Accepted to the First World Conference on Integrated Design and Process Technology, Austin Texas, 1995

THE DIALECTICAL APPROACH TO SYSTEM DESIGN

Lars Taxén
System Design Process Dept.
Ellemtel
S-12525 Stockholm, Sweden
e-mail: Lars.Taxen@eua.ericsson.se

ABSTRACT

This paper describes an approach to system design called the dialectical. In this approach, the design and its environment, including the people involved, are assumed to form an inseparable whole. The whole and its parts are subject to interaction and change throughout the entire design process. Using this approach, a number of aspects of system design are covered, and a model for design processes is presented. An example from the design of telecom equipments using this model is given.

INTRODUCTION

Telecom equipments, such as systems for mobile communication or fibre optical transmission, can be characterized as complex systems:

- Their environment is rapidly changing; the market situation is very dynamic, and new technologies and methods appear for each new generation.
- Their parts are heterogeneous: digital, analog, software, mechanical etc.

The design of systems of this kind shows some characteristic features:

- Sophisticated design support is needed; tools, methods and information handling.
- Skilled designers with domain knowledge are required; teams from different disciplines must coordinate their efforts.
- Parts become interrelated across design disciplines; software design may for example influence electromagnetic emission from the hardware.

To some extent, this situation is due to the fact that not enough effort is spend on making good, decoupled designs. Even so, the complexity of the design situation makes it necessary to review how the design work is carried out. Today, most existing process models are geared towards a specific type of design, such as software or hardware. Furthermore, they are usually based on a certain methodological principle throughout such as the waterfall or spiral model (Carmel and Becker, 1995). These inherent features make them less suitable for usage in dynamic, interacting and heterogeneous development contexts. The process models must either be adapted to purposes they were not intended for, or interaction aspects between domains must be neglected since there is usually no systematic way to combine the models. The traditional split in the telecom area between software- and hardware design disciplines, is a striking example of the latter.

This state of affairs has been found inadequate for the design of complex systems, partly because it relies on the assumption that only the system is affected by the design process. It is more or less taken for granted that the environment of the design is well defined and stable, although real experiences indicate the opposite.

The dialectical approach is an attempt to grasp the entirety of the real design situation. The term “dialectical” has been used by a number of philosophers such as Platon, Aristoteles, Kant, Hegel and Marx. It expresses a view on reality based on contradictions and their resolutions as the driving force for progress.

In this paper, some basic thoughts behind the dialectical approach are described. We discuss some aspects of system design from this point of view, including implications for the social organization of the design work. A model for design processes based on the dialectical approach is presented. The key feature of this model is the *process module*, a package containing all the support needed to perform a certain design task. We also give an example of system design using this process model.

BASIC DIALECTICAL THINKING

The dominant way of thinking about systems can be traced back to Cartesian reductionism, which can be characterized by the following:

- High level phenomena can be explained by reduction to low level, “atomic,” phenomena.
- There is a natural set of parts which make up the whole.
- The parts are homogeneous within themselves.

- Causes are properties of subjects, and separated from effects, which are properties of objects.
- Higher dimensional objects are in some way "composed" of its lower dimensional projections.

This approach has been very successful in many areas, but it fails to cope with some fundamental problems associated with complex systems. The main drawbacks are that it overlooks the dynamic effects of interacting phenomena, and that emergent properties on higher levels of organization are reduced to lover level phenomena.

The dialectical development approach tries to capture the whole richness of a complex, multidimensional and multilevel phenomena interacting with its environment. It takes its starting point in the changes and interactions that the system and its environment undergoes during the process of design. Some characteristic features are:

- The properties of parts are acquired by being parts of a particular whole. These properties come into existence in the interactions that make the whole[1].
- A system is considered to be a relation of heterogeneous parts that have no prior existence as parts.
- Cause and effect are interchangeable, subject and object may change.
- Change is characteristic of all systems and all aspects of a system.

