Agile SYSTEMS ENGINEERING versus AGILE SYSTEMS engineering

Prof. Reinhard Haberfellner
Technical University, Graz
Austria

Prof. Olivier de Weck
Massachusetts Institute of Technology
United States of America

Fifteenth Annual International Symposium of the
International Council On Systems Engineering (INCOSE)
10 July to 15 July 2005

Abstract. This paper explores recent developments in agile systems engineering. We draw a distinction between agility in the systems engineering process versus agility in the resulting system itself. In the first case the emphasis is on carefully exploring the space of design alternatives and to delay the freeze point as long as possible as new information becomes available during product development. In the second case we are interested in systems that can respond to changed requirements after initial fielding of the system. We provide a list of known and emerging methods in both domains and explore a number of illustrative examples such as the case of the Iridium satellite constellation or recent developments in the automobile industry.

1. Introduction

The recent focus on "agility" in Systems Engineering is a manifestation of the increasing speed at which new products and systems are designed and introduced into the market place. More than speed, however, it is the existence of uncertainty in future user needs and operating conditions and the resulting ambiguity in the "true" requirements that drives new ways of developing systems. An important purpose of this paper is therefore to clarify the use and to find a proper understanding of the term "agile", which is being used in various, sometimes confusing, ways in the context of systems engineering. The title of the paper "Agile SYSTEMS ENGINEERING versus AGILE SYSTEMS engineering" highlights the fact that agility may be found both in the process of engineering systems, as well as in the resulting systems themselves.

In the first case the emphasis is on agility in SYSTEMS ENGINEERING. This way of thinking about agility focuses on flexibility and speed in the upstream process of conceiving, designing and implementing products and systems. A generic "agile" product development process can be characterized as being:

- nimble, dexterous and swift
- adaptive and response to new, sometimes unexpected, information that becomes available during product/system development
- opposite the traditional belief in engineering design that requirements and design solutions should be frozen as early as possible.

We will investigate different approaches that have been suggested to achieve an agile SYSTEMS ENGINEERING process and illustrate the relevance with a few examples.

AGILE SYSTEMS engineering, on the other hand, puts the emphasis on embedding agility in the systems themselves. This is usually done when the ability to predict the future demand or functional requirements of a system is severely compromised. An agile system is both flexible and has the ability to change from one state or operating condition to another rapidly, without large switching costs or increases in system complexity. Agile systems can be characterized as:

- flexible, reconfigurable, extensible
- scalable in the sense of capacity. An example are flexible manufacturing systems that can change the capacity (output/unit time) rapidly to adapt to actual market demand (Asl, Ulsoy 2002).
- flexible in terms of functions and performance levels. Such systems can be modified after initial deployment by addition of modules or modification of performance levels

It should be clear that these two approaches, agility in the design process and agility in the product itself, are not mutually exclusive. Agility has previously been defined by (Fricke et. al. 2000) as:

Agility = Property of a system that can be changed rapidly.
As such it represents an additional qualifier on systems that exhibit flexibility:
Flexibility = Property of a system that can be changed easily.

Alternative, but similar, definitions have been offered in the emerging field of Engineering Systems (Moses et. al., 2004).

It is important to recognize that robustness and adaptability are distinct from flexibility and agility in that human agency is not usually necessary in order to initiate or implement system change in the later two situations:

Robustness - ability to perform under a variety of circumstances; ability to deliver desired functions in spite of changes in the environment or internal variations.

Adaptability – the ability of a system to change internally and autonomously to follow changes in its environment. In our definition above a flexible system is usually modified from outside the system by an agent. An adaptable system on the other hand may undergo self-modification (e.g., a thermostat controlling the heating of a subsystem).

2. Agile SYSTEMS ENGINEERING

Agile Systems Engineering is an important consideration in situations where there are significant uncertainties during product development and manufacturing. These uncertainties can be due to ambiguities in customer requirements, the viability of new technologies or the appropriateness of one manufacturing process over another, among others. Usually there is the expectation that these uncertainties can be resolved before the product or system is shipped and operations start. Thus, mature industries that focus on process innovation rather than product innovation (Haberfellner et. al. 2002) might be most amenable to agile SYSTEMS ENGINEERING.

Example 1: (Food Processing Industry) Nestlé

The food industry is characterized by relative maturity in terms of consumer markets. Major product innovations are marketing driven, unique selling propositions, differentiation etc. are major topics. Most technical innovation is focused on food processing, packaging and distribution. A manufacturer like Nestlé will be very careful in choosing new technologies and factory configurations and will have a strong need for agility during the phase of his planning, evaluation and decision and even for the design and manufacturing phase of this equipment.

In such industries the industrial customers are in a very sensitive phase in terms of time needed to observe developments, to learn new information during product development and to make careful and deliberate investment decisions.

Another case are industries that may experience rapidly changing customer needs, but where the products/systems themselves may have rather short lifetimes and a well-defined purpose.

Example 2: (Toy Industry) Mattel

The toy industry is very competitive and dynamic with customer needs and wants changing from season to season. Many companies, such as Mattel, depend on the holiday season and the success of a few “blockbuster-products” to survive. Because it is difficult to predict what will be in high demand, toy companies produce a large number of different product ideas and many prototypes. They subject these to user “clinics”, and conduct extensive marketing surveys, using conjoint analysis and other methods. Typically, production decisions will be delayed as far as possible to understand what competitors will be offering and what the latest emerging fashion might be.

In all these cases agility is an essential part of the product design and implementation (production) process, but not necessarily a part of the product itself.

2.1 Different SE-processes

When talking of SE-processes, we are focusing on the procedural model and the process logic. We will discuss different models and compare it to a “reference model” in order to show different degrees of flexibility.

2.1.1 INCOSE-approach

Table 1 shows different descriptive levels for the SE-process

Level	Description	Examples
1	Life Cycle Phase	Concept Definition, Development, Production
2	Program Activity	Mission Analysis, Prelim. Design, Detail Design
3	SE Process	Reqts. Analysis, Architecture Definition, System Design
4	Eng. Specialty Area	Software, Human Factors, Mechanical Design

Table 1. Descriptive Levels for the Systems Engineering Process (INCOSE 2004):

2.1.2 Hall-ETH-approach

Similar to the INCOSE-approach is the Hall-ETH-model. The origin is Arthur D. Hall (1962), taken up by A. Buechel (1969) and R. Haberfellner et al. (2002), both from ETH-Zurich. We will use this model as a “Reference Model” in order to compare different models and to show, how flexibility may be installed at different points in the SE process.

