An Analyst's Perspective on Systems Analysis

The Essence
of
an Undervalued Discipline

Version 1.0 (1/12/2016)

Axel Vanhooren

1 Introduction

The landscape of software systems forms a huge and critical part of companies and organisations. These systems perform a tremendous amount of work. Analysis has an important impact in the way companies and organisations work, in how they perform, in their manageability, in their responsiveness and in their adaptability.

Analysis has evolved through the years. The discipline, as it is practiced today, seems to have been moulded to suit the business stakeholders, the project managers and the developers. It is applied accordingly to their vision and expectations. It ensures the transition between the business demand and the programming work. Everything seems to be fine.

But are the stakeholders really satisfied of the It implementation and of the IT community? Does the overall IT implementation meet the expectations? Does the business community trust their IT department?

A gigantic gap is being formed between the way 'analysis' is commonly practised and what Analysis is really about. The essence of the discipline has been lost. And consequently, the discipline isn't and can't fully be exploited. The Analysis discipline is largely undervalued and the real benefits the discipline has to offer aren't reaped. The present document will explains the essence of the discipline and clarify this gap.

Since, put in very general terms, the goal of Analysis is to improve the company or organisation, an improvement in the Analysis discipline should logically benefit to the company or organisation. Given the importance of the discipline, it is certainly worth time and thoughts.

What will be discussed in this document?

> This document considers Analysis within the context of Business Informatics and thus corporate IT.
>
> A comparison is a good way to present a gap in a clearer way. Therefore, a picture is drawn of analysis as it is commonly understood and widely practised today in companies. The next chapter discusses why, in software development projects, different disciplines or groups of activities are actually necessary. It describes, as a kind of “what if”, the types of problems the project would run into if these activities are neglected or ignored. The purpose, the driver, the essence, of the Analysis discipline defines its actual value. Then the discussion will be about the core activities and how they relate to each other. The Analysis discipline has still many other benefits to offer. It’s interesting to know them as well. The role of the Analyst, the knowledge domains and the skills will complete the picture.
>
> A lot of problems are caused by our perception, by misapprehension, by confusions, by wrong interpretations or by incorrectly fill in the unknowns. This document tries to remedy to this. However, not all of them can be rectified by this text.
>
> The generic term of ‘Analysis’, as used in this document, covers different variants of ‘Analysis’ like Business Analysis, Process Analysis, Information Analysis and Functional Analysis. They are all included in the umbrella term of ‘Systems Analysis’, or shorter, ‘Analysis’.

Rethinking the ‘Analysis’ discipline means also redefining the role of the Analyst. Both are intrinsically linked. This shift in comprehension will influence other aspects, like the role of the IT department, its competencies and how it is staffed.

There are some difficulties in presenting such a more thoughtful vision. First, the domain of Analysis is rather abstract. So it is difficult to create a sense of feeling about difference between amateurish analysis and professional Analysis. It’s a matter of level, of applied norms. And it is difficult to explain and to create awareness of the true power of our discipline, and thus of its untapped potential.

The presented vision differs from the mainstream picture, from the “what everybody knows”, from widely spread habits and expectations. The common understanding is based on some false assumptions and distorted beliefs which exist already for decades. They are deeply rooted and widely spread. It is mandatory for such a more profound discussion to question these well-known deeply rooted beliefs and assumptions. It is through a very objective and critical eye and an open mind that a deeper and true reflection about Analysis can happen. Things may differ from what we think.

The true nature of things and the natural laws have to be respected. If we exercise a too big force on an object, the object will break. Using a screwdriver as hammer won’t give a good result. If a farmer omits to plough or sows too late, the harvest will be meagre. Building something on shaky ground may lead to its collapse. We can delude ourselves, but we can’t cheat with nature. Not respecting the nature of things and the natural laws is taking a risk for suffering problematic consequences. Problems of the same nature will keep arising until the lesson that has to be learned is learned and until nature is respected. Today, analysts are supposed to know standards and methods, to be able to apply them and to have some field experience. This is assumed to be sufficient to perform an Analysis. Companies take hereby an immense and dangerous shortcut. A more profound understanding of the discipline and crucial skills are neglected. Without this insight and these competencies the nature of things and the natural laws will continue to be ignored and violated. Problems of some types will continue to occur.

Adopting a new picture of Analysis implies a change. And change requires overcoming obstacles and performing a substantial effort. It is much easier to ignore the more solid, more profound and more professional definition of Analysis and to stick with the more easy to grasp, cheaper and weaker version of Analysis as it is known today. It is easier to leave everything as it is and to follow the main stream. But the benefits will remain limited at best.

Small, basic and simple software initiatives are easy to perform successfully and don't require a profound understanding of the discipline and advanced competencies. Larger and complex initiatives are the real challenges that test a discipline, methods and skills. Therefore, this document is about larger and more complex information systems and processes in corporate environments.

There are different types of software applications, like smaller personal software applications, websites, apps, embedded software, games or system administration tools. They may apply the Analysis discipline in a reduced or otherwise adapted form. This document doesn't focus on such special cases. This document concerns information systems in corporate environment.

The term 'analysis' (with a lowercase 'a') points either to the common term as defined in dictionaries or to the common and present understanding and application of analysis activities. The term 'Analysis' (with an uppercase 'A') points to the real professional discipline. The same indication is applied on the function title of (A/a)nalyst.

Hopefully, this vision may provide new insights or some matter to reflect upon. The discipline of analysis is still in its adolescence. It has still to evolve a lot. Breakthrough is often realised, not by simple improvement, but by questioning what we know and the underlying beliefs and by a radical change. This might be a step in this journey.

2 The Widely Spread Understanding of Analysis

This chapter describes the form of analysis and the role of the analyst (both with lowercase 'a') in the context of software development projects in a corporate environment as it is commonly understood and practiced. This understanding forms the basis of how software development projects are embedded and executed in companies. It defines the methodologies, roles, authority, objectives, expectations, and relations between concerned business people and the project team. The interpretation of analysis and the role of the analyst may slightly differ from organisation to organisation and even from person to person. There is also a difference between what a person knows and thinks it should be and it is applied in the field.

The analyst acts as a bridge or interface between the business community and IT people and ensures that the final product matches the business demand, respects the established scope and solves the stated problem.

The business demand forms the initial impulse and the foundation for the analysis activities. Commonly, this demand contains the objectives of the business organisational unit, a problem description (what is perceived as being wrong, missing or needed), the scope, stakeholder's desires and expectations and a basic set of requirements. It provides a global picture of the desired solution. It depicts the nature of the solution, like a new system, a new database, a new information exchanges or some new software features. It may also describe the overall structure, the main logic and interactions. And it may even provide some further details of what business people see as the solution they want.

The analyst satisfies the business needs by responding to the business demand. Therefore, the analyst analyses the demand. Analysis is about taking a closer look at all this information coming from the business community. It is about looking at the details, about clarifying the ambiguities and about detecting gaps and inconsistencies.

The analyst obtains (business) requirements from business stakeholders. These requirements are usually gathered, received and sometimes imposed. They fit within the scope and objectives defined by the business demand. The analyst analysis the requirements, refines them, manages them and ensures their validation by business specialists and by the stakeholders.

Business knowledge, explanation and additional information are provided to the analyst. This input helps the analyst to understand the business demand, the vision of the solution and the expectations and to maximise the chance of building the solution as it is demanded.

The analyst translates all this information into specifications. Often, the analyst produces BPMN or UML-models. The specifications and models which describe the product to be built are passed to the developers.

The analysis translates the business demand, the mentioned business needs, and the requirements into artefacts allowing the developers to build the requested system or solution. The analyst is also active in testing. But testing is actually a different discipline.

Basically, the business demand, its objective, the stated problem and the expressed solution define the scope and the product. They form the basis, the foundation, for the analyst's work. Analysis refines the business demand. By building and delivering what the business stakeholders asked for, business value is created and satisfaction of the business community is achieved.

Let's call a spade a spade. This approach can be termed as "Business Demand Analysis" performed by what could be called a "Business Demand Analyst". It contains some analytical aspect. But it is actually a distorted and much diluted form of **A**nalysis. It misses the very essence of the discipline and it doesn't respect the philosophy of **A**nalysis and its real purpose.

Analysis is not a discipline that can be considered as an obligatory step in projects, as add-on to other disciplines, as a secondary discipline which solves all the remaining questions that weren't answered by others.

3 Analysis in the Software Development Process

Analysis is applied in software development projects. Companies develop software applications of different sizes and of different degrees of complexity. Depending on these factors, different types of problems may occur. Analysis and Design are an answer to these issues. This chapter clarifies some of the purposes of Analysis. Most of these issues are probably known, but they are too often ignored when judging, developing or applying a methodology.

Programming

The most basic software development methodology consists of a single phase or activity: Programming (or coding). Basically, this is the only activity that can't be left out. When the solution to be built is simple, the developer can easily and quickly form a mental picture of the solution and can start programming right away. The development of small and simple software applications doesn't require an elaborated preparation or subsequent activities.

Design

When the software application to be developed is somewhat larger, a developer will quickly encounter some new kinds of difficulties.