Thus, no part of the system or the environment can be considered fixed until the design is completed. The environment and the system influence each other and must be modelled together. Furthermore, since parts are assumed to be internally heterogeneous at every level, the "correct" division of the whole into parts varies, depending on which particular aspect of the whole is in focus.

The system and its environment, including the people involved, are assumed to form an inseparable whole in this approach. Design cannot be reduced to the realm of methods and tools. It is rather a social process which transforms system, environment, organization and people. This is of course recognized by the design community, but usually in a fragmented and unstructured way.Thus, the dialectical approach reflects a different frame of mind, rather than brand new insights.

DIALECTICAL ASPECTS OF SYSTEM DESIGN

When designing a system, we often tend to focus on the system and neglect its environment. This may have severe consequences. We may rely on a allegedly stable specification which is actually changing, or we may miss the effect that the system has on its environment. The dialectical approach is to assume that the model of the system and the model of the environment are determined by the interaction between them. We can model the system alone only if the environment is stable.

Let's define some terms:

- *context*; represents all phenomena relevant to the particular system design situation. These phenomena influence and become influenced by the system and its characteristics.
- *context horizon*; a conceived boundary separating relevant phenomena from those not relevant.
- *item in focus, I_F*; the system we want to design.
- *item in context, I_C*; those phenomena that are visible above the context horizon and interacts with the I_F.
- *association*; represents the interaction between an I_C and I_F.

Let's take an example, the design of an ATM (Asynchronous Transfer Mode) switch shown in Figure 1. Clearly, it is a matter of judgement to decide the context horizon for an I_F. What items are important enough to be considered in the context? For example, should previous switches influence the development of the ATM-switch? What about the control processor? Other items are obviously less relevant, e.g. the technology developed during the World War II.

All phenomena within the context horizon will influence the model of the ATM-switch and become more or less influenced by it. Now, if we put another phenomenon in focus, say for example the broad-band application, other phenomena will appear above the context horizon and others will disappear below it. One I_C will most certainly be the ISDN user, and the previous I_F, the ATM-switch, will now become an I_C. Thus a web of phenomena and interactions are formed, where the focus may shift constantly during the design process.

Needs - Specifications

A major cause of development failures in large software projects is the inability to meet requirements[2]. The standard approach is to negotiate the requirements with the customer and fix them as early as possible in the process. However, in practice it is rare that the customer can state his needs in a distinct way before the system becomes tangible.

The dialectical approach would be to regard the requirements as an association between the needs and the specifications of the system. This emphasizes the view that the requirements cannot be determined before the interaction between needs and specification of the system is stable. In general, this is not the case before the architecture of

1. As a simple example, consider an aeroplane, where not only the whole (the aeroplane) but also the parts of that whole (e.g. the engines and the pilots) are given the ability to fly through the social organization that makes it possible to design and manufacture that aeroplane.

2.An illustrative example is given by Sheldon (1992), where requirement translation and interpretation accounted for 36% of the defects on a large software project, whereas the logic design accounted for 28% of the defects.

the system is set (Jirotka and Goguen, 1994). A good example of this approach is given by Carmel and Becker (1995), in which an iterative loop involving customers and developers take place before the specifications are frozen.

Function - Characteristics

The concept of "function" is usually associated with objects. One says, for example, that the function of a saw is to cut logs. In the dialectical approach however, a function is considered to be an association between an object and a subject to achieve a desired result. The characteristics of an object and the intentions of a subject interact to give new properties to both[3]. Thus, the function of an artifact cannot be an inherent property of that artifact, but can only be defined in a context. This is however not very obvious in a well established design discipline, where the functions are more or less obvious.

Information Modelling

By information modelling, we mean the way we define and structure the information produced and acted upon. The dialectical approach will put some specific requirements on this:

- It must be possible to define contexts dynamically. Any phenomenon may in principle become an I_F. New phenomena may become I_C's as the context horizon changes, and new contexts may be defined as the need for them become evident.
- The structuring must be dynamic to cope with changes in either the system or the environment. It is fruitless to try to foresee all needs in advance. Static structures are not possible to use, and ordinary "composed of" structures are necessary but not sufficient.
- A model item can take part in different structures and have different meaning in these structures. A specification may, for example, be a prescription in one context and a presumption in another.
- The associations must be easy to model. This will probably mean some kind of extension to a hyperlink oriented mechanism.

