3.5

SYSTEMS ENGINEERING IN THE WORLD OF TRAINING AND CONSULTING

3.5.1 "COMBINING ENGINEERING AND MANAGEMENT SKILLS"

An Interview with Dr. Eric Honour

Systems engineering is a new discipline, created by the need for it, which originated in the field. As part of its evolution process, it is gradually establishing itself as a profession and an area of academic research and study. We talked with an experienced systems engineer, who has lived through this field's evolution alongside that of his own professional career, which included field work as well as teaching and research over the years.

Background

After graduating high school, Eric Honour joined the US Naval Academy, where he obtained a bachelor's degree in systems engineering.

Eric Honour: "Back then, during the 1960s, we had no courses in systems engineering processes, like they do today. Systems engineering was mostly about control system theory. Our studies focused on system analysis, a combination of mechanical

Managing and Engineering Complex Technological Systems, First Edition.
Avigdor Zonnenshain and Shuki Stauber.

and electronic systems. The software field was making its first steps at the time, and computer studies were included in the systems engineering curriculum. We did a lot of experiments and mathematical calculations. We received a combination of officers' and system analyzers' training."

He was discharged from the military 9 years later, in 1978, having served as a pilot in the navy for a large part of those years. During the last years of his service, he served as an instructor on engineering fields, like electronics and thermodynamics.

This basic training is unusual for a systems engineer. Most systems engineers are trained in one of the basic engineering fields and only then, having acquired some hands-on experience and having recognized their tendency towards systematic work, become systems engineers. Eric Honour's basic training, however, was in systems engineering from the start. This is because he began his career path in the US navy, an organization with unique characteristics and needs. This experience proved advantageous for him later on.

Eric Honour: "The fact that I had a military career before I started practicing systems engineering had two advantages for me. The first: it allowed me to acquire leadership skills, as an officer. Most engineers don't get that kind of training. The second: I had acquired knowledge about the operative use of complex systems, so that when I designed systems later on, I kept thinking about how people would use them. People purchase the system to use it in combination with the other systems in the operative environment. I was able to look at the systems not only as a developer and designer, but also as a user."

(See interview with Henry Broodney for a discussion on a similar topic.)

(In this context, see also the discussion with Hillary Sillitto about the mobile phone becoming the number one cause of car accidents.)

After completing his military service, he spent a few months working in software development and then joined E-Systems, a company engaged in military communications systems, where he worked until the late 1980s. During this time, he had served five different positions, most of them in system engineering on various organizational levels.

In his first job as a principal engineer, he designed electronic circuits for signal processing systems. Afterwards, he was promoted to an administrative position as the systems engineer in charge of system design, where he had to manage the engineers who performed the detailed design of the systems' components. With a job title of Engineering Supervisor, he managed an engineering team of nine. His job was to coordinate between the different factors in the design of systems for various projects in the communication intelligence field – military communication systems that allowed one to listen to enemy communication traffic.

He continued to advance, and for the next 3 years, he served as an engineering manager with nearly 50 engineers in three groups under his authority. In this position, he also continued to be actively involved in the systems engineering of larger systems worked by those groups, including being the designated lead systems engineer for two major systems. After this experience, he was offered to leave the systems engineering and take up project management.

Eric Honour: "I began working with costs, schedules, stakeholders' involvement, preparing proposals for clients. I did this for 3 years, but after all that, I chose to return to systems engineering, because I realized that I loved the professional, technical work more than administrative work. During my time at E-Systems, I got to work on projects at all stages, from requirement formulation to delivery."

Being well aware of what he wanted to do, Eric Honour began working for Harris Corporation, a company that also specialized in military communications systems, while also working on major systems for NASA and for the FBI, where he spent another 10 years, this time, in only one position: Senior Principal Systems Engineer. As part of this job, he was integrated into various projects within his area of expertise, including projects in communications intelligence, smart guidance systems and more.

In the late 1990s, the company encountered some economic difficulties and was forced to make company-wide pay cuts. This circumstance led Eric Honour to his decision to resign and establish his own business, a company that offers consulting and training in systems engineering fields.

Eric Honour: "I realized that job security in large companies was no longer guaranteed. I was also the president of INCOSE at the time; I had an international reputation and good connections, so the timing was good. At first, I had aimed toward consulting only, but over the years it became evident that training courses were better business."

The company employs seven experienced instructors who lead dozens of focused courses a year, mostly on client-site. It also provides consulting and research services. Recently, Eric Honour has also completed his dissertation and received his PhD.

Professional Background and Systems Engineer Training

Eric Honour believes that the basic engineering disciplines that provide a good knowledge infrastructure for a systems engineer are electronics engineering, mechanical engineering, and software engineering; because most systems include these three core fields within them.

