8

Design Management

Understanding the roles and interactions of different agents in design should be of interest to those who manage design in industry. In this chapter we return to the problems of integration introduced in section 5.1. We consider the role of the design team in design management before discussing how definitive script manipulation can be used to develop an approach suited to a multi-agent scenario.

8.1 The Design Team

8.11 Development of Design Management

It is known that the design phase influences some 80% of the product price whilst the design activity itself is a small cost to that product (Helldén, 1987). It is therefore not surprising that there has been markedly more management interest in the design process in recent times. While that might be thought to be a good thing, the initial results have not been encouraging. Managers have tended to make management decisions about design without a proper understanding of the process. Particularly in the traditional industrialised countries such as UK, US and Germany, companies have not regarded designers as being in line management and so there has been a poor history of design management. One symptom has been that the introduction of computer based design processes has been hampered because managers are somewhat chary about allowing access to Company financial information. That has meant that designers could commit firms to inappropriate expenditure or not take advantage of favourable trading conditions with friendly firms. Conversely, management policy might get formulated without reference to the traditions built up by designers with customers and contractors. For example, “bread and butter” jobs that keep firms going in lean times are chopped by management as giving a poor contribution to profits.

As management has moved from keeping designers at "arm's length" to forming design teams, it has proved to have been a rather rough journey. The tendency has been to try to keep the compartmentalised approach but to reorganise to permit greater use of technology such as CAD.

Evidence of that management development is contained in a survey reported by Wilfred Miller [Miller, 1989] and concerns the implementation strategies for CAD in West Germany over the 5 years 1984-89. He reports that 40% of medium sized companies surveyed who used CAD had no clearly defined management of CAD implementation. The consequences in those cases were that CAD managers were not suited to their tasks, CAD was not being integrated with manufacture, there were sharp divisions between CAD and manual methods, and there was very little training. In the remaining 60% of the survey there were interdepartmental groups set up by senior managers to implement CAD. Significantly however the managerial emphasis was computer orientated rather than process or design orientated. As a result there was no proper authority structure to carry out a co-ordinated policy on CAD. Although CAD was successfully introduced, with better drawings and reduced iteration between design and manufacture, there remained conflict, particularly at the interface between design and manufacture. In only a relatively few cases were companies appointing managers with engineering design qualifications who had high responsibility for integration, particularly at that interface between CAD and CAM.

The problems of implementation of CAD highlight the fact that industry still suffers from Islands of Automation. Many activities in the product generation process are still having to be done by hand or by different groups of independent agents. The design-manufacture interface is a particularly acute one, as has been identified technically by many working in CADCAM but is latterly being realised by managers too.

The management difficulties identified here arise from an historical practice of specialisation, where product generation tasks are grouped such that the responsibility of one function is with one phase of many products. Each function was dealt with independently, with little interaction with other functions; each produced documents that provided the input to the next phase as a kind of “message passing”. Dealing with feedback of negative comment from other functions led to a long time-span from conception to production. That time lag becomes apparent if one compares specialisation with an alternative approach: to

integrate task responsibilities from concept to production along a single product. In the latter case the time lag is much shorter as noted by Swedish investigators [Wærn, 1986] where such "vertical" integration is the norm.

It was the attempt to get the best of both specialisation and integration that Forward or Concurrent Engineering has been developed. Design teams accept joint responsibility for a single product and different products have different teams, albeit with considerable overlap. That makes use of specialist skills whilst limiting feedback to iteration rather than criticism. "Message passing" still exists, but tends to be more constructive. For concurrency to operate each member of the team has to have up-to-date information of the state of the design. That makes for many meetings and multiple copies of documents appertaining to the design. Design management in such circumstances becomes a significant problem because of the potentially damaging prospect of copies not being in step with one another. Chris Voss and Graham Winch identified these problems with their observations on organisational links for CADCAM implementation. [Voss, Winch, 1989]. Use of a common data base, where data is accessible across functions is essential. Putting different groups of people in physical proximity also helps. However the significant feature identified by Voss in his empirical study of a motor vehicle company was the need for a co-ordinating person with an overall knowledge of the CAD system with multiple functional and integrating skills.

These observations concerning integrative methods are highly significant in connection with the ideas described in this thesis. If we can identify and co-ordinate all the agents in the design process and have them working on a single document, analogous to simultaneous working on a physical prototype, then we have the potential for true concurrent engineering. We can then address the real difficulties of concurrency, namely the identification of agents and how they interrelate. Those problems are extremely difficult and are the subject of current research programs. [Beynon, Cartwright, Joy, & Godfrey, 1993]. The following section describes the background and current progress in identifying who the agents are in a prototype design and how a multi-agent orientated definitive system might be implemented.