One example of an information model along these lines, is the Specification Based Data Model reported by Gandhi and Robertson (1995).

Modules and Reuse

By reuse, we mean that the same part is used in different contexts[4]. If these parts can be configured to systems without additional design, we call them modules.

If we think about a reusable part as an I_F, then the concept of reuse can be interpreted as finding the contexts over which that I_F is invariant. One dialectical aspect of this would be to consider changing the environment in order to keep the invariance of the I_F. In other words, we would look for situations where it makes sense to design the environment rather than the system.

As has been pointed out earlier, one dialectical principle is that the division of a whole into parts depend on the particular aspect of the whole that is in question. Modules define the division of the systems that can be configured from them. It is then important to know the contexts of those systems to find the proper modules. If, for example, the contexts do not change, there is no need for a modular approach at all. The system should then be designed as a monolith to optimize its properties.

Design subjects - Design objects

There are many examples of well-intended process models that never come into use. One reason is that the difficulties to introduce and get acceptance for new ways of working are grossly underestimated. We must take into consideration the needs, desires, conflicts, hopes and fears of the people involved, and we must understand how they act and become acted upon in their work.

In the dialectical thinking, this realm is characterized by a concept called "praxis". It stands for the general social activity in which people come together to produce things. Subjects are acting on objects to bring about a desired result. In a well established design community, the designers and their instrument — the design process — are subjects acting on the design object. Together they form a totality.

At some point however, a contradiction between the motivation and the achievement become apparent. New technologies may appear, old design paradigms may be outdated or new market conditions may emerge[5].

This leads to a refusal to categorize things the usual way. New solutions are looked for. The totality between subject and object is challenged and broken up, or to state it in dialectical terms, "one is divided into two". This opens up a process where new knowledge is produced. Finally, this new knowledge is assimilated as new insights and understandings, and a new totality is formed.

Thus, designers are both the subjects and the objects of the design praxis. This is important to recognize if we are to make sure that new design process models actually come into use.

3.Take for example a torch, which is lit by somebody. The function may be to light up the dark, to heat a room or to inaugurate the olympic games.

4.One obvious aspect of reuse is the reuse of designer knowledge, but we will not discuss that here.

5.One good example is the evolving field of hardware/software codesign and computer-aided software/hardware engineering, which is considered to be the single area where the biggest improvements of lead-times and quality of electronic systems can be made (Rozenblit and Buchenrieder, editors, 1995). Today, cultural and organizational problems are barring the way for codesign, not technical difficulties.

A MODULAR DESIGN PROCESS MODEL

In this chapter, we will discuss a process model which has been inspired from findings in several areas. One such area is the evolution of living systems (Gel-Mann, 1994). It seems that complex systems, which are capable of adapting their behaviour to a dynamic environment, can only evolve if they are subject to certain restrictions between chaos and complete order.

Another area is the development of stochastic, adaptive optimization algorithms (Kjellström and Taxén, 1981), where the efficiency of the algorithm is maximized, also between chaos and complete order.

Thus, if we want a process model which can be adaptable to a dynamic design environment, it makes sense to try to organize that model in accordance with these findings. Furthermore, this model should adhere to the dialectical approach described earlier. The resulting process organization is called the *process architecture*, which will be described next.

The Process Core

The first step in defining the process architecture is the observation that a process defines a certain degree of control of the design work. One extreme is no control at all, which obviously leads to chaos. The other extreme is when every activity is controlled, which might be characterized as plan economy. At both extremes progress is slow, if any, which is illustrated in Figure 2.

Thus, there is obviously an optimal level of control which maximizes the efficiency of the organization and the creativity of the individual. This level is defined by a set of restrictions, applicable to the entire organization. Examples of such restrictions are basic information models, naming conventions, principles for quality assurance etc.

