Chapter 19

Mechatronics as Science and Engineering – or Both

19.1. Introduction

The Internet upheaval is established in the 21st Century. This has affected a number of new areas of science and engineering. Basic engineering ancestry has begun to grow into a second-level approach. Most of the imaginary "things" would now become true with effective and heavy mathematical tools. This is because of new technological advancements in human society and their innovative creations and thinking for the next century. Today's scientists and engineers are sure to join our club to celebrate the outcome.

We begin here with the theories of mechatronics as science and, at the same time, as engineering studies and topics. We flavor some of our experiments with the real-world of businesses. We explain how we did it and why. In this respect we would invite you to go through our initiated cases to the main body of this chapter. The following chapters introduce experiments and case studies which are implemented in the Finnish industrial environment. We believe that it is practical and a learning experience for the readers of this handbook today and for the future.

Chapter written by Balan PILLAI and Vesa SALMINEN.

Let us first become acquainted and established with mechatronic systems in more detail. The encyclopedia of Wikipedia[1] has defined that a mechatronics engineer, is able to blend the principles of mechanics, electronics and computing; to create something simpler, while adding value and a reliable system. It is also centered on mechanics, electronics, computing, control engineering, molecular engineering (from nanochemistry and biology) and optical engineering. Through a group effort, the mechatronic modules perform the production goals and inherit flexible and agile manufacturing properties in the production scheme. Figure 19.1[2] shows an enlarged version of the interconnectivity to grasp the veracity of mechatronics in general.

Figure 19.1. *Veracity of mechatronics*

In other words, mechatronics engineering and science is mastered in almost all the sectors that we know so far [HAB 07]. Therefore, it is now time for us to enter into, with a bit more detail but with limitations, the field of mechatronic products[3], its development schemes and implementation methodology scenarios. The engineering of cybernetics welds with the question of control engineering of mechatronic systems. Modern production machines consist of mechatronic modules that are integrated according to the

1 www.wikipedia.org.

2 Adapted from the Aerial Euler diagram from RPI's Website, the figure describes the various fields that combine the mechatronics engineering.

3 The portmanteau "mechatronics" was coined by Tetsuro Mori, a senior engineer of the Japanese company Yaskawa in 1969. An industrial robot or forest harvesting machine is a typical example of a mechatronics system; it includes aspects of electronics, mechanics and computing to do its day-to-day jobs.

control engineering architecture. The most common architectures involve hierarchy, polyarchy, heterarchy and hybrid. The methods for achieving a technical consequence are described by control algorithms, which might or might not use formal methods in their design. Hybrid systems, important to mechatronics, include production systems, synergy drives, planetary exploration rovers, automotive subsystems, such as anti-lock braking systems and spin assist, and everyday equipment, such as autofocus cameras, video, hard disks and CD players.

When we enthusiastically focus on mechatronic products and their service aspects, it is necessary and practicable to know how those are in the first place planned, designed and anchored into the manufacturing environment that delivers the end product to the final customers. On the one hand, the customer could be a middle or intermediate one, and at the same time, it could be a participator in other large systems, such as a paper machine parts or systems, or turbo engines, or, say, passenger ships. Product design masked with precise or specific technical and ergonomic disciplines that are systematically followed by an A to Z[4] are set at the "idea-phase" on a blackboard for physical product. The product creation generates from "zero-level; that is the idea", and goes through an immense number of technical and commercial phases [HAB 07, HAB 06], before it could be grounded as a kind of physical product at the process end or this end leads to a second or third level in large systems or even simple and stand-alone systems. In the process, "*the idea*" turns from a "skeleton" to "flesh". At this phase, there are a number of unanswered or even unquestioned elements that are directly attached to or concerned with the product and its manufacturing environments. This may sometimes lead to, in some sense, the engineering one, constructed on "if-and-then" notions. Occasionally, but not always, the situation required a superior technical touch in setups, and its configuration routes perhaps do not exist or are clearly specified, while the future possibility in modification tracks is missing or demands the usability trajectory failure, maintainability, functionality and operability. Nevertheless, when we take a shortcut directly to design, the design procedures are very pragmatic, which involves multi-scientific and engineering knowledge, a lot of drafting, drifting, modeling, versioning, prototyping including interoperability scheme checking, tests on functionality, performance and so on [HAB 08]. Therefore, it is absolutely necessary to know the state of the art so as to know how the design and planning process are met in systematically going forward until the manufacturing of a typical mechatronic product is in hand.

---

4 An A to Z is namely a set of many actions that are organized in a series to be implemented at sequences.

19.2. Theories and methods of design, planning and manufacturing

We will now figure out the theories and methods including the design patterns, solutions and structures of mechatronic products, a paradigm in its manufacturing point of view and the product release strategies. Figure 19.2 shows the business operations, where manufacturing and its connected activities are clustered into each other.

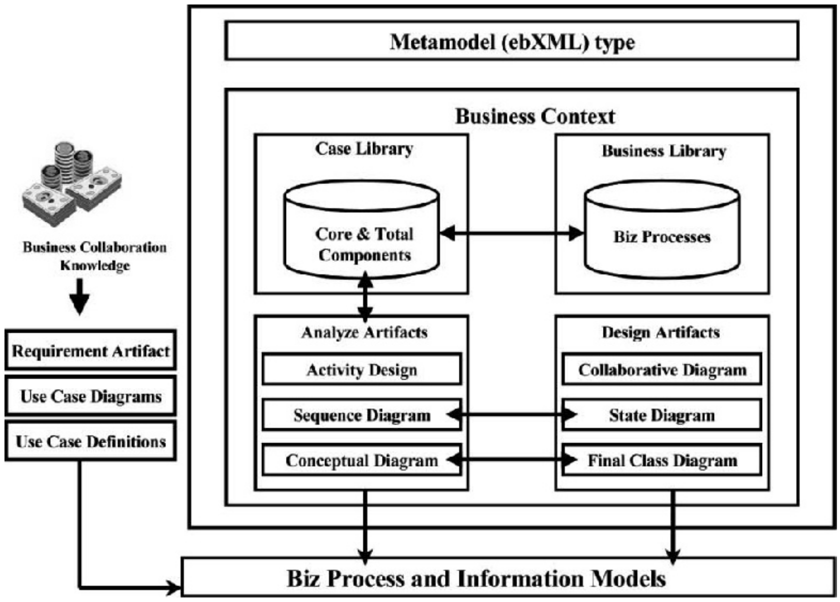


Figure 19.2. *Business operations and modeling structure*

Figure 19.2 is a typical modeling diagram that shows the entire business process and how its information flows can be modeled on an ebXML[5] platform. The metamodeling with electronic business concept is not a new one and is implemented in many industrial environments. This system approach would facilitate, perhaps, "the missing-links" between product system architecture and knowledge depositories. In addition, the modeling concept would assist the peers, their spare part-vision targets, or cover the design trajectory envelops and depositories of the data environment.

5 ebXML = electronic business extendable mark-up language.

To be successful in business, a company must increase the frequency and speed at which it comes out with new marketable and producible models and constructions. The driving forces behind product development are the demands of customers that are based on their present requirements, and may be the new products supplied by other competitors. There are also other factors that are built in such as involving new investments in manufacturing technology by the main supplier company and also a supplier's capability in improving their overall productivity.

Product design has such a huge effect on the operation and economy of the entire company that more versatile tools must be used for measuring its efficiency. At least the following key factors should be taken into consideration:

– accuracy and relevancy (external and internal);

– meeting the needs of the market;

– keeping the production costs down;

– speed;

– efficiency in the use of research and development (R&D) resources.

It is important to know the interfaces of these factors while modeling the business processes of the company (see Figure 19.3).

Figure 19.3. *Business processes and vision*

When we begin to speed up the product development process, we must bear in mind that we cannot afford to lower the quality of the work and the reliability of the end product. On the contrary, they should be improved even when we work at a higher speed. Because the majority of the costs related to an engineering product are already determined at the design and planning stage, it is absolutely essential that the actual production costs are kept on a competitive level even when we are developing new products faster. This is only possible if the product designers and the people responsible for production system development work in close cooperation [AND 83]. Generally, the marketing experts, with a deep knowledge of the market requirements, must participate in the development work right from the beginning. These requirements already make intensive teamwork necessary in the future product design work as well. Because the logistics experts and subcontractors must be included at a relatively early stage, the project manager is faced with a really hard task, when the aim is higher development speeds.

The interaction between modules to be connected with each other has an essential impact on the development speed. The module interactions are basically: on, functional, physical, chemical, geometrical and related to the information content. The greatest loss of product development time is caused by the physical, geometrical and informational aspects. At its various generation and creation phases, the product is treated in different ways, advancing from product orientation to the process orientation. At the different phases, the organization will also be different.

Furthermore, we proceed ahead in handling the product development strategies, such as speed and the ergonomic aspects, where we might encounter, sometimes purposely or otherwise, several imaginary or visionary steps for the sake of arguments at executing the product strategy. Normal procedure predicts at this phase of the process or sees more than one complication, or it tends to be complicated by passing a step further into an alternative constructive version that enhances further complexity. Let us see how we could theoretically avoid or survive within such situations.

19.3. Complexity versus complicatedness

Tang and Salminen [TAN 01] illustrated how the complexity versus complicatedness can be handled. They assume in their work that global, dynamic and a competitive business environment has increased the complexity in product, service, operational processes and on the human side. Much engineering effort goes into reducing systems’ complexity. We argue that the real issue is reducing complicatedness. This is an important distinction. Complexity can be a desirable property of systems, provided that it is *architected complexity* that reduces *complicatedness.* Complexity and complicatedness are not synonyms. Complexity is an inherent property of systems, and complicatedness is a derived function of complexity. We introduce the idea of complicatedness of complex systems, present equations for each and show that they are separate and distinct properties. To make these ideas actionable, we present a design methodology to address complicatedness. We show examples and discuss how our equations reflect the fundamental behavior of complex systems and how our equations are consistent with our intuition and system design experience. We discuss validation experiments with global firms and address potential areas for further research. We close with a discussion of the implications for systems design engineers. As engineers, we believe that our strongest contributions are to the analysis, design and managerial practice of complex systems analysis and design.