- At a certain point in an on-going software development project, some software components; like menus, screens, features and data exchanges; have already been developed, while others still have to be developed and integrated. This integration may be not as easy as expected. The part of the software application developed so far hasn't been prepared to integrate the components still in the development pipeline or it has been vaguely and superficially prepared. When the developer builds one of those components, he or she has to think about how to integrate the new component in the software application already built. This integration may require an adaptation of already developed components. Components, which were considered as being finished, have now to be modified to allow this integration. This means rework. As the software application progresses, integrating new components into the software application may become gradually more difficult and more time consuming.
- The developer will face another type of issues. When starting the development, the developer has a rather imprecise picture of the solution in mind. Some areas or aspects are clear, while other areas or aspects are fuzzy or they still contain some uncertainties. As programming progresses, the developer has to consider aspects and details he or she didn't really think about earlier. The picture of the solution the developer has in mind will clarify. The details of the solution are being determined. The clarification of the fuzzy areas or aspects may influence the areas or aspects that were assumed to be clear. The clarification of an unsettled area may bring about changes in the areas and aspects presumed to be clear and defined. Consequently, what was already developed and considered as being finished needs now to be adapted.
- A similar effect occurs when choices and decisions concerning some logic have been kept pending. Just before the development of the concerned chunk of software, an agreement

has to be reached, a choice has to be made or a decision has to be taken. This may also require the adaptation of already developed program logic.

In each of these cases, the developer needs not only to program the new logic, but also to see how existing code has to be adapted, and sometimes even restructured, in order to integrate the new logic while maintaining the coherence.

It can be explained with a simple example. A screen capturing and displaying a lot of information has been developed. Now, some additional information elements need to be captured on that screen. Adding the required buttons and fields would make the screen too large and overloaded. The solution to this issue is to split up the large screen into two smaller screens. A new screen is created and some fields have to be moved from the original large screen to the new screen. This operation forces to review the whole screen management of the first screen. The retrieval of the information and the process of writing information in the database have to be reviewed. The places from where the screen can be accessed have also to be verified. Maybe menus or the user access rights have to be adapted. If this had been foreseen from the beginning, a lot of rework could have been saved.

Even if we don’t care too much about some rework, modifications and adding new pieces of code have a detrimental effect on the software system. The quality and internal organisation of the source code will suffer when developers are lesser capable, when they don’t care too much, when modification are executed under time pressure or when the applied norm is "as long as it works". The software application work well. The quality issue can be present. Business stakeholders and users can’t judge if the job is well done or not. This type of issues occurs inside the code and is completely hidden to them. So even though, everything may be fine from an end-users perspective, the software application may not meet decent quality standards.

As changes are applied and new things are carelessly added, gradually, the source code becomes unreadable, inconsistent, misleading, inefficient, unorganised, duplicated, overloaded with unused code, weak and overly complex. The architecture becomes quickly clumsy and the internal logic and the source code become chaotic. Adding or changing a piece of code will take much more time and will also be more risky. A reorganisation and a clean-up is then required. This is called refactoring. It doesn’t matter whether this reorganisation and clean-up happen along with debugging or with new developments, occasionally in small steps, under the radar or as single huge planned task. It may also require the software application to be retested and redeployed. If the logic is conceptually changed or adapted, then manuals and trainings may have to be adapted as well. It’s a lot of rework. It’s there. It takes time and it costs. Adapting a small software application is still easy. But adapting or reengineering larger software systems is much more difficult.

Any new information, new or changed requirements, clarifications, additional agreements, choices or decisions brought up or being considered only just before the source is code about to be programmed and hereby generating rework, are not “just in time”. Something inappropriate has been built and has to be adapted. This means rework. Some waste has been created. These elements are thus obviously too late.

Design (and also architecture) solves these problems. It aims to reduce this type of rework by thinking upfront about, for example, what menus, screens, features, data, interfaces, information flows, technologies are required and how to organise them. It also aims to increase the internal organisation and standardisation of the source code.

The methodological process contained already the programming activity. It is now expanded with the 'Design' discipline and activity.

Analysis

The methodological process consists now of a design and a programming (and testing) stage. Simple and local issues can be solved with this approach. But companies have even larger software systems to be developed. The developer will again face new types of issues.

Note that, at this point, we can already observe that an approach that works for small software applications, doesn't guarantee whatsoever to be also usable for larger and more complex software systems.

The business demand and the additional information and explanation received from the business community are the main input to the development of a software application product. All this information is depicts and is aligned with the vision of the business community, with their objectives and with their expectations. So, the product is fundamentally based on the business perspective and on the end-users' perspective. It is assumed that this approach creates business value. Probably, the software product facilitates the work of the end-users and contribute to business objectives.

Nonetheless, after deployment of the solution, various problems may arise. Possibly, the 'solution' does not even solve the problem or need. Some problems may arise immediately after the implementation. Others may occur at any time, triggered by an unusual event or within specific conditions, which are sometimes labelled as 'exceptional'. Other problems may grow progressively and covertly. They may appear as problems with information or as problems among systems. These problems may arise more downstream in a process or they may pop up elsewhere in the organisation or at a different hierarchical level. Some services may not fit or they don't support some needs. The company may suffer from hindrances, limitations, risks or from unsurmountable obstacle. They may impact the company's operations, the employee's morale or it may result in loss of clients, in higher cost or in difficulties in the transformation of the company. The different types and areas in which they may occur are discussed hereafter. And none of them are technological.

- Most software systems fit in an arrangement of existing systems. They are a part of a supra-system. The integration of a new or adapted version of a system into the ensemble of co-systems and in the supra-system may cause conceptual or architectural issues. A change in one system may cause problems in another system. For example, they may become dysfunctional or inconsistent. Or their performances may drop. Many things may happen.

- Inconsistencies and gaps in the implemented business logic may occur. For two reasons. It is very hard for business people to manage the business logic implement in software systems. The second reason is that software systems may be used by different organisational units. Their processes may cross organisational boundaries. They may serve the objectives of different organisational units and may contain logic of very different business domains. There is not necessarily a 1-to-1-to-1 relation between a business domain, an organisational unit and a software system. Ignoring this mix of different business domains and the situation and the needs of the various concerned organisational units is a source of problems. The requested solution may suit one organisational unit or one business domain and causing issues in another one.

- Problems appear when a solution or system doesn’t fit in its environment. A system’s logic may seem to solve a problem. The environment has a huge influence on the type and design of the solution. If the solution doesn’t respond to the true challenge posed by the environment, if it doesn’t respect its complexity and if its logic does not correspond with that of the real world, is likely to be inappropriate. If the environment is not correctly taken into account during the conception of a solution, this solution is unlikely to solve the intended problems and is likely to create new problems.

- Having as objective to obtain a system that functions well, having this as norm and be satisfied with it, is a good way to get into troubles. A system that is designed to perform a certain job, to do what it has to do, is a weak system. Such a system may work fine for a short period of time. A system which has to be viable over a longer period of time entails specific functions, mechanisms, a structure, qualities and abilities. When they are lacking or too weak, the system is fragile and incomplete. It will run into trouble and won’t last.

- The usage of archaic, inappropriate or simplistic concepts in systems design is another common problem. There are many concepts which seem very practical but which actually constitute a trap. Some concepts, mechanisms and characteristics may set a damaging tendency in motion. These tendencies, but also limitations and obstacles, are then embedded in the design. Information duplication, hierarchical classification of information or unstructured information (like ‘documents’) are such potential traps. Some of these concepts are inspired from the paper administration. This is a limiting, inefficient and obsolete way of thinking in the world of computing. The automation of existing processes and ways of working is another typical case. Existing inefficiencies will remain. They will then simply be automated. All these problems are then experienced as hindrances in implementing new services, in processing information, in responding to some events, in reuse and broader exploitation of information as well as in adapting the system. This happens particularly when amateurish analysts do the design or when the design is influenced by laypersons. Using the right concepts in solutions is important to have efficient systems. It is also a key factor for progress and innovation.

- Software applications are often based on ideas and concepts. These ideas and concepts may not yet be mature. Stakeholders may have different visions, different expectations and disagreements. Conflicting objectives, different visions and expectations, lack of insight, inappropriate priorities, diverse opinions, ill-conceived concepts, immature and changing ideas and ignored aspects are issues that have to be solved. Some of these issues will be clarified, settled and maybe even agreed upon during the building process, or maybe later, through feedback. Others won't. These issues lead to unexpected issues, unforeseen obstacles and bad decisions. They will generate a lot of 'changes'. But the reality is that these aren't real changes. Indeed, the environment or the functioning of the business didn't change. Only the insight and the mind changed. The cause - this isn't even the root cause! - is the lack of insight and the still hazy and unstable picture of the solution. Changes and iterations are then (ab)used as remedy to the consequences of this lack of decent and stable insight.

Solutions have tooth problems and iterations are necessary. Environments, products and policies do change. Good ideas may pop up at any moment and they may take time to mature. Changes do occur. And a change can be beneficial. But a change implies also additional effort (rework), longer delay, a higher cost and risks. Changes may also lead to confusion, uncertainty, frustration and demoralisation. They may undermine credibility and trust. Changes keep valuable resources busy. Some changes, but not all, indicate that something unsuitable is being or has been implemented.