To find the optimum level of control, it is necessary to define the context within which the process is relevant. Should the process be used for one specific type of designs, or should it be valid across a broader spectrum of designs? Once this has been decided, the restrictions can be established. They should be determined by the experiences found in the design praxis.

There is a need to establish a location where to manifest this. We call it the *process core*, which is the first item in the process architecture.

Process Modules

Another observation is that there are certain design tasks which are performed repeatedly within a project or across projects. In other words, they are invariant over a number of contexts. Examples of such tasks are the design and manufacturing of printed circuit boards or integrated circuits. The support for performing these tasks can be contained in packages consisting of an information interface including information structure translation, tools, methods, quality assurance, control flow for the information refinement and application support. We call these packages *process modules*, which is the second process architecture item. In order to find the optimum module division, it is necessary to define the contexts where the modules are to be used. Furthermore, to make sure that the modules are configurable, they must adhere to the restrictions in the process core.

Status Points

In order to monitor the project and assure the quality, an association called Status Point is defined between the process- and information models (see Figure 4). From the information's point of view, a Status Point specifies a set of information elements and a corresponding set of status values, which shall be reached at some point during the process execution. From the process' point of view, a Status Point is related to the process module where the last prescribed state is reached. The assignment of the status is done by an inspection team, which is part of the overall quality assurance strategy. In order to monitor the project, the Status Points are linked to the project process when the project is established.

Applied Processes

The final item in the process architecture is tailor made processes defined for each particular type of system design. For example, an application consisting mainly of switch control software may need quite a different process from a hardware oriented line access product. This is achieved by configuring the relevant process modules through a control structure, which is implemented as a network of Status Points. This network expresses the growth of the information through the project.

Organization

To achieve a maximum of efficiency, an organization should be structured according to the architecture of the product it is handling. Following the dialectical approach, the organizational structure should be competence centres with teams organized around the process modules. Each team is responsible for the complete module, and may be invoked as needed in different design contexts implemented as applied processes. Thus, the organizational structure will be market oriented where each competence centre provides a well defined service to whomever needs it. This is summarized in Figure 3.

Advantages

The process model defined by this architecture has a number of advantages over traditional process models:

- It enables a fast set-up of application specific processes. This can be done by local design support in close interaction with the designers, which promotes user involvement.
- The customers of the process will get cheaper products. Only those modules that are needed have to be delivered to a design site.
- It is easier to keep up with the technology evolution, since the release of new methodologies, tools and information models can be confined to modules, rather than to the entire process.

- Development, maintenance and application support of a process module can be handled by a team, which promotes team productivity and encourages a non-hierarchical business organization.
- The overall quality is improved, since modules can be independently tested before delivery.
- Modules can be reused and exchanged within the organization.
- Module development can be done uncoordinated as long as the principles and design restrictions defined in the process core are followed.

AN EXAMPLE

A number of process modules have been developed at Ellemtel. These cover the entire design spectrum, including validation of customer specifications, software design, hardware design and initial manufacturing of volume products. These modules are supported both by tools developed in-house and purchased support systems.

To test the process concept, a pilot project designing a Multi-Chip Module (MCM) has been carried out. Four process modules were developed and configured into an applied process for this purpose; Design, Production Planning & Setup, Test Planning & Preparation and Initial Production.The design was done in-house, whereas the production and test preparations was done in close cooperation with the manufacturer. The initial production was done at the manufacturer.

The development of the process modules themselves was done by a design team, which had no previous acquaintance with the process concept. The experience was quite clearly that team building was promoted, and process knowledge and thinking encouraged. The team had a solid experience in designing MCM circuits, which assured the quality of the process modules. The development was done without coordination with other groups. Thus, the modular approach was working as intended. Other experiences gained was:

- It was possible to handle the multitude of information associated with the MCM design in a disciplined way.
- Several design errors was found by the inspection teams using the quality assurance concept.
- Cooperation with the manufacturer was facilitated. A common understanding was built around the process model.
- Several improvements of the process model was suggested by the team.
- Money was saved.

