changed and impacted area. Rest of the artefacts were unchanged. Review and validation should go swiftly.
6) Repeat for artefacts ‘b’

**Difficult? Strange?** Developers do this all the time in waterfall-approaches

Bugs found during the test phase are corrected by programming activities without having to move the whole project back to “programming phase”. Only the corrected software features are retested before the software code gets clearance to be released in production. Sometimes regression tests are required.

This same process can be applied to analysis, design, architecture, ...

Sufficient verification and testing has to be implemented during the whole course of the project. Often, a flaw may already be detected in the next phase.

*If a previous phase has been very badly executed, it might be necessary to redo it. This is a drastic decision.*
Approaches based on a restricted vision and developed for smaller goals aren’t suitable to develop larger more significant solutions.

Developing Information Systems requires a different approach than developing a collection of features. Similarly, developing software systems for the enterprise requires a different approach and thinking than developing “users solutions”.

Each initiative of different orders of magnitude faces new types of challenges.

- **Scalability** (managing several and/or larger teams)
- **World view** adapted to the new challenge
- Awareness of new type of **risks and issues** that may be met
- Awareness of new type of **aspects** to be considered
- Ability to enrich methodology with new **activities, methods, techniques, solutions, ...**
- Additional **customer’s and engineering knowledge areas, disciplines and skills**

**Scalable Methodology?**

Can the same philosophy and methodology be used to build these different projects? Why not?
A few strategies and tactics

- Avoid first creating more chaos
- Avoid first to continue to allow bad input
- Identify the garbage and remove it
- Map what is known and gain understanding by further analysis and mapping
- Work follow-the-flow, top down or from fundament to what’s built on top (bottom-up)
- Separate what’s good and in order from what’s bad. Protect what’s good.
- Identify areas, clean up area by area (or system by system, process, by process, DB by DB, ...)

Avoiding creating chaos

- Put in controls
- Build upon truth and understanding of reality
- Think of consequences, impacts, ..
- Is the result an increase in order or complexity (hiding complexity also increases complexity) or increase of order and clarity.
- Consider a decision, an action, a behaviour, an exception, an issue, an event, a project, anything and imagine it to happen 1000x, or more. If order is maintained, then it is fine. It the expected result is a dysfunctioning, imbalance, waste or chaos, then it is a bad practice that shouldn’t happen or happen with prudence and sparsely.
- ANALYSIS -
Key Elements of Analysis

Problem → Solution

DETECTION

EVALUATION → DIAGNOSIS

SOLVING → LEARNING
Three Key-activities of the Analysis

- Diagnosis
- Study - Learning
- Solution Engineering

All three
• are equally important
• must be performed by a professional Analyst

A demand, requirements, an analysis or a solution outline not resulting from these activities performed by a professional Analyst, should be considered as unreliable.

Building upon it is building on quicksand.
Key Activities in Analysis

• Learning the systems, processes, structures and their environment.

• Diagnosis: Identify gaps, problems, needs, obstacles, issues, opportunities

• Align demands, intentions, expectations with the NECESSITY

It is not about what people wants ("not mainly", "not always", ...; What people want does not always solve the problem or contribute positively to the whole organisation).

Systems Analysis is not a discipline that aims to please people, but to conceive systems, structure, processes to solve problems.

The alignment of the expectations with the necessity is important in order to create motivation, aligned collaboration and to avoid disillusion.

• Align conflicting interests, demands, priorities, rules, ...

• Conceive a solution

• Get consensus about the solution

Analysis should be ideally) right and complete. Usually, a fairly right and fairly complete works also.

But an Analysis that aims to solve the wrong problem or which is insufficiently based on the reality, on the truth, like superficial insight, is likely to do more harm than good.
Needs Analysis

Cause of the need, driver, links with other needs, goals, objectives

Need
  - Want, desire
  - Expectation
  - Intention
  - Obligation
  - Goal, objective

  Non-essential
    - Important
      - Vital, critical

Unknown, unidentified
  - Known, identified
    - Unarticulated
      - Articulated

Not understood, wrongly understood
  - Partially, vaguely understood, somewhat unclear
    - Understood, clear

Justified
  - Partially justified
    - Unjustified

Beneficial to a part
  - Beneficial to the whole

Short term
  - Long term

One-shot
  - Some occurrences
    - Many occurrences

Unjustified
  - Stable
    - Variable

Local
  - Global
Needs Analysis

<table>
<thead>
<tr>
<th>NEED</th>
<th>SOLUTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>“I need a drill”</td>
<td>Drill</td>
</tr>
<tr>
<td>Why?</td>
<td></td>
</tr>
<tr>
<td>“I need to make a hole in the wall”</td>
<td>A hole</td>
</tr>
<tr>
<td>Why?</td>
<td></td>
</tr>
<tr>
<td>“I want to hang a painting on the wall”</td>
<td>Hanging an painting</td>
</tr>
<tr>
<td>Why?</td>
<td></td>
</tr>
<tr>
<td>“To make to room cosy in order to relax”</td>
<td>To relax</td>
</tr>
<tr>
<td>Why?</td>
<td></td>
</tr>
<tr>
<td>“I am stressed.”</td>
<td>Not being stressed</td>
</tr>
<tr>
<td>What stresses you?</td>
<td></td>
</tr>
<tr>
<td>Work? Relationship? Family? Conflicts?</td>
<td>Not being stressed</td>
</tr>
<tr>
<td>Problems? Uncertainty? ...</td>
<td></td>
</tr>
</tbody>
</table>

*Questioning can continue to get to the deeper cause*

Wrong solutions: Drill, hole, painting, making a room cosy, relax
Bad diagnosis: Stress (=consequence)
Right diagnosis: Cause of stress

More ‘needs’ and solutions appear.
What is the Problem?

1. Lack of solution (Absence of solution)

Unpleasant experience
- Annoyance,
- Nuisance,
- Embarrassment,
- Discomfort,
- Pain,
- Irritation, ...