Tang and Salminen [TAN 01] illustrated the difference between complexity and complicatedness. Relative to a manual transmission, a car's automatic transmission has more parts and more intricate linkages. It is more *complex.* For drivers, it is unquestionably less *complicated*, but for mechanics, which have to fix it, it is more *complicated*. This illustrates a fundamental fact about systems; *decision units* act on systems to manage their behavior. Complexity is an inherent property of systems. Complicatedness is a derived property that characterizes an execution unit's ability to manage a complex system. A system of complexity level, $\mathbf{C_a}$, may present different degrees of complicatedness, **K**, to distinct execution units **E** and **F**; $\mathbf{K_E = K_E\ (C_a) \neq K_F = K_F(C_a)}$.

There is no research on complicatedness and complexity as distinct properties of systems. Research seems to cluster around engineering management and physical products. The focus is on modularization and interactions with a bias to linear systems and qualitative metrics. There are efforts on methodologies and tools, but theory, foundations and software have a demonstrably lesser presence. The work of Ferninand on software systems complexity is a happy exception [FER 93]. It is analytical, rigorous and elegant. The services and enterprise solutions are barely addressed. This is a serious omission given the high proportion of services in industrialized economies. Although layering of abstract systems and reintegration has a long history, the literature is skewed to decomposition rather than integration.

Overwhelmingly, the literature considers a system with a large number of elements as complex. Very few address the linkages among the elements, and no one considers their bandwidth. All these factors are inherent characteristics of systems. Therefore, Tang and Salminen [TAN 01] argued that the number of elements, the number of *interactions* among them and the *bandwidth* of these interactions determine the complexity of the system. As any of these increase, we expect complexity to increase. For example, a system $\mathbf{N} = \{\mathbf{n}_i\}_{i = 1,2,\ldots,p}$ with binary interactions among the elements. Complexity, $\mathbf{C_N}$, of this system does not exceed p^2, we denote this by $\mathbf{C_N} = \mathbf{O}(p^2)$. System $\mathbf{M} = \{\mathbf{m}_j\}_{j = 1,2,\ldots,p}$ can have complexity $\mathbf{C_M} = \mathbf{O}(p^k)$ where $k > 2$. When $\mathbf{M}$ admits $\{\mathbf{m}_j\mathbf{x}\mathbf{m}_r\}_{jr}$ and $\{\mathbf{m}_j\mathbf{x}\mathbf{m}_r\mathbf{x}\mathbf{m}_s\}_{jrs}$ interactions, $\mathbf{C_M} = \mathbf{O}(p^3)$. If $\mathbf{M}$ admits $\{\mathbf{m}_j\mathbf{x}\mathbf{m}_r\mathbf{x}\mathbf{m}_s\mathbf{x}\mathbf{m}_t\}_{jrst}$ interactions, $\mathbf{C_M} = \mathbf{O}(p^4)$. This characterization of complex systems admits systems with feedback loops of arbitrary nesting and depth, and high bandwidth interactions among system elements. Complexity is a monotonically increasing function as the size of the system size, number of interactions increases and bandwidth of interactions increase. In the limit, complexity $\rightarrow \infty$. We define complexity by $\mathbf{C} = \mathbf{X}^n \mathbf{\Sigma}_b\, \mathbf{B}^b$

where $\mathbf{X}$ is an integer denoting the number of elements $\{x_e\}_{e = 1,\ldots,p}$

n is the integer indicated in the relation $O(p^n)$

and $\mathbf{B}_1 = \Sigma_{ij}\lambda_{ij}\beta_{ij}$

λ_{ij} is the number of linkages between x_i and x_j

β_{ij} is the bandwidth of the linkages between x_i and x_j

$\mathbf{B_2} = \Sigma_k \lambda_k^{ij}\beta_k^{ij}$

λ_k^{ij} is the number of linkages between x_k and (x_i,x_j)

β_k^{ij} is the bandwidth of the linkages between x_k and (x_i,x_j) and, in general,

$B_n = \Sigma_n \lambda_p^{ijk\ldots n-1}\beta_n^{ijkl\ldots n-1}$

$\lambda_n^{ijkl\ldots n-1}$ number of linkages among x_k and $(x_i,x_j),(x_i,x_j,x_k),\ldots,(x_i,x_j,x_k,x_{k,\ldots},x_{n-1})$

$\beta_n^{ijkl\ldots n-1}$ linkage bandwidth among x_k and $(x_i,x_j),(x_i,x_j,x_k),\ldots,(x_i,x_j,x_k,x_{k,\ldots},x_{n-1})$

B is a measure of the information capacity among the elements of the system. Note that the monotonicity properties are not violated. In Figure 19.2, an example is given.

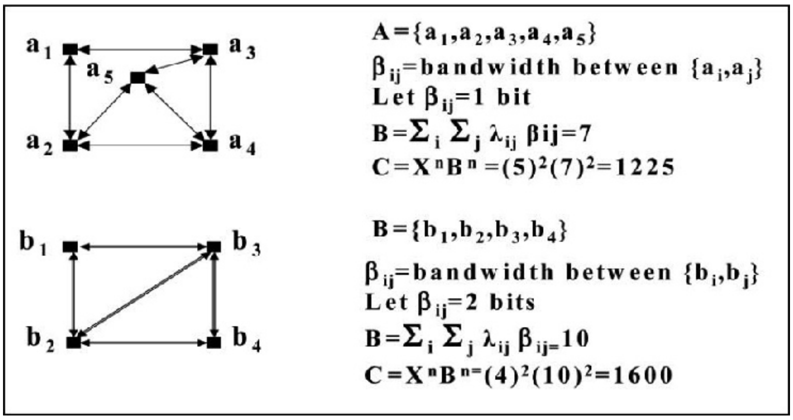


Figure 19.4. *Complexity example of two systems*

Complicatedness is the degree to which a decision unit for the system is able to manage the level of complexity presented by the system. The decision unit can be another system or a person. Complicatedness is a function of complexity, $\mathbf{K} = \mathbf{K}(\mathbf{C})$. Let us explore the properties that we expect from a complicatedness function. We expect that monotonicity of complexity is imposed on complicatedness, but do not expect that they are identical. Clearly at $\mathbf{C} = 0$, $\mathbf{K} = 0$. Consider $\mathbf{K}$ when $\mathbf{C}\rightarrow\infty$. Intuitively, there is a level of complexity at which the decision unit can barely cope with the system. The system is becoming unmanageable. For example, most people can visualize a graph, $\mathbf{g} = \mathbf{g}(x,y)$ of $\mathbf{C_g} = \mathbf{O}(p^2)$, but it is harder for $h = h(x,y,z)$ with $\mathbf{C_h} = \mathbf{O}(p^3)$. Few can visualize a surface with four variables, although complexity has only reached $\mathbf{O}(p^4)$. Consider equally incomprehensible systems **A** and **B,** where $\mathbf{C_A} = \mathbf{O}(p^{100})$ and $\mathbf{C_B} = \mathbf{O}(p^{100,000})$, respectively; $\mathbf{K_A} \gtrsim \mathbf{K_B}$ although $\mathbf{O}(p^{100,000}) >> \mathbf{O}(p^{10,000})$. Therefore, when $\mathbf{C} = 0$, $\mathbf{K} = 0$ and when $\mathbf{C}\rightarrow\infty$, $\mathbf{K}\rightarrow\mathbf{K_{max}}$ asymptotically.

Accordingly, Tang and Salminen [TAN 01] systems are designed to operate and to be managed approximately at an optimal point of complexity, say $\mathbf{C^*}$. For $\mathbf{C} < \mathbf{C^*}$, although the complexity increases, it is well within the interval of manageability. At $\mathbf{C} = \mathbf{C^*}$, the system complexity is optimal for the decision unit. For $\mathbf{C} > \mathbf{C}^*$, complexity is increasing and the decision unit can manage the system with decelerating effectiveness. Mathematically, $d\mathbf{K}/d\mathbf{C} > 0$ in the open interval $(0, \infty)$. At $\mathbf{C} = \mathbf{C^*}$, $d\mathbf{K}/d\mathbf{C} = 0$ and $d^2\mathbf{K}/d\mathbf{C}^2 = 0$. Complicatedness has reached an inflection point. So that for $\mathbf{C} > \mathbf{C^*}$, $d^2\mathbf{K}/d\mathbf{C}^2 < 0$, that is complicatedness is reaching saturation. The decision unit's ability to manage complexity has reached diminishing returns. For $\mathbf{C} < \mathbf{C}^*$, $d^2\mathbf{K}/d\mathbf{C}^2 > 0$, complexity is growing faster than complicatedness. Because the logistic function is one of the simplest mathematical expressions that have all the above properties (Figure 19.3). We adopt it to express complicatedness. $\mathbf{K}(\mathbf{C}) = \mathbf{K}_{\mathbf{max}}/(1+e^{-\alpha C})$

where **e** is the transcendental number e = 3.2718 2818 284…;

α is a constant specific to the decision unit;

C is the complexity of the system.

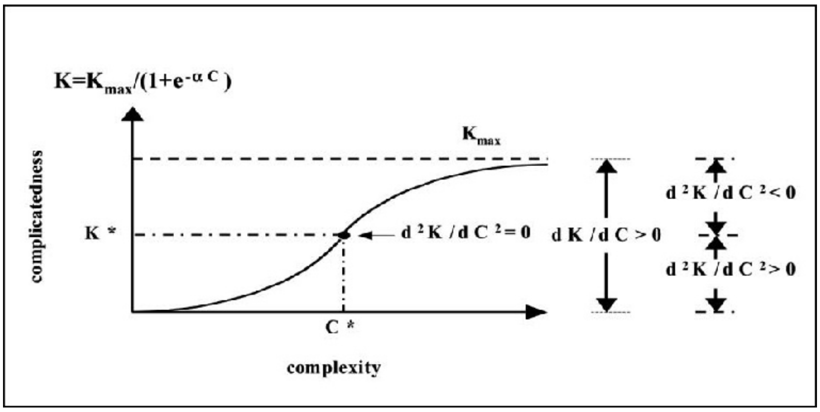


Figure 19.5. *Complexity and complicatedness*

Without loss of generality, we set $\mathbf{K}_{\mathbf{max}} = 1$ to indicate the abject complicatedness. There are other functions, such as the Gompertz curve, Weibull distribution and log-reciprocal function, that can be used. The major differences are the location of the inflection point, the growth pattern before and after the inflection point, and the symmetry around the inflection point.