4 basic ideas are characterizing the action model shown in Fig. 1 and should be regarded as components to be combined for use.

1. Proceeding from the general to the particular and not the opposite way ("**Top Down Approach**"). Esp. large systems should not be designed in detail without checking different variants of solution

principles and their feasibility (in a preliminary study) and having developed different overall concepts or a master plans (main study)

2. Observing the principle of **Developing Variants**: you should not to be satisfied with a single ("the first") variant but always look for alternatives.
3. Dividing the process of system development and system implementation into **Project Phases**. The phases define the macro-logic, the management-approach to SE. After each development phase there is a decision point, which means that you ought to have
    a. elaborated alternative ways to reach the goals on the given level of treatment
    b. stepwise decisions as commitments (within the project team, but of course with the steering committee), which way to go
4. Using the **problem solving cycle** (PLC) as a kind of working- and thinking logic (= micro-logic), no matter what kind of problem it is and in which phase it appears. The PLC is composed of 3 steps
    a. search for objectives
    b. search for solutions
    c. selection

These 4 components can be combined with each other, as graphically shown in Fig. 1.

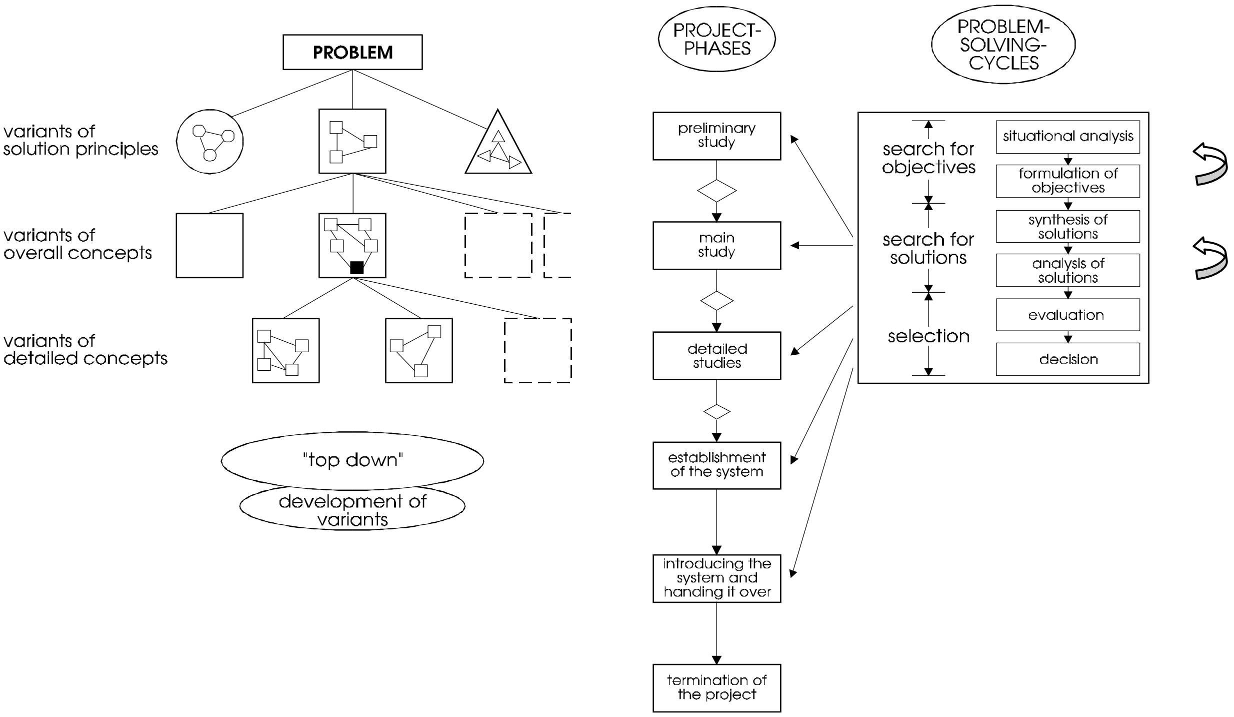

Figure 1. Action Model (Haberfellner et al., 2002)

2.1.3 Spiral Development (Barry Boehm 1986, 2000):

Technology evolution has provided software development with a flexibility rarely seen in other engineering fields. Indeed, applications such as Computer Aided Software Engineering (CASE) tools and Graphic Programming Languages have made possible not only to produce systems in shorter periods of times but also to incorporate specifications with greater facility. This flexibility has originated the emergence of iterative development processes such as Rapid Application Development (RAD) see Fig.2 (left). These processes encourage an early beginning of the codification tasks in order to provide the users with tangible results and thus incorporate their feedback through several iterative cycles under a spiral process model. This is opposed to traditional development approaches in which coding tasks are initiated only after the previous phases of definition, analysis and design have been completed, as described in the waterfall process model (see Fig 2, right). Barry Boehm is credited with the first crisp definition of the spiral process

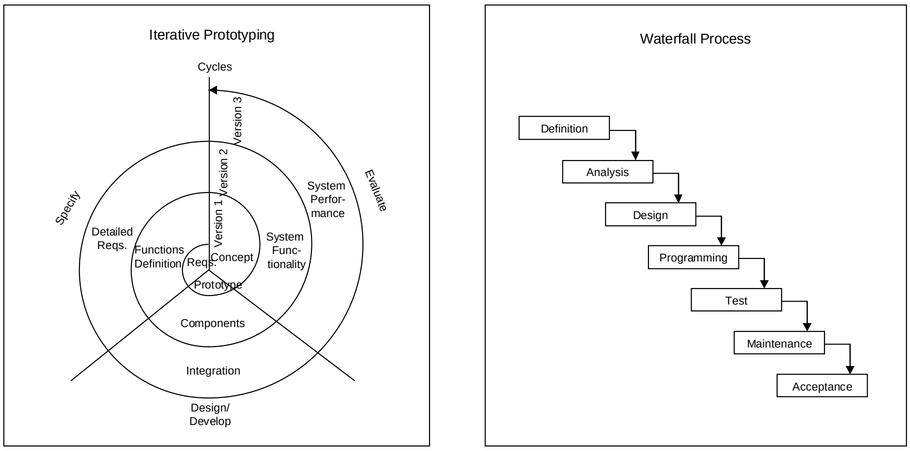

Figure 2. Iterative Prototyping (left) and Waterfall Process (right)

Comparing the Spiral Model (Fig. 2 left) to the "Reference-Model" in Fig. 1, the spiral model may be seen as a combination of component 3 (Project Phases) and component 4 (Problem Solving Cycle). The steps in the spiral indicate a kind of "phases", the sectors (i.e. specify, design, evaluate) can be mapped to the "Problem Solving Cycle".