Is a problem ...

1) the lack of something, the missing solution, ...
   “we need a database”; “we need a system”; “we must have (object, product, solution)"

2) getting rid of the unpleasant experience, unpleasant feeling, irritation, inability, ...

3) eliminating the cause

4) the whole concept

What do the stakeholders ask?
What do they want?
What is the motive of the demand?
What is their expectations?
What is their demand about?
What is required?

Are the answers related to 1 and 2?
What to Analyse

Understanding System and Environment and determine the NECESSITY

- Organisation
  - Systems
  - Structures
  - Processes
  - Dependencies
  - Flows
  - Resources

- Environment
  (& interactions with and within environment)

- Constraints
  - Prohibitions
  - Impossibilities
  - Possibilities
  - Risks
  - Probabilities

- Various business knowledge and expertise domains
  - Business Perspectives
  - Aspects

- Capabilities
  - Capacity
  - Aptitudes
  - Resources

- Value
  - Purpose
  - Role
  - Ranking, Position
  - Significances
  - Criticalities
  - Priorities

- Objectives
  - Intentions
  - Motives
  - Complaints
  - Issues
  - Needs
  - Obstacles
  - Lacks
  - Results
  - Inefficiencies
  - Ineffectiveness
  - Wastes
  - Costs

- Evolutions
  - Tendencies
  - Inclinations
  - Forces

Unknown needs and unarticulated needs

Personal insights

- Personal Perspectives
  - Insights
  - Conflicting views
  - Belief systems
  - Assumptions

- Articulates needs, partially articulated needs, known but not expressed needs, unjustified needs and expectations, ...

What the stakeholder, the customers or the end-users want

- Opinions
  - Subjective diagnosis
  - Ideas, Suggestions
  - Disapprovals
  - Dislikes

- Demand
  - Wants
  - Desires
  - Preferences

* : for concerned business domains, business stakeholders, ...
Weak & Strong Analysis

**STRONG Analysis**

- Study Context, systems and environment
  - \(\rightarrow\) necessity: finding out what is necessary
  - Business goals or outcomes
  - Customer’s intentions
  - Customer’s expectations
  - Customer’s goals
  - Customer’s wants
  - Customer’s demand

**LOWER RISK FOR FAILURE**

- Much more reliable diagnosis

**WEAK Analysis**

- Customer does the diagnosis, 
  & missing insight in context and existing situation

- **HIGHER RISK FOR FAILURE**

  - WRONG, AWKWARD, INNEFFICIENT, INEFFECTIVE SOLUTION
  - SOLVING CONSEQUENCES
  - MISSED OPPORTUNITY
  - NO INNOVATION
**Wants vs Reality**

Limited mono-disciplinary perspective, limited scope, reduced objectives, biased background

Satisfying People ≠ Solve Real Problems

- Wants & Desires
- Preferences
- Choices
- Dislikes
- Demands
- Expectations

Trying to please others; being dependent of other people’s thinking, desires, expectations, …
Subjective diagnostic: Subjective, emotionally influenced, from single perspective, distorted, partial, vague, biased by belief system (eg. View on IT)

- What the situation, the reality, requires
- What is possible & impossible

Thinking critically
Being autonomous
Taking responsibility
Doing what you decided is necessary
Suits the situation
More objective diagnostic & view
Performed by competent analyst

Serving Clients REALLY by FINDING OUT what is NECESSARY and solving the REAL Problems with REAL Solutions

**Notes:**
Doing what is necessary may, in the end, suite and satisfy people much more. Doing what is necessary does not mean people choices should be ignored. It’s more a matter of having different priorities.
Studying Reality

Business & Users Perspective and Interests

Partial, vague, … knowledge

Business Stakeholders

Business Demand

Information: Business Knowledge, Business domain Expertise, Field expertise, complaints, …

Building a business, users, and systemic perspective and interests

Analyst

Study: (more) holistic and objective perspective

REALITY MATTERS

I want … I need …

What has to be changed in reality and will truly solve the problem/needs and meet the company’s systemic requirements:

Business, company’s environment, real world,

INCREASED RELIABILITY and STABILITY

UNRELIABLE - UNSTABLE

INCREASED RELIABILITY and STABILITY

30/10/2018

652
Analysis aligns cause, necessity and expectation.
Conflicting Views

Causes of different views and different expectations

• Different knowledge
• Partial understanding of the whole solution, its obstacles and its constraints
• Different perspectives
• Different goals and priorities
• Different solutions in mind

Four possible solutions (ordered from best to worse)

1. Listening to the explanations and arguments to create a final global insight – merging the knowledge and perspectives

2. Finding a solution that meets all different perspectives, interests, goals and priorities

3. Making a compromise among the different parties to come to a final solution, hopefully the best solution

4. Choosing the solution of the most authoritarian party
We know how to test/detect programming bugs.

But how to detect (on time)

• Analysis flaws?
• Architecture flaws?
• Design flaws?

If we don’t detect them, and handle them as simple ‘changes’, how can we improve?
1. Follow the Flow - Chronological

Activities in chronological order.

2. Top Down

Start at the top and work down.

3. Begin with the end in mind - Goal oriented

Start with a clear goal, then build.

4. Radial Expansion

Start at the center and expand outward.

5. Build based on / around stable aspects

Start with stable, clear elements.

6. Foundation First.

Build from the foundation up.

**Note:**

“Top down” means to look first at the broader picture, a drone-view. “Top” does not point towards the top organisational level (management).
Analysis Tactics

7. Greenfield Approach or Reengineering

8. Approach for Adaptation OF Existing System(s)

9. Aspect-based

Study specific aspects, domains, perspectives one by one. Then putting it all together when conceiving the solution. Example: process perspective and information perspective of a software application.