Earlier, we presented the automobile transmission as a complex system that is uncomplicated. Neural networks are more interesting as a systems engineering example. Typically, they are applied to situations where there are an intractable number of data points to analyze in order to set a course of action. To solve this difficulty, the neural network is layered, see Figure 19.6. The complexity is increased relative to the input vector. Many new elements, new interactions and their bandwidths have all increased the initial complexity. But *architected complexity* has reduced an intractably complicated input vector to an output vector that now makes the system manageable. This approach has been proved effective for engineering paper machines [PIL]. This is a non-trivial example. The purchase price of a paper machine ranges around \$50 million and more. The mill generates about 10^9 data points, which are processed in real time by adaptive and distributed neural networks embedded in the machine. We experienced this case practically on a running paper machine in Finland and elsewhere to which we may revert at the end of this chapter, where we bring to the readers' attention the "*live experiments*".

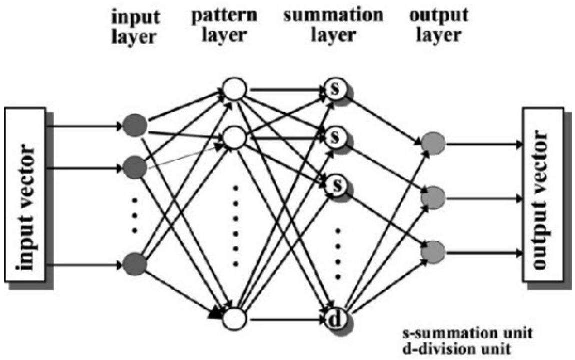


Figure 19.6. *Use of neural network as architected complexity*

The telecom infrastructure is one of the most largest systems in the world. On demand, it interoperates an immense array of networks, products and computers. The system complexity is enormous, yet we routinely make transcontinental telephone calls and download music and pictures from the Web. *Architected complexity* has made telecom networks manageable. Engineers created the OSI (Operating System Interface) reference model by partitioning the network functions into distinct layers. This architectural innovation creates, at each level, a distinct presentation of the network that is more abstract at each successive layer. Each layer presents to decision units a specific *system image* of the network that is vastly less complicated. Layering system images is a widely adopted doctrine in computers, for example programming languages. With the first computer, applications programming was very difficult. Programmers had to embed arcane hardware details into their algorithms. High-level languages were invented to present an abstract, but domain specific, system image for programming. A layer of software hid and encapsulated, transparently to the programmer, all machine specificities.

Architected complexity is a very effective complexity management strategy; it reduces complicatedness.

It suffices to present three examples. The typical video cassette recorder (VCR) control panel is a classic example of complex and complicated design. Another example is PC software or “bloatware”. So many application packages are functionally so extravagant that the average person can learn only a fraction of their functionality. Cellular phones are in danger of becoming examples of complex and complicated products.

There is good and bad cholesterol. Similarly, there are *architected* and *unarchitected complexity*. The former reduces complicatedness, whereas the latter does not. There are two important principles in *architected complexity:* partitioning the system into modules and reintegrating them while maintaining the systems integrity. Many decomposition schemes address the first principle; Karnaugh maps for digital circuits, Djysktra architectures for computers, design structure matrix for mechanical products [GHA 99], etc. They are effective tools, but when the decomposition creates a large number of new components and interactions, the result can become intolerably complicated and make reintegration impractical. Reintegration is less visible in research, although widely practiced by engineers.

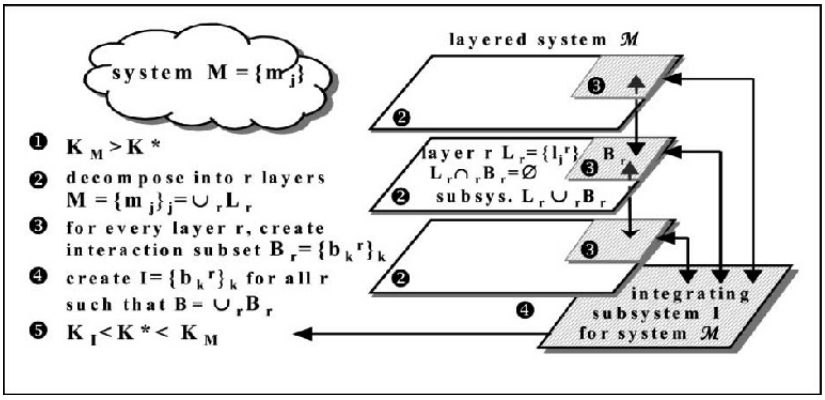


Figure 19.7. *Architected complexity to reduce complicatedness*

According to Tang and Salminen [TAN 01], the goal is to architect complexity so that **M** is transformed into $\mathcal{M}$, such that $K_{\mathcal{M}} < K_{\mathbf{M}*}$, although $C_M > C_{M*}$. Partition **M** into layers, $\mathbf{L}_r = \{\mathbf{l}_j^r\}_j$ such that $\mathbf{M} = \cup_r \mathbf{L}_r$, that is all the elements of **M** appear in some specific layer. Functional decomposition is an engineering application of this principle. For a paper mill, these can be the mechanical, control and process domains [HUB 88], or for a computer, the arithmetic unit, main memory, the I/O units, etc. Design the layers so that there are only intra-level interactions among the elements of a layer. Create $\mathbf{B}_r = \{\mathbf{b}_k^r\}_k$ for ever layer $\mathbf{L}_r$, so that $\mathbf{L}_r \cap_r \mathbf{B}_r = \varnothing$. Design **B** so that only elements of **B** communicate with each other. $\mathbf{B} = \cup_r \mathbf{B}_r$ is ***M*** 's communications subsystem. For ***M***, design a system integration unit $\mathbf{T} = \{t_x\}_x$, which on one side interfaces with **B** and on the other side with the decision unit. Note that T presents the decision unit with an *image* of the system ***M***. This is a hallmark of a good architecture. Good design always presents a less complicated *system image* to a decision unit.

We have seen here that when mastering complexity versus complicatedness it seemingly is a challenging aspect, though very scientifically interesting. This cast part of the text gives you an insight about its nature and solution procedures when one enters into this area. It is time to turn the topic of methods and solutions for design, a product development scheme; earlier, we mentioned an original story in this area. When we plan to produce a faster product development scheme, we would imagine that there must be benefits to this phenomenon soon or at the end, when we are able to execute it into production, at a level before or later. These benefits, that we are now fashioning, have some impact on overall productivity and an alternative selection criteria to draw upon.

19.4. Benefits of fast product developments

When aiming at a higher product development and design speed, a company must, as a rule, change its traditional internal working methods. This will put a heavy strain on even the most positive of corporate climates. Furthermore, in such a case, the main contractor must coordinate its product development activities with its network of subcontractors.

The transfer from research to pragmatic design and manufacture is difficult when extensive entities and complex matters must be dealt with. There are phases during which the development work must be cyclically accelerated, alternating with phases for setting the sights for future times, for creating new visions. When creating breakthrough products, the development process itself should be developed so that the information and knowledge gathered through experience, can be documented and retained in the active data storage. The essential part of the historical information on the various evolution versions of the product must be reusable to speed up the evolution process as much as possible. The modular structure of the multi-technical product is then extremely essential so that in the development work it is possible to concentrate the resources to the most important area.

The fast product development brings in at least two main benefits:

– When we get the product on to the market before the competitors come out with their own equivalent or better due to an early access benefit which, at best, gives a higher price and a larger market share (Figure 19.3).

– In time we develop new products faster than our competitors; we can start its R&D project later and still make a better success because the quicker products would be able to use the latest technology and fresher market information.

While launching a new product on the market, timing is often a key factor in whether or not the faster production process would make a better profit. A correctly timed introduction gives us a large market share right from the beginning and also helps us build up volume throughout the entire lifecycle of the product. Luckily, the speedier a product development project is, the less unforeseen problems are likely to occur during its course. The reason is that neither the market requirements nor the rival products can change much during a short development process. Consequently with a higher speed, the predictability improves. However, when developing the new products and building up timetables, one must always make provision for redesign, because it is inherent in this work that one must face the "new" and therefore be prepared for some degree of uncertainty. To understand the problems while building up a timetable, we should mainly deal with certain special characteristics of product development.

By developing its technology and product strategy, a company may significantly improve the proper prioritization of its R&D projects, and thus the efficiency of the entire production design process [CAR 93]. Speeding up of product development may be examined in two ways:

– developing the product development process (that is recognizing the value-added functions, flexible team work and continuous improvement);

– developing the product concept (i.e. especially modular concept).

The combination of these two is very important for the future. It gives us a possibility to use the sourcing of technology in a controlled manner. The division of labor can be developed more speedily within the company and when working with subcontractors. This division offers a natural cause to speed up the product development scheme when part of the responsibility for product development could also be shared with the subcontractors.

A mechatronic product would be described according to Figure 19.8, when the interfaces between different sub-areas are emphasized [SAL 94]. Dividing the product with the help of interfaces is also fundamental to the modularization procedure.

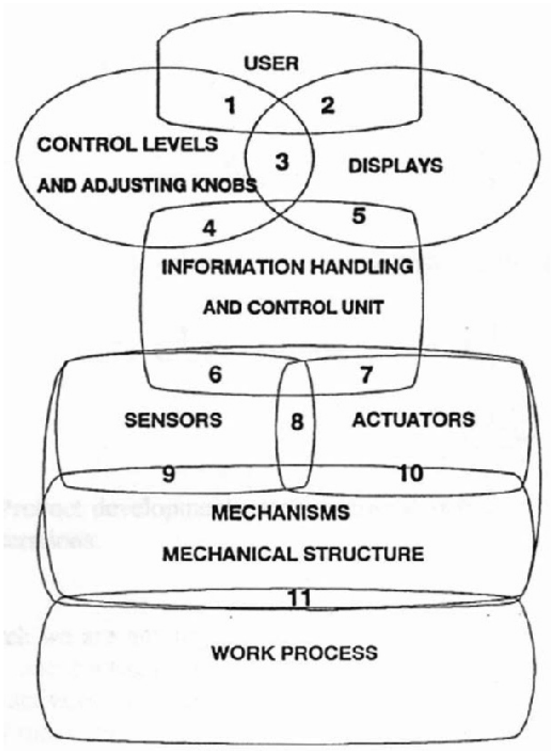


Figure 19.8. *Model of the major interfaces of a mechatronic product*

In Figure 19.8, interfaces 1 and 2 form the actual man–machine interface. It should be as user-friendly as possible. Sometimes these two interfaces are combined, see at interface 3. Interfaces 4–7 represent the distribution of information handling, when intelligence is integrated to controllers, displays, sensors, actuators and mechanisms. On the other hand, interfaces 8, 9 and 10 characterize the integration of sensor functions and actuators to the mechanisms. However, the control electronics today are often integrated into the mechanisms compared with the 19th Century setups.