The spiral model indicates flexibility, if you compare it to the Waterfall Model (Fig. 2, right), which represents a rough sketch of a process (only phases), without indicating any kind of iteration.

2.1.4 Agile Manifesto

The Agile Manifesto was formulated by a group of software developers in order to describe their values and approaches. The following statements describe the philosophy[1].

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Principles behind the Agile Manifesto

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

[1] http://agilemanifesto.org/principles.html

- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity - the art of maximizing the amount of work not done - is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

2.1.5 Simultaneous Engineering, Concurrent Engineering

Simultaneous Engineering is basically an overlapping phase concept (Fig. 3). This action model is very common in the automotive industry. The basic idea is to shorten the “time to market” by arranging the project phases not serial but overlapping.

An **advantage** of this approach is of course that one can shorten the duration of projects.

But there are also **risks**: one has to pass on results to next stages without having completed the ongoing stage (in terms of traditional planning). This needs another form of culture, including cooperation, frequent communication with other groups etc. And one cannot deny a higher risk for wrong investments and rework, resulting from imperfect information.

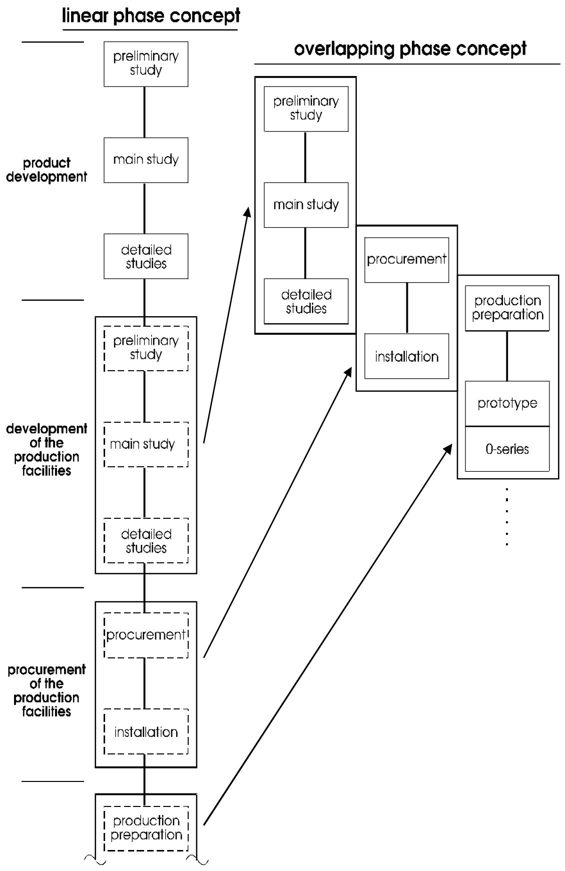

Figure 3:
Simultaneous Engineering (Concurrent Engineering)

2.2 Embedding agility into the SE-process

As mentioned before, agile SYSTEMS ENGINEERING puts emphasis on the SE-process, which should be flexible in terms of rethinking and modifying solutions and concepts and so adapting them to recent developments and findings.

Agility in this process is described in a number of ways. The approaches to agile SYSTEMS ENGINEERING we deem most relevant are outlined below. All methods attempt to include agility (flexibility at speed) [2] in the product development and implementation process in one form or another.

But let us first state some remarks that seem to be important in this context.

Agility in the SE-process is not for free:
If you want to install agility into the SE-process, of course you should be aware that this is not for free, because:

- Intentional and purposeful agility means more effort in thinking, planning, rethinking, modifying
- Why is it so? You have to find out
  - Where in your concept and in what respect do we need flexibility?
  - What assumptions may be questionable, unstable, unsettled, doubtful, incorrect or even false?
  - Which facts or influencing variables may change?
  - In which respect? What might be the impact on the solution you are working on? What negative or positive effects may arise? What should be done in such a case?
  - Which components of your system may be affected?
  - Can we sell this need for flexibility to our contracting body?
  - All SE-processes which intend to reduce the “time to market” – like all Simultaneous- or Concurrent-Engineering-concepts - have to scale down the range of variants in early phases of a project. And unfortunately exactly this openness for a range of solutions would give flexibility to a SE-process. Simultaneous-Engineering means working on a system on different levels of detail at the same time. You have to release a concept to the next team without the chance for much effort to analyze its properties concerning flexibility or similar aspects. And the more intensely you are working on detailed concepts the lesser are the possibilities to change the overall concept.
- An early design freeze may increase the speed of development but it is obviously difficult to modify or change a frozen concept.

Conclusion: There is a basic conflict between an agile and a fast running SE-process.

How and where install agility into the SE-process?
There are different options to do that, but of course no cut and dry answers. Therefore we define different **grades** of agility.

Basically one should be aware of the type of system which should be developed and – where appropriate or even necessary – should add the demand for flexibility to the catalogue of requirements or objectives. So there is a higher likelihood for bearing in mind flexibility in the design process and its control.

The following questions may help to identify the need for installing agility in the process.

- How stable are the requirements you have in mind?

[2] It may not be apparent if we define “agility” as “flexibility at speed”: It seems to be obvious, that flexibility in terms of a later modification of a concept may – in contrary - even slow down the SE-process. Therefore we have to state more precisely that we mean “at speed” only compared to a specific situation

- where we have to modify or even make major changes in a concept (and don’t avoid changes by pointing at decisions which may really be design-freeze-decision or only be interpreted as such) or
- where our SE-process that has not been designed as being flexible

- In which respect and in what direction may the requirements change? (concerning function, environment stable or changing, purpose of use, timing and pacing, etc.)
- How expensive is a broader interpretation of requirements? What would be the additional costs for this generosity? etc.