10. Mixed Top-down and Bottom Up

11. Initiate Bottom Up and Go Top-down

Detecting, identifying, study, ... something at the bottom. Bring the subject to the top. Work with it downwards by taking this new subject, intentions, plans, insight, ... into account. While the detection and start is below, the top-down process facilitates the integration and alignment.
12. Concrete to Abstract

![Diagram showing Real world to Abstract to Concrete reflection of real world]

13. Top-down w/o big bang - Phased

![Diagram illustrating Phases 0, 1, 2, 3, and 4]

Phase 0 provides an overall framework, organisation (and architecture). It shapes the whole initiative.

Then (sub-)projects can be launched one after another as phase 2, 3 and so on. These phases can overlap. They can be executed in parallel or they can be delayed if necessary. This offers greater flexibility.

14. Follow the Structures

Notes:
An approach for chaotic and undocumented environments can be to make first inventories. Get rid of the obvious waste. Then compartmentalise accordingly to previous approaches. Prioritise them and then clean up one compartment after another.
Sub-Optimal TACTICS

1. Risky Parts First

A project can fail on a difficult and risky part. This part is dealt with early in the project. If it fails, the project can be stopped or an alternative solution can be searched. If this difficulty is postponed until later in the project and it fails, then time and resources have been wasted and the lost investment will be larger.

2. Quick Wins First

Quick wins show usefulness and may create buy-in and motivation. The quick wins may correspond to simple issues that can easily be solved (see easy parts first). Or, it may be motivated by short term results orientation rather than by a system/long term ambition.

3. Following Established Priorities

Priorities may be defined by using different criteria and with different motives. How have these priorities been established? For what motives? Who defines the process? Based on what competencies? Or what aspects have to been taken into account and which have been underestimated or neglected?

4. Easy Parts First

Postponing the more difficult part can be driven by fear of complexity. This is not a good sign.

5. Known Parts First

Postponing the lesser well-known parts can be driven by fear of the unknown. Risks and true challenge may lay in the unknown (or lesser known) area’s.
Analysis Tactics

Sub-Optimal TACTICS

6. Bottom - Up
   ...

7. Follow the Boss, the Demand, the Sponsor, ...
   ...

8. Smallest First
   ...

9. Highest Business Value First
   ...

10. Organic expansion based on Usage and User’s perspective
    
1. Need-driven
2. Problem-driven
3. Short-term-goal-driven
4. Result-driven
5. Demand-driven
6. Priority-driven (based on usage)
7. Opportunity-driven

WORST
Reactive, symptomatic and organic growth based on defined by persons having only superficially knowledge of information, system, information /software systems and systems engineering (NOT SE as defined in IT).
Analysis Perspectives

Analysts investigate and think about the environment, system, problem from different perspectives:

- Objectives - Plans - Intentions
- Causes - Consequences
- Opportunities
- Priorities – Importance - Criticality
- Value
  (not just stakeholder’s appreciation)
- Stakeholders
  (their domain, objectives, ...)
- Necessity – Preferences & Desires
Analysis Perspectives

System and System Environment

Evolution (past, present, future)

Short and Long Term

Forces - Tendencies

Hierarchy of Systems

Whole - Parts

Capability and Attributes

Different levels of details

Structure & Mechanisms
(behaviours, dynamics, processes, ...)

True Nature of Things

Interactions - Influences

Flows of Energy, Information, Matter, Money
Analysis Perspectives

Abstract - Concrete

Lifecycles of systems, information, products, services, documents, ...

Mainstream - Exceptions

Balance & Harmony

Effectivity - Efficiency

Flexibility - Evolvability

Usage – Maintenance – Manageability

Integration
(all levels, aspects and perspectives)

Risks & Security

Various other perspectives: Company culture, values, vision, organisation, policies, geographical locations, innovation, progress, speed, strengths and weaknesses, viability, ...

Cost
Three Key Perspectives of Thinking

*Thinking in terms of ...*

1) A WORLD OF SYSTEMS

- Supra-systems, systems of systems, systems and sub-systems,
- Systemology, Systems Thinking, Systems Design, Systems Behaviour, ...

2) MULTI-DIMENSIONAL THINKING

Purpose, role, meaning, intentions, reasons, context, implications, knowledge areas, skills, perspectives, subjective versus objectivity, usefulness, location, focus, alternative usages, appropriateness, similarities, commonalities, differences, cohesiveness, coupling, characteristics, parts – whole - greater whole, object versus environment, structures, systems, processes, lifecycles, flows, concrete and abstractions, mechanisms, balance, harmony, action – reaction, nature of things, cause-effect, different levels of detail, evolution, small amounts versus larger amounts (numbers, repetitions,...), what-if’s, strengths and weaknesses, opportunities and limitations, obstacles, importance, criticality, options, generalisation versus specific case, mainstream and exceptions, forces, interactions, influences, relations, certainty (probability), precision versus vagueness, approximation, variability, classification, priorities, order, and so on. (see previous slides)

All these aspects can be applied iteratively.

3) INFORMATION EXPLOITATION

- Presence, Availability
  - Creation, capturing, gathering, dissemination
- Value and Quality
  - Includes relations (connections, links, ...)
- Processability
  - Format, structure, organisation
- Usage, Innovation
Models

- Business Model
- Business Canvas
- Business Strategy Model
- Five-Forces Model
- Business Domain Model
- Value Network
- Value Chain Model
- Value Stream
- (System) Context diagram
- Organisation chart
- Goal Model
- Stakeholder Map
- Product / Feature Roadmap
- Business Function Model
- Business Activity Model
- Business Scenario Model
- Business Role Definition
- Business Function/Process Models:
  - Process Model / Swim-lane
  - Role Activity Diagram
  - Data Flow Diagram
  - Process Flow Diagram
  - Workflow diagram
  - Use Case Model
  - Flow Chart
  - Function Hierarchy
- Functional Dependency Diagram
- Sequence Diagram
- Business Event Model
- Interaction Diagram
- Timing Diagram
- State Transition Diagram
- Information/Data/Database Models:
  - Information Architecture
  - Logical Data model
  - Conceptual Data Model
  - Entity Relation Diagram
  - Physical Data Model
  - Class Model
  - Business Object Model
- Implementation Diagram
- Component Diagram
- Composite Structure Diagram
- Deployment Diagram
- Physical Model
- System Architecture Diagram
ANALYSIS Traps

Assumption:
THE DEMAND DESCRIBES THE PROBLEM
AND/OR THE REQUIRED SOLUTION

You ask, I deliver – Your wish is my command
Giving Users What They Want

We are tired of having to input these documents manually. Can you find a solution to this?