Changes due to incompetence, to ignorance or to immature ideas aren't real changes and should be avoided by working smarter. The occurrence of some real changes can be forecasted and thus possibly avoided by anticipation. How to deal with a forecasted change or need is a matter of factors like the probability, the risks, its importance, the conceptual impact, plans, workload and the availability of resources or time. Some of the forecastable real changes can also be avoided or facilitated.

- A solution can be correct and well-designed, but it may solve the wrong problem. The focus is often directed on visible and concrete elements and on the result. But we also tend to focus on unpleasant events or on annoying experiences. They cause suffering. But these are often only the consequences of an underlying problem. So, maybe what has been identified as the problem was only a consequence of the real problem or a part of it.
- The solution has to be implemented and it will produce results. It may influence, positively or negatively, the output of the company. As it is integrated into a set of co-systems, in a supra-system and operating in an environment, it will exercise an influence on these systems and the system's environments. It may affect other areas in the company and the people in those areas. It may create new business opportunities or eliminate some of them. It may produce undesired effects or create new problems. It surely has limitations which may or may not be relevant to the business. It may impact work of people, the flexibility in their work or it may require specific skills. New risks may appear. It may create the risk for misuse and opportunities for abuse. It may increase or decrease the company's flexibility. It may facilitate or hinder the management of the systems. And if it solves the problem, does it also make the company more efficient and robust? Many other aspects have to be considered by the Analyst.

The Analysis discipline tries to ensure that the right problem is solved correctly. It seeks to ensure a good integration of the system in the landscape of systems (or system of systems) and in the environment. It seeks to maintain harmony within the different business domains. It seeks to support the operations of all the organisational units and to maximise the exploitation of information to the benefit of the company. It seeks to keep systems understandable, organised, manageable and flexible. It seeks to ensure that right concepts are designed and assembled and to prevent systems from decaying. It prevents a lot of rework due to immature or unsettled ideas.

Developing business software applications is a bit more complicated than implementing business logic or getting technology to work. This challenge can’t be solved with only business domain knowledge, programming skills and technological knowledge. Approaches and methodologies assuming this are doomed to fail. And adding skills and experience in Use Cases, User Stories, UML, requirements or social skills won't fill the gap.

4 The Purpose of Analysis

What is the purpose of Analysis? Accordingly to the name "Systems Analysis", the discipline is about analysing systems. Analysis is often understood as looking inwards, looking at the details, refining something. It is about studying systems. That's a short and, regrettably, also very incomplete answer.

Systems Analysis is about studying and conceiving(!) systems. It's about thinking in terms of systems. It's about looking at the world as if it was composed of systems.

Traditionally, analysis is used in software system development. It focusses on the system to be built: the software application or software system. The software system isn't the system that matters the most. This is just a component, a sub-system, of a larger system.

Systems Analysis is about the study and conception of a particular system: the company.

And in this paper, the term 'company' has a broader meaning including other similar organisations as well. It may also mean a very high-level organisational unit functioning rather independently like a business unit. Sometimes it may also mean an enterprise. Systems Analysis is not responsible for the conception of the whole company, but only a part of it.

A company, organisation or enterprise, is a socio-technical system. It has all the characteristics of a system. This system has areas that are organised and even formalised, while others are anarchic. It is a system of systems. It consists of systems having different purposes, fulfilling different functions and processing different raw materials. These systems can be of different nature. They can be concrete or abstract. They are made up of constituents of different types. A company has an architecture. These systems may be hierarchically organised. But, at the same time, they can be networked. They may overlap each other and can even be interwoven. All these entangled systems have to work together as one single, coherent and balanced system. All these systems use information.

A company can't function without information. It can't function with bad, unreliable or incomplete information. Information is really critical. It is a matter of death, survival or prosperity. The company needs *Information Capabilities* to deal with this information. The *Information Capabilities* concern the company's ability to exploit and to manage information. This implicates all the activities applicable on information, like capturing information, storing it, organising it, information transfers, finding information, its retrieval, its selection, its processing and also its usage. *Information Capabilities* are implemented by the ensemble of all the company's systems and by the involved people.

The *Information Capabilities* concern the ability and capacity of the enterprise's systems to deal with information. But it also concerns the information abilities of people and the ability of engineering information solutions.

Implementing these *Information Capabilities* and *Information Systems* and integrate them correctly in the company's systems is the domain of *Business Informatics*.

Systems Analysis is a discipline that can be applied for all kinds of systems, like business systems, economical systems, ecological systems, industrial systems and financial systems. Commonly, it is considered as a core engineering discipline of Systems Engineering. It is a core discipline applied in Software Systems Engineering, Information Systems Engineering and thus also in Business Informatics.

> The discipline of Systems Analysis, as a sub-discipline of Business Informatics, is concerned with the study, the conception and the improvement of systems dealing with information, computerised or not, implementing information capabilities.

It may also be concerned with how information is and can be used in systems to the benefit of the company or organisation and to its clients.

Systems Analysis is not about software development. The work area of the Systems Analysis discipline is information and information systems. To avoid any confusion, information systems are not the same as software systems. Information systems are organised and often formalised systems which may include people, software systems and computers. They may include manual processing as well as automated processing of information. They can be manual, partly-automated or fully automated. A software system includes one or several computer applications and may be used by people, but it doesn't include them.

Systems Analysis is about equipping the company with the necessary information capabilities essentially by evaluating, diagnosing, conceiving, improving and implementing information system able to deal with information.

Not all information processing is organised and not all information processing is automated. The tendency is to get as much of this information and information processing organised and automated. Furthermore, Analysis may also provide value to the lesser structured and spontaneous information usage as, for example, made possible by office automation software.

So, information and (information) systems are the core business of Systems Analysis.

Systems Analysis is concerned by solving information needs and information problems. But Analysis itself doesn't solve information needs. Information needs can only be solved by information. Analysis studies and conceives an organised, equipped and managed environment supporting the fulfilment of information needs.

Information problems shouldn't be confused with business problems. The business community may experience or identify an information problem. It may describe it and request it to be solved. This doesn't mean that the problem is a business problem. Business needs aren't necessarily information needs. A business system is very different from an information system. And, unfortunately, information processes are frequently confused with business processes. When information is a product or a service to the client, then an information process can be a business process as well. This confusion is created by the inappropriate use of the labels. Here it concerns the label ‘business’. This is not just a matter of semantics. It creates fundamental misunderstandings about the very nature of the wrongly labelled element. This has a serious practical impact. Dealing with business problems, business systems or business processes require different domain knowledge, approach, methods and skills than if it were information problems, information systems and information processes. The fact that some business knowledge is used in an information system or information process doesn’t change the true nature of this system or process. And the involved objectives and the responsibilities are different as well.

The company has to be able to work efficiently and effectively. The company is a complex system of different systems which has to work seamlessly together. The Analyst needs to understand this ensemble of these systems. Therefore, Systems Analysis requires a close collaboration of different engineering disciplines and domains of expertise.

The company or organisation strives to be prosperous and also to exist infinitively. It operates in a dynamic world of collaborating, competing and directing systems. To exist in this world of competition, of plenty of opportunities and with limited resources, it has to play a meaningful role. It has to be strong, skilled, efficient and effective. It has to strive, to strengthen itself, to improve, to evolve and to progress in order to survive, to grow and to prosper. Information systems are part of the company. And d chain is only as strong as its weakest link. So this reality plays a role in the conception of information systems. It is fundamental for Systems Analysis to take this into account and to contribute to this goal. If the analyst ignores this, weak systems will be conceived. The company will become sluggish and/or fragile.

The company really matters to Systems Analysis. Consequently, it is by far the main stakeholder.

Information has value. But this value is volatile. Over time, the value of information tends to decrease. But people can also destroy its value. Companies are sitting on a mountain of information and probably more useful information is available outside the company. Precious information may be stored on a company’s servers and not used. This sleeping resource is unexploited. Systems Analysis can be employed to preserve the value of information and to increase its exploitability.

The company is also a managed and guided system. Hence, it must be understood. Systems Analysis offers a method to study systems. Systems Analysis clarifies the internal structure and the functioning of system, of the supra-system (like the company) and of the relations with their environment. This information is not only important for developing the system but also to the functioning, to the management and to the transformations of the company.

Systems Analysis is not simply about making a company more efficient. Information is a source of innovation. Information can be used to drive the business activities by allowing new working methods, by providing new opportunities, by offering new services and new products. Analysis can be a very important source of innovation. This is particularly true in the information society we live in. But it won’t be and can’t be if it is considered as a reactive refinement process triggered by a prescribing demand emanating from the business community.

Systems Analysis is not about finding out what software application to build to satisfy the demands and expectations of the business stakeholders. It is about thinking how to facilitate the functioning of the company or organisation, how to support its life in a sustainable way (in particular the aspects of responsiveness, adaptability and manageability), how to create value for the company or organisation and for the clients through information. Systems Analysis is a critical step in the overall development process of the company, of information systems and of software systems.

