INTEGRATING DESIGN TOOLS INTO THE DESIGN PROCESS

Damian Mac Randal
Rutherford Appleton Laboratory
Chilton, Didcot Oxon. OX12 0QX UK
email: damian@inf.rl.ac.uk

ABSTRACT

There are two almost orthogonal aspects that need to be considered when looking at the use of design tools by the profession, the integration of the design tools "around" the product being designed and the integration of the tools "into" the process using them. In the building design field, the "Product" aspect of this has received much more attention than the "process" aspects. However, in the business arena, Process Support is the latest "hot topic" and is spawning lots of new systems. This paper looks at the current state of some of the outstanding problems in both areas, and stresses the potential benefits to be gained from a more serious consideration of the process dimension when developing an integrated design support environments.

INTRODUCTION

In common with a number of other fields, the building design profession is finding it difficult to exploit the full potential of the computer tools available to them. While there are many technical, organizational and cultural reasons for this, one of the major stumbling blocks is the difficulty in integrating the disparate design tools into the current design process. There are two separate aspects to this, the integration of the design tools "around" the product being designed and the integration of the tools "into" the process using them. While not completely orthogonal, these two aspects are normally considered independently - the product centred view providing the necessary data and data structures for the process operation and the process centred view providing the dynamic behaviour needed to achieve the process goals.

PRODUCT MODELLING ISSUES

Over the last few years, there has been a number of projects addressing the general area of design tool integration, mainly with the objective of ensuring a selection of tools can interoperate smoothly - or even seamlessly. The main trend here has been to devise a common product model to which the various design tools can relate, and provide mechanisms/interfaces to ensure a semantically consistent operations on this central model. Although this is not easy, as evidenced by the slow progress towards an AE Application Protocol for STEP, recent projects focusing on a narrower spectrum of design tools have been making encouraging progress. The main focus of attention is in ensuring the central data model is consistent, complete and efficient. However, one core problem that remains is that not all information is actually held in or communicated via the central data model. Most tools have certain data that is specific to the tool, and is effectively meaningless outside that context. Where this causes problems is when that data creates, or even just implies in the mind the user, that further relationships exist in the stored data that are not explicitly presented in the data model. The user then "carries" (or doesn't carry) this information from tool to tool, possibly subverting any semantic checking on the exchanged data. This suggests that more attention should be paid to the users mental models(s) when operating the tools and the data models should be designed to be symbiotic with the users mental model, rather than attempt to completely pre-define inter-tool information exchanges.

The next phase in this tool integration approach is to ensure that the tools themselves co-operate as well as inter-operate. This requires taking control of the tool use, for example by controlling the sequence in which the tools are used. The immediate objective is to ensure the tools do not interfere with each other, either directly by overwriting data or indirectly by making changes to the data model that invalidates input to or output from another tool. However, in many cases, the tools are actually designed to be used in a particular sequence, and it can be of value to the designer to be guided through that sequence. The problem here is that the focus is on the tools, what they can do (and not do) singly and does not allow for any synergism due to tool interactions. Furthermore, ignoring the designers purpose in using the tools risks getting in the designers way, forcing them into a pre-ordained sequence of operations. This is a throwback to the "user serves the computer" modus operandi.

A good exemplar of this tool integration category is the COMBINE system [Augenbroe 94]. In this a central Integrated Data Model has been defined and implemented in an object oriented database. This database stores the data representing the current building design and is used to "initialize" any design tools that the designer wishes to run. The tools update the central data model, either by adding new information to the current description or modifying existing data. The information stored includes the performance assessment results relating to the current building as returned by the various tools. On top of this basic central database functionality, the COMBINE system provides two further mechanisms to help address the problems raised earlier.

Firstly, tool interaction with the database is via sub-schemas pre-defined for each specific tool. By describing the data inputs and outputs of the design tool in terms of the central data model, information obtained from and returned to the database is guaranteed to be semantically consistent with that already there.

Secondly, when information in the database is updated by a design tool, any further data that has been derived from the previous value of that data is flagged as invalid. The provision of input and output schemas for each design tool allows such dependencies to be detected automatically. This helps prevent future decisions being based on out of date information.

COMBINE is more than just a data exchange system, it also provides the means to manage the use of the tools. To do this, it allows the permissible sequence(s) of design tool invocation to be specified. Thus a tool can be made available to the designer only when its use would be appropriate, and conversely, the designer can be prevented from skipping the operation of some essential tool. Combined with the above mentioned ability to check if some analysis output has been invalidated by subsequent work, it then becomes possible to "manage" the design process.

Looking at the main problems highlighted earlier, it is clear that while the pre-defined schemas can automatically "map" from the central data model to the design tool's input, they cannot supply any of the further, tool specific data which falls outside, or is incompatible with, the central data model. On the other hand, the collection of this data cannot be left completely to the design tool, as it no longer has the complete context against which to check the new data. What is required is some sort of filter between the central database and the design tool that supports the user in inputting the necessary supplementary data and acts as a monitor/advisor to the user while they are running the tool. While the COMBINE system provides assistance in writing C++ filters to carry out the mapping part of this, substantially more work is required to clearly identify the functionality and support required by real users.