He is not among those who denounce basic systems engineering studies.

Eric Honour: "Today, the convention regarding systems engineering training is to study for a master's degree in systems engineering, having already gained some experience; but there are undergraduate programs available in the United States, and I believe it is entirely possible to study this field without prior experience. After graduation, one can gradually be integrated into the practical field. It is not very different from an electronics engineer who studies for 4 years and then starts working at an organization, having no prior experience. Clearly, his first assignment will not be designing a complex electronic system. He will first be attached to a mentor and charged with a relatively simple task within the framework of designing a complex electronic system. Similarly, a newly graduated systems engineer can first be assigned less complex tasks and aided by a mentor. Much of the knowledge I acquired in this field was gained by experience and through mentoring."

Eric Honour, owner of a company that provides systems engineering courses, explains the difference between the activity of universities and companies like his in the systems engineering field: "Most of the universities' activity in the field consists of long education programs that grant academic degrees to those who complete them. Many of the lecturers in these programs are academics with no hands-on field experience. There is no competition between these studies and the ones provided by training companies; the two complement each other. However, in recent years, in order to generate more income, universities have begun to offer on client-site short-term courses. For this purpose, they also offer instructors with field experience. This particular activity indeed competes with private training companies."

The question, then, is: if in the past, the added value these companies possessed was lecturers with vast experience, which academics could not offer, what added value can these companies offer their clients today?

Eric Honour: "Because of their structure, universities can't always pay good, experienced lecturers salaries worthy of their status and are, therefore, often unable to employ the best ones. Private training companies do not suffer from this limitation, being able to bring the best lecturers and customize short and focused courses to suit the needs of a specific company's employees. It's not just about teaching processes or administration, it's about thinking systematically. It's not easy to teach, and so, training has to be left to those who are experienced and able to bring good examples from their personal experience."

It is his belief that there is a fundamental mistake in the way systems engineers are trained today. According to him, too much emphasis is placed on processes, rather than instilling system analysis skills: "That is how systems engineers become process engineers. Instead of telling others what to do, they do things themselves. This is wrong. It is impossible to construct a system without system analysis. That is the systems engineer's job, because none of the core engineering disciplines include the knowledge of how to perform a system analysis."

He says that a systems engineer needs to be trained as an engineer, but also needs knowledge in administration and project management, namely, the management of tasks and schedules.

He also stresses the importance of leadership: "A good school will teach more than just systems engineering courses, but project management courses as well. And if it's a really good school, it will also teach leadership, the ability to motivate others, to make a group work together. Leadership may be hard to teach, but some leadership skills can be taught, despite it being an inborn trait."

Eric Honour divided systems engineer training into three areas: technical, administrative, and leadership. In his opinion, both the technical and the administrative areas can be taught in courses and through implementation in the field – by gaining experience at work. However, the best way to instill leadership skills and teach one to think like a leader is mentorship combined with experiential teamwork. He says that solving problems together helps build teamwork skills and helps the students learn how to work with other people.

He testifies that he had learned a lot from a mentor who had accompanied him at E-systems: "Mentorship is a powerful training tool, and it's a pity companies don't use it more widely."

The Evolution of Systems Engineering

Eric Honour says that 40 years ago, when he had taken his first steps as a systems engineer, systems engineering had been radically different from what we know today. During those years, it had undergone some significant changes, yet the basic patterns designed in the 1960s were similar to those used today: "In the late 50s, a book was published, that laid the foundation for systems engineering. About a decade later, in 1969, the first systems engineering standard was published, that set standards and presented processes similar to those we see today. When I studied at the academy, the emphasis in systems engineering was more on systems analysis, rather than processes and standards. In the 50s and 60s, systems engineers wrote textbooks and standards for what was rapidly becoming a professional field based on analysis and common sense – the right way to do the job. By the early 70s systems engineering had full structure as an engineering discipline."

However, according to Honour, the SE discipline largely fell into disuse through the 1970s and 1980s, except in some narrow areas. What caused this change?

Eric Honour: "In the early 1970s, managers were viewing systems engineering as 'just common sense,' and they began to put less emphasis on it in projects. Funding for SE was reduced. There were some areas where it continued strong, such as the Navy's Aegis (Naval combat systems) program and the NASA space shuttle development. During those years, these organizations were deeply engaged in systems engineering, but the field did not evolve outside of them. Only during the 1980s, after the breakthrough in the software engineering field, did awareness of systems engineering begin to grow again. This change was due to a significant problem encountered in the late 1980s: many software failures were discovered, because software personnel had not received the information they needed, at the quality they required. They began looking at ways of receiving better requirement specifications, and that put systems engineering back into people's minds."

However, many claim that software specialists only care about software.