19.5. Nature of product development process

Twiss pragmatically synthesized the nature in saying that research is the antithesis of certainty. All that the formal scheduling systems can do is that they may raise the ratio of objective and subjective criteria in planning and decision-making [TWI 80]. The real innovative ability of a company is based on the information and know-how accumulated or generated in the company. For this reason, the innovative ability must be developed over a longer period of time. The usability and accessibility of the accumulating information are also key factors when we are aiming at a faster implementation of a product design project. Information gathering must, therefore, be managed in a purposeful way, bearing the needs of future product design projects in mind.

In research, we are moving on a partly unknown territory and hence, must be prepared for unexpected surprises, even unpleasant ones, in the planning and steering of research activities. Furthermore, there is a particular problem with scheduling and building up of timetables in R&D, that are not found in routine projects. This is a problem of related interdependencies (cross-coupling), where coupling of tasks is not serial or parallel, but a more complex one (Figure 19.9). Finding a solution for such awkwardly coupled tasks requires iteration. How much time this takes depends on two factors: the time taken by one cycle of iteration and the speed of convergence. If we must include research work in such an iteration cycle, we may have to waste a lot of time. Therefore, it is important to complete all search and exploration early enough to allow us to perform the actual product design project as fast as possible.

Practical product design projects and numerous studies show that there is a somewhat fuzzy phase of preliminary studies and surveys at the beginning of a design process. Too often, these studies are conducted as part-time jobs, between other tasks. Furthermore, they often lack an objective and a timetable. The working methods, too, may be erratic and somewhat spontaneous. All this wastes the precious calendar time reserved for the project [REI 91]. An extensive study [COO 90] conducted in the United States states that the product development projects leading to successful businesses are different from the rest in that distinctly more work was put in at the early stages of the project. We should note particularly that the total amount of work at these early stages is very small, and, consequently, the costs are fairly low, at most a few percent of the total expense of the entire product development project.

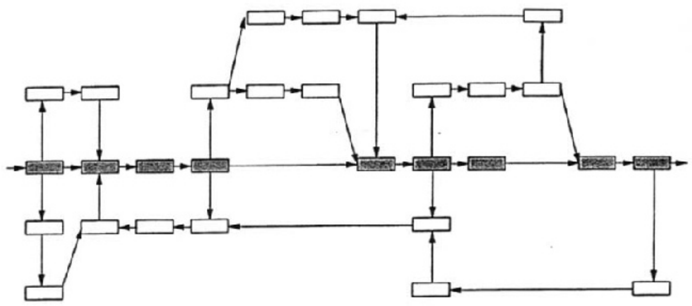


Figure 19.9. *Product development process, critical path and feedback loops caused by iteration*

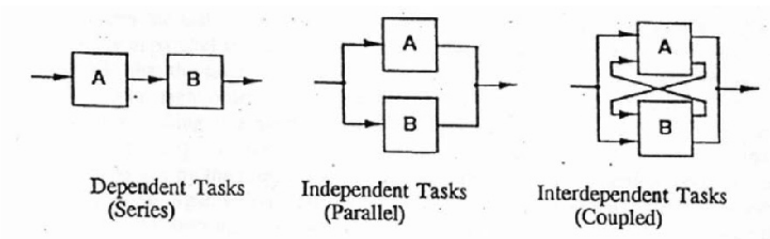


Figure 19.10. *Three possible sequences for two design tasks [EPP 91]*

When building up a concept, a critical, most problematic point or area is circumvented too easily. It is difficult to recognize critical subprocesses. Usually, these are the ones where the widest technology leaps are made and also where the greatest risk is taken. The most sensitive points on the critical path should be solved first if we want to shorten R&D time and accelerate the process. When increasing speed, the degree of integration on the critical path must be defined precisely. When deciding which functions should be developed parallel to which others, special attention must be paid to the weak links, i.e. the manufacture of the prototype, testing and the utilization of the results. With a little extra work input at the beginning of a product design project, it is possible to accomplish a lot, both shortening the time required and guaranteeing the business success. This stage provides one of the cheapest and best chances of speeding up a product development project [COO 90]. The projects studied show that about half of the product design time is lost before the nomination of the project group [SMI].

19.6. Planning the timetable of a product design project

The same scheduling principles apply to a product design project as to engineering projects in general. The design project must be divided into tasks, and the interdependencies between these must be distinguished. The relations between the tasks may be of three types as explained earlier, i.e. parallel, serial or coupled. A genuine parallel relation between the tasks makes the scheduling easy because these tasks may actually be done in parallel and without interdependencies. Even the serially connected tasks are easy in that the tasks may be absolved independent of each other as long as they are performed in the right order. On the other hand, the coupled tasks are difficult from the viewpoint of making up a timetable and keeping it because solutions to these problems must often be sought by some kind of iteration process. Therefore, they often require special measures by the project manager. For instance, if such coupled tasks are performed within different organizations, there is a risk of the parties only waiting for other partners to give the start-up information, with no progress in the actual project. Hence, it is important that such issues between the tasks are recognized at the project planning stage.

The goals of product development and product design should be business-oriented. Although the product should be excellent from the viewpoint of the selected customers, the product designer must not merely optimize the functional characteristics of the machine; he should mainly aim at optimizing our business. It follows from this principle that both the requirements of the customer and the business requirements of company should be taken into account when setting goals. From the viewpoint of speedy product design, it is important to keep the goal-setting as clear and unambiguous as possible. It should contain the following objectives:

– operational objectives;

– quality objectives;

– production objectives;

– marketing objectives;

– maintenance objectives;

– economic objectives;

– objectives of logistics;

– objectives of interest groups;

– the company's objectives;

– timetable objectives.

These objectives should be expressed as clearly as possible, preferably quantitatively, even in the case of quality objectives. In setting the goals for product design, we must pay special attention to the degree of change that must be made in the product to secure its competitiveness. Whether or not a conceptual change is made in the product generally makes a great difference in the speed at which the project is carried out [HOL 90]. Industrial products vary greatly in terms of design speed. From the viewpoint of customer flexibility, they may be divided into the following categories:

– serial products, delivered to all customers, identical in form and function;

– products assembled from standard parts, according to the customer's order;

– products designed and manufactured fully according to the customer's order (tailor-made);

– various combinations of the above; most investment goods fall into this category.

Factors affecting the economy of the development of products and services are, for example, the time at which they are launched on the market, their characteristics and the technology used to develop them.

Figures 19.8 and 19.9 schematize the interfaces of a mechatronic product and product development process, respectively. Figures 19.11, 19.12(a) and 19.12(b) schematize the implementation of development and design projects based on either a static or a dynamic product development concept. See Figure 19.11.

The design of a product based on a static concept would be accelerated and developed also by automating the design process. On the other hand, a product based on a dynamic concept mostly requires more iteration and development time.

Figure 19.11. *Product design based on a static product concept*

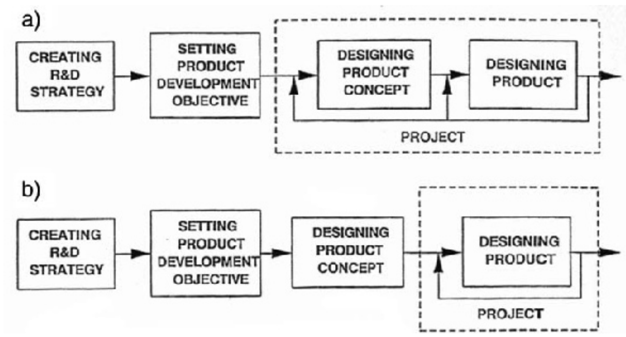


Figure 19.12. *Product development based on a dynamic concept [SAL 94]: a) fully dynamic concept with strong interaction between the concept and its elements and b) semi-dynamic product concepts with interactions between the concept and its elements, weak enough to allow changes without extensive iteration.*

19.7. Designing the product concept

Creating a product concept is in many respects an extremely important phase of the design process. It largely determines the flexibility of the product entity in meeting the special requirements of each individual customer. The product concept also decides how fast, reliably and economically we will be able to serve the customer after he has given us his order. The choice of a product concept often has a very strong impact on the speed of the actual product development.

19.8. Enhancing conceptual design

The objective of a conceptual design should be fewer and simpler interfaces. The number of technical elements in a modern product may be high. Consequently, the number of interactions between the elements is very high as well. For instance, in the system in Figure 19.7 with $3 \times n$ elements, the theoretical number of interfaces between the elements is $3 \times n/2$ $(3 \times n - 1)$. Although in actual systems, interactions do not occur between all the elements, the risk of making the concept too complicated at the system design phase remains. The reason usually is that not enough attention is paid to the interfaces and their number (see Figure 19.13).

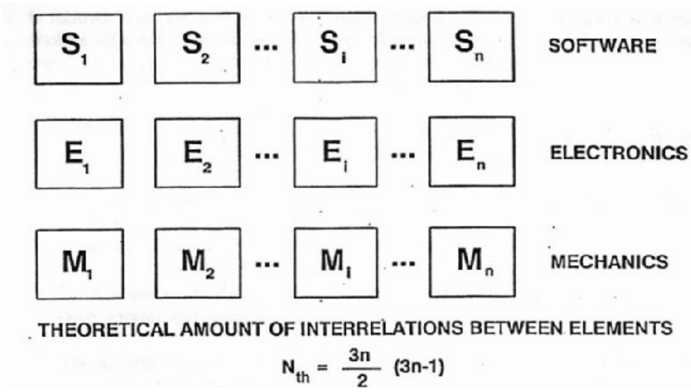


Figure 19.13. *Theoretical number of interfaces between elements in mechatronic technical systems with 3 × n elements [SAL 94]*

In speeding up and scheduling a product development project, key questions related to conceptual design are the following [SAL 94]:

– What functions should be included in the product?