We now want to sketch different grades of agility in the SE-process:

Grade 1: Apply agility to existing SE-process-model
Execute the "situational analysis" and the "analysis of solutions" in the SE-problem-solving-cycle mindfully:

- The situational analysis particularly with regard to iterative loops back from objectives / requirements (see Fig. 1, Problem Solving Cycles)
- The analysis of solutions particularly in terms of claimed flexibility (or not claimed but essential)

Grade 2: Piecemeal Engineering (Karl Popper, 1944):

In "The Poverty of Historicism" the philosopher Popper mounted a major attack on large scale "all at once" (or revolutionary, utopian) social engineering and proposed "piecemeal social engineering". Popper prescribed piecemeal reform because we can better monitor and eliminate our mistakes in the small than in the large; he rejects revolutionary reform because "we can neither easily monitor the society-wide ramifications nor reverse our leaps".

Let us translate this philosophy into the idea of staged product development:

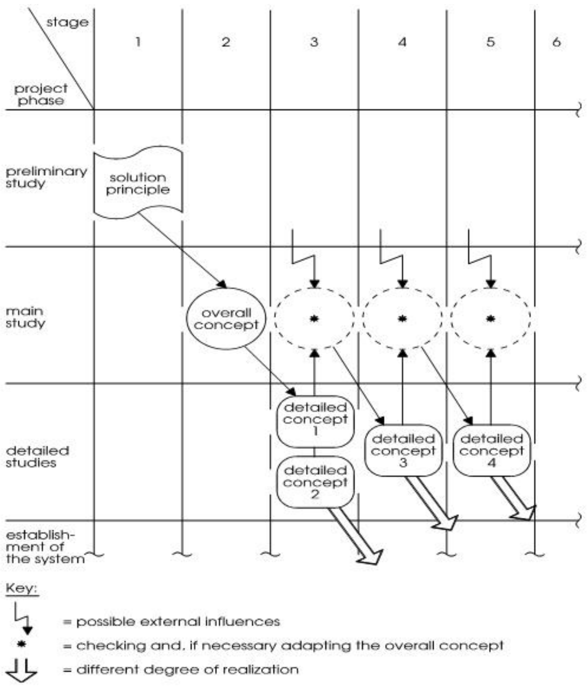

Figure 4.
The dynamics of the overall concept
(Haberfellner et al. 2002)

As indicated in Fig. 4, possible reasons for later reconsiderations may be located inside or outside the system:

- Intrasystem reasons are about difficulties (or unexpected options) which are arising in the process of detailed design (arrowheads pointing upwards from "detail concepts")
- Extrasystem reasons may be unexpected financial, technological, political, personal or whatever changes in the environment which have an impact on the overall concept (lightning bolts into "overall concept")

Regular checks of defined and monitored internal or external influences are part of this approach.

The area of application for this piecemeal engineering approach may be characterized as follows

- Changes to existing systems
- A system consisting of modules which show a value by themselves. One may introduce and use the modules and get advantages out of this
- For many situations it makes sense, not to change every module in a new system. Emphasizes the use of proven modules.
- When designing new modules, test and use them first in already existing systems etc.

Of course, the possibilities for changes of the overall concept decrease as the number of already worked out and maybe already realized detailed concepts increases. At a particular time it may not anymore be possible to accept any wishes or needs for modifications of the concept. Then you have only the alternatives: go on with the selected overall concept – or terminate the project.

Grade 3: Set-Based Design (Shigley 1989)

Set based design is based on the philosophy that one should consider working on a set of design solutions in parallel until one is forced to reduce the set of options to a smaller set. This reduction may occur because new information such as feasibility constraints become available, because long-lead items have to be made or ordered or because of a lack of time or resources that would otherwise allow carrying a larger set. In set-based-design an overall concept is not necessarily chosen ahead of time, but usually a sequence of decision steps is defined and followed. The overall design emerges over time as the design space is narrowed. It has been shown that Toyota relies heavily on the idea of set-based-design. Even though many of the Japanese manufacturer's design teams are decentralized, they have been successful. Toyota makes extensive use of clay models and prototypes before selecting a final vehicle for production and implements set-based design extensively.

The advantages of this approach rest on

- The awareness of the designers and managers in an early stage of not having found and are not fixed to the ultimate solution.
- The attempt to explore the total solution space, which has similar effects and facilitates minor or major changes, which may be necessary later on.
- Positive effects of internal competition between design teams to prevail.

So far we assume a decision for a particular overall concept in a rather early stage. This overall concept may be adapted or modified if necessary and still possible (Fig. 4).

Now we drop this assumption and keep several options open for a time span as long as possible. A concept that was applied in practice is shown in Fig. 5.

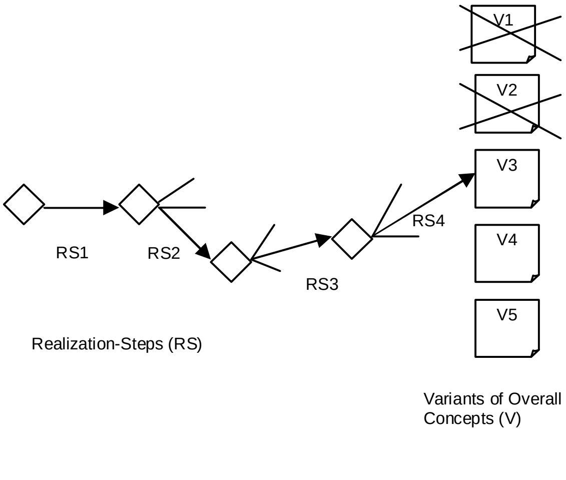

Figure 5.
Decisions for Realization-Steps, not for Variants of the Overall Concepts

We manage this by designing several overall concepts, each of which makes sense for a

special set of assumptions which are likely to occur (= V1 – V5).
Then we plan – separately for each of the overall concepts - the activities (steps of realization), which are necessary in order to realize them. We pick out activities which have to be done for a large number of options (common tasks, preferably for every variant). We decide to realize and execute them. As time goes on, some of the variants become irrelevant because they don't match any more with the realization-steps. This is indicated by the crossed out (eliminated) variants V1 and V2 in Fig. 5. So you have to be aware that there are no implicit decisions for or against variants.