Sure! I will analyse your requirements.

If this helps you. I will analyse your demand.

Customers are satisfied.
We produce value.

Improved Process

I need to be able to print this in a list

Sure! I will analyse your requirements.

If this helps you. I will analyse your demand.

Customers are satisfied.
We produce value.

Traps:
• Missing the detection, diagnosis, learning, evaluation
• Trusting the demand, confusing the demand or complain with the diagnosis; assuming a diagnosis has been done.
• Limited scope, local thinking
• No analysis is done, only a refinement, translation of the demand

I print data

Great! No manual data input anymore. Now, I scan documents and process them with OCR.

I don’t need this data
Hey, I need something like a good sweater? I have so cold. Can you help me? I have such a cold. I am shivering.

Yes. Sure! Hmm.. Clothes keep warm.

I will feel so much better when having warm again.

Ho yes! I have jumpers, shirts, ... I have really nice looking ones. They will be delighted.

Hey, I need something like a good sweater?

Yes. Sure!

Problem statement: Someone has cold
Demand: Something to get warm
Expectation: Feeling fine once problem is solved
Need: Something to get warm
Objective: Feeling warm (normal)
Requirement: Must keep person warm or deliver warmth
Solved: When demand is satisfied, when person feels warm (normal) again.
ANALYSIS Traps

How About Alternatives?

- Go into the sun
- Going inside the house
- Switch the heating on
- Take a warm bath or hot shower
- Get a blanket
- Drink a hot drink
- ...

How about questioning?

```
Having cold
  ↓
Fever
  ↓
Lung infection
  ↓
Weakened immune system
```

- What caused that feeling of having cold?
- What caused the fever?
- What caused the lung infection?
- Why is the immune system weakened?

```
Whole set of causes
```

The real problem can be solved and real solutions can be found.

**Questions** provide a totally different view on the problem and on the problematic situation.
TRAP: No LIMIT

The Business Stakeholders have **NO LIMIT** in their demand of:

**Maximising**
- Control
- Pace of development
- Adaptability
- Number of features
- Instant result, knowledge, ...

**Minimising**
- Cost
- Risks
- Effort

Informatics (or IT) seeks maximal time, resources, autonomy and reward.

Maximising benefits, Easier and lesser effort And masking the own shortcomings and limits
ANALYSIS Traps

Confusions

- Understanding the demand ≠ Understanding the problem
- Understanding requirements ≠ Understanding the situation, the context, ...
- Understanding requirements ≠ Understanding the required solution
Some Dangerous Assumptions

- The complaints or the demand tell us what the problem is.
- The complaints must be resolved.
- The demand or requirements tell us what is needed.
- A demand or requirements are based on an analysis.
- A demand or requirements are reliable.
If a diagnosis has been performed at the level of BA, then the FA doesn’t have to do a diagnosis. However, it is possible that the FA refine the diagnoses, new information about the problem change the diagnoses or new problems are detected (like dead bodies in the closets).
- ARCHITECTURE -
A Few Types of Architectures

- Single User
- MF
- Client-Server
- 3-Tier
- Service-oriented
- Agent Based System
- Peer-to-Peer
- Component-based
A Few Types of Architectures

Data Warehouse System

- Operational Systems
- Staging
- Data Warehouse (DWH)
- Data marts
- Reporting, Analysis Tool

Central Information Exchange Platform

Organisational units and roles

Software applications

Software application servers

DB servers

Operational Systems
Factors Determining the System’s Architecture

Indicative List of Factors (not exhaustive)

- Number of actors
- Geographical spread of actors
- Geographical spread of usage of information (capture, processing, usage)
- Other existing systems and infrastructure
- Functional decomposition/organisation
- Different involved independent parties
- Distribution of responsibilities among the involved parties (sponsors, stakeholders, users)
- Nature of information (type, structure, subject matters, ..)
- Degree of connectedness of information
- Required systems qualities (flexibility, evolvability, reusability, scalability,..)
- Emphasis on workflow, on processes, on data, on security, on system’s qualities
- Degree of dynamism in information change
- Degree of spread and fragmentation of information
- Push or pull philosophy (in information distribution)
- Nature of processes (formal, ad-hoc, cohesive, event-driven fragments of processes)
- Degree of desired integration
- Usage of information
- Ownership of information
- Degree of control over systems, processes and information
- Degree of sharing information (desired, achievable)
- Required level of standardisation and compliancy to some laws, policies, agreements or rules
- Amount of information
- Future plans, intentions, ...

30/10/2018
- DESIGN -
Good Design

- In line with the **purpose**
- **Simple**, no unnecessary complexity and no oversimplification
- Easily **adaptable**
- **Expandable** (horizontal)
- Can be built on top (vertical)
- Reflects **reality**

Requires excellent abstract thinking skills
System’s design has to reflect the real world (reality) (as much as possible)

Understanding the world, the reality, the “what is”, the objective reality (versus partial subjective perspectives)

Analyst’s key abilities
- Learn and understanding the reality
- Discerning the signal from the noise, the essence from the details, the truth from the distortions
- Abstraction and Synthesis skills
Major Types of Design Flaws

Solid foundation, harmony, balance, stability, ...