Eric Honour: "True. And so, to write better software, they wanted better requirements specifications. In time, their approach changed. In the last 20 years, they switched from an approach based on the desire to receive better requirements to one relying on client relations, striving to structure the requirements together with the client."

He believes the evolution of systems engineering also depends on it becoming a scientific discipline: "Take electronics for example. In mid 19th century, people built electric engines. They had no idea how they worked, they just knew that they did. It had taken many years for the link between electricity and magnetic fields to be discovered and mathematical tools to be formulated, allowing the mechanisms of generators and engines to be analyzed. These discoveries created a new world of electronics, radio and frequencies.

So if we want systems engineering to get better, we need scientific research that includes three stages of development: analyzing events that show us what systems engineering is (your book is a good example of this), empirical research and, finally, theoretical research. Today, systems engineering is somewhere between the first and second phases."

More Insights on Systems Engineering

On the essence of systems engineering

- "Systems engineering is several things: it is a discipline, because it articulates a list of processes that, if implemented, provide us with better results. It is also a way of thinking, because if we don't think correctly, the processes by themselves will not be much help to us, because the core of systems engineering is systematic thinking – the ability to see the big pictures and base decisions on it. It is also a profession; because there is a group of people who have embraced this way of thinking and these processes in order to use them to do things better. When people ask me what systems engineering is, I explain that each part in an airplane is made by a different engineer, while a systems engineer looks at how all the parts come together. He tells each engineer what to design within his specialization. This means systems engineering combines engineering and management. A systems engineer needs to possess both administrative and engineering skills. He can't tell the engineers what to design, if he doesn't know how it fits into the system; and he can't tell them what to design, if he doesn't have good relations with them."

On systems engineering and project management

- "Systems engineering and project management differ when it comes to priorities: project managers focus on the tasks, schedule and budget. The technical manager (or chief systems engineer, or whatever you call him) is responsible for the results the tasks yield.

  The project manager wants to accomplish the mission. The systems engineer wants to make sure it is accomplished well. Their goals are the same, but their priorities are different. The project manager looks at the cost first, then at the schedule and only then at the technical aspects. The system's engineer's priority list is reversed."

Eric Honour demonstrates these differences

- "Let's say a problem is discovered in the link between software and hardware. The principal systems engineer comes to the project manager and says 'there's a problem here, because the software engineer and the mechanical engineer have chosen different paths, and now we need to perform a forward analysis and decide where we go from here.' The project manager then asks how much this will cost, and the principal systems engineer says 'fifty thousand dollars, because we need to make a software simulation to assess the gaps.' And then, the project manager says he doesn't have that much money."

On systems engineering and intercultural differences

- Eric Honour finds that intercultural differences are mostly differences in language and the different conceptualizations adopted by systems engineers in different countries, or even different industries. However, the systems engineering thought patterns, the ways we look at the big picture, are fairly similar.
- He notes a relevant, significant difference between the United States, a culture that pushes forward to get results, and European culture, prevalent in countries like the United Kingdom and Germany, where the cooperative approach is more common. He also mentions Israel and its high motivation and entrepreneurial passion.

On a systems engineer's skills

- People choose to study engineering because they love things more than they love people. So, at first, as engineers, we learn about things. Then people grow to become systems engineers, because they realize that it's impossible to build complex systems without understanding people as well as things. They understand that it's more important than mechanical or electronic or software design.

3.5.2 "MODEL-BASED SYSTEMS ENGINEERING"

An Interview with Sanford (Sandy) Friedenthal

Systems engineering is an evolving discipline that requires one to possess, among other things, hands-on experience and training in many facets of engineering in order to be applied effectively. In this chapter, we describe how an engineer became a systems engineer in US aerospace and defense industry, and we discuss the path he took to eventually become an expert consultant in this field.

Background

Hughes Aircraft Company:

After graduating from a general engineering studies program at UCLA in 1973, Sandy Friedenthal began working for the Hughes Aircraft Company, Missile Systems Division in Canoga Park, California. Hughes Aircraft was a major US aerospace and defense contractor that was owned by the Howard Hughes Medical Institute, a non-profit medical research organization founded by Howard Hughes. The Missile Systems Division developed and produced many state-of-the-art missiles including Maverick, Phoenix, and TOW. The company maintained a culture of technical excellence and a long-term view to advance critical technologies.

Sandy joined the missile guidance lab, where the focus was on developing and applying advanced missile seeker and guidance technologies to missile system design. Sandy gained hands on-experience working in a high-technology engineering lab and was mentored by technical experts in the field. The work included development of prototypes using radio-frequency traveling wave tubes, electro-optics sensors, and guidance algorithms.