The typical important questions Analysis considers are:

- What information does the organisation need to be able to function and to contribute to its prosperity and to a long life?
- How should this information be organised and how can it be used?
- What information processes and information systems are required to benefit to the organisation and how should they look like?
- How can this information be captured with the least effort?
- How can the value of information be enhanced, preserved and exploited to the benefit of the organisation?
- How can information facilitate and improve the functioning, the management and the reactivity of the company?
- How should that part of the company that deals with information be conceived, built and maintained while ensuring its consistency, its reliability, its manageability, its responsiveness, its adaptability and its expandability?
- What are the weaknesses, the unsophisticated concepts, the risks and the hazardous tendencies of the implemented information supra-system (the company's global information solution)?
- What information services and information products can be provided to clients?
- How to bring innovation in the field of information to the company? How can information support innovation in the company?

These are the (types of) questions that should define the objectives and the brainwork of the Analyst. They drive the Analysis activities. This purpose is very different from and much more beneficial than serving the business community with the implementation of requested pre-defined 'solutions'.

The company is a system created and designed by man. This implies that it should be and can be engineered. Systems Analysis is basically a problem solving and an engineering discipline. These terms shouldn't be taken lightly (see chapter about the Analyst). Analysis offers problem solving and engineering frameworks.

A problem can be solved by luck or it can disappear by itself. But these are not the most reliable problem solving techniques. And it is possible to build something, try it, improve it and repeat these steps until the product solves the problem. This requires little engineering skills. It is also a very uncertain and slow approach. It is probably not very economical. Moreover, it is not always possible and may cause much harm. It's very risky. Trial and error is normally used in engineering when no other method works. And it is probably not used as main approach, but rather as a sub-method for a specific issue.

A widely used approach to conceive engineered systems is a structured approach comprising the acquisition of understanding, the conception, the building and testing. But since the tasks of Analysists are diverse, different frameworks and approaches should exist.

The fastest and most economical way to solve a problem is to solve the right problem rightly from the first time. This is probably the most fundamental principle driving Analysis. It is its raison d'être.

Systems Analysis is about identifying the right problem. The term 'problem' must be interpreted here in a very broad sense. It can, for example, be understood as 'issue', 'inefficiency', 'need' or 'opportunity'. A problem can't be solved, if it is not detected and not correctly identified. Solving consequences doesn't solve the problem itself.

A problem can't be solved if it is not understood. Ignorance and a superficial comprehension are a fertile ground for mistakes and inefficiencies. It may create even more problems. Analysis seeks to avoid or to limit these mistakes caused by ignorance. It seeks to avoid spending time, resources and money to the construction of experimental transitional solutions and non-solutions.

Insight in the problem and in the actual situation shouldn't be confused with the understanding of the demand and of all the received information. They are different. Similarly, we may have the feeling of understanding by listening to a person. This emotion can be misleading. In fact, the given explanation has been understood. This doesn't mean we understand the situation. The explanation may sketch a distorted picture of the reality. The received information can be partial or erroneous. It is the understanding of the real situation that matters to the Analyst.

Learning, studying and analysing are key activities to acquiring insight. This insight is necessary for a proper diagnosis and for the conception of a solution. Information gathering, investigation, dealing with information and managing information are crucial for Analysis.

Information systems, and a fortiori software systems, are sub-systems of a company. They exist in a world of systems. Some of them belong to the company, while others exist outside the company. The company itself is actually a system of systems. This whole arrangement of systems influences the design of the system. Each system operates in an environment which influences the design of the system. The company and its environment influence also the design of the company's subsystems. Systems, processes, information flows and information usage often cross organisational borders.

And, in the opposite direction, a system influences its supra-system and its environment. The influence, the consequences and the impact of a system on the other systems, on the supra-system and on the environments have also to be understood. This means that Systems Analysis can't be limited to looking only inwards. Its focus shouldn't be limited to the problem or to the solution. In general, the scope of Analysis can't and should be limited to a single business domain or to an organisational unit.

Without a more global understanding of the overall situation, an analyst can build something. An analyst can do what is asked and expected. He or she may know what he or she does. But he or she won't know whether the product will solve the problem, what value is created or destroyed, what the consequences of the 'solution' for the supra-system will be, and so on. It is then also impossible to apply real Analysis expertise. For example, the analyst can suggest some variations to the requested solution, but it will be very difficult to thinking about real innovative alternatives, alternatives which are radically different from the demand and of significantly greater value to the company.

Mistakes have to be detected as early as possible. Programming source code is a very slow and very labour intensive job. Adapting this source code and cleaning it up is often much harder than producing new source code. However, it is much easier to change a thought, a document, a model, a set of requirements than to change source code. Detecting a mistake and applying the change on analysis and design artefacts is thus much faster and cheaper. It's better to make a mistake in a stage of 'thinking', and detecting and correcting it during this stage, than when it happens in a building stage. Analysis reduces the amount of mistakes and allows the detection before they reach the developer's activities. Therefore communication, verification (testing) and feedback are important aspects in Analysis.

Analysis is certainly not just a problem or demand driven activity. It doesn't need to be limited to projects. It can be and even should also be a continuous activity within the company.

Analysis, as a process, can transform gathered information about the company into knowledge and knowledge into insight and insight into innovative concepts (creativity). These concepts are then, usually by means of projects, turned into a beneficial reality.

Hereby, Analysis is concerned about how to change a company and how to equip it with the necessary information capabilities in order to deal with information rightly and efficiently and to maximise its exploitation.

5 Principles, Pre-determined Elements and Directives

There seems to be a great confusion about how to interpret principles, predetermined elements and directive elements and how to deal correctly with them. This chapter is just a practical clarification to avoid some misunderstandings about Analysis.

Principles

Systems Analysis, like all disciplines, has its objectives, principles, rules and methods. Objectives are not always entirely reachable. Principles and rules are not always true. Methods aren't always applicable. These elements don't fit in every single situation. It makes no sense to apply them literally, systematically or regardless of the specific situation. But considering them therefore as worthless and thus choosing to ignore them or to abandon them and even deciding to do the exact opposite, isn't very wise either. Often, it is not the literal meaning that has to be understood. It is the underlying true meaning, the meant intention, which is of great value.

- Analysis is based on the idea of doing the right things right from the first time. However, wanting to conceive systematically from the first time the right solution solving the right problem, seeking to always succeeding for the fully 100%, is inefficient and it is often even impossible. Many projects would be stuck forever in the analysis and/or design phase. Projects have to progress. Mistakes can never be excluded. We may conclude that the principle is utopic, thus invalid and unusable and that making mistakes should be acceptable.

Well, this is a misunderstanding of the principle and a wrong reasoning. But it doesn’t invalidate the intention behind the principle.

The real value of the principle lays in its interpretation and in a sound application. Systems Analysis seeks to solve the right problem rightly from the first time. The intention is to do a genuine and reasonable effort in the direction of this utopic goal. Analysis increases the likelihood of getting close to this objective.

The Analyst has to do a genuine effort to do a decent analysis. At a certain point, the Analyst will become aware or estimate that the problem is rather correctly identified. The subject is mastered. Continuing to work on the Analysis will yield gradually lesser and lesser benefits. By then, the acquired insight and the main part of the solution should be right. The new information or insight will confirm what is already known. It will concern details or the new information isn’t really relevant. It doesn’t change the solution. There is a lot of confidence in the proposed solution. The mistakes and unknowns should be limited and will mostly concern only minor elements and details. The project and the Analyst should be able to deal with the additional, refinements, corrections and improvements. And the stakeholders should be open for these changes.

So, reaching 80% can be fine. And reaching 90% or even 95% is probably even better. But trying to reach 98% or the full 100% may be not efficient anymore. Of course, for some systems the full 100% is possible and even required.

A first release often still has tooth problems which can be corrected after implementation or in a next release. A solution should be evaluated after the correction of these tooth problems being solved.

The fact that the principle can’t always be achieved cannot be an excuse to assume too quickly a situation is understood which would result in a sloppy job. And it is certainly not a free rein for making mistakes. Some amounts of mistakes and some mistakes are unacceptable.

- In common Analysis approaches, analysis precedes the design. Design precedes programming and programming precedes testing. Wrong and meaningless interpretations are that these are tasks are strictly sequential, that each of these tasks should be entirely and fully completed before the next one can start and that no overlap or iteration is possible. These interpretations reveal an absolute misunderstanding of the different software development activities and a lack of sound judgment about these activities. Basically, it’s better to first identify the problem, understand it as well as the environment before thinking about a solution. It’s better to first think about how the solution looks like before starting to build it. Source code can only be tested after it has been programmed. In this context, ‘testing’ concerns only the software code. It doesn’t concern the analysis or the design which can and have to be verified (tested) much earlier. This is the meaning of the principle. Once we understand the more profound purpose, the role, the nature, the benefits, the prerequisites, the risks and limits of these activities, then activities can be organised in many different ways with for example overlaps, iterations, sub-projects, releases, phases and activities. It is possible, for example, to have programming activities very early in a project without violating the stated principle. It will just depend of what is being programmed and what activities have been performed before.

Predefined Elements

Some elements, like the scope, delivery date, deadlines, a picture of the overall solution, a plan, a budget, and requirements, are determined early in the project.

Some of these elements will determine or describe the future solution. Because the insight in the situation progresses during the project execution, we may become aware that on or several of these elements, like some requirements, have to be changed. We may become aware that an improved or even a different concept for the solution would lead to a more beneficial solution. Two decisions have then to be taken. Do we take advantage of this opportunity during this project? And if we do, how will we respond to it and how do we adapt the plans for the future solution?