Major types of design flaws

1. Solving symptoms, local problems
2. Lack of controls
3. Inappropriate structures
4. Incomplete solution
5. Imbalances
6. Internally build-in limitations preventing a full exploitation
7. Internal mechanisms undermining the system (may create imbalances)
8. Inappropriate internal organisation
9. False beliefs, assumptions, principles, values, ... (foundation)
Two mind-sets / perspectives to consider and engineer processes

- **Focus**: Ability to perform a function, a role and to manage information
  - Function/feature-oriented
  - Stronger expandable functions, systems, components, agents,..
  - Responsive to events
  - Harder to get optimised, controlled seamless horizontal cross-border processes
  - Understanding and mapping end-to-end-processes?
  - Better for systems supporting ad hoc and unstructured actions (like managing data).

- **Focus**: Ability to produce
  - Easier to engineer seamless horizontal cross-border processes
  - Easier to measure and to optimise these processes
  - Greater mix of different business domains in the logic
  - Reduced flexibility to react to different types of events, exceptions, ...
  - Aspect of role, agent, .. is weaker.
  - More suitable for business processes
1. Preparation

- Identify Trigger
- Identify Outcome (all products, changes, ...)
- Existing Process Exists?
- Identify start of process, end(s) of process (across the entire organisation)
- Identify Stakeholders
- Gather information
- Determine the established performance goals

2. Modelling of the HL Business Process

- Identify order and add:
  - main stages / phases
  - the different obligatory intermediary results across the process
  - the main business decisions
  - the purpose and intentions
  - the concerned locations
  - the pre-conditions
  - the criteria
  - the end-criteria
- Verify the process
- Record specific key business expertise to take into account during the detailed modelling

3. Detailed Business Process Modelling

- Model the process
- Main case
- Exceptions – Exception handling
- Assign executors (ppl, machines, software apps/ computer)
- Check inter-process interactions
  - Ex.: Delivery process & Client’s address change
- Check & define user access authorisations
- Add controls
- Add measures
- List involved information entities
- Are they present in the information architecture and databases?
- Check the states and the events triggering state changes of these entities for effects on the process or if the process changes the states as noted?
- PROGRAMMING -
Internal Organisation of Program Logic

Unreadable, confusing, incoherent, unorganised code increases effort to adapt and risks

Organisation of code increases clarity

• For understanding
• For reducing risks
• For people needing to maintain the code
• For re-use
• To avoid increasing the entropy
• Increases maintainability and evolvability
• Lowers cost and increase the longevity of the system

A few methods:

• **By component, package, library**
  - Loosely coupled externally and with a high internal cohesion
  - Often about a same subject
• **By Layer**
  - Higher layers have lesser details and use code from the lower and more detailed layers.
• **By Kind**
• **By Technology**
• **Or a mix of the previous methods**

Check the “SOLID principles”
Many assume knowledge of programming languages and technologies suffice to be a programmer.

As a consequence

**Software Engineering**

and

**Programming Techniques**

are ignored by many
Brief Overview

Types of Programming

- Structured programming
- Procedural-oriented programming
- Functional programming
- Logic programming
- Event-driven programming
- Object Oriented programming
- Scripting

Programming Techniques

- User Interface Design
- Interface design
- Multi-user
- Multi-language, multi- ...
- Distributed computing
- Data formatting
- Patterns
- Logging, tracing
- Messaging
- Buffering techniques
- Commenting
- Sorting Algorithms
- Linking, Chain algorithms
- Encryption and decryption techniques
- Graphics, Sound
- Fuzzy logic
- Transaction
- Recursively
- Back tracking
- Dead lock
- Immutability
- Lambda
- Safe Call
- Closures
- Concurrency
- ...

Environments

- Apps
- Web
- Mainframes
- ...

686
Is code optimisation overdoing?

An optimisation reduced the processing time of a transaction by 0.1 second.

<table>
<thead>
<tr>
<th># Transactions</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.000</td>
<td>16,6 minutes</td>
</tr>
<tr>
<td>1.000.000</td>
<td>27,7 hours</td>
</tr>
</tbody>
</table>

Since computers work extremely fast, 1 hour of computer time means a lot of work.

A company has more than one computers, running several applications day and night, executing many different transactions and operations.

The global optimisation is more than code optimisation. An efficient data architecture, usage of performant technologies, optimised systems, and correctly configured databases are also part of the optimisation.
- TESTING -
Differentiating ‘Test’

TESTING ACTIVITIES

Synonyms: Verify, Control, Check, Test, Feedback, Evaluate, Review

Testing activities include all kinds of verifications (even those done as part of the analysis), simulations, trials, prototype, Proof of Concept, reviews, ...

These activities happen always during the whole course of the project, from the very beginning, regardless of the project phase.

The objective of testing activities is to know the obtained information is correct, the understanding is correct, the decisions are right, a concept will work, the design is fine, the work is well done, and so on.

Software can only be tested once it exists, after it has been programmed. But many other things can be tested earlier.

ORGANISED TESTS

Specific tests can be organised during the course of the project (regardless of the project phase)

• Proof of concept
• Simulation
• Reviews
• Validation
• Unit tests
• Integration tests
• Data quality tests

TESTING PHASE at SDLC-level

The Test phase is the project phase that focusses on executing a global set of tests right to make sure the product can be released and no bugs make it to the operational level.

“TESTING PHASE” in the WBS

The project’s WBS’s often contains a branch called “Test Phase” or “Testing”. This branch contains the planned activities of the “Test-phase”. However, this Testing section of the WBS also include a lot of other testing activities which are executed outside the “Test phase”.
GIGO in Projects

GIGO-prone process

Project = Progressive Elaboration Process

A house is built upon its foundation. Its roof rests upon the walls.

Development Process

while the project is on track and deliver the demanded product within scope, on time and on budget.