– What are the inter-relationships between the functions?

– What functions will be performed and by which part of the machine or module?

– What are the performance data of the product modules and what are the safety margins of the design?

– What are the interfaces between the elements? These may be both geometric-physical contacts and signal transmission methods as well as protocols related to the flow of information.

– How will the reliability objectives of the overall product be divided among the various modules?

Although we are faced with asking and finding solutions for such essential questions at the phase of designing the product concept, it is often largely neglected; many of these questions may never even be subjected to explicit consideration. This involves considerable risks for the future of the product. Even from the viewpoint of fast product development, the decisions made at this phase are very far-reaching. Decisions related to the product concept may essentially slow down the product development project. The implementation speed of such a project is largely decided by the choice of the product concept and especially the module construction linked with the design of the interfaces. What is perhaps worst is that these decisions also affect the speed at which this product can be updated during subsequent product development cycles. The determination of the product concept may predestine the development speed of future product generations as a slow one.

It thus follows from the above that we, when designing a product concept and making choices affecting it, must necessarily pay attention to the effect of the concept on the speed of product development. Below, we take a look at the most important factors affecting the development speed through the choice of product concept [SAL 94]:

1) An appropriate module construction may promote the progress of parallel design teams working on different projects at the same time.

2) By using standard components and modules, we may concentrate our efforts on developing the product characteristics that are the most important ones for the customer and for our own competitiveness.

3) By the use of modules, we may also promote subcontracting, both in product development and manufacture.

4) The appropriate division of product development risks among different modules gives us a better chance to control the risks involved in both the product itself and in the scheduling of the product development project.

5) If we divide the product into modules in a way that allows the fastest-developing technologies to be contained in their own modules, we will be able to update the product design quickly without having to change too many details.

6) As the product concept, we should choose a solution that is as clear and simple in its interconnections as possible.

7) The concept should be insensitive to minor changes in the interconnections in the adjacent modules.

These principles and possibilities are of great importance and should be taken seriously because they give us extensive chances to make both the product design and, in part, the production more efficient. In the conceptual design of a machine, we must make several important decisions concerning the structure of the machine. This structure may be looked at on several levels and from the viewpoint of several different interactions. The key interactions are [HUB 88]:

– interaction in the functional structure;

– interaction between the functional organs;

– interaction between the constructional elements.

The constructional elements may be divided into three categories as follows [HUB 88]:

– internal constructional elements;

– border elements forming a link between the internal subsystems of the total system;

– boundary elements located on the outer surfaces of the technical system.

Among these, the second category, the border elements, is important when we consider the constructional elements of a machine that have an effect on the speed of product development.

19.9. Interaction between the parts of the machine

The interfaces between the parts and modules of the product and the complexity of the interactions across them are very important for the speed of product development. It is in the interfaces between the machine parts that the functional, physical and constructional interaction happens, making the intended function of the whole product possible. Essential parts of this interaction are the information, energy and material flows[6]. All these flows have an effect not only on the function of the product but also on the process of developing it and, in particular, on the complexity and speed of product development. Another extremely important factor in the interaction of the machine parts is their geometry.

6 This is mentioned earlier in relation to Figure 19.8.

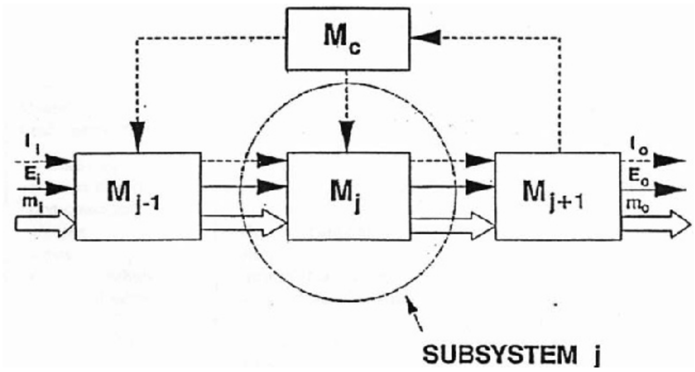


Figure 19.14. *Development of a module with consideration of the information, energy and material flows between it and the adjacent modules (a product consisting of serially coupled modules)*

The interaction between the machine parts can be looked at from many viewpoints. Below, we try to find those features of interaction that have an effect on the speed of product development. A particularly important question is "how strongly a change in the structure of an adjacent element (machine part or module) affects the elements connected to it and how large a volume of design work is made necessary by this change to modify these elements according to the new requirements?"

19.10. Effect of the strength of interaction between product parts and development speed

The interaction between modules to be connected to each other and the development speed has an essential impact on the process [SAL 94]. It is divided into three categories: weak, semi-strong and strong. The interaction suggested here may be functional, physical, chemical, geometrical and related to the information content that are in transition. At most, loss of product development time is caused by the physical (and chemical), geometrical and informational (control, information handling) aspects. Components and modules with weak interaction are characterized by the fact that they hardly cause any delays in the convergence of the product development process. Components with strong interaction, on the other hand, slow down the convergence in product design and development, which leads to the lengthening of the entire product development process when the elements in question happen to be on the critical path of the design process. This discussion is briefly summarized in Tables 19.1–19.3.

Machine parts with weak interaction	Machine parts with semi-strong interaction	Machine parts with strong interaction
Wireless signal transmission methods Non-contacting sensors Ropes Electrical cables Pipes and tubes Chains Belts and other flexible bands and strips	Flexible shafts Articulated shaft lockings (universal joints, Hooke's coupling, etc.) Springs (coil spring, rubber and plastics, leaves etc.) Actuators (hydraulic and pneumatic)	Machine body/frame Shafts & spindles Bearings, Screws Welded, flexible, solder and glued Joints Electromagnetic power transmission parts (when a good efficiency is the objective)

Table 19.1. *Classification of typical machine parts according to strength of geometrical interaction*

Functions with weak interaction	Functions with semi-strong interaction	Functions with strong interaction
On/off signal transmission Signal transmission in slow measuring Transmission of sensor signals of a scalar quantity Slow transmission of signal	Inter-module signal transmission in a modular control program Rapid transmission of simple signals	Rapid measuring and control Navigation in a changing environment Graphics, pattern recognition Embedded control (interface between electronics and software modules)

Table 19.2. *Classification of informational functions according to strength of interaction*

The interaction between the informational functions and the geometrical interaction between machine parts are the areas of product development that usually require most design work (Table 19.3).

Informational interaction Geometrical Interaction	Weak	Strong
Weak	On/off signal (overpressure sensor) Warning light cable	Gear wheel
Strong	Embedded control Image handling and transmission Navigation	Electrohydraulic actuator with control Electromagnetic bearing with control Electropneumatic actuator with control

Table 19.3. *Interaction between informational functions on one hand and geometrical interaction between other machine parts on the other hand in various parts usually found in a machine system*

The stronger the interaction is, the harder it is to work out the design. This means that the design work gets laborious and slows down as we move down and to the right in the field of Table 19.3. The interaction between the information system and the mechanical parts is today gaining more and more importance. This is especially true of the interaction between the control and adjustment algorithms of mechatronic systems and the mechanical parts of the machine. This interaction may, at its most difficult, be quite strong and complicated.

So far, we came up with terrifying problems in identifying and tracking the idea to a physical mechatronic product, and it is possible now to lead you in configuring and isolating mechatronic products. Mechanical engineers, who think of a product as a complete "*end-to-end one*" and stand-alone system, need not necessarily specify its surrounding features that would need backup and updating processes, including continuous maintenance through spares and other intimated agendas. They consistently assume that a physical product has other agendas that are included in the specification and do not repeat them. We simply look at a physical product and perhaps its other agendas such as after-sales service or other lifecycle coverage potentials. Instead of going into details, we define what the product is and what service it gives and then combine them to understand the phenomenon much better than before.

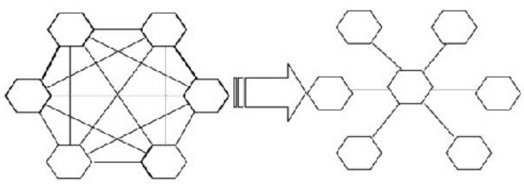
19.11. Definition of product and service

The product and service need to be defined. An engineer assumes when he produces a technical product, which has internally a maintaining service domain. On the other hand, a service provider assumes that the concept consistently offers not only the product but also services. Design engineers and product users are misunderstanding; or assuming, what is a product and what is a service. It is like the fundamental law (Coulomb's law[7]), which states that the electric force of attraction or repulsion between two point charges is directly proportional to the product of magnitude of each charge and inversely proportional to the square of the distance between them. The force also depends on the permittivity of the medium in which the charges are placed. If Q_1 and Q_2 are the points of charges at distance d apart, the force is:

$$F = \frac{1}{4\pi\varepsilon}\frac{Q_1, Q_2}{d^2}$$

where ε is the permittivity of the medium.

The force F is attractive for opposite charges and repulsive for the same charges. In this context, it is addressed that product and service are scientifically attracted. They are treated or understood to be one and the same. One may interpret it either way because of the purpose paradigm. It is further defined so as to understand the phenomenon in the form of a molecular structure, see Figure 19.15.



$$\frac{n!}{r!(n-r)!} * 2 = n(n-1)$$

where r=2

r = number of systems
n= number of application

Figure 19.15. *Product centric versus service centric*

7 Dictionary of Science, University of Cambridge, UK, 1998.

It is shown in the above figure that product and service are two different entities that are not well defined but can be seen when distilled. Let us look at the same phenomenon from a different angle. In physics, the rate of change of the heat Δ h of a process with temperature, υ, carried out at constant pressure, is given by:

$$(\sigma \Delta h/\sigma \upsilon)\ v = \Delta\ (\sigma h/\sigma \upsilon)\ v = \Delta C_p\ t,$$

where ΔC_p is the change in the heat capacity at a constant pressure for the same process.