After a few years working for Hughes, Sandy decided to continue his education towards a master's degree in control systems.

Sandy: "I chose the control field based on personal interest and practical considerations. The field of study was highly relevant to my work at Hughes, but had an analytical orientation that was of interest to me. As I found out later, the control systems theory turned out to provide an excellent foundation for systems engineering."

In his job, he was able to apply the analytical techniques he learned in school. The application of control systems techniques was all about modeling the system and analyzing its performance. However, the work remained focused on the control aspects of the system with limited emphasis on other multi-disciplinary aspects of the system. He was using systems engineering methods, but only within the framework of my specific discipline.

The company's organizational unit included specialized engineering departments that focused on different aspects of the missile, such as guidance and control, propulsion, aerodynamics, and warheads. "From my perspective as a young engineer, there was not a lot of cross-disciplinary integration. The integration occurred at the program level by special teams in charge of integration and test."

Martin Marietta Corporation, the Early Days

Sandy left Hughes Aircraft in 1981 for a new opportunity at Martin Marietta in Orlando, Florida. Sandy had been exposed to Martin Marietta as part of a proposal effort at Hughes Aircraft, and was impressed with their proposal contribution.

Martin Marietta was a large and diversified US aerospace and defense contractor. Sandy joined the Advanced Concepts Group in their Electronics and Missile Systems Division, where he worked on new concepts for missile systems. He had the opportunity to work on an advanced concept for an air-to-air missile that integrated technologies across the entire missile system including seekers, guidance and control, propulsion, airframes, and warheads. The advanced concepts projects were not just on paper, but involved conceiving, developing, building, and testing the concepts.

Sandy found the culture of Martin Marietta quite different from that of Hughes Aircraft: "Not having to focus on profit, Hughes Aircraft maintained a long term time horizon. They often planned their research programs out 10 years. In contrast, Martin Marietta, which was a for profit company, planned 3 years ahead. The difference between a 3 year time horizon and a 10 year time horizon had a significant impact on the pace of work. (Today, planning even 3 years ahead is considered long term). There was a constant sense of urgency; where schedule mattered, even within the Advanced Concepts Group.

Additionally, unlike Hughes Aircraft, where the work was performed as part of the specialized departments, Martin Marietta had a matrix organization where people were assigned from their specialized functional department to work on a project with other technical and non-technical disciplines. In addition, at Martin Marietta, the manufacturing plant and the engineering development were located in the same place, unlike Hughes Aircraft, where the engineering and manufacturing were performed in different states. As a result of these business differences, the nature of my work was significantly different at Martin Marietta. I enjoyed the pace at Martin Marietta, and the exposure to other disciplines and programs, which helped me gain a broader perspective."

Sandy emphasizes: "This environment created partnerships between disciplines working on the general design of systems. Systems engineering methodologies, as well as the term itself, were used at Martin Marietta. I was able to support overall systems development on a daily basis."

Evolving as a Systems Engineer at Martin Marietta

Sandy transitioned from the Advanced Concepts Group to work as a systems engineer on a complex electro-optical system that was transitioning to manufacturing. There, he helped to support engineering changes, and to document and improve the way the system was built and tested on the production line. Over the next several years at Martin Marietta, he had increasing levels of responsibility in systems engineering. In the mid-1980s, he became manager of the Systems Requirements, Design, and Integration Section, and later became Director of the Systems Engineering Department.

At the request of the VP of Engineering, Sandy then took the responsibility to lead a new company initiative in concurrent engineering. Concurrent engineering is an approach that is fundamental to good systems engineering. It is all about involving the right disciplines early enough in the design process, so that down-stream processes such as manufacturing and support are adequately considered. Engaging these disciplines early in design fundamentally changes how the work is done, and the methods, tools, and training have to be adapted accordingly.

Sandy Friedenthal: "We formed a work group of senior-level executives from different disciplines, and together, we formulated a strategy for how to implement concurrent engineering across the organization and on projects. We also worked with our customers to involve them in this strategy. It was an all-encompassing initiative with many facets: from planning, development of the methods and tools, training personnel, and working with projects to implement the approach. The application of concurrent engineering to the Navigation and Targeting System on a particular advanced helicopter project was widely publicized across Martin Marietta and their customer community, and was considered a successful early adopter of this approach. In the late 80s, Martin Marietta acquired another company's operations that developed reconnaissance systems. I joined the program as the technical director (one of the names organizations use for lead systems engineers. See, for instance, the interview with Mimi Timnat from Elbit Systems). Much of the work was subcontracted out. I applied systems engineering to the management of the subcontractors and to ensure the overall system requirements were satisfied. There were also many organizational challenges, due to mix of cultures between Martin Marietta and the acquired company."