Notes:

All work requires an input. And often this input is the result of previous work. Garbage may not be inputted in the project or be created by the project and to be dragged all the way downstream through the process.
Avoiding GIGO in Projects

No GIGO in project process

No creation of garbage

Principle: Test at the end of a each unit of work

For example:
- A problem has been diagnosed \rightarrow test the cause
- An area has been analysed \rightarrow test the understanding
- A part of the software has been designed \rightarrow test the design part
- A set of screens has been designed \rightarrow test the screens
- A component has been developed \rightarrow test the component
- An integration has been done \rightarrow test the integrated components
- An artefact has been produced \rightarrow test the artefact
The Test phase in a project SDLC is the final complete ensemble of tests before the product is released to production environment.

The goal of this ultimate large scale testing is to verify whether the product is ready for production environment.

- The parts of the product function as expected. These products can be integrated. The whole product meets the requirements, and has to be able to meet the goal (or to contribute as expected). Although, at this stage, there has to be already a great confidence in these aspects.

- The product has to function as expected.

- Ensuring the product is ready for being released in the operational environment. It is the last chance before going into operations to detect and remove bugs.

- Idea: Maybe we should have a
  - “Regular Testing”: normal testing of the software being developed. This testing occurs as soon as it is possible (unit testing, integration tests, deployment tests, ..)
  - “Pre-release test phase”: final tests aimed to get the approval for deployment and release to the operational environment. This phase follows the Regular Testing phase.
  - This does not eliminate the need of testing thorough the whole course of the project.
Test Methodology

Main steps:

1. Plan Tests
2. Design Tests
3. Prepare Tests
4. Execute Tests
5. Evaluate Tests
6. Take Measures

Following activities can be added:

- Reviewing test scenarios
- Validating test scenarios
- Acquiring test tools
- Developing test scripts
- Establishing test environment check list
- Preparing test environment
- Testing the test environment
- Test the tests
- Preparing test data
- Training the testers
- Controlling test execution
- Analysing test results
- Analysing defects
- Test Environment Clean-up
- Elaborating a Test Report
- ...

Related:

- Acceptance Criteria
- KPI's
- Metrics
- Validation Matrix
- Test Log
- Test Strategy
- Test Model ((network)model of subsequent tests to be carried out)
- Bug reporting

- Tests incidents
- Defect communication
- Review Matrix
- Test Data, Test Data Management
- Test automation
- Cycle Runs, Manual Runs
- Evaluation meetings
- Learned lessons
V-model shows the order in which some tests are being performed (and why)
### W-Model

**Variant 1**
- Write Requirements
  - Specify System (Specification)
  - Design the System

**Variant 2**
- Write Requirements
  - Specify System (Specification)
  - Architectural Design
  - Detailed Design
  - Elaborate, Plan and Prepare Integration Test
  - Elaborate, Plan and Prepare System Tests
  - Elaborate, Plan and Prepare Acceptance Tests

**Notes:**
- Improved V-model - Many variants of the model exist
- Easily to be incorporated in Waterfall-type approaches
- Understand the idea
- Activities related to testing start way earlier than at the end of Programming phase.
Misconceptions

Testing is **NOT** limited to the Testing phase

Various forms of testing (including checking, verifying and reviewing) are done during the whole course of the project. Specific tests (like proof of concept, simulation, unit tests, integration tests, data tests) can be organised through the project. The Test phase is only an ensemble of tests aiming to prevent faulty software to make it to the production environment. It is the ultimate chance to catch bugs before reaching the operational phase.

Testing Phase is **NOT** a “Proof of Concept”

The goal of the Test phase is not to find out if the product works or not. Other techniques, like a “Proof of Concept” serve this purpose. When a product reaches the test phase, there must be a fair amount of confidence that the product is ready for production. The Test phase is a final test that should detect the “last” issues and bugs and, once corrected, to provide confirmation of its readiness for release.

Testing Phase is **NOT** a “Product Discovery Phase”

Business stakeholders and representatives often see the product for the first time during the Test phase. They should have seen and tried the software product (or any product) during the design activities (prototype, screens mock-ups, ..) and during the programming/building (even partially, non-working software applications can be showed). Major flaws have to be discovered early. The goal of such reviews is not to capture latest preferences and new ideas. If they are important, they should be treated as real changes (depending on their importance and impact).

Testing Phase is **NOT** a “Learning Phase”

Learning happens all the time. However, one of the major goals of the Analysis is to learn (not just to be told). Then, a lot should be learned when presenting artefacts, models or the product in construction to the stakeholders. Finally, some learning happens also once the product is in production, leading to adjustments.
Types of Tests

**ACCEPTANCE TESTING** Testing to verify a product meets customer specified requirements. A customer usually does this type of testing on a product that is developed externally.

**BLACK BOX TESTING** Testing without knowledge of the internal workings of the item being tested. Tests are usually functional.

**CHAOS TESTING** Testing by deliberately injecting failure into their services and systems (tools: “Chaos Monkeys”).

**COMPATIBILITY TESTING** Testing to ensure compatibility of an application or Web site with different browsers, OSs, and hardware platforms. Compatibility testing can be performed manually or can be driven by an automated functional or regression test suite.

**CONFORMANCE TESTING** Verifying implementation conformance to industry standards. Producing tests for the behaviour of an implementation to be sure it provides the portability, interoperability, and/or compatibility a standard defines.

**FUNCTIONAL TESTING** Validating an application or Web site conforms to its specifications and correctly performs all its required functions. This entails a series of tests which perform a feature by feature validation of behaviour, using a wide range of normal and erroneous input data. This can involve testing of the product’s user interface, APIs, database management, security, installation, networking, etc. testing can be performed on an automated or manual basis using black box or white box methodologies.

**INTEGRATION TESTING** Testing in which modules are combined and tested as a group. Modules are typically code modules, individual applications, client and server applications on a network, etc. Integration Testing follows unit testing and precedes system testing.