CONCLUSION

The dialectical approach to the design of complex systems makes it possible to find new solutions to problems encountered in this area. Some examples are:

- By emphasizing the dynamic interaction between designs and their environments, an adaptable process model has been defined.
- By emphasizing the totality between a design and its context, a modular process model and an flexible information structure has been described.
- The concept of praxis makes it easier to understand how new processes are incorporated in the design community.
- Emphasizing the mutual influence of requirements and design makes it is easier to ensure that the requirements will be in accordance with the design.

Several process modules and applied processes have been implemented using this approach. These have been validated in actual telecom design work with good results. The process model is now being developed further in close cooperation with the designer community.

REFERENCES

Sheldon, F.T., 1992, “Reliability measurement: from theory to practice,” IEEE Software, vol. 9, No 4, pp 13 - 20.

Jirotka, and Goguen, 1994, “Requirement Engineering,” Academic Press.

Carmel, E. and Becker, S., 1995, “A Process Model for Packaged Software Development,” *IEEE Tr. on Engineering Management*, vol. 42, No 1, pp 50 - 61.

Gandhi, and Robertson, 1995,“SDBM as a Model for Codesign Data” in “Computer Aided Software/Hardware Engineering,” IEEE Press, Piscataway.

J. Rozenblit, and K. Buchenrieder, eds.,1995, “Computer Aided Software/Hardware Engineering,” IEEE Press, Piscataway.

Gel-Mann, M., 1994: “The Quark and the Jaguar,”.

Kjellström, G., and Taxén, L., 1981, “Stochastic Optimization in System Design,” *IEEE Circuits and Systems,* vol. CAS-28, no7, pp 702-715.

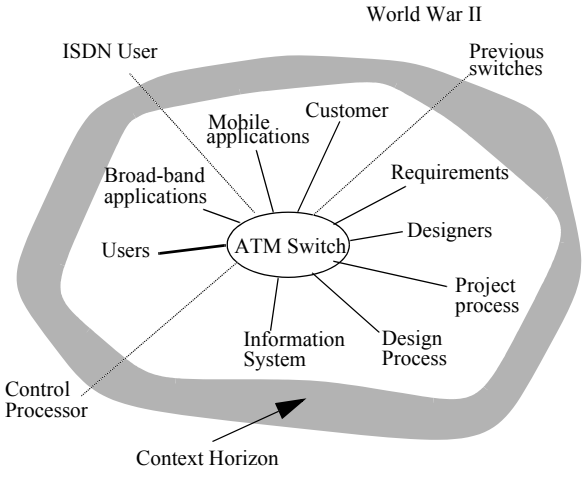

Figure 1 The system - environment interaction

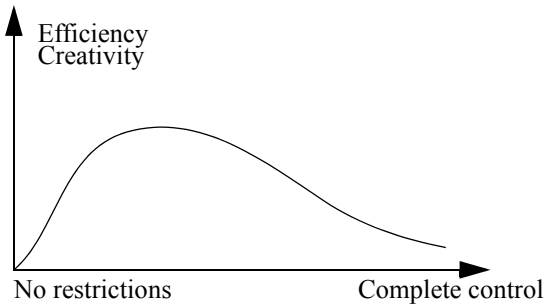

Figure 2 Efficiency as a function of process control

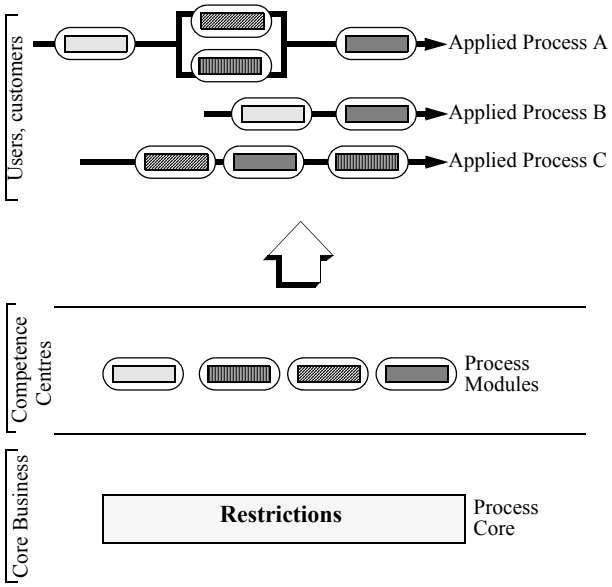

Figure 3 The organizational picture

Legend

The horizontal lines represent data used or refined in the process module.