Continuing Career at Lockheed Martin

Martin Marietta merged with Lockheed to become Lockheed Martin in the mid 1990s. Sandy and his family moved to the Northern Virginia area to continue his career with Lockheed Martin. There, he had many opportunities to grow as a systems engineer and work in a variety of technical and management positions. Sandy performed in many different systems engineering roles, some on projects and others leading engineering improvement initiatives.

In his last years at Lockheed Martin, Sandy led up the Corporate Initiative on model-based systems development (MBSD), where he was responsible for developing strategies to deploy a model-based approach across the Business Units, and to provide direct MBSD support to the programs. MBSD, or model-based systems engineering as it is more generally known across industry, involves formalizing how systems engineering is performed through the use of system models. Sandy left Lockheed Martin after a long and interesting career at the end of 2010, and became an independent consultant in model-based systems engineering

Sandy did not begin his career as a systems engineer. Rather, he evolved to become a systems engineer, and it became his job title later. The systems engineering role continues to evolve and broaden throughout his career.

Based on his personal experience, Sandy explained to us how he evolved to become a systems engineer. "You don't typically start off as a systems engineer. For me, it was an evolution where I started with a particular technical focus in the lab primarily dealing with electronics, then my technical focus shifted to guidance and controls, followed by increasing involvement with other technical disciplines, and different phases of a system's development from conceptual design through manufacturing. As I moved through my career, I had increasing exposure to other aspects of the system. The challenge for how these different aspects of the system and the project work together continued to present itself. I was intrigued by this challenge."

Sandy continues: "My background in control systems gave me a starting perspective for how to think about systems. My exposure to multiple disciplines enabled me to gain a diverse perspective on the nature of systems. Since I had the opportunity to support a variety of programs, I also began to see how systems engineering could be applied to different systems."

Sandy found that the principles of systems engineering, could be applied to things other than technological systems. Many of Sandy's assignments were associated with leading organizational initiatives. The systems approach was extremely useful approach to deal with large complex organizational challenges, where many stakeholders are involved with widely varying needs. The systems approach can be applied to help create some order from the chaos. As the lead for the MBSD initiative at Lockheed Martin, Sandy found it useful to think of the overall modeling infrastructure of modeling methods, tools, training, and project support as a system that has its own life cycle and interfaces. The systems approach enabled him to deal with the various complexities and aspects of the modeling infrastructure in a holistic manner.

More Insights on Systems Engineering

On the evolution of a systems engineer

– "Formally, I became a systems engineer when I joined the Systems Requirements, Design and Integration Section at Martin Marietta (during the first half of the 80s, see the preceding text). The systems engineers recruited into the department came from various fields of engineering. Back then, there was no one prominent field that produced more systems engineers. Today, we are seeing more software engineers and architects transition to systems engineering. This is in part due to the increasing scope of software in systems.
  Eventually, what leads people to systems engineering is, for the most part, their personalities. Analytical skills, teamwork ability, and a passion to understand how the pieces work together are of great importance.
  It is possible to learn how to be a systems engineer. It is a continuous learning process. Every time I work with a new discipline, I try to examine how I can adopt their perspective into my thinking about systems. In order to succeed in this, you need to establish some framework for thinking about systems."

On System Thinking

– "Today, when I work with systems engineers on a project using model-based systems engineering, I try to employ classic systems engineering methods: requirements, architecture, trade-off, the 'illities' (a generic suffix used for various system abilities; see also interview with Olivier de Weck) and verification. However, I also try to apply system thinking in everything I do, as a way of approaching problems. The focus of this approach is to understand different stakeholder perspectives and concerns, and define a problem first before jumping to a solution. Then establish value from the perspective of the stakeholders, determine alternative approaches to address the problem, evaluate the alternative solutions, and validate the solution addresses the need. System thinking provides a way to think about how the pieces of the solution fit together to address the problem."

Opportunities and Challenges

– "One of the difficulties in learning and communicating the benefit of systems engineering is the abstract nature of what systems engineering produces. Mechanical engineers focus on the mechanical aspects of a system and produce mechanical designs. Likewise, electrical engineers focus on the electrical aspects of a system, and produce electrical designs. The products these disciplines produce are very tangible, such as a computer aided design of a mechanical assembly or a circuit card. Software engineers also produce a tangible product, namely, code. On the other hand, a primary product of systems engineering is the specifications of the system components and the supporting analysis and data that demonstrate how these components integrate to accomplish the system objectives. The systems engineering products are more abstract than the products produced by many other engineering disciplines.

A critical characteristic of a systems engineer is the ability to think abstractly. This is a challenge for many people who either do not want to or are unable to think this way. A systems engineer must strive to see the picture in its entirety."