The ultimate reason for this approach is to play for time for the decision, which variant should be chosen. This approach is driven by the uncertainty of the situation which has to be communicated to everybody in the team.

3. Manifestations of Inflexibility in Systems

The previous section focused on embedding agility in the Systems Engineering process. The underlying assumption was that uncertainty would be resolved before the product is released or the system is fielded. This is often true for commodities (example 1), short lived products (example 2) or systems operating in stable, unchanging environments. For many large scale, long-lived systems uncertainty cannot be resolved ahead of time. In these situations one must examine the need to embed flexibility in the systems directly. The history of large infrastructures and systems is littered with technically well executed, but strategically inflexible systems (de Neufville 2004). Some high capital investment systems fail, even though they meet technical performance targets and – in some cases – are designed to cost and schedule (Browning 2002). This is often the case because forecasts of future usage or demand are easily confounded. Even when technical requirements are initially met, there is no long-term guarantee of system success (success = long term sustainable survival based on a positive balance of value delivered to society).

Example 3 (Space System): Low Earth Orbit Satellite Constellations

Iridium and Globalstar pioneered mobile space-based telephony in the late 1990s (Lutz 2000, Fossa 1998). Despite extraordinary technical breakthroughs, these systems were commercial failures, respectively resulting in losses of roughly $5 and $3.5 billion (Iridium LLC 1999). The proximate cause of these failures is that prior forecasts and expectations were confounded, in particular that the market for wireless telephony went to ground-based competitors that arose between the conception in 1990 and the launch of the communication satellites in 1998 (Inkpen 2000, Christensen 2000, de Weck 2004).

Example 4 (Automotive): Automotive Manufacturing Plant

Inflexibility can also lead to missed opportunities. A case in point is Daimler Chrysler's PT Cruiser. Based on the Neon compact car, the PT Cruiser was a big hit in the 2000 and 2001 model years; demand quickly exceeded the capacity of the Mexican plant where it was built. But Daimler Chrysler was unable to shift overflow production to its Neon plant in Belvidere, Ill., which had capacity to spare. Why? Planners had neglected to make the Belvidere paint shop tall enough (or reconfigurable) for the PT Cruiser, which is a few inches higher than the Neon. The oversight meant the company lost approximately $ 480 million in forgone pretax profits according to estimates by Prudential (Brown 2004).

Traditional practice is to generate a most likely "forecast" for demand. Systems are then optimized for this expected future. However, forecasts are almost always wrong by either overestimating demand (example 3), or by underestimating it (example 4). Agile systems engineering acknowledges explicitly that future demand patterns are uncertain and that embedding flexibility in systems might, in some cases, be a more effective strategy than designing a rigid system for the expected or “worst case”. Unknown customer preferences and demand can be modeled using Geometric Brownian Motion (GBM), see Fig. 6, lattice trees or discrete scenarios.

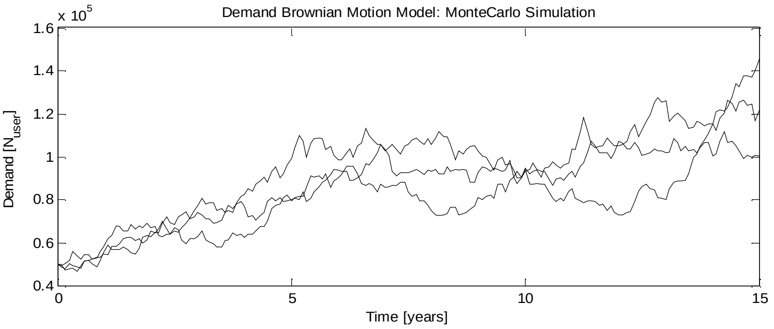

Figure 6. GBM model of uncertain demand, ΔT = 1 month, D_o = 50,000, μ = 8% p.a., σ = 40% p.a. – 3 scenarios are shown (de Weck et al. 2004)

Macro Systems	Dominant Uncertainties	Inflexibility Manifestations	References
Space Systems			
Communications Satellites	# of subscribers, activity level, service type (data, voice, …)	System capacity is inadequate, cannot switch type of service	(Christensen 2000, Chang 2004)
Exploration Systems	science targets, technology readiness, govt. funding	Cannot land humans/robots to locations of interest, can’t substitute technologies, inability to scale down to a smaller budget	
Vehicle Systems			
Commercial Aircraft	travel demand, demographic shifts, regulations, competition	Aircraft cabin size, range, emissions are locked in	
Cars & Trucks (Automob.)	fuel prices, shifting customer preferences, competition, reg. numerous intangible criteria in customer decisions such as styling preferences	CAFE[3]standards violated, can’t derive new vehicle variants from existing platforms Inadequate models, less attractive, inferior to competitors models	(Carney 2004)
Infrastructures			
Radio Telescope Arrays	new discoveries, weather, electromagnetic interference	Telescope operates in polluted frequency band, can’t switch to different type of science	(Cohanim 2004)
Oil & Gas Exploitation Sys .	fuel and gas prices, reserves contained in reservoirs, interest rates	cannot take advantage of a previously unknown oil field	(Kulatilaka 1993)
Commercial Buildings	Utilization patterns, land prices, rental rates $/sqft/month	High vacancy rates of commercial real estate	(Kalligeros 2004)

Table 2. Uncertainties and ‘failure modes’ for different classes of large systems

[3] Corporate Average Fuel Economy. URL: http://www.nhtsa.dot.gov/cars/rules/cafe/

Table 2 shows the major types of (external) uncertainties that can impact large systems after their initial deployment, along with typical symptoms resulting from insufficient flexibility or agility to adapt. Failures here are not primarily of a technical-operational nature, but rather failures in delivering the desired functionality and generating enough value (= revenue for commercial systems) for long-term sustainability and evolution. Inflexible systems cannot adapt to the future either to capitalize on new opportunities or to avoid market risks as by permitting gradual deployment of the system, which permits truncating costs if the market fails to materialize. Agile Systems Engineering points the way to a number of potential remedies.