The refinement is done in the activities **(1)**. Downward arrows indicate input/used data, and upward arrows output/refined data. In most cases, the activities are supported by CAD-tools **(2)**. The data refinement network **(3)** shows the logical dependencies between status values of the data. The Status Points **(4)** are quality assurance points, where a specified set of data elements are checked at specified points in the process.

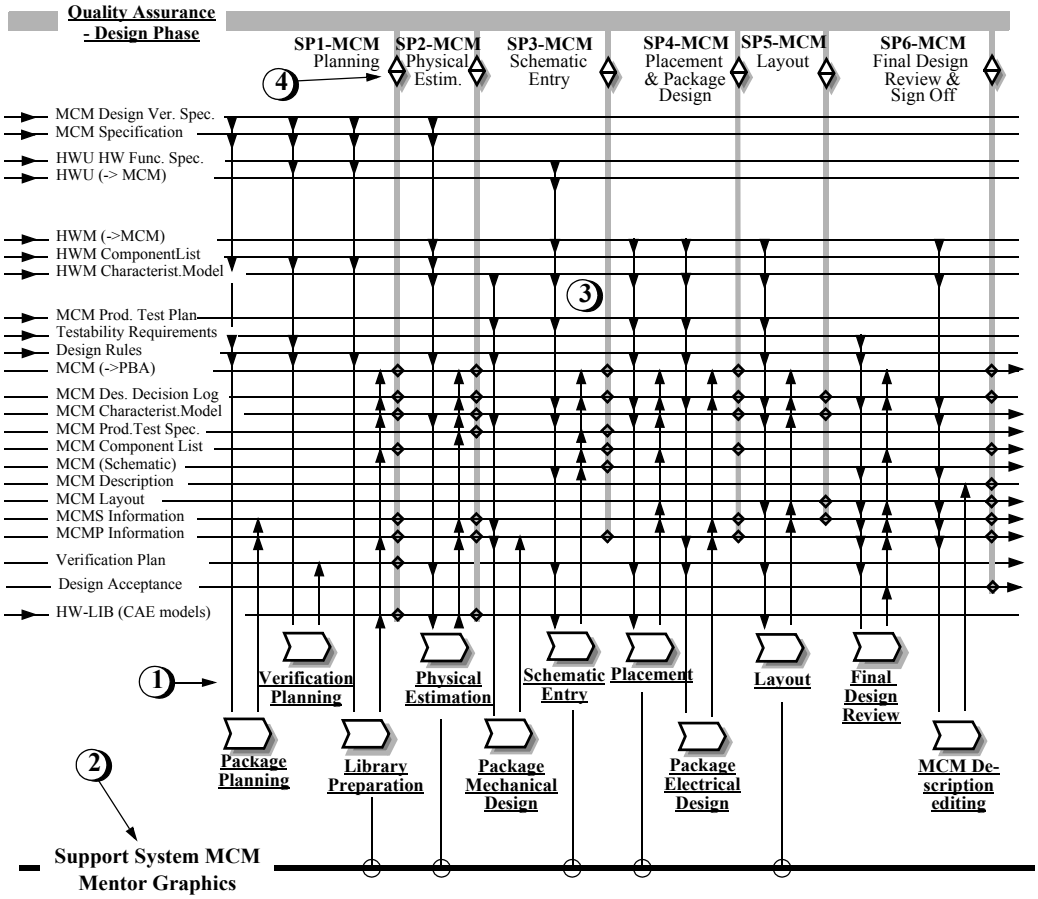

Figure 4 Information refinement flow of a Process Module for MCM Design