Some other elements serve to organise and to manage the project. But when it turns out that these elements diverge too much from the reality or from an achievable and valuable objective, they can’t support project management anymore. They can’t play their role and they even become harmful.

Some of these elements are determined early in the project. In the beginning of the project, this information is vague, partly ungrounded and uncertain or some information can be missing. These predetermined elements are thus very approximate. Also, some of these elements are estimates and are based on this approximate and incomplete information. Freezing these elements is rather unwise. It ignores that a typical characteristic of projects is the progressive insight and clarification as well as the increasing certainty. The other extreme, changing them continuously, is not the way to go either. The intention is to obtain the greatest possible stability in the project. But adaptation must be possible if beneficial.

To be useful, these elements have to be based on precise information. The situation and the product to be built are or should be fairly well known. The design of the solution describes the product to be built. Work, plans and estimations done before the design are then rather approximate. Only after the design, they should be precise and stable enough.

These elements need to be realistic and achievable during the whole course of the project, taking into account a variable degree of approximation. To keep them realistic and achievable, these elements need to be reviewed and corrected during the course of the project. Only then they can again play a positive role in the management of the project and contribute to its success.

It is important to understand on what predefined elements are based, how reliable, how variable (or stable) and how (im)precise they are, when and how their quality and usability can be improved. Then their value can be used to the maximum benefit of the project.

Directive or Guiding Elements

The Analysis discipline contains many different frameworks, approaches, methodologies and methods. They may not be applied as if they were prescriptive. They aren't. They simply can't be prescriptive.

The assumption that they are prescriptive means that frameworks, methodologies or methods can be applied regardless of the situation and project. It would mean that these tools would fit to projects without being adapted. But every situation is different. And every project is different. Consequently, these elements have to be adapted to the project. Or, at least, the Analyst has to verify whether and how these tools have to be adapted to the project and to its specific situation. So, these elements are just suggestions. They are meant to assist the Analyst in his job. It is the responsibility of the Analyst to select these guiding elements which are usable for the project. The Analyst decides whether such a guiding element is applicable or not, how to adapt it and how and when to use it. Possibly, the element or the situation, or both need be adapted in order for the element to be usable.

These elements do not and cannot replace the Analyst's knowledge. They are no substitute for professional insight and skills. They don't solve a lack of them. They only provide a support to this knowledge and to the competencies. All the responsibility remains in the hands of the Analyst. It is never the methodology that is responsible for a failure. Knowing a framework, methodology or method, without understanding it, is a dangerous game. Fully relying on, for example, a methodology, because of a lack of professional insight, is a receipt for disasters.

The real mastery of a discipline requires the ability to apply frameworks, methodologies, methods, concepts, principles and tools correctly. This implies deciding when or when not to apply them and how to adapt them. To yield results, it requires the understanding of the prerequisites, the limits, the risks, the underlying principles, the intention, the purpose, the value, the benefits, the weaknesses of these elements. It is important to understand what concept, principle or method works, why it works, when it works, what doesn't work and why and how to make it work to get results.

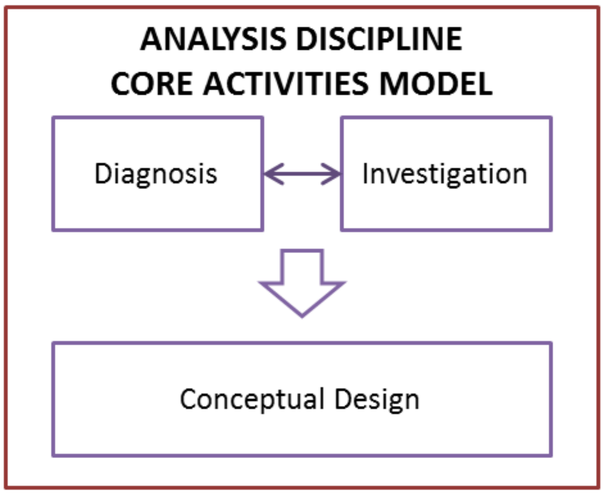
Most elements provide support. They help sometimes as guidance. They can be right for 80% right and usable in 80% of the cases. They are for 80% reliable or they can be stable for 80%. A real professional knows how to maximise the certainty, precision and stability and how to deal with the remaining limited amount of uncertainty, approximation and variability.

Fully trusting and blindly relying on guiding elements and applying a standard, a framework, a method, a principle or concept literally, as 'prescribed', is rather unwise. It's a sign of weak skills or of missing a basic insight in the discipline. It leads to failures and, ultimately, to the rejection of precious principles, concepts, methods, techniques or tools. An incorrect understanding or wrong usage of a method doesn't make that method erroneous. The same is valid for other elements of the discipline.

6 Analysis Core Activities

The Analysis discipline contains the three following core activities or main types of activities.

- Diagnosis
- Investigation
- Conceptual Design



1. Diagnosis

It's obvious that an information problem or opportunity can't be solved without first detecting and identifying the problem. Then it is important to ensure that what is being experienced or perceived as a problem is the real problem. And finally, the problem has to be understood. There is a tremendous difference between identifying the presence of a problem, localising it and understanding it.

The diagnosis may detect one or several information needs, shortages, gaps, discrepancies, quality issues or other information problems. It may detect inefficiencies, wastes, hindrances, obstacles, limitations, inconsistencies, risks or other weaknesses in information systems. It may point out a need for clarification, organisation, formalisation or standardisation. The list is not exhaustive. The overall diagnosis or evaluation may raise the awareness for a need to rethinking or re-engineering of an information system or it may lead to detailing an innovative idea and resolving the unanswered questions.

The Analyst solves mainly information problems. An information need or problem expressed otherwise, like "we need a new database" or "we need a new software system", makes simply no sense.

The diagnosis requires specific skills and should to be done by an information specialist. A diagnosis posed by a non-Analyst is questionable.

The diagnosis is pretty much neglected today. It is a waste to build a solution based on a wrong diagnosis.

2. **Investigation**

The investigation (analysis) is the study of the situation. It's about learning. This investigation is not limited to the problem, to the inside of a system or process, or to that part that is directly related to the problem, the limited immediate area closely surrounding to the problem. The investigation requires a broader and overall vision. It has to consider different levels of details, sometimes even those that are much higher than there where the problem has been situated. The investigation studies concepts, flows, mechanisms, active forces and tendencies. It studies normal cases and exceptions. It is important to study the overall system, the environment, the other information systems, the concrete world and its abstractions, the dynamics of systems and the evolution going from the past to an expected future. Analysis studies different aspects in information, information usage and information systems. It looks at it from many different perspectives as well.

We can differentiate a problem oriented investigation from a solution oriented investigation. The first serves the diagnosis. The latter supports the conception of a solution. Both, diagnosis and solution are needed. During the investigation, gradually the focus will shift from the problem to the solution.

3. **Conceptual Design**

Surprise! Analysis is not about analysis only. Design, and more specifically the conceptual design (or functional design, logical design, the non-technological design) is part of the Analysis discipline as well. This design is the logical conception, the definition, of the solution. It is imagining the future solution.

Note: Some demands express the need for a new software application, a new system or a new database. These descriptions confuse the need or problem with a missing product or solution. But a need or a problem to be solved is not the missing solution or product. This is treacherous because it shifts the mind to the solution and away from the real problem or need. Furthermore, a business demand may define the requested solution as a system, database, a set of features or anything alike. This implies that an architectural and/or design decision has been taken. Namely, maybe there is an alternative solution. Maybe that need can be resolved by adapting the existing systems, by consulting systems or by extracting the data from existing databases. Such decisions should be taken only by informatics specialists like Architects and Analysts. And this also implies a diagnosis and investigation has been done before.

The logical conception of systems and solutions is the most or one of the most creative activities of information systems engineering.

The technical design is a kind of prolongation of the functional design. But it doesn't concern the main logic anymore. Technical design is concerned by the concrete implementation and the technological choices and issues. It's a complete different domain with a different purpose and focus. It makes sense to clearly separate the professions of technical and technological design from the professions of functional design.

The diagnosis happens simultaneously with learning (investigation). Posing a correct diagnosis; which consist of detecting, identifying, localising and understanding the problem; requires a decent more problem-oriented investigation. And the diagnosis will determine what has to be learned and understood to solve the problem. It provides a basis for the solution-oriented investigation. Insight is the result of learning, thinking and verifying. The evolving insight clarifies. But it may also shift our view on the actual problem. So, both feed each other. This leads to a more complete, accurate and reliable insight which is used for the conception of an appropriate solution.

Only when the problem is identified and understood and when the broader situation is thoroughly (!) understood, a solution can be conceived.