Sandy concludes: "I believe the more widespread adoption of systems engineering and systems thinking can provide an important contribution to help solve current and future problems. However, based on my experience, it takes a long time to build an effective framework to think about systems as a whole. We see evidence every day in systems development, in business, and in government, where considerations of the whole are compromised at the expense of considerations for particular pieces or facets of the whole. Perhaps what is needed is to introduce systems thinking beginning in early education, and then reinforce this framework throughout the educational process."

3.5.3 "THE MAIN REQUIREMENT: KEEPING UP WITH SCHEDULES"

An Interview with Niels Malotaux

One of the main problems that arise during projects is failure to meet deadlines. Consequently, a large number of projects are not completed on time. So, a large number, in fact, that many, including quite a few systems engineers, believe such situations to be acts of fate, often brought on by developments that cannot be foreseen. This makes it a self-fulfilling prophecy.

Niels Malotaux, working as a Project Coach, challenges this perception, because his experience tells him that projects do not have to be late. Over the years, he has developed the techniques to put this into effect.

This chapter will deal with Malotaux's perception of systems engineering, emphasizing the issue of failing meeting deadlines and what to do about it.

Background

As an electronics engineer, Niels Malotaux devoted the first half of his career to the development of systems with integrated hardware, electronics, and software. For 18 years, he has been managing an electronic systems design company in the Netherlands, alongside his brother, followed by 15 years of Project Coaching, developing, and honing techniques for successfully running projects in real practice.

Niels Malotaux: "As electronics engineers we specialized in systems design. When farmers approached us with the need to develop a system for climate control of pig-stables, feeding the pigs, as well for example weighing chickens, we looked at the whole system, including the interfaces, the sensors, and actuators, the PC at the farmer's bedside table, how it would be used, and how it could easily be programmed by local software people without knowledge of the details of the electronics hardware."

Niels Malotaux does not view himself as a systems engineer: "Systems Engineers accuse 'normal engineers' of 'silo-thinking,' not minding if their developments don't work with other parts of a system. Now, what's the point of completing part of a system if it doesn't work together with the other parts? So, any engineer must at least take into account how his development will work together with the other parts. When I first heard about systems engineering I was surprised. I thought that was something every engineer should do, so why would we need the extra word? Therefore I wouldn't call myself a systems engineer, but I think I do meet the definition of one.

Engineers must be able to communicate with one another. First, they must be experts in their own fields, but at the same time, they should be able to communicate with people in other disciplines, to make sure the system as a whole works as it should. This is what I have been doing all my life and now I'm teaching systems engineers how to do the right things at the right time. In short: 'Quality on Time'."

Malotaux's words suggest that this issue receives insufficient attention in basic engineering studies. This is one of the reasons for the rise in the importance of systems engineering, a discipline meant to bridge over gaps.

One of the fundamental practices in systems engineering is defining a system's requirements. According to Niels Malotaux, a systems engineer has to understand the client's *real* requirements. In many cases, clients do not know how to define their needs correctly, leaving it to a systems engineer to figure them out: "We used to design electronic products for our clients, but it was not enough to merely do what was asked of us. For example, working with a client I asked him what his problem was. He answered: 'Just do as I tell you,' to which I said: 'But what is the problem?' He said: 'What problem? I don't have a problem'. So I said: 'If you don't have a problem, we don't need to do anything'. 'Ah. Well, the problem I want to solve is … ' This is an example of the 'ask five times *Why*' technique and an example of my work method, as a systems engineer. First and foremost, I look for the client's real need."

In fact, we too have witnessed this approach during our conversation with Malotaux. When asked certain questions, he enquired as to what we really wanted to know. For instance, when we asked when he had first encountered the term 'systems engineering,' he wondered why we were asking. Having learned that our purpose was to discern the root of the term, he proceeded to tell us his version of it.

The Essence of Systems Engineering

When Niels Malotaux told his father he was planning to participate in a systems engineering symposium, his father asked him what that was. Niels told him: "Systems

Engineering is what you lectured your students on at University. You only didn't use the term Systems Engineering. Only you taught them much more than what Systems Engineering curricula teach nowadays."

Niels Malotaux: "Other than we see at other universities, my father didn't want to create specific Systems Engineers. He argued that every engineer should know about Systems Engineering, how organizations work, how R&D works. While discussing about the whole system with fellow engineers, they still have to be able to drill down to their specialized field to be able to see and communicate the consequences for their field. Optimizing the total system, which may mean suboptimizing one's own contribution to the system.