Similarly, for a process carried out at a constant volume:

$$(\sigma \Delta h\ /\ \sigma \upsilon)\ v = \Delta\ (\sigma U\ /\ \sigma \upsilon)\ v = \Delta Cv.$$

Product and service platform, when optimized, give opportunities to align the patterns from one perspective to another. Requirement and dependency threads the guidelines to form metadata for product data management (PDM).

Modern complex human–machine systems are truly mechatronic in nature, such as the control room in nuclear power plants, the autopilot or pilot-aid-systems in the cockpit or aircraft, and a paper machine in generic terms. Lin *et al.* [LIN 01] identified and explained through their experiments that the parts displayed in physical laws or effects in the ecological interface design (EID) framework appear problematic. When the mental workload of a human being is higher; compared with his self-assessment, the complexity increases that threat of the complicatedness. We defined previously that the entire product coincides with a service manifesto, which is encountered as a separate "thing[8]" and has no effect on design and interoperability terms. We will proceed further to express the previous discussions on mechatronic design methods and isolate mechatronics as a physical product and shed new light into the services. The use of industrial robots in manufacturing plants has facilitated the massive production of goods with high quality and low costs. This chapter introduces some of the basic ideas used in our previous discussions through a couple of case studies; they are classified as mainframe mechatronic products but also include the new phenomenon known as the service.

8 The term "thing" here is meant as an axiomatic in its behavior, or say designers or creators are of the opinion that this is not important although the product itself is the matter.

19.12. The case studies

We will introduce a couple of cases studies, as mentioned in section 19.11, that are focused on a *paper machine* and addressed to a *model-based implementation case* based on *provisioning of live data* to a portal provider company. Pulp and papermaking involve numerous steps, many of which rely on chemical activities. Strategies to solve elementary schemes in the wet-end process of a paper or paperboard machine are of a vital interest within the global paper industry. Figure 19.16 shows a paper machine with control points and interaction needs. The papermaking is a real-time, stochastic and an intelligent learning process.

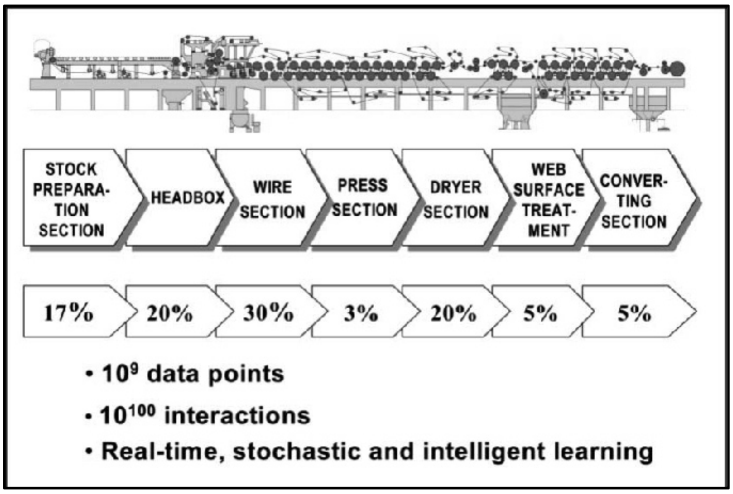


Figure 19.16. *Paper machine concept*

All the nonlinearity that appears in the chemical process of the wet-end (e.g. short circulation and headbox) of the paper machine is difficult to include in a global model. Machine wet-end must be decentralized and managed in small modules. The product design plays a significant role in the iterative and complex engineering process. Decisions made at the engineering design stage have a considerable impact on the product's lifecycle costs. With our real world, hands-on training, we learn to use new technology right the first time rather than waste time with costly trial and error. Today's technology advances at a relentless pace, while the "implementation gap" continues to widen.

The proportional, integral and derivative (PID) controller has been in use for over a century in various forms. In a typical PID controller, these elements are driven by a combination of the system command and the feedback signal from the object that is being controlled in a plant (or say paper mill). Figure 19.17 is a block diagram of a basic PID controller. In this case, the derivative element is being driven only from plant feedback. The plant feedback is subtracted from the command signal to generate an error. This error signal drives the proportional and integral elements. The resulting signals are added together and used to drive the plant.

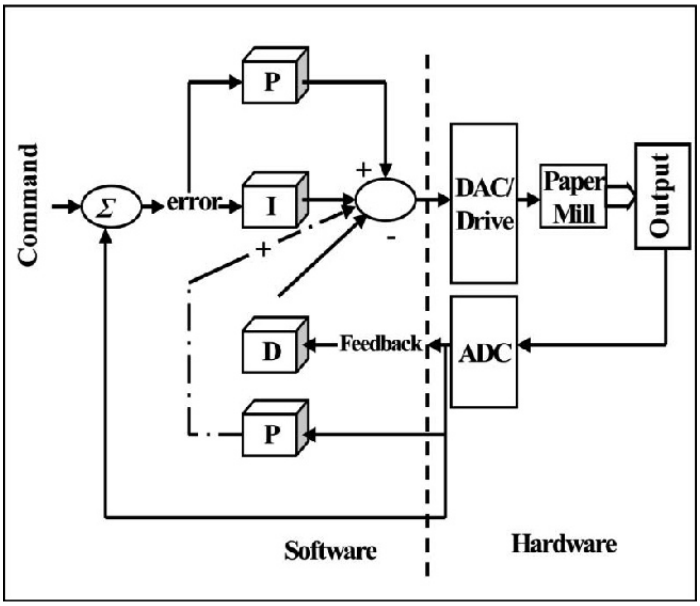


Figure 19.17. *Basic PID controller scheme*

A complex system communication is a collection of nodes and links that communicate by defined sets of formats and protocols. Within the network, there are usually three layers: transmission, switching and service. The transmission layer consists of systems, for example cables, radio links and their related technical equipment. The switching layer consists of switching nodes with generic and application software and data. The service layer, distributed among the switching network elements, consists of special hardware and their application software and data. This concept applies to robotic systems or complex generic product deployment systems in the processing environment.

19.13. Networking systems and learning mechanism

Digital integrated services based on asynchronous transfer modes is the suitable technology for transmitting information from any media spectrum and applications. A cell-based, high-bandwidth, low-delay switching and multiplexing technology is designed to deliver a variety of high-speed digital communication services. These services include local area network interconnection, imaging, and multimedia applications as well as video distribution, video telephoning and other applications. The learning algorithms we executed to a robotic concept are of asynchronous form because the recurrence of cells or movements containing information from any degree of freedom is not necessarily periodic in nature [KOH 89]. Therefore, they can handle both connection-oriented and connectionless traffic with adaptation layers and operate at either a constant bit rate or a variable bit rate connection. Each module or cell sent into the network contains addressing information that establishes a virtual connection from the source to the destination. All are then transferred in sequence over this virtual connection.

Papermaking (Figure 19.16) is modeled onto subsystems that have a system of damping elements, and the equivalent damper can be shown using the same logic and similar steps. As is, in the event of a robotic structure, the number of degrees of freedom of a dynamic entity, such as papermaking, is defined as the number of independent generalized coordinates that specify the configuration of the system. Generalized coordinates need not be restricted only to the actual coordinates of position.

The position coordinates are physical coordinates. On the other hand, a generalized coordinate could be anything, e.g. positional coordinate, translational displacement, rotational displacement, pressure, voltage, fiber, chemical component or current. Generalized coordinates need not be of the same type. The generalized mechanical configuration can be a mixed set of translational displacement and rotational displacements. The physical layer has two sublayers: transmission convergence and physical medium-dependent. The physical medium-dependent sublayer interfaces with the actual physical medium and passes the recovered information (bit stream) to the transmission convergence sublayer. This sublayer extracts and inserts the module cell with the synchronous digital hierarchy time-division multiplexed frame and then passes them to (and from) the basic model medium layer. The model medium layer performs multiplexing, switching and controlling actions based on information in the model medium cell header. This is what we experimented with in the wet-end configuration of a liquid paperboard-making machine environment. Before we switch to discussing the experiments, we address the non-holonomic constraints involved in the framework. There are two types of constraints: holonomic and non-holonomic constraints. Systems with holonomic constraints are relatively easy to deal with, whereas those with non-holonomic constraints are more difficult. Fortunately, many engineering systems that we often encounter contain holonomic constraints. For papermaking systems, the number of degrees of freedom is the same as the number of independent generalized coordinates, which is also the same as the number of independent differential equations. Holonomic constraints are equations expressed in terms of the coordinates and time. The classical wandering effect defined by Pillai [PIL] is an excellent example of holonomic constraints.

We experimented on a mechanical system that requires two independent coordinates to specify its configuration; this is called a two-degrees-of-freedom system in the papermaking process. The two-degrees-of-freedom is a special class of multiple-degrees-of-freedom systems. We express the equations without the standard second-order matrix as:

$$\begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} x_1 \\ x_2 \\ \dot{x}_1 \\ \dot{x}_2 \end{Bmatrix}, u_1 = f_1(t), u_2 = f_2(t), y = x_1 \qquad [19.1]$$

Here, the equation contains the mathematical body and physical entity, where the stated equation is a set of four first-order differential equations, where many intelligent people prefer to apply Newton's second law for the mass m_1 and m_2, respectively,

$$\rightarrow \sum F_x = m a_{cx} \rightarrow \qquad [19.2]$$

$$
\begin{aligned}
&f_1(t) - k_1 x_1 + c(x_2 - x_1) + (x_2 - x_1) = m_1 x_1 \\
&f_2(t) - c(x_2 - x_1) - k_2(x_2 - x_1) = m_2 \ddot{x}_2
\end{aligned}
\qquad [19.3]
$$

A check on the accuracy of the differential equations of dynamic systems is always useful. All the elements on the main diagonals are non-negative (either positive or zero). All the mass, damping and stiffness matrices are symmetrical (with respect to the main diagonals). The off-diagonal elements of both the damping and stiffness matrices are non-positive (either negative or zero), and the off-diagonal elements of the mass matrix are non-negative. A pure system is defined as strictly of one type, e.g. mechanical systems with purely translational or rotational motions. Liquid or fiber mix level systems are also dynamic and pure systems in this context.