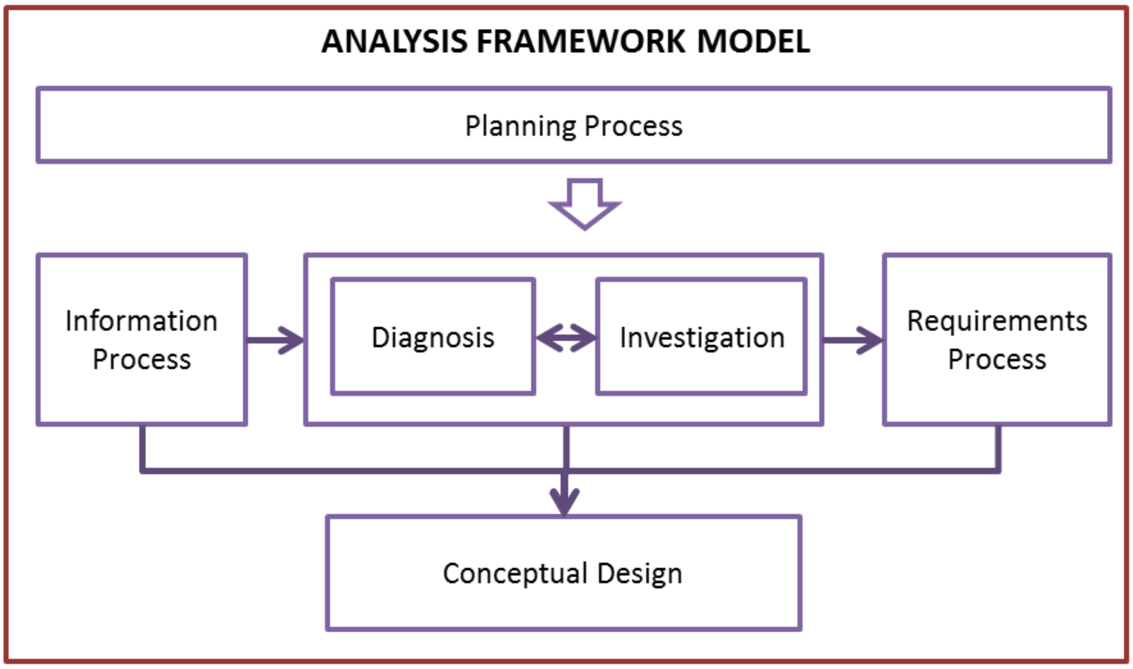
This order is valid at the intellectual level.

Examples:

- An Analyst learns about the situation of a stakeholder. An idea for a solution may pop up in his or her mind.
- An Analyst is working on the design of the solution on Monday but got stuck. He or she decides to get more information and to clarify some aspects on Tuesday. On Wednesday, when (s)he learned what was needed, (s)he can continue with the functional design. The learning (analysis) on Tuesday preceded the functional design of Wednesday.
- A 'solution' is designed based on superficial knowledge. Once implemented the flaws appear. The design is corrected in a subsequent iteration. Here also, learning preceded the (final) design. The problem in this example is that implementation has been tried earlier with insufficient insight resulting in an inappropriate solution, additional risks and waste.

It's all different when we consider this explanation from the perspective of the formally defined activities of a methodology or the project phases. The level of the intellectual activities shouldn't be confused with, for example, the project phases. A project phase is a segment of the project's overall process during which the emphasis is on a certain type of activities and the main objective is to produce an output with a specific purpose. A project phase doesn't exclude the execution of activities of other types. Thus, it is possible, for example, to do some design or programming during the analysis phase. Executing tasks of another type than indicated by the phase is actually quite common. If not, no bug correction, which is done through programming, could be done during the test phase. So, there are three distinctive levels: project phase level, activity level and intellectual level.

A high-level framework model can be elaborated by enriching and expanding the model of the core activities with some other groups of activities. This framework model respects the basic model with the three groups of core activities.



The Information Process contains activities like gathering information, evaluating and selecting information, classifying information, verifying information, managing information, and so on.

The Requirements Process is well known. The standards BABoK of IIBA and Volere of the "Atlantic Systems Guild" show both requirements processes.

An expanded framework model can be different. For example, it is possible to come to a solution without planning or without requirements. This model can also be further enriched with, for example, stakeholder management, testing activities and feedback processes, a solution evaluation and selection process, a product evaluation and an iteration mechanism.

7 Pursued Benefits of Analysis

The main purpose of Analysis is to identify issues and opportunities and to gain understanding in order to increase the likelihood of solving the right problem and solve it rightly and efficiently – reduction of effort, time and cost -, while aiming to maximise the benefits and limiting the risks. Since the conceptual design (functional design) is a part of the Analysis discipline, it also defines the solution.

The Analysis discipline allows posing a diagnosis, creating an overall picture of the system and its environment and it provides more detailed information about the solution as well. This offers various benefits:

- Detection, identification and localisation of problems, issues, opportunities, risks, …
- Getting a decent diagnosis to solve the right problem
- Identification of priorities (not only business priorities or end-user priorities)
- Identification of the area concerned by the problems or opportunity

- Identification of the involved business areas and determination of all the stakeholders. This limits the risk of not considering the whole issue and of omitting an impacted or concerned stakeholder.
- Facilitates interdisciplinary communication and collaboration
- Detection and clarification of ambiguities, hazy situations and issues
- Getting an understanding that can serve as foundation for engineering a structurally integrated solution
- Identification of things which may change and those which must not be changed
- Allows new ideas to pop-up and existing ideas to mature. Immature ideas are an important cause of 'changes'.
- Identification of alternatives and evaluation of these alternatives
- Early detection of mistakes even before they reach programming
- Identifying the required changes and the area in the organisation that will be impacted
- Increases the coherence across the company
- Choice of appropriate technologies
- Prevention of creating new issues elsewhere
- Identification of implications, limitations, new opportunities created by the solution and other consequences.
- Implementation of performance monitoring, systems administration and management mechanisms and of a system recovery mechanism.
- The solution is more likely to be a real solution and to last longer.
- Facilitates communication within and outside the project
- Creation of consensus among the different stakeholders. Disagreements are a cause of 'change'.
- Facilitates documentation writing and training
- Supports project management
- Opportunity to organise and plan projects and/or activities. Maybe different problems can be solved together, some can be solved later or the present project may take future developments already into account.
- Planning of resource needs (required skills and people)
- Planning of budget
- Identification of risks
- Facilitates standardisation
- Facilitates the internal organisation of the source code. This leads to an increased manageability and flexibility.
- Facilitates reuse
- Facilitates product testing
- Faster development
- Saving budget and resources
- … and many more benefits

The Analysis provides a clear picture of the overall situation and of what has to be built. Having this picture provides a great number of benefits. Without it, the project would progress in the dark with little or even no idea of the product to be built and of the goal. The project benefits of advantages which wouldn't be obtained when the diagnosis is absent or wrong, when only a limited portion of the problem is taken into account, when only a part of the problem area is considered, when little insight is present or when only a vague picture of the future solution exists. It allows to plan, to organise, to communicate, to collaborate, to limit risks and so on. It provides some stability and allows to progress more steadily towards that goal. It increases the level of control in the project.

8 The Analyst

The Analyst is a very valuable role in larger companies. Selecting, forming and recruiting should be done with the greatest care. Also, there is no quick fix to become an Analyst. I have reasons to believe that it is much harder to become a proficient Analyst than to become a skilled developer or project manager.

The objective of Analysis is to equip the company with information capabilities and information systems. Without them and without information, the company is dead. That's what is at stake.

Maybe a more suitable name for the general Analyst should be "Information Systems Engineer" or "Information Capability Engineer".

Activities versus Discipline

Anybody can learn standards and methods and develop the skill to apply them. Everybody can analyse, solve problems or engineer something in some way. And anybody can understand the business demand and work out the details. But this doesn't transform a person into an Analyst.

Analysis is not a set of administrative tasks that can be learned simply by being explained what to do, by imitating the working habits of colleagues and by accumulating 'experience' in doing what is asked or expected. Analysis is not a set of activities that can be performed by a project manager, a manager or any other business stakeholder. Analysis is not simply the set of activities in the project's work breakdown structure (WBS).

Analysis is a professional discipline within the domain of Informatics. Since Analysis is a discipline, the norms are higher than if it were simple administrative activities. It requires a much broader and deeper insight and training. There are more prerequisites.

Everybody doesn't have appropriate analysis skills. Everybody can run, but not everybody can run 10 km or more at a good speed. Everybody can fight, but few people have the fighting skills of a trained martial art practitioner.

Role versus Profession

If analysis is considered as a set of activities, then it is not surprising that the analyst is considered as a role.

This role is to be a Business-IT interface or bridge, a refiner, a translator. The analyst is meant to fill the gap between the business community and business domain and the developers.

Considering the true nature of the jobs analysts perform today and what is expected of them, it seems that titles like '*Business Demand Analyst*', Requirements Manager, UML-ist or BPMN-specialist, Modeller, Software Feature Specialist, Specification Writer or Technical Writer would be more appropriate. Even when these roles and skills are combined, the essence of the role of the Analyst is still missing.

Analysing something isn't the same as performing an Analysis. And drawing some models representing a software application is not the same as engineering software.

'Analyst' isn't simply a role. It's first of all profession. There is a huge difference between a person playing a certain role by performing activities and a professional applying his discipline.

Reaching a professional level requires mastering the discipline and being able to reap results through its application. This requires the appropriate skills. It takes decades to master the discipline, to acquire the skills and to become senior in the discipline.

Considering Analyst as a role and assign it as if it was an average job that can be learned on the field, is, for the company, taking a huge risk. Lesser competent analysts may pose a threat to the company. A company should have sufficient professional Analysis capabilities. This is crucial for the long term survival and prosperity.