Systems and *engineering* are just words, and the problem with words is people attribute different meanings to them. In Dutch, an engineer is someone with a university degree in engineering, where I feel that in English an engineer is anyone who's using a screwdriver professionally. Systems engineering is everything that makes a system successful. The word *engineering* can be attributed to the act of organizing things in a way that makes them work. This can include anything. The name systems engineering is simply the label given to all the things systems engineers must do in order to create a successful system. It is much like when people talk about terms like *lean*: sometimes these words help us talk about our fields. This is why I accept people talking about systems engineering. If it helps, it doesn't matter what it's called, but using a common term helps to communicate better."

As he sees it, systems engineering is an attitude that can be adopted by anyone, but, in effect, focuses on technological systems, which is why a systems engineer must first be an engineer. Still, a lot of engineering as well as systems engineering knowledge can be used to organize social systems.

He adds: "I teach systems engineers that when working on a project, failure is not an option, because the knowledge of how to routinely make projects, even very complicated projects, successful and on time is known, although apparently not too much practiced. As soon as we see that things can go wrong, we don't call it fate (as in Murphy's Law). We do something about it before it goes wrong."

In the late 1990s, when Niels Malotaux first heard the term *systems engineering* ("people invent the term and make a hype"), he found that the newly defined discipline lacked sufficient attention to two important areas. The first of these was awareness of schedules, mostly concerning on-time delivery, on which we shall extrapolate later.

The second area was the human factor. It took quite a few years for systems engineering to understand the importance of the human factor, which only recently began to receive the attention it deserves.

Niels Malotaux: "People are always part of a system. A system that doesn't include people is a useless one. A systems engineer needs to understand how people behave: not how he thinks they should behave, but how they really behave – the two are not the same. In projects we have to think about the behavior of people working in the project, as well as people who will be using system developed.

They say people are unpredictable because they never do what you expect, and maybe, to a certain degree, that is true. But, for the most part, people are a lot more predictable than we realize. What we call 'unpredictable' is often simply different from what we expect. It is therefore very important for systems engineers to learn how people *really* behave. If we model the users of our systems according to what we *hope* they do, the system will not work."

He demonstrates: "Our company developed a monitoring system for controlling the freezing and defrosting of dough in bakeries. Historically, a baker had to get up at 3:00 in the morning to prepare the dough for the bread, to switch on the oven and put the carts with the dough into the oven. Now the dough has been prepared and frozen the day before, and the next morning it has to be slowly defrosted (if the dough is defrosted too quickly, it becomes wet and its quality suffers). We developed a system that performs the freezing and defrosting up to the point when the baking could begin. When the baker arrives in the morning, the oven is already switched on and all he has to do is drive the cart with the dough in the oven, and in half an hour's time, the bread is ready.

First we designed the user interface technically, with a menu-structure that allowed the baker to see the status and change settings systematically, we thought. However, when bakers saw this user interface they said: 'That's too complicated. We want to see the status at a glance from a distance and be able to operate the daily functions without going into menus.' So we redesigned the user interface accordingly. To see whether it was intuitive enough, every time we got a visitor in our office, we showed him the display and said 'Do something'. The visitor then said: 'What should I do?' We said: 'Whatever. Just try what you can do'. This way we learnt a lot of how to make the user interface as intuitive as possible.

To developers I usually say: 'We are already too much distorted. We cannot imagine what normal users find normal. So we have to find out by giving it to them and *observe*. We need the feedback from normal people.' "

An Insight on the Importance of Planning Ahead

Niels Malotaux: "Intuition is how we automatically react on situations. Intuition is fed by experience (if it's from before birth, we call it instinct). If intuition would be perfect, everything we do would be perfect. As not everything we do is perfect, apparently our intuition sometimes steers us in the wrong direction. So, instead of just doing things intuitively, we better first plan *what* to do and *how* to do it most efficiently. The plan must be doable and we have to do as planned. Once we did that we can check whether the result was according to the plan and whether the way we did it was as efficient as planned. If it wasn't, we think whether we should and how we would do it better. If it was, we think whether and how we would do it even better. Then we decide, based on this analysis, what to do differently the next time we plan and do. This way we continuously improve what we do, how we do it, as well as the processes we find to do it best. If we do this very quickly and frequently, we quickly learn how to do things better all the time. This is what Deming already taught with the Plan-Do-Check-Act cycle. It's a very powerful technique."

Meeting Deadlines

Many project leaders will testify that a project's three main objectives are to meet the specified requirements on time and on budget. However, if the time and the budget are fixed, the result is by definition not fixed, whether we like it or not. Once we realize that a lot of things we do in projects later turn out not to have been necessary, there is a lot of opportunity to keep what is really required within the time and hence within budget.

One of the main reasons for the evolvement of the systems engineering field is the fact that engineers mostly focus on achieving the first objective, namely, meeting the project's requirements and even only their own part of the requirements. Systems engineering is the bridge that leads to the two remaining objectives. Nonetheless, having emerged from the engineering world, a large part of the discipline still aims to address the technological complexity, rather than the last two objectives: meeting the budget and on-time delivery.