With those dynamic entities, we have experimented with a non-holonomic colloidal search in the wet-end of a liquid paperboard-making machine. A robotic function was invented. We had installed seven high-speed digital cameras directed towards certain activities. Those activities are based on control parameters, chemical reactions and time. The time domain is based on the action initiated, action processed and action terminated or continued to the next stage, which is detected through some rules. Those rules are configured and adapted through fuzzy interpretations. The configurations are based on model-based domains. The adaptation layer configures the module-dependent actions. The model passes protocol data units to and accepts them from higher layers. Protocol data units may differ in variable or fixed length from the model length. The physical layer corresponds to the defined layer 1:3, and many are subconfigured protocols. The subconfigured protocols at the module configuration survive as an open system interconnected model, while the modular layer and adaptation layer correspond to parts of the open system interconnected model. The physical model consists of two logical sublayers: the physical medium-dependent and the transmission convergence. This is better understood by calling it the physical layer of the paper machine headbox. The headbox itself contains several sublayers. Those layers are comfortably operating in this case as independent model-based modules or sub-sub-layers. The transmission convergence layer functions and interprets the interfaces between the layer submissions. It also provides bit transmission capability, including transfers, alignment, line coding and electrical-optical conversions. The flow information between fiber bond including the surface value changes are transmitted and received over a physical medium. The transmission frame adaptation function performs the actions necessary to structure the model flow according to the payload structure of the transmission frame, or conveying direction, to extract the module flow out of the transaction frame (receive direction).

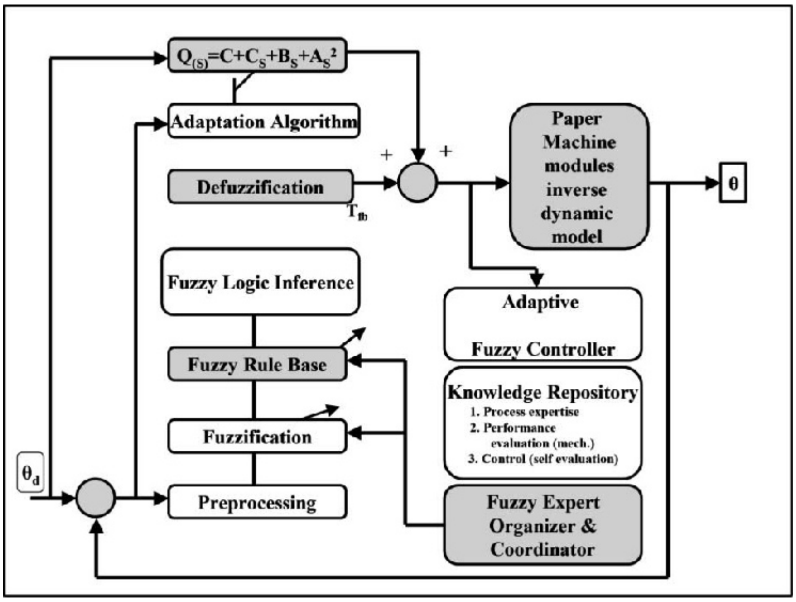


Figure 19.18. *Adaptive fuzzy logic control scheme [PIL]*

In the adaptation of network- and rule-based fuzzy logic [PRI 98] also, fuzzy modeling has been used in constructing a model for the papermaking and process, which equally accesses the referendum from the combination of the mechanical, process components with its control strategy. Figure 19.19 shows an approach for adapting a control strategy in the decentralized system as discussed earlier. The system is considered for all the players in the sequence where mechanical, process and control components fit neatly. This function is one of the several adaptive fuzzy controllers. $Q_{(s)}$ is the module system at any given moment. The subscript (s) indicates frequency domain. In the time domain, it will be:

$$Q_{(S)}=C+C_{(s)}+B_{(s)}+A_{(s)}2 \qquad [19.4]$$

where θ is a vector of control variables; A, B and C are coefficient matrices.

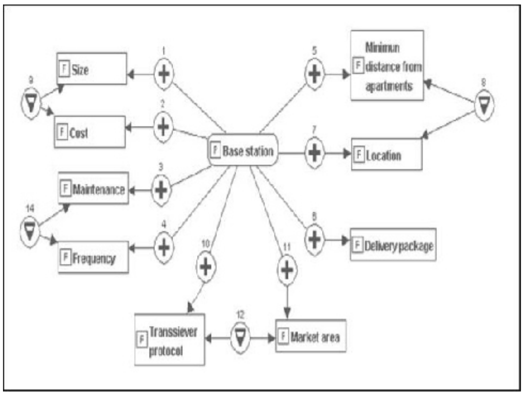


Figure 19.19. *The dependency setups of a telecom portal (a screenshot)*

We have seen that the complexity of a system like the one experimented upon is difficult to detect or converge. Salminen and Pillai [SAL] have defined the complexity for the purposes as follows:

> Any system comprising of the varieties of interdependent and compound netting or parenting suffix, we call complex system or systems. A complex system, as we have defined, has many activities surroundings, which are not linear members of the functions. It performs as linear state functions. This linearity shifts, when process non-stability appears, due to gray-areas in the netting. This gray-area evokes the process wandering. This phenomenon we call nonlinear dynamics. This nonlinear syndrome is seen in human organizations, man-machine-interfaces, partnership setups and breathing. These systems are physical cell bodies that acquire, store and utilize knowledge. The connection weight is adjusted to achieve a desired output. The process is embedded in such a manner that it computes the derivative of the energy function with respect to the connection weight as an increment of the connection weight. This way, the derivative determines the rule of changing connection weights in order to minimize the descent of the participating energy function along the gradient. Therefore, we weigh here the feedback-error-learning and not linearizing the complex system. Obviously, also the product development with tools and skills is a complex system, which needs adaptive interpretations.

This discussion is not meant to offer any software solutions but to interpret the mismatch of engineering disciplines. We have a number of cases where syntax is present, but purely missing the ontology of the concept. We suppose that the engineering disciplines need to grasp a better understanding of the engineering ontology of all related or non-related entities for this purpose. Here, we introduced a case study using a computational intelligence technique, fuzzy logic and a learning network system for solving complex problems of papermaking process. There are many similarities in applying this hybrid methodology for the purposes of robotics. We could say now that the experience shows that the same approach could be used in most of the complex nonlinear, systems.

19.14. Model-based methodology: an implemented case

We conduct a case study based on the above framework. The study executed with a telecom portal provider [SAL 03]. We have dynamically integrated the product and service concept into an industrial reality. We used also the above mentioned software tool. We modeled the portal-provisioning live data, while acquiring at random, a number of unspecified customer requirements and its new features. The following figures would explain the use of methodology and the software tool.

Figure 19.19 (an actual computer screenshot) indicates the dependency of dismantled provision automation systems. The dependency; is further expanded with performance rules. The methodology of rule creation categorized is shown Figure 19.20.

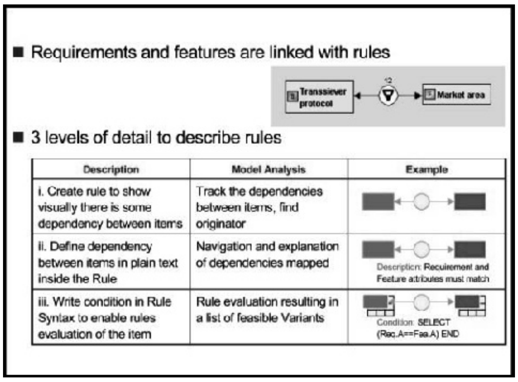


Figure 19.20. *Rule creation system*

While considering the lifecycle Challenge Management [SAL 03], LcC_{Mgt} of a product and service in this case is invisible and faster than the solid entity. Here as a research team, we had a problem in justifying this concept in general. However, Lifecycle is very short and fully packed with high technology. Here, the challenge makes sense. Therefore, we made an archive (Library) of requirement and used trajectory for tracking to build the "Backbone". The software tool is then applied to interpret and view the "Backbone". It is further scanned to pick up the required information from the archive to produce three-dimensional (3D) figures as to how a document or component is published or released collaboratively, "looks-alive", before the decision portfolio. All these steps are shown in Figures 19.21 and 19.22. While creating methodology, we scheduled an adaptive learning algorithm.

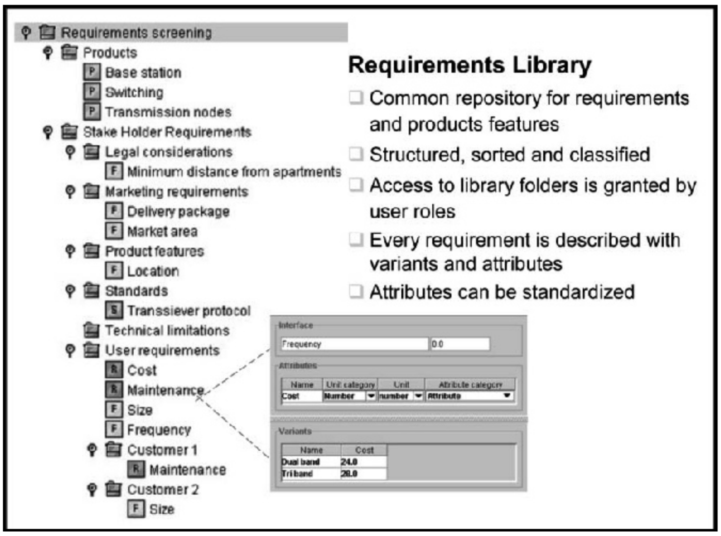


Figure 19.21. *The use of requirements library*

Integrating products and services into the new framework attracts the industrial needs in fast-growing technological areas. In the context of knowledge sharing, we use the term "ontology" to mean a specification of a conceptualization. That is, ontology is a description of the concepts and relationships that can exist for an "agent" or a community of agents. When knowledge of a domain is represented in a declarative formalism, the set of objects (e.g. service and product) that can be represented is called the universe of discourse.