4. Engineering AGILE SYSTEMS

AGILE SYSTEMS are those that have flexibility embedded in them such that they can be changed in response to unfolding user demand after they are initially fielded. This requires essentially three elements: (a) the necessary flexible elements inside the system that allow it to be changed easily and quickly, (b) a set of sensors to monitor external attributes to alert decision makers when changes might be warranted and (c) a decision mechanism by which the benefits and costs of system adaptation are compared and system state changes are triggered. The main approaches to AGILE Systems can be found along a number of thrusts:

- **Capacity Adaptation**: The total size or throughput capability of a system often requires that an estimate be made of the amount of future demand for a given type of product or service. Capacity adaptation in the initial build-out of the system is typically referred to as "**Staged Deployment**". We have previously shown how staged deployment could have been employed in the implementation of large satellite constellations (de Weck et. al 2004) to reduce investment risks due to unknown demand. Recent research in flexible manufacturing systems also focuses on capacity adaptation (Asl, Ulsoy 2002)

Optimization of Staged Deployment of Satellite Constellations (de Weck 2004). The contribution of this work is that the evolution path of a satellite constellation was optimized, using a staged deployment approach (Fig.7 left).

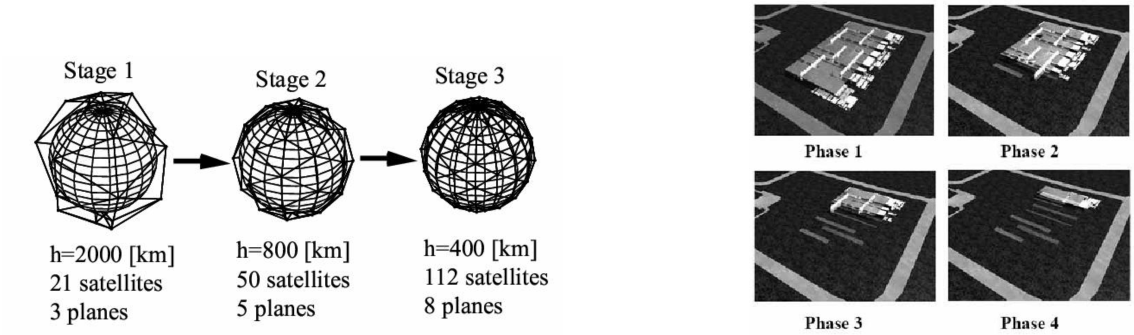

Figure 7: (left) Optimal staging of satellite constellations for growth (de Weck 2004a); **(right) Optimal modular facility design for divestment** (Kalligeros 2004), figure courtesy: J. Fernandez

This goes beyond optimizing the system for a 'cost per function' metric. It was demonstrated that economic risk reduction could be achieved by phasing the system into existence in discrete steps, rather than building it all-at-once using a real options approach. Traditionally, satellite constellation design had been viewed as a static coverage optimization problem. This work showed that staged deployment of Iridium and Globalstar could have resulted in lifecycle cost savings on the order of 20-45%, averaged over future scenarios.

Flexible Commercial Bulding/System Design: Managers of large ("macro") systems want to have the ability to shut down parts of systems, re-deploy assets or expand the system as needed by unexpected growth. (Ahmed 2003a, 2003b) have recently analyzed the problem of capacity expansion under uncertainty. An example of recent work on modular building optimization and valuation (Kalligeros 2004) is shown in Fig. 7 (right). Whitney has clarified, though a case study on strategic design at Nippondenso (Whitney 1993) that good design for flexibility allows for only a finite set of carefully chosen reconfiguration options.

AGILITY in systems is especially needed, when the systems are:

- Expensive, involving significant upfront investment cost
- Long-lived, e.g. >10 years. User requirements may change significantly during the lifecycle.
- Significant switching costs exist, i.e. the expense might be too large for building an entirely new system each time the requirements change.

If these three factors are present one may think about designing flexibility into systems as both an insurance against downside risks, but also as a way to take advantage of unexpected upside opportunities. Non-dimensional relationships between these three factors across a range of domains remain to be analyzed in future.

Example 5 (Automotive): Magna Steyr Fahrzeugtechnik, Graz MSF

The relevance of agility in systems is evidenced by the necessity of automotive manufacturers to run their factories near capacity (a break even point often mentioned is a load at or above 85% of capacity). BMW und PSA (Peugeot/Citroen) are exemplars in Europe for this flexible strategy with Toyota, once again, leading the way among Asian manufacturers.

The MSF plant in Graz/Austria, has been designed for flexibility. Quite a number of different vehicles may be assembled in the same production line in any order. A special level of sophistication is that production data may be transmitted from the USA as well as from Germany.

In the **assembly line** until now have already been simultaneously assembled:
- Chrysler Voyager + PT Cruiser
- Chrysler Voyager + Jeep Cherokee + Mercedes M
- Chrysler Voyager + Jeep Cherokee + Chrysler 300

Bodyshell work is always done separately because of different work cycles.

Body painting is possible in one paint shop for: Chrysler Voyager, Jeep Cherokee, Chrysler 300, Puch G, Saab Cabrio and Mercedes E-type.

American car manufacturers are also restructuring to allow production of a large variety of automobiles from a relatively small set of platforms (NZZ 2004, Carney 2004). The most successful manufacturers are able to combine several platforms into one production line and have the ability to profitably manufacture products that are at different points in their lifecycle.

Of course flexibility may be expensive (higher investments) but unemployed equipment may be even more expensive. And it is more likely to be worth designing for flexibility, if one cannot fully load one line with one platform.

It appears that much of the work on AGILE SYSTEMS is converging in a field described as **Real Options Theory**. The use of options has recently been extended to physical facilities under the rubric of “real options” (Myers 1977). Real options analysis applies the financial principles of options (Black and Scholes 1973) to “real” or physical systems such as power plants, copper mines and so forth (de Neufville 2000). For the most part, the use of “real options” is almost identical with that of financial options. The project is treated as a “black box” and the conceptual and analytic effort focuses on trying to work the available data into forms suitable for the tools of financial analysis. A case in point is the Black-Scholes (1973) formula for pricing European call options:

$V = N(d_1)A - N(d_2)Xe^{-rT}$	*A* = current value of the underlying asset
$d_1 = \ln\frac{\sigma}{X} + \left(r + \frac{\sigma^2}{2}\right)T/\sigma\sqrt{T}$	*X* = exercise price
	T = Time of expiration
$d_2 = d_1 - \sigma\sqrt{T}$	*R* = Risk free interest rate
V = value of call option	σ= Volatility of the underlying asset
	N(d) = Cumulative value of normal distr. at d

Such an option gives the right, but not the obligation to buy a stock, which is currently traded at price *A* on a particular day *T* at strike price *X*. It is not obvious what the equivalent values should be for physical systems, which are not traded on the equity markets. In this case the real options are “on” technical systems (Fig. 8). Engineers designing AGILE SYSTEMS however, are primarily interested in real options “in” systems (de Neufville 2004).