8.12 Agents in the Design Process

The term *agent* is used in Definitive methods in the manner described in chapter 4. However it is useful first to have an intuitive understanding of agents in the design process. In management terms an agent may be thought of as being in charge of a

task or tasks that can be carried out largely independently of other tasks in the overall process, as for example in a particular line sequence of a PERT or critical path analysis chart. At the top level an agent is a person with functions normally found in industry. At that level the members of a design team are the agents. For example, we can group the functions in the design of a mechanical engineering product into agent areas as follows.

Market research
Ascertaining the market demand
Forming the preliminary product or systems specification:
Defining essential and desirable functions,
performance, target price, environment, appearance, service and maintenance, product functional life, product run life.

Research and development
Examining the physical principles to be exploited and coming up with novel processes and product possibilities, or refining ideas generated in response to user enquires

M**echanical design**
Developed from the specification in consultation with R&D. Functions include
Preliminary design - establishing the "design space" relations
Feasibility studies - using tools such as a graphical sketchpad to layout ideas
General Arrangement - arranging the design layout with reference to mechanical analysis.
Materials selection with respect to parametric property requirements
Detailing - using databases with catalogue and other data of standard features and components

Mechanical analysis
Compares product requirements (the demand) with the performance of real materials and systems (the supply). Analytical tools might be:
Power analysis: power flows in the system, interactions of effort and flow systems
Geometrical modelling and properties
Finite element analysis: stress, strain, thermal, static & dynamic stiffness, vibration

Electrical and Electronic Engineering
E.g. design of drive and control systems and devices.
Links with mechanical and other hardware. Heat transfer analysis.

Systems:
Computer system hardware: input and output devices;
Software for low level support and high level control.

Industrial Engineering
Aesthetics, environmental issues, anthropometrics

Manufacturing:
Process planning: group technology families. process features, tolerances.
Manufacturing technology: process: machine tools, tooling, jig and fixture design, materials handling, scheduling
Assembly requirements - assembly sequence, robotics programming

With that diversity of independent interests, interactions can occur in uncoordinated ways, resulting in designs with particular weaknesses and strengths. The co-ordination identified by Voss needs to be done by a design manager with an understanding of the current state of interactions. The design manager would perhaps coordinate the design team interaction in the context of regular meetings to give mile-stones to the design process, to agree strategic decisions that constrain future developments in areas in the concurrent system such as

- the specification of the product,
- common requirements of team members,
- subsequent requests to modify the prototype.

The support that the computer can give the design manager in may be correlated with the extent to which representations in the design process may be successfully integrated. We have already pointed out in earlier chapters how difficult that process is with current tools. The most obvious difficulty is that of data representation. When processes are specialised we can, and frequently do have a situation where data internal to the specialisation has a representation that is peculiar to that discipline. Only in the message passing is data presented in a common format; although even then the format may be governed by the specialisation that other functions simply have to learn, rather like some English assuming that 'the French can jolly well learn English if they want to communicate with us". Thus CAD has its own internal data representation (in fact numerous different ones!). Standards such as IGES help to provide a common output but that output is not necessarily helpful to the other stages in product generation. CAD to CAM has already been identified.

The way pointed up by IGES is that it may be possible to translate ideas or data into useful formats, albeit with some loss of definition in many cases (*e.g.* translating 3D to 2D). An integrated model can operate using existing systems provided such translation takes place. For example some standard CAPP (Computer aided Process Planning) techniques use expert systems and an extensive database of proven process plans to generate suggested manufacturing plans for a given drawing using tolerances to imply particular processes and functional properties to imply materials and heat treatment. The input to such systems still needs to be in machine readable form such as IGES code. If the information is less precise, *e.g.* from a scanned machine drawing then the intervention of human skills may then be necessary.

A considerable difficulty with translation is to have two-way message passing. Translation is normally one way: the drawing imported into the Word-processor from a CAD system cannot be passed back for amendment in the light of examination in the later process. Ideally we would want to have a common data representation for an integrated system. But even then we can get into difficulties. If we seek an integrated model that should support processes that are normally peculiar to a specialist function, the representations have to serve a multiplicity of functions. It may then be that there will be problems in separating the roles of different agents.