To be able to practice the discipline correctly, some fundamental prerequisites have to be respected.

What knowledge areas does an Analyst need to master and what are the key skills of the Analyst?

8.1 Analyst's Knowledge Areas

The Analyst solves information problems. He or she studies and engineers information processes and information systems and deals with information in order to get information needs solved. The Analyst has to understand the true nature of these things. (S)He deals essentially with information, systems, information processes and information systems and with organisational frameworks, approaches and methodologies.

Information

An Analyst has to understand the resource 'information'. Understanding information is crucial to understand information needs, to diagnose information issues and to know how to engineer sound information solutions. (S)He has to fully understand the different functions information has within a company, the role information plays in a company. The Analyst has to master different domains within the broader domain of information science.

Systems

The Analyst must be an expert in systems and processes and especially man-made or engineered systems and processes. He or she has to study the science of systems. Without understanding systems it is impossible to diagnose and engineer them.

Information Systems and information Processes

Information systems are the practical combination of information and systems. Information systems implement information processes. The Analyst has to understand information processes and information systems. He or she must understand how to engineer them.

Information systems are not the same as business systems. An information system doesn't become a business system because it is designed by, owned by or used by the business community. This doesn't change its nature.

Similarly, information processes executed by the business community or serving business purposes doesn't become a business process. It remains an information process. Wrongly labelling elements lead to confusion and abuse. Terms should be used with greatest care.

Methodologies and Methods

Not one methodology fits every project. A standard methodology is like a template that serves as a base to the creation of the practical methodology of the specific project. This base is then adapted to the project. The selection of the methodology, its adaptation and the selection of methods and artefacts is a task of the Analyst. He or she has to decide what has to be done, how and when in order to come to a real analysis. This adaptation, as well as the planning of tasks is then done in collaboration with the project manager. Sometimes, the input of business stakeholders is required as well. The Analyst has, to some degree, also to be a methodologist.

The Analyst, as an expert, has really to understand why (s)he does something or why not, when to do it and why, how to do it and why, and so on.

An analyst having project experience, knowing standards, knowing how to elicit and manage requirements and knowing or methods like UML, BPMN or Use Cases but who misses this more profound understanding won't be able to produce work that meet professional norms and which over the long term significantly benefits to the company.

8.2 Analyst's Skills

The Analyst is an engineer. This is a conceptual job, a job of problem solving and methodical creation. This requires different skills than

An Analyst should possess the following skills

- Engineering thinking
- Analytical thinking
- Systemic thinking
- Abstract thinking
- Critical thinking
- Investigation skills
- Information skills
- Learning skills
- Ability to seek to understand the true nature of things and natural laws

- Problem solving skills (which include the critical diagnosis skills)
- Common sense, Sound judgement
- Creativity
- Professional awareness

The listed skills are really critical in the conception of solutions. The Analyst has absolutely to excel in these skills. The Analyst must be a thinker. The knowledge of methods and the years of experience aren’t much worth without these thinking skills. An analyst with average skills may work on simple, independent, peripheral systems. But the more important a system is, the more critical these skills are.

The role of Analyst is first of all intellectual. It requires a general aptitude which is a combination of a certain philosophy, an intellectual attitude and a certain capacity of thinking skills. The way the brain is wired, the thinking patterns, is crucial. Knowledge, insight and experience can be acquired. But it is much harder to acquire thinking skills. Some people may succeed in sharpen their thinking skills, while for others it may be a hopeless attempt. The person may learn methods, train and gain experience to no avail. This knowledge may well mask the incompetence and may lead to worse catastrophes. Methodologies and methods help capable Analysts to be more efficient, but they won’t prevent unable analyst to fail. The analyst can build successfully inappropriate solutions. While, given the lifespan of a system, it is better to ‘unsuccessfully’ build a suitable solution.

The Analyst must certainly be able to function and to think autonomously. This is vital. However, he or she has to work in group as well.

Many analysts do what is demanded and expected. They submit issues to others and expect answers. They work based on solutions being given by others and conceive the demanded solution. The Analyst is dependent of many elements, like the dependency of information and knowledge provided by others. But the attitude and thinking shouldn’t be one of dependency.

Together with the independency comes the professional awareness. Professional awareness means here the consciousness of the real reason why an Analysis activity has to be done or why not, why it has to be done at a certain moment and why it should be done in certain way. It is about the ability to deliberately make these choices and to make the right choices. It is about having a more profound understanding of the nature of information, processes, systems but also, for example, of organisations and to know how to deal correctly with them, to know what may work or won’t work, to know what solution is appropriate and which not. Professional awareness is also about understanding the purpose and true nature of Analysis and the real role and value of the Analyst. This role can be much broader and valuable than the one stated in the job description.

An analyst may performing activities because it is the way things should be done and because they are expected, considering it self-evident to do them. Everybody does it like that. He or she may heavily rely on standards and methods and perform tasks because they are prescribed in the methodology. These are examples of little independence and of a lack of professional awareness. Acquiring this professional awareness is a learning process. It requires the right thinking habits and may easily take decades. This is what the mastery of the discipline is.

Additionally, other intellectual skills, social skills and communication skills are also important. Social skills are often considered as being the most important. They aren't much worth if the skills necessary to come to a solution are mediocre. Social skills are actually supporting skills.

8.3 The Role of the Analyst

Trying to satisfy the business community is definitely a positive objective. Collaborating in building software applications demanded, defined (which can be understood as 'designed') and decided by business people is not the role of the Analyst. His role might be redefined as conceiving software applications facilitating the work of end-users which help them to reach business objectives. This is really too low. We need to leave the frame of thinking of software applications - end-users – developer - software production. This level of thinking is appropriate for developers. It isn't suitable for Analysts. The real role of the Analyst is so much more important.

An Analyst, as a professional, is first of all an information expert and a systems expert. The Analyst's real challenge is to conceive the whole formally, organised and managed part of the company that deals with information. He or she is also responsible for the conception of a skilled and equipped information environment propitious to a maximal exploitation of information to the benefit of the company and its clients. The Analyst seeks to create a stronger company. The true value delivered by Analysts is for the company. So the company's interest predominates over the interests of its organisational units, and even more over interests of individuals. It's a matter of greater priority, not a matter of ignoring. Simply, it is vital to have a well-functioning, robust, manageable, adaptable and scalable company. The analyst's work has to support the company's longevity. This is the way to contribute to the company's prosperity.

The role of the analyst is to change the structure, the components and the functioning of the company. A software development project building a new software system does not simply create a new software application. It doesn't just add a component to the company. It also changes the way the company, as a supra-system, functions. The starting point for the Analyst is always this existing supra-system that has to be adapted.

The Analyst does more than producing plans for software applications. The solution can be of different types. They may concern adaptation or creation information processes or information systems, reorganisation of the information, cleaning up information, creation or adaptation of policies related to information, training or even the creation of new information products and services. And (s)he can propose more business related solutions as well.

Even though the objective of a particular project is to conceive a software application for the business community, the Analyst's concern to preserve and to contribute to the capabilities and characteristics of the company remains. For example, implementing a lot of information duplicating features may imperil the information resource. The Analyst has to safeguard the customer from using inappropriate and deficient information concepts; from wrong or ignored software engineered; from applying principles; from following simplistic software development philosophies; from unjustified objectives, norms and expectations; from neglecting implications, and so on.

The Analyst has to comprehend where the company is heading to. This concerns objectives, strategies, plans, tactics, policies, business drivers, tendencies, evolutions, forces, or even intentions and opportunities. He or she will also pay attention to indications of issues like symptoms, negative tendencies, complaints, results going down, numbers showing a negative trend, and so on.

An Analyst can work on continuous basis or with projects, or both. And each project is different. There is not a unique Analysis approach that fits to all projects and situations.

The Analyst defines the analysis approach, plans his work and defines the methods to be used in Analysis. Because the analyst is greatly dependent on the input and collaboration of others and of his/hers findings, because of the uncertainties and because the mind doesn't work at a regular pace, it makes little sense to freeze this plan. This plan serves as guidance and is likely to change. As an expert, the Analyst has the responsibility and authority to define also the prerequisites, his needs related to the working environment, collaboration and to the chosen approach. Normally, this should concern access to information, availability and collaboration of people, usage of time, an environment propitious to think work and creativity, tools, and so on.

The Analyst is responsible for the diagnosis and for the identification of opportunities. He or she determines the boundaries of the areas to be investigated. The boundaries are normally determined by the diagnosed issues and by all the elements that one way or another are concerned by these issues.

The Analyst determines the impacted areas. He or she determines the different knowledge areas and disciplines required to conceive the solution and that are exploited rightly at the right time in the project. And he or she also ensures that all the required aspects and perspectives are considered and taken into account. Business knowledge and technological knowledge form only a part of the required knowledge areas. Analysis, as a discipline, provides an important input itself. The Analyst determines all the concerned stakeholders. They have all to be involved in the project. He or she orchestrates their collaboration so that all the right people and right disciplines play their role at the right time.

The Analyst has to gather relevant information and has to verify it. Commonly, the Analyst has to acquire advanced business knowledge. However, he or she doesn't really need to become an expert in the business domain.

The Analyst evaluates situations and conceives the logical solution or the logical adaptation to systems and necessities for the implementation.