**LOAD TESTING** Load testing is a generic term covering Performance Testing and Stress Testing.

**PERFORMANCE TESTING** Performance testing can be applied to understand your application or web site's scalability, or to benchmark the performance in an environment of third party products such as servers and middleware for potential purchase. This sort of testing is particularly useful to identify performance bottlenecks in high use applications. Performance testing generally involves an automated test suite as this allows easy simulation of a variety of normal, peak, and exceptional load conditions.

**REGRESSION TESTING** Similar in scope to a functional test, a regression test allows a consistent, repeatable validation of each new release of a product or Web site. Such testing ensures reported product defects have been corrected for each new release and that no new quality problems were introduced in the maintenance process. Though regression testing can be performed manually an automated test suite is often used to reduce the time and resources needed to perform the required testing.

**SMOKE TESTING** A quick-and-dirty test that the major functions of a piece of software work without bothering with finer details. Originated in the hardware testing practice of turning on a new piece of hardware for the first time and considering it a success if it does not catch on fire.

**STRESS TESTING** Testing conducted to evaluate a system or component at or beyond the limits of its specified requirements to determine the load under which it fails and how. A graceful degradation under load leading to non-catastrophic failure is the desired result. Often Stress Testing is performed using the same process as Performance Testing but employing a very high level of simulated load.

**SYSTEM TESTING** Testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

**UNIT TESTING** Functional and reliability testing in an Engineering environment. Producing tests for the behaviour of components of a product to ensure their correct behaviour prior to system integration.

**WHITE BOX TESTING** Testing based on an analysis of internal workings and structure of a piece of software. Includes techniques such as Branch Testing and Path Testing. Also known as Structural Testing and Glass Box Testing.

(Src: various websites on the web publish this list.)
List of Techniques

- 5W1H
- 5Why's
- MoSCoW method
- Business Case
- CATWOE
- Critical Success Factors
- de Bono's 6 Thinking Hats
- Force Field Analysis
- Heptalysis
- Interviews
- Ishikawa
- KAOS (Knowledge Analysis in autOmated Specification)
- Mind Maps
- MOST analysis
- Net Present Value
- Job Observation
- PESTLE/PESTELI
- Proof of Concept
- Prototyping
- Questionnaires
- Requirements Analysis
- Requirements Engineering
- Rich picture
- SCRS
- Soft Systems Methodology
- Stakeholders Analysis
- Storyboarding
- SWOT
- Task Demonstration
- Use Cases
- Theme / Epic / User Stories
- WBS
- Workshops
- CRC
- ...

Models

ERD
Finite State Machine

Languages

UML
BPMN
Specification and Description Language (SDL)
User Requirement Notation (URN)
Use Case Maps (UCM, a URN language))
Goal-oriented Requirements Language (GRL)
Systems Modelling Language (SysML)
Requirements

Objectives → Diagnosis → Knowledge of situation, context, broader area, ...

- What must the solution?
- What must the solution comply with?
- What do we know for sure?
- ...

First step: Establishing an as stable information as possible fundament that allow starting design. Using most sure and stable information.

Requirements give an indication about the future solution to the design process. These requirements will be implemented in the final solution. Thinking about what the solution should be able to do and comply with, is solution–oriented thinking. On the other hand, requirements are deduced from and by the analysis activities. They are not created as a result of creativity.

A requirement is a statement expressing a function, a key aspect, attribute, principle, characteristic, capability, constraint or limitation that the future solution should implement or comply with.

The Analyst works with a set of requirements.

The Analyst extracts requirements through investigations, interviews of stakeholders and analysis. The requirements are the result of an Analysis, of an investigation.

The Set of Requirements is an artefact serving as a criteria. The usage of this set of requirements is a method.

Notes:
Techniques: Volere, Planguage, EARS (Easy Approach to Requirements Syntax)
Requirements

The Requirements technique is a Problem Solving technique

The following metaphor illustrates the functioning of requirements in the problem solving. Each dot can be seen as a requirement.

Elicited (not gathered) requirements provide some information about what has to be incorporated in the future solution.

The more requirements are elicited, the more become known about the future solution.

Sometimes, in the end, a pattern showing the future solution may emerge (in the illustration: a bull) from the set of requirements. The requirements help the mind to connect the dots and to fill in the blanks. If no pattern emerges, it is up to the Analyst to conceive a solution. They also help the Analyst excluding very early on solutions that do not meet the requirements. In any case, requirements help conceiving a solution.

After requirements have been elicited, the solution has still to be defined (conceived). Requirements facilitate this task.
Requirements

- Requirements represent needed capabilities, constraints, limits or other characteristics.
- A requirement should state what a system must comply to, not or rarely how. However, this is likely to be in terms of business logic (in particular as business rules).
- Requirements without a right prior diagnosis are unusable.
- Requirements have to be feasible / attainable.
- Requirements should be deduced from the objectives, from the business knowledge and field expertise, from the business forecasts, from the supra-system and from the environment.
- A requirement is the result of an analysis, a deduction, an observation, a conclusion. If a requirement doesn’t comply with this and is the product of creativity, then it is not a requirement but already a design statement. Such early design statements, as false requirement, narrow the range of solutions. This is not the role of requirements.
- An approach in which different subjects, like an area, a domain, an aspect or a level of detail, are subsequently analysed is motivated by the objective to elicit requirements about that subject. The assumption is often made that only requirements about the subject being analysed will be elicited. It can’t be excluded that when analysing a subject A, requirements impacting subject B or impacting the solution or system as a whole.
- Requirements are by their very nature incomplete.
- The future solution has to comply with the ensemble of final requirements.
- The set of final requirements does not represent “THE SOLUTION”.
- Requirements help to solve problems. It’s a (possible) stage in the problem solving process. But it is not the magic bullet that replaces problem solving skills.
- Requirements are not a substitution for insight in the situation, in the context and in the broader area. Conceiving a solution based on a set of requirements without understanding the supra-system and the broader environment is a good receipt for disasters. It’s better to have this insight and no requirements than to have requirements and no decent broader understanding.
- Business requirements are requirements concerning the business domain, field experience, business activities. It does not mean “owned by the business” or “expressed by the business”.
- A future solution not complying with the final set of requirements, provided they are right, guarantees that this solution won’t function properly or, at least, be significantly lesser valuable.
- Any solution meeting the set of final requirements doesn’t guarantee the right problem is being tackled or contributing to the right goals. It doesn’t guarantee the problem will be solved, the solution resulting from it being right and the best, the solution being valuable nor does it guarantee customer’s satisfaction.
- A solution matching the requirements can still be refused.
Requirements

Three Main Purposes

• To facilitate the process of conception of the solution by providing key elements that must be incorporated in this solution. It is the first information we know about the future solution.