Niels Malotaux: "Delivery time often is one of the most important requirements of a project. How come most projects are late? Apparently all other requirements are treated as more important than the most important one. Weird.

The project manager is *responsible* for the right results at the right time. The systems engineer, however, *determines* the time and what it will cost. This makes the systems engineer (as well as all other workers in the project) at least as responsible for delivering Quality on Time."

Malotaux believes on time delivery is more important than staying on budget, because one brings about the other anyway: "One of the first things I ask when starting to coach a project is: 'What is the cost of one day of delay?' Usually people don't know. Then I ask: 'What is the cost of one day of the project?' Usually they don't know. Then I ask: 'What do you cost per day?' Most people don't know.

When discussing these questions in a project, I just saw, trough the small window of the door, the boss passing by. I opened the door and said to the boss: 'Boss, these people don't know their cost, the cost of the project nor the cost of one day of delay. How can these people make *design decisions*?' He said: 'I don't know their cost, but I'll find out!' An hour later he returned, saying: '€400 per person per day.'

The benefit of the project should be huge, otherwise we should do another project, but if we don't know the benefit I always suggest to assume that the benefit is about 10 times the cost of the project.

Using this figure we calculated the cost of one hour of delay: In this case it was 7 people × €400 × 10 per day. This is usually a lot more anybody had imagined and it is a good basis to start making much better founded design decisions.

Some engineers complain: 'We may be delaying, but this way the product will be even better!' My reply: 'What is the return you get for that improvement? Does it justify the added expense?'"

Believing that about 50% of the work done on a project later will prove not to have been necessary, there is a lot of opportunity to save a lot of time. Niels Malotaux uses several techniques to deliver the right things at the right time. For example, to improve our efficiency, he lets people plan every week: "Many people use checklists of what to do. However, hardly anyone checks whether what he thinks he has to do fits in the available time. So we determine how much time we have available in the coming week. Then we take about 1/3 of that for all the routine interrupts we get, like planning, meetings, email, telephone, drinking coffee, helping each other, etc. For some people these interrupts consume 100% of their time and hence they never have time for what they are supposed to do. Therefore we time-box it to 1/3 of the available time, leaving 2/3 for whatever we plan to do. We estimate the tasks we have to do and fill the available planed time exactly. Every hour we do not plan will evaporate. Every hour we plan to do more than we have time available, will not be done anyway, so don't waste time planning something that we already know won't be done.

On a spacecraft project with many sub-contractors, the engineers were complaining that they didn't get enough time to properly do what they had to do. In this project all engineers were trained systems engineers with a lot of experience, except for how to be on time. I asked a systems engineer why he was complaining. He said: 'I have so many documents to review before we have a meeting with the sub-contractors. We cannot do a proper reviews if we only get two weeks to do it. This causes a lot of stress'. I asked 'How many reviews?' 'Seven'. 'How much time for each?' 'I don't know'.

'Is it more than two hours and less than seven?' 'Well we have heavy and light documents'. After some more discussion we found that four heavy documents would take about 15 hours each and three light ones about 2. OK, that's $4 \times 15 + 3 \times 2$ is 66 hours. What else do you have to do? After some discussion we ended up at 99 hours of work to do, while the available time was 46 hours. At this time we knew that just starting and doing our best wouldn't be successful. As 'failure is not an option,' I suggested to organize the reviews more cleverly, and I gave them some suggestions how to do that. The result was that all documents were reviewed properly on time before the meeting with the sub-contractors. Everyone was satisfied.

Before having coached this project, they never met any deadline. It's now about a year later, and they haven't missed any deadline since. No tools, just some simple techniques."

A conclusion easily reached from Malotaux's words is that delays are not an unavoidable obstacle brought on by fate, but problems that can and should be resolved. "If in *retrospect* we see that we spent more time than necessary, the time is already lost. If in *prespect* we see that we are going to waste time, we still can refrain from wasting it. Isn't it strange that the word 'prespect,' while so important, doesn't exist in English?" First, however, the problems must be defined. The situation in the aforementioned example shows us just that. Niels uses more techniques, like bi-weekly deliveries to improve our effectiveness, and a TimeLine process to see what will happen and to do something about it if we don't like what we see.

Niels Malotaux: "I tell them to solve the problem as systems engineers, to treat the timely delivery problem as a design problem, and then use their expertise to solve it. In the example I gave, once the team became aware of how much time they had, what they believed they had to do (and whether they should do it) the problem was resolved, and the schedule was met. It is important to work one step at a time, to ask ourselves what needs to be done next and move on to planning the next phase."