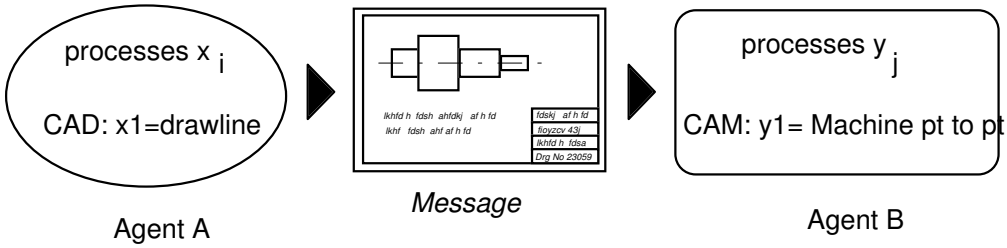

Fig. 8.1 Data Representation and message passing

In *figure 8.1* we consider data representation between two respective functions of Agent A, a designer, and Agent B, a production planner. Information peculiar to agent A (for example, process x_i where x_i might be to specify a line) would apparently not be of interest to a reader of the drawing produced as the output message since it relates to the method of producing the line on the drawing, not what that line represents. Similarly agent B, would have "private" ways of dealing with machine tool operations, part of process set y_i. In a fully integrated model we would wish support

- an intelligible interface for agent A to influence process y_j and for agent B to influence x_i
- the protocol for A to influence processes y_j and vice versa.
- For some information to be not being fully specified *e.g.* for prototyping

In the "private" method the computational approach may be regarded as largely born out of batch processing. In parallel processing there needs to be a more fundamental approach to the computational modelling; one that allows for communication, interaction and concurrent action. We need to have a computational model of the product that permits the kind of private activity that characterises much of an agent's normal work whilst also allowing access to other

agents, albeit within the constraints of the specialist's discipline. That model might allow different representations of the data such as graphical image, functional model, physical or catalogue data and properties, manufacturing process planning schedule, simulation of behaviour during manufacture or in ordinary operation.

This is the style of programming that is espoused in this thesis. Agent oriented definitive script manipulation allows the agent approach to be developed meaningfully in management terms. The definitive manager might then correspond to Voss's suggested "Design Manager". The difference would be that the function would have to include fluency in the definitive method as well as competence in the management of design decisions. That is clearly a major hurdle whilst operating at code level but in the development of the idea the system will eventually be simplified by suitable user interfaces.

8.2 An Agent Oriented Approach to Design Management

Within a single management function we can identify a multiplicity of tasks with the same pattern of specialisation and integration scenarios described for the whole process. So we reduce the scope of the problem in order to highlight how progress can be made in terms of formulating agents for sub-tasks and their interfaces with other agencies.

If we take a single function such as the mechanical design of a component, [Wærn, 1986] differentiates the tasks that have that degree of independence. He quotes the following breakdown of activities involved in design, from a study of three companies in Scandinavia.

	% of Total Time
Administration and planning	10
Retrieval of information	11
Problem solving	18
Computing	6
Drawing and changes	32
Assembling information	8
Checking	6
Other	9

Ave. time for different activities in design work, from [Wærn, 1986]

That analysis provides the basis for breaking up the design process into domains under the control of independent agents. Suppose for example we take the biggest

task, "drawing and changes". Here we may have a number of different people occupied with different aspects of the design, each with particular areas that are totally independent, but all with some commonality at their interfaces. If we return for our example to the shaft design problem promulgated in the previous chapter we can identify Agent A, responsible for analysis of the shaft design, Agent B a detailer and Agent C the manufacturer. In overall charge we have the design co-ordinator.

Now the analyst and the detailer may wish to experiment with different materials for their own reasons. The first wants a particular strength and stiffness pattern for the ideal material; the latter wants the cheapest and most readily available standard material within or close to the property range specified. All have excellent reasons for choosing their particular material and each choice has different effects on the design. Thus each needs to have the liberty to explore on the current prototype the consequences of different choices. Clearly at some points there will exist numerous scenarios, exactly as in normal design developments. The task of the design co-ordinator is now to adjudicate at those places where a decision has to be made on what constitutes the "current prototype" and what are local variants. Those choice points are the milestones of the design identified above. So we can write families of definitions that represent the different patterns of work that are currently allowed. Those constitute the constraints on the design. The constraints would not necessarily address which agent is allowed to choose the material but maybe who is responsible for a particular issue from that choice. The issue might be who determines the maximum deflection of the shaft. That makes the decision scenario demand driven rather than supply driven. Clearly the choice is ultimately one of policy, directed by the specification of the initial design.