• Purpose of foundation for further thinking. They form the stable basis for creative thinking. (stability of requirements!)

• To keep the conception of a solution in the right direction. Ensuring that the important aspects which have been identified are taken into account in the design, or, preventing to forget or ignored some of them. This can be verified by requirements-based tests.
Requirements Norms

Qualities

• Unitary, atomic
• Clear
• Complete
• Cohesive
• Consistent
• Non-conjugated
• Traceable
• Current
• Feasible
• Unambiguous
• **Necessary**
• Importance specified
• Verified
• Verifiable, testable

A requirement has first a temporary and uncertain status, like draft or preliminary. Later its status may become more definitive or final when it is found to be true, certain, accepted, validated or agreed (depending of the organisation).

Let’s call these the "**final requirements**".
Types of Requirements

1. Business Requirements
2. Goal Requirements
3. Stakeholder Requirements
4. User Requirements
5. Program Requirements
6. Project Requirements
7. Process Requirements
8. Management Requirements
9. Enterprise Requirements
10. Environmental Requirements
11. Solution Requirements
12. Product Requirements
13. Architectural Requirements
14. System Requirements
15. Software Requirements
16. Change Requirements
17. Regulatory Requirements
18. Quality Requirements
19. Quality-of-service (non-functional) requirements
20. Functional Requirements
21. Non-functional Requirements
22. Usability Requirements
23. Indirect Requirements
24. High-Level Requirements
25. Detailed Requirements
26. Information Requirements
27. IT Requirements
28. Data Requirements
29. Design Requirements
30. (non-)Technical Requirements
31. Testability Requirements
32. Security Requirements
33. Documentation Requirements
34. Implementation Requirements
35. Transition Requirements
36. Requirements Specifications

... and there are many more
Sources of Requirements

- Management’s Objectives, Plans
- Company’s Environment
- Business Domain Knowledge, Business Knowledge, Field Experience
- Supra-System, Co-systems, infrastructure
- Society
- Regulations
- Standards
- Theories, Research
- Natural Laws, Fundamental Principles, Nature of Things

ANALYSIS

REQUIREMENTS

(investigation?)
Notes:
This model shows a general idea. It is not the only way of working and can be adapted, for example, the selection and validation can be swapped.
Requirements

Requirements ??

• The new database must be online 24h/24h – 7d/7d.
• The software application must be able to print the customer’s data.

These are NOT requirements

• What if the problem can be solved without new database?
• What if another software application would print the customer’s data? Or what if the customer’s data could be made available in another way?

Whether a new database is required, what software application should print or if printing is requires are all design decisions.

If a statement is based on a design decision, then it is not a requirement.

Correct requirements:

• The information should be available 24/h/24h – 7d/7d
• The customer’s data should be made available to the customer.

Notes:
Requirements precede any design decisions. They are not the result of a design decision of any kind.
Requirements are the result of an Analysis Process. Good requirements result from the understanding the environment, the situation, the business domain, the objectives and systems.

Requirements are NEITHER GATHERED NOR RECEIVED !!
Some requirements are obvious.

Generally, spoken it is hard work to get sufficient relevant requirements. Some critical requirement are well hidden.

Obtaining bad requirement / bad practices:

• Gathering requirements
• Asked, received or expected from the business community, SME, ...
• Accept requirements at face value
• Deduced from “what the business wants”
• Asking the business community to confirm the requirements they provided

Bad requirements result from an insufficient insight in the context and the situation.

A “requirement” stated by the business community CAN be valid, but it must be verified by the Analyst checked with the reality, with what “is”, by investigating the broader context and see if this requirement matches and fits in this broader contexts.
Requirements

WRONG PURPOSES of REQUIREMENTS

Requirements as a goal

“The goal is to meet the requirements.” This shows little care about whether the solution will solve the problem or need or not. Basically, we don’ mind to create value. Meeting requirements is good enough. Or we assume that by meeting requirements, value is created. Requirements are not a goal. Meeting requirements is an important criteria among others.

To replace a diagnosis

Requirements are a post-diagnosis activity. Requirements result from investigation activities (Analysis). The diagnosis should have been done by the time requirements technique is started.

To understand the problem or solution

Understanding requirements is not the same as understanding the problem or the solution. It does not discharge the Analyst from understanding the situation and the context. Requirements may describe a fraction of the problem or solution. And requirements are incomplete and may be wrong.

To describe the solution

Requirements describing a solution are called “specifications”. Seeking to have all the requirements is impossible and senseless.
Questions

1. Why?
2. What?
3. When?
4. Who?
5. Where?
6. How?
7. Which?
8. What kind of?
9. What for?
10. Whose?
11. How long?
12. How much?
13. How old?
14. How come?
15. How far?
16. Why not?
17. What if?
18. Whom?
**5 Why’s**

**Purpose:** Finding the root cause

- The number of 5 shouldn’t be taken literally. The idea is to repeat the question as many times as possible.
- The answer may consist of several reasons → branching
- It is simply a matter of answering the question based on the present knowledge. If necessary, it should lead to further investigation (new knowledge) of the question.
5W1H

• Who was involved?
• What happened?
• Where did it take place?
• When did it take place?
• Why did that happen?
• How did it happen?
THE END