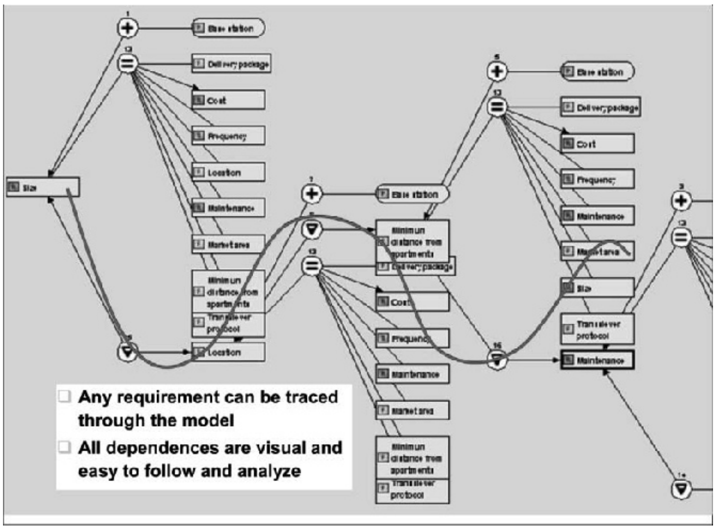


Figure 19.22. *Tracking from the trajectory*

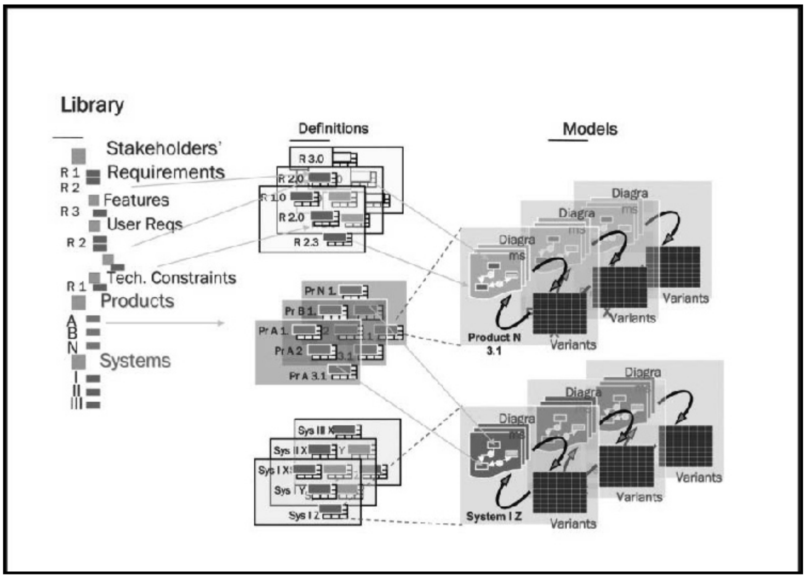


Figure 19.23. *Summary of the methodology used*

On the practical side, we automated the workload of a telecom provisioning system. The search engine and browser robot were created on a semantic infrastructure. The piloting results calculated are shown in Table 19.4 as an actual representation of the telecom portal. This project was piloted in a team of 25 employees of a portal provider. An immense number of database processes are integrated and deleted offline. Monetary value generated, is simply to understand here, as a saving through the workload of the pilot employees. A great many files or documents were opened to trace a combination for certain class integration or to add as default repositories. The results indicated that the traditional linear theory with product data management (Bill of Material)-oriented implementation of product structuring becomes difficult and expensive. We stress here that the requirement has to be captured before product versioning in its physical mode or *invisible* format. The experimental outcome is given in Table 19.4:

Experimental results of a telecom system				
Item	Methodology before use	Methodology after use	% Time saved	US$/1,000 K value generated
Semi-automated provisioning	Manpower employed mainly	Electronic format-model-based	40 – Regular man-hour	500
SCADA-Integration	Databased in different sources	Easy integration, no filtering, but models, interpretation and visualization	50 – man-hour	650
Customer requirements	Manually collected, data not used and storage capacity large	Totally modeled, integrated, agent-based browser access to view and integration to product offering	80 – man-hour	750

Table 19.4. *Experimental results of the pilot study at an IT-corporation*

On the other hand, a very small cause that escapes our notice determines a considerable effect that we cannot fail to see, and then we say that the effect is due to chance. The evaluation tool and tracking elements made life easier when implementing the project.

The inspiration for this project came from the initiatives of an IT-corporation in Finland, which failed in applying giant software which were expensive and difficult for piloting purposes. What we learned here is that the most complex things in the known universe are living creatures, such as human beings. These complex systems are made from the most common raw materials known to exist. Those raw materials naturally assemble themselves into self-organizing systems, where simple underlying causes can produce surface complexity.

During the actual implementation of the product development project, working in parallel is essential. This requires flexible service-mindedness from all participants (marketing, R&D, production, logistics, customer service, etc.) in spite of the fact that the goal must be pursued through several iterations, on the basis of information that, at least in the beginning, may be somewhat inaccurate. The speeding up of a product development project is always so dependent on the product and type of task that its efficient implementation cannot be guaranteed without detailed preparatory work and constant replanning according to the current situation. Especially, in designing a modular product concept, we should aim at making the interfaces between the modules insensitive to changes in the adjacent modules. This can be achieved by choosing interface elements with weak interdependences related to those structural elements of the modules themselves that take the longest time to design.

19.15. Conclusions

Separating complicatedness and complexity improves the clarity by which systems can be described and analyzed. In this way, it is possible to clearly separate an inherent property of the system, complexity, from a derived attribute, which is complicatedness. The mathematical expressions were formulated to capture additional properties of systems that have heretofore remained largely unaddressed. It is possible to derive results that give valuable insight into the behavior of systems. These insights are useful in the analysis and design of very large complex systems and also move us towards a theory of complicatedness.

The speeding up of a product development project is always so dependent on the product and type of task that its efficient implementation cannot be guaranteed without detailed preparatory work and constant replanning according to the current situation. Especially in designing a modular product concept, we should aim at making the interfaces between the modules insensitive to changes in the adjacent modules. This can be achieved by choosing interface elements with weak interdependence related to those structural elements of the modules themselves that take the longest time to design.

Industrial experimentations and cases shown at the end of the chapter illustrate and validate some of the mechatronics theories as science introduced in the beginning of the chapter.

19.16. Bibliography

[AND 83] ANDREASEN M.M., "Integrated product development – a new framework for methodological design", *International Conference on Engineering Design, ICED 83*, Copenhagen, Denmark, pp. 289–297, 1983.

[CAR 93] CARRUBA F.P., "Designing people-pleasing products", *International Conference on Engineering Design (ICED 93)*, The Hague, Netherlands, 1993.

[COO 90] COOPER C.G., "New products – what distinguishes the winner", *Research and Technology Management*, vol.33, pp. 27–31, 1990.

[EPP 91] EPPINGER S.D., "Model-based approaches to managing concurrent engineering", *International Conference on Engineering Design (ICED 91)*, Zurich, Switzerland, 1991.

[FER 93] FERDINAND A.E., *Systems, Software and Quality Engineering*, Van Nostrand Reinhold, New York, 1993.

[GHA 99] GHARAJEDAGHI J., *System Thinking: Managing Chaos and Complexity*, Butterworth-Heineman Publishing, 1999.

[HAB 06] HABIB M.K., "Mechatronics engineering the evolution, the needs and the challenges", *Proceedings of the 32nd Annual Conference of IEEE Industrial Electronics Society* (*IECON 2006)*, pp. 4510–4515, 2006.

[HAB 07] HABIB M.K., "Mechatronics: a unifying interdisciplinary and intelligent engineering paradigm", *IEEE Industrial Electronics Magazine*, vol. 1, no. 2, pp. 12–24, 2007.

[HAB 08] HABIB M.K., "Interdisciplinary mechatronics: problem solving, creative thinking and concurrent design synergy", *International Journal of Mechatronics and Manufacturing Systems (IJMMS)*, vol.1, no.1, pp. 264–269, 2008.

[HOL 90] HOLLINS B., PUGH B., *Successful Product Design*, Butterworths, London, UK, pp. 1–167, 1990.

[HUB 88] HUBKA W., EDER W.E., *Theory of Technical Systems*, Springer-Verlag, Berlin, Germany, p. 275, 1988.

[KOH 89] KOHONEN T., *Self-Organizing Map*, 3rd ed., Springer-Verlag, Berlin, Heidelberg/New York, NY, 1989.

[LIN 01] LIN Y., ZHANG W.J., WATSON L.G., "On proposal of function-behavior-State framework as refinement of EID (ecological interface design)", in ARAI E. (ed.), *Human Friendly Mechatronics*, Elsevier Publication, Tokyo, Japan, pp. 61–71, 2001.

[PIL] PILLAI B., "Adaptation of neural network and fuzzy logic for the wet-end-control and process management of paper or board achiness: a tool making approach", *Actia Polytechnica Scandinavica Publication*, Mechanical Engineering Series No. 138, Finland.

[PRI 98] PRICKETT S.E., MAVROVOUNIOTIS M.L., "Construction of complex reaction systems, part 1–3", *Computational Chemical Engineering*, p. 15, 1998.

[REI 91] REINERTSEN D.G., SMITH P.G., "The strategist's role in shortening product development", *Journal of Business Strategy*, vol. 4, pp. 18–22, 1991.

[SAL 94] SALMINEN V., VERHO A., "Systematic shortening of the mechatronic product development cycle", *The International Conference on Machine Automation, Mechatronics Spells Profitability*, Tampere, Finland, 15–18 February 1994.

[SAL 03] SALMINEN V., PILLAI B., "Methodology on product lifecycle challenge management for virtual enterprises", *4th IFIP Working Conference on Virtual Enterprises, PRO-VE'03*, Lugano, Switzerland, 2003.

[SAL 00] SALMINEN V., PILLAI B., "Non-linear dynamics of interacting mechanisms in distributed product development", *International Conference on Complex Systems, ICCS 2000,* Nashua, NH, 2000.

[SMI 91] SMITH P.G., REINERTSEN D.G., *Developing Products Half the Time*, Van Nostrand, New York, p. 287, 1991.

[TAN 00] TANG V., SALMINEN V., PILLAI B., "Extending DSM to very complex systems. Design structure matrix", *World-Wide Conference*, Massachusetts Institute of Technology, Cambridge, Boston, MA, 2000.

[TAN 01] TANG V., SALMINEN V., "Towards a theory of complicatedness: framework for complex systems analysis and design", *13th International Conference on Engineering Design*, *ICED 2001*, Glasgow, Scotland, UK, 21–23 August 2001.

[TWI 80] TWISS B.C., *Managing Technological Innovation*, Longman, London, UK, p. 234, 1980.