The computational model of the current prototype can be thought of as a "virtual prototype" a term that has echoes of virtual reality and so seems apt for this purpose. The virtual prototype, or VP, begins its "life" as the script of definitions that picks up the initial specification of the product deduced from the decomposition stage outlined above. As it gets thought about in the creative ideas stage, the VP may run off in a number of directions as each agent gets to work. If ideas flow a number of possibly independent "solutions" may be suggested. In *fig 8.2* those are shown by the different agents branching off the main VP. In the real world these ideas would be recorded (or at least the less outrageous ones!) in a design folio. In Definitive modes these ideas would be represented as separate

scripts of definitions, each describing a solution in terms of the initial specification. To make sense of these scripts, each agent keeps its set separate from the principal VP until an idea becomes accepted at the milestone points. If a set is accepted then that becomes part of the Virtual Prototype to be worked on from that point on. If the design is concurrent then all agents agree to use that common VP. The variants are not discarded but remain in the design folio as alternative routes in case the current VP proves infeasible.

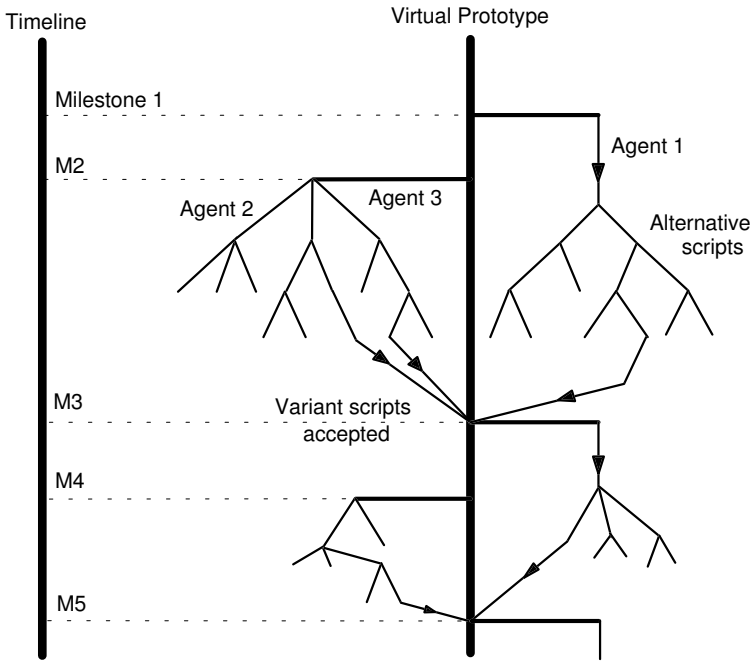

Fig. 8.2 Design of Virtual prototype by concurrent scripts

Different designs can be developed independently until sufficiently defined to make a selection. Thus each agent can have a number of design histories in a folio and argue for the one that is to become the VP for development at the milestone point. Further, the interactions of various parameters and groups of parameters become apparent as the design proceeds and it becomes necessary to group such interactions into separate scripts. Since the parameters will in general be interdependent it is necessary to identify within a script those variables that can be changed without affecting other scripts, those that can be changed only within constraints imposed by another script and finally those that would impose constraints upon variables or groups of variables in other scripts. It is here that the notion of agents needs to be developed.

With different anticipated decision patterns, families of definitions can be included or excluded from the current prototypes by simply having functions that do that by means of setting their guards on. If guard A is set then Agent Detailer would have the responsibility of providing the definition script for that area and also putting up the constraint pattern that permitted limited access for further experimentation. In that way the analyst would not suddenly discover that the material had become cream cheese! Some of these issues are discussed in [Beynon, Cartwright, Yung and Adzhiev, 1994] where the idea of virtual prototype is explored further.

The snag with recognising patterns of decisions is the risk of an explosion of alternatives. It needs careful organisation so that decision patterns are defined strictly on the basis of particular product specification. In our example if three agents have decision making access for say the material choice then, with both supply and demand scenarios we could end up with 6! = 24 choices merely to make one script part of the prototype. Unless automatically generated it becomes a serious piece of programming to anticipate all these scenarios.

The approach using Actions described in the last chapter could provide a way to structure such choices to prevent the combinatorial explosion. Decisions could be delayed or made more abstract by the generation of definitions from text. This is the subject of further work.