This distinction affects the most appropriate valuation techniques as well as the degree to which the technical aspects of the underlying the system must be understood.

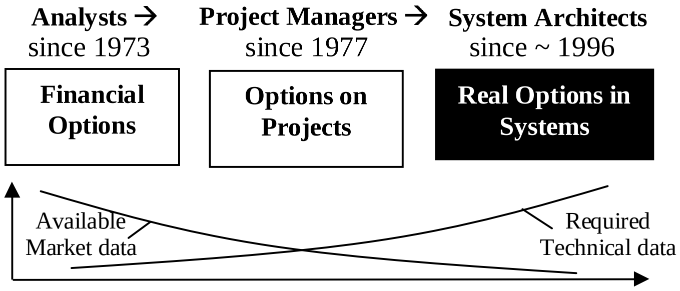

**Figure 8.
Three types of options.**

Real Options give weight to the upside opportunities associated with uncertainty, in addition to the traditional concern with downside losses and risks. A real option gives the holder the right, but not the obligation, to adjust the design of a system at a later date in a significant way that enables the system managers to either avoid downside consequences or exploit upside opportunities.

An alternative to valuing flexibility in systems to the financial options approach of Black/Scholes is to compare the relative Value-at-Risk curves of a rigid, non-flexible system relative to a flexible system across a set of stochastic future scenarios, see Fig. 9.

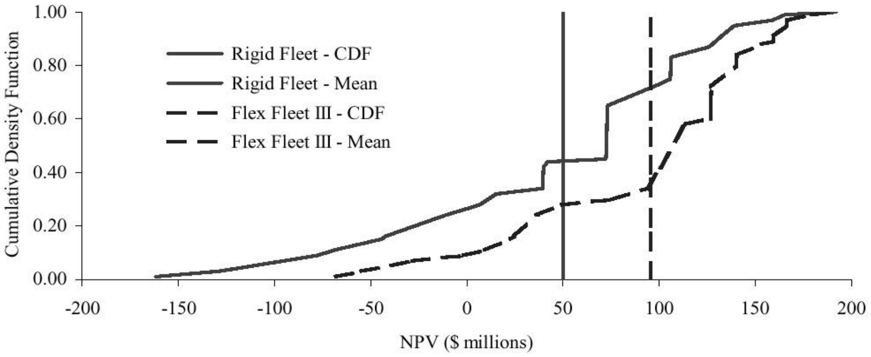

Figure 9: Flexibility valuation via relative comparison of E [NPV], (Hassan et al. 2005)

5. Concluding Remarks

A major focus of this paper is to distinguish between agile SYSTEMS ENGINEERING and the engineering of AGILE SYSTEMS. These two concepts are often confused in the practice of engineering design and systems engineering. In the first case flexibility is embedded in the design process itself, in the second case it is the system that exhibits the ability to be changed easily and quickly.

We argue that AGILE SYSTEMS are beneficial in cases where the systems have a long lifecycle, and significant switching costs exist coupled with substantial uncertainty in the environment (customer functional requirements, demand evolution, …). In some cases uncertainty may be resolved before product release in which case focusing on agile SYSTEMS ENGINEERING alone may be sufficient.

It must also be acknowledged that system flexibility and agility often carry a price in terms of increased system complexity, cost, mass/weight or the introduction of unwanted interfaces and other technical penalties. One must carefully analyze whether such penalties are worth accepting upfront, relative to the value that agility gives the system users, operator and owners during later parts of the lifecycle. The field of Agile Systems Engineering is still evolving and both academic research and practice are in flux. As such the methods and conclusions presented here should be regarded as “work-in-progress” rather than a converged theory. There is no doubt, however, that systems engineers can no longer be content with drawing narrow boundaries around their technical systems and then ignore the dynamics and uncertainties associated with elements and systems outside those boundaries.

6. References

Ahmed, S., King, A.J., and Parija, G.: *A Multi-stage Stochastic Integer Programming Approach for Capacity Expansion under Uncertainty*, Journal of Global Optimization, 26, pp. 3 – 24, 2003a

Ahmed, S. and Sahinidis, N. V.: *An approximation scheme for stochastic integer programs arising in capacity expansion*. Operations Research, 51(3):461–471, 2003b

Asl, F.M and Ulsoy, A.G.: *Capacity Management in Reconfigurable Manufacturing Systems with Stochastic Market Demand,* IMECE2002-MED-32872, 2002 ASME International Mechanical Engineering Congress and Exposition, New Orleans, Louisiana, November 17-22, 2002

Black, Fischer and Myron S. Scholes: *The pricing of options and corporate liabilities,* Journal of Political Economy, Vol. 81, 637-654, 1973

Boehm, Barry, edited by Wilfred J. Hansen: *Spiral Development: Experience, Principles, and Refinements,* Software Engineering Institute, Carnegie Mellon University, Special Report CMU/SEI-00-SR-08, ESC-SR-00-08, June, 2000

Brown, Stuart F.: *Toyota's Global Body Shop,* Fortune Magazine, 2/9/2004, Vol. 149, Issue 3, pp.120-123, 2004

Browning, Tyson R., Deyst John J., Whitney Daniel E., Eppinger Steven D.: *Adding Value in Product Development by Creating Information and Reducing Risk,* IEEE Transactions on Engineering Management, Vol. 49, No.4, pp. 443-458, 2002

Buechel, Alfred: *Systems Engineering*, industrielle organisation, 38(1969) 9, pp. 373-385

Carney D.: *Platform Flexibility*, Automotive Engineering International, February, pp. 147-149, 2004

Chang, D. and de Weck, O.: *Basic capacity calculation methods and benchmarking for MF-TDMA and MF-CDMA communication satellites*, Int. Journal of Sat. Communications, 2004, (in press)

Christensen, C.B., and Beard. S.: *Iridium: Failure & Successes*. International Astronautical Federation, Proceedings of the 51st Internat. Astronautical Congress, Rio de Janeiro, Brazil, Oct. 2-6, 2000