The Analyst is responsible for the proposed design, the proposed alternatives, their evaluation, the characteristics of the product, how well it fits into the environment, and so on.

The Analyst brings a huge benefit to the company. But if the required Analysis skills are missing or if the role of analyst is restrained, it can harm the company very much as well. This can be a waste of time, money and resources. There can be a waste of opportunity by implementing a mediocre solution. Since it requires the understanding of this mediocrity and much more of how much better another solution could have been, few are even aware of the waste and missed opportunity.

A chaotic or little known and unmanaged or mismanaged work environment generates a lot of uncertainties. Many unexpected issues may surface. This won't facilitate the Analyst's job.

The analysis has failed when the wrong solution has been built or if new problems are created. However, this doesn't necessarily mean that it is the a/Analyst's fault. The a/Analyst depends heavily of different people and of the overall situation (e.g. the quality of existing systems and documentation). People may, even unwittingly, withhold or distort information. They may disagree with each other or regularly change their mind. They may, in many ways, have a very detrimental influence on the Analyst's work in many ways.

The Analyst is an Agent of Change. Agent of change is the person that invents the change, not the one that is limited to apply the change. He is a key person in innovation. Innovation requires the understanding of information, systems, the possibilities and limitations of software and computers and the work environment and the business combined with a great dose of creativity and the ability to rethink the world freely, not bound by the existing paradigms.

There is a great confusion about the difference between a Project Manager and an Analyst. The Project Manager is responsible about the project. The Analyst is responsible about the product.

The Project Manager is responsible for the temporary organisation, the process, the plans and the resources of the project. The Analyst is responsible of every aspect defining the product. Both have to collaborate.

Examples:

- The Analyst has to respect the scope. However, it may happen that scope doesn't correspond with the diagnosed issues or with beneficial ideas and opportunities. A common agreement should be reached through consultation with the business stakeholders and project manager.
- The conception of the future solution may introduce risks or require an adaption of the plans, the budget or the needed resources. This should also be discussed with the project manager and business stakeholders.

Therefore, the Analyst should have at least some basic but sound knowledge of project management as well.

It is interesting to note that, because of the specific knowledge and skills, the Analysis discipline is also of great value in domains and disciplines other than informatics.

9 A Few Other Issues

9.1 Requirements

Many software development approaches focus on requirements. Requirements are an important concept in software development approaches.

The concept of requirements has two purposes. The first and main purpose is a conceptual function. The second is the role of criteria.

- The idea of the requirements process is to record statements (the requirements) expressing capabilities and characteristics of the future solution. These are key statements about the solution. A few requirements will tell little about the future solution. But the more requirements are recorded, the solution is known and the more it takes shape. As the amount of requirements grows, a picture of the future solution gradually arises and crystallises. The main line presents themselves to the Analyst. It is then only a matter of clarifying still hazy areas and to fill in the gaps. The solution emerges from the requirements.

- The collection of requirements can be served to perform some key tests of the future solution and, provided the requirements are kept up-to-date, to keep the system aligned.

Requirements are not an obligatory tactic to come to a solution.

Requirements are an output of some Analysis activities. They are the result of the investigation of the problem, problem area and the broader situation. Requirements shouldn't be received from the business stakeholders or collected. It is the responsibility of the Analyst to determine them based on the investigation and on the information received from the business stakeholders.

Requirements don't guarantee that the product will solve the problem. A solution, product or system may meet the requirements and these latter can all be right, yet the problem may still not be solved or new problems may be created. A test proofs only what is being tested. Testing a product against its requirements proofs only the correspondence between the requirements and the product. It doesn't proof the system will solve the problem, fit into the environment or causing no problems. Requirements can be right but irrelevant and requirements relevant to the real problem may be missing.

The requirements approach still requires a right diagnosis. Requirements don't solve the problem of wrong diagnosis, wrong scope and wrong objective. Requirements are based on business knowledge, but they can also be based on a certain insight of a context and on an understanding of a problem. Or, a different diagnosis of the same issue or a different insight on a same situation may lead to different requirements. Requirements do not replace the diagnosis or a decent investigation. Requirements are only a help to come to a solution. Requirements are solution-oriented. The requirements process comes after the diagnosis.

An analyst may have registered a lot of requirements, but lacking of insight in the situation. Or, the Analyst may understand the situation very well without having established a list of requirements. The risk for developing a non-solution is greater in the first case than in the latter.

The Analyst has to understand the organisations, domains, areas, mechanisms, elements, aspects and forces influencing the conception of the system and which are impacted by it. They form a source of requirements. All of them have to be taken into account. Basing design solely on requirements of the business domain or business community means that a lot of information necessary to an appropriate information solution has not been taken into account.

Gathering requirements should not be confused with doing an Analysis. And failing to do a sound analysis leads to false and very changing requirements.

Requirements are produced by analytical activities. They are the input to the conception of the solution. They are an input neither to analysis, nor to programming activities.

Requirements should increase the likelihood of building a solution with the right capabilities and characteristics. It should not serve as substitute to a lack of insight in the subject. Neither should it replace problem solving skills or the creative work in the conception process.

The objective of requirements approaches is not to get all the requirements. This can never be the objective because getting all the requirements is impossible. By their very nature and by the role they play, they can never be ‘complete’. The intention is to obtain sufficient requirements to come to a conceptual solution.

If two or more alternative ways to achieve a goal, we may expect mutually exclusive requirements. Their status of being obligatory can then be changed accordingly to the chosen way.

Vigilance is required. A requirement may implicitly hide a design decision. Such decisions shouldn’t occur through requirements.

Just-in-time is a concept that is recently being promoted in the requirements approach. The requirement must be determined to know whether, when and where to use it.

The overall solution can be divided into parts. And the overall investigation area of the project can be divided into sub-areas linked to these solution parts. And we may assume that all the requirements for that part of the solution will be found in the sub-area. Normally, the determined requirements are related to the subject and area of investigation. But this is not always the case. Investigating a subject or area may lead to the detection of requirements that has little or even no relation with the investigated subject or issue. The investigation of one subject and area may lead to requirements for another subject. The overall solution is more than an assemblage of its parts. Another reason is that a lot is interlinked.

A new requirement which hasn’t been triggered by a real change in the business domain or in the business environment and which forces an adaptation of what has already been built is actually a requirement that is too late.

Trying to receive as many requirements as possible in order to obtain a complete and consistent set of requirements that fairly well describe the solution to be built, without diagnosis, decent analysis and little understanding of the real situation and of the broader environment is definitely not the way to go.

9.2 Analysis Artefacts and Documentation

Analysis artefacts are documents. They can be delivered as documents. The Analyst documents processes and systems. He or she may document existing processes and systems, although in an engineered, organised and manged environment, this documentation should already exist.

Analysis artefacts are not project documentation or system documentation. They are, or should be, elaborated with a complete different mind-set and purpose. Performing an Analysis is by no means the same as producing documentation. Or, writing documentation has nothing to do with Analysis.

Documentation is created to describe something. Analysis artefacts also describe something. But the primary purpose of Analysis artefacts is different. Its main purpose is to support the process of thinking focussed on a gradual elaboration of a concept. This thinking occurs not only before, but also during the elaboration of the Analysis artefacts. The different forms of thinking have been listed in the chapter about the Analyst's skills. Analysis artefacts are not meant to passively absorb its content. The reader of these artefacts is supposed to think critically about aspects like the clarity, the truthfulness, the coherence, the feasibility, the implications and the risks. These artefacts feed discussions and help to come to agreements.

Only afterwards, when the thinking and discussions are over and agreements are made, the Analysis artefacts can be used as documentation describing the new system for experts or they can be used as input to write different manuals. These artefacts form then the basis for other and future work, like the Analysis of a next release.

10 Conclusion

Information and systems form the core knowledge of the Analysis discipline. These two elements pervade our society and modern companies. This indicates how crucial and powerful Systems Analysis is. Analysis brings the thinking about the problems and solutions to a much higher level: abstract, in terms of systems, mechanisms, concepts, structures, holistic, integration, interactions, long term, opportunity, efficiency, innovation, reuse, and so on. The sphere of thinking is often limited to business thinking and thinking about what and how to program the software application. Analysis broadens this sphere significantly.

In this document the essence and the purpose of Analysis have been discussed. A certain mind-set of the Analyst has been drawn. Misapprehensions, ignored principles, confusions and common traps have been clarified. This document pleads for higher and appropriate norms for the discipline. Furthermore, the role of the Analyst has been described, as well as the required knowledge area and skills.

The discipline hasn't reached its full maturity yet. It has to evolve from a purely software oriented discipline to a broader discipline. In the world of Analysts, but even more outside this world, a distorted picture about the discipline exists. This makes the discipline extremely vulnerable. Practitioners have to safeguard it from the forces pushing it on to an undesired path. Analysis reflects on information issues and solves these problems. Now Analysts have to analyse and to reflect on Analysis.

Axel Vanhooren

Freelance Consultant
Business Informatics and IS Methodologies
Axel.Vanhooren@taurus-ee.com

Download latest version here   (Free to distribute)

Other related documents:
Problem Solving: Concepts and Approach for Systems and Strategies
The Waterfall: A Commonly Misapprehended Powerful Methodological Concept