Cohanim B. E., Hewitt J. N., and de Weck O. L.: *The Design of Radio Telescope Array Configurations using Multiobjective Optimization: Imaging Performance versus Cable Length.* The Astrophysical Journal, Supplement Series, Vol. 154: 705-719, October 2004

Crawley, E., Hoffman, J., de Weck, O. et al.: *Paradigm Shift in Design for NASA's New Exploration Initiative*, Final Report, 16.89/ESD.352 Space Systems Architecture Class, Massachusetts Institute of Technology, 2004, URL: http://web.mit.edu/spacearchitects/1689report.htm

de Neufville R.: *Dynamic Strategic Planning for Technology Policy* - International Journal of Technology Management, Vol.19, No.3/4/5, pp.225-245, 2000

de Neufville, R. et al: *Uncertainty Management for Engineering Systems Planning and Design*, MIT International Engineering Systems Symposium, Monograph, MIT, Cambridge, MA. March 2004. http://esd.mit.edu/symposium/pdfs/monograph/uncertainty.pdf

de Weck, O.L., de Neufville R. and Chaize M.: *Staged Deployment of Communications Satellite Constellations in Low Earth Orbit,* Journal of Aerospace Computing, Information, and Communication, Vol. 1, No.3, pp. 119-136, 2004

Fossa, C.E., e.a.: *An overview of the Iridium low earth orbit satellite system*, Proceedings of IEEE 1998 National Aerospace and Electronics Conference; (A99-17228 03-01): pp. 152-159., 1998

Fricke, Ernst; Gebhard, Bernd; Negele, Herbert and Igenbergs Eduard: *Coping with changes: Causes, findings, and strategies*, Systems Engineering, Vol. 3, Issue 4, Date: 2000, pp.: 169-179

Haberfellner, R. et al.: *Systems Engineering*, 11th ed, Industrielle Organisation, Zurich, Switzerland, 2002

Hall, Arthur D.: *A Methodology for Systems Engineering*, Princeton N.J., 1962

Hassan R., de Weck O., de Neufville R.: *Value at Risk Analysis for Real Options in Complex Engineered Systems*, IEEE Systems, Man and Cybernetics Conference, 2005

INCOSE, *SE Handbook,* Version 2a, 2004

Inkpen, A., Martin, M. and Fas-Pacheco I.: *The Rise and Fall of Iridium*. Thunderbird, the American Graduate School of International Management, Case Study: http://www.t-bird.edu/faculty_research/case_series/cases_2000/iridium.htm , 2000

Kalligeros K.C. and de Weck O.L.: *Flexible Design of Commercial Systems under Market Uncertainty: Framework and Application,* 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, AIAA-2004-4646, Albany, New York, 2004

Kulatilaka, N.: *The value of flexibility: the case of a dual-fuel industrial steam boiler*. Financial Management, 22(3), 1993

Lutz E., Werner M. and Jahn A.: *Satellite Systems for Personal and Broadband Communications*. Springer-Verlag, Berlin, Germany, 2000

Mehrabi, M.G, Ulsoy, A.G, Koren, Y, and Heytler, P.: *Trends and Perspectives in Flexible and Reconfigurable Manufacturing Systems*. Journal of Intelligent Manufacturing, 13, 135-146, 2002

Moses, J., et al.: *Framing Paper for Engineering Systems,* Engineering Systems Symposium, March 2004, URL: http://esd.mit.edu (Monographs)

Myers, S.: *Determinants of Corporate Borrowing*, Journal of Fin. Economics, 5, Nov., pp. 147-175, 1977

N.N. *Zur Flexibilität verdammte Autokonzerne,* Neue Zuercher Zeitung NZZ, 16./17. Oktober 2004

Popper, Karl: *The Poverty of Historicism*, Routledge Classics, 2002. Routledge & Kegan Paul, 1957. Originally published in Economica, 1944/5.

Ribbens, J.: *Simultaneous Engineering for New Product Development*: Mfg Applications. Wiley, 2000

Schulz, Armin P.; Clausing, Don P.; Fricke, Ernst and Negele, Herbert; *Development and integration of winning technologies as key to competitive advantage*, Systems Engineering, Vol. **3,** Issue 4, Date: 2000, pp. 180-211

Shigley, J. E., and Mischke, Charles R.: *Mechanical Engineering Design,* 5th ed, McGraw-Hill, 1989

Whitney D., Nippondenso Co. Ltd: *A Case Study of Strategic Product Design*, Research in Engineering Design, 5(1), pp. 1-20., 1993

8. Biographies

Reinhard Haberfellner is currently a Professor of industrial management at the Technical University of Graz (TUG) in Austria. From 1959-65 he studied Mechanical Engineering and Economics at the Technical Universities of Vienna and Graz/Austria, where he earned his Diplom Ingenieur degree (MS equivalent) . He obtained his doctorate in 1973 at the Swiss Federal Institute of Technology (ETH) in Zurich, Switzerland. From 1965-1979 he was a senior research scientist at the Institute of Industrial Management (BWI) at ETH Zurich. In his research and consulting activities he focused on design of organizations, strategic planning and systems engineering methodology. Since 1979 he served as full professor of industrial management and organization at TU Graz and also served as Dean of the School of Engineering. The book on Systems Engineering he has co-authored is currently in its 11.th edition and is the most successful German publication in SE. From 1995-2000 he served as CEO of Styria, a large publications company in Graz. He returned to the faculty of TUG in early 2000.

Olivier L. de Weck is currently an assistant professor with a dual appointment between the Department of Aeronautics and Astronautics and the Engineering Systems Division (ESD) at MIT. His research interests are in Integrated Modeling and Simulation, Multidisciplinary Design Optimization and System Architecture. In 2001 he obtained a Ph.D. in Aerospace Systems from MIT. From 1987 to 1993 he attended the Swiss Federal Institute of Technology (ETH Zurich) in Switzerland, where he earned a Diplom Ingenieur degree (MS equivalent) in industrial engineering. From 1993 to 1997 he served as liaison engineer and later as engineering program manager for the Swiss F/A-18 program at McDonnell Douglas (now Boeing) in St. Louis, MO. Since 2002 he is a member of the AIAA Multidisciplinary Design Optimization (MDO) Technical Committee (TC). He won two best paper awards at the 2004 INCOSE conference.