The problems associated with controlling design tools use on the basis of the underlying data model and the facilities the COMBINE system provides to overcome this are more appropriately described in the next section.

While COMBINE concentrates on the use of product modelling to support only the design process, the problems identified above apply right across the design, construction, operation, decommissioning lifecycle. The ProcessBase project addresses the data management and exchange requirements for process plant design, construction and operation. Process plant, clearly has the same design stage problems as building design, only more so, since plant functionality is inherently more complex. In addition, because of the intimate relationship between designed functionality and operation, there is the major issue of the storage and management (for the very long lifetime of the plant) of the huge volume of CAD and functional data generated. In particular, this information has to be made available to the operational, refurbishment and decommissioning specialists, and the details of any plant modifications reintegrated into the original. Naturally, the STEP standard features highly in the solution to these problems, but as in COMBINE it is the data/process management aspects that require the most attention.

PROCESS MODELLING ISSUES

Any higher level of design support than the above has to tackle the design process itself. It has been found in practice that it is not enough to just to provide powerful and sophisticated tools in order to improve productivity and enhance the quality of the product. It is crucial to ensure that the tools fit seamlessly into the production process. The key to this is to provide computer based support for the process itself. But this runs headlong into a completely new set of issues, nothing to do with the tools being used, but concerned more with the design process, and in particular the management of the design process, the needs of the designer and their organization, the need for quality in the process as well as in the product, and so forth. Since these issues are exactly those being tackled in other fields, it is worth briefly looking at what other domains are doing before considering what directions should be explored in the building performance assessment domain.

Under the impetus of the latest management fad, Business Process Re-engineering, a number of non-manufacturing industries are starting to define and examine the processes that constitute their business. A business process is "a sequence of independent tasks and functions which together produce outcomes that contribute to the [business] success of an enterprise" [Scott Morton 91]. While the primary objective of BPR is to identify profitable and non-profitable processes, and distinguish core from peripheral processes, one of the spin-offs is a growing interest in computer support of the process itself, as opposed to merely supporting the process components. One common category of support is 'workflow', which views the process simply as a the flow of data/information between worksteps. This flow may be managed by an IT system, even where the data itself may still be paper-based and routed using mail/messenger services. The IBM Flowmark system [IBM 95] is an example of such a system. But, while this might be thought suitable for fairly prescribed businesses like mail-order or insurance claim handling, even there it is recognised that rigid pre-defined processes are not sufficient in a rapidly changing world. This leads to the concept of 'process support', where the IT system supports dynamic, data dependent processes that are clearly more than just the sum of their parts. In general, explicit process modelling actively encourages the users to become involved in defining/refining the process to suit the changing needs of the organization and its customers. Examples of this category of system are the ICL ProcessWise system [ICL 93] and the HICOS system [HICOS 95]. A lot of the issues these systems are tackling are found even more strongly in the building design process.

The advantages to be gained from process support fall into two main categories.

Firstly it provides the mechanisms for much better management of the processes, not least because the processes is now explicit and thus more easily communicated, designed (rather than just evolving), reasoned about and adapted to changing needs. This leads to more timely, and most likely better, management of the process, and hence to improved quality of product.

Secondly, it gives everyone involved in the process a clearer overview, both of where their work fits in to the organizations operation, but also of where and how the process has to be changed to accommodate exceptional circumstances. This allows more "empowerment" of the individual participants in the process and leads to greater productivity, while at the same time enhancing the security and accountability that are necessary to guarantee a high quality of product.

By integrating process modelling capabilities into the process support system, initial development of the processes and subsequent modifications to cope with a changing environment becomes much easier. This greater responsiveness is also the key to building design support, where the process is highly dependent on the building being designed.

However, there are several obvious caveats. Most importantly, any IT system involved in, let alone running, an organization's core business management must be completely trustworthy. For this reason, the process modelling and process support system must be based on a formalized process modelling language. It is also essential that some sort of simulation/reasoning about a proposed process is possible, or at the very least, some sort of small scale prototype testing mechanism is provided. Finally, the system must be capable of being run directly by the participants. The need to involve even in-house IT specialists would place too large a barrier between the managers/participants and the process, losing the sense of involvement, control and responsibility that generates most of the productivity and quality gains.

Turning to look at existing systems which provide process management or at least workflow support, it is clear that the field is much less mature than the data modelling described earlier. While there are lots of systems targeted at business process management appearing on the market, there is very little experience of using any of these "for real". However, some useful, if rather sweeping, generalizations can be made, provided it is recognised that most existing systems are new, closely focused on particular business issues and changing rapidly. For the purposes of this paper, existing systems can be grouped into two main categories, "Workflow" and "Process Support" to use these terms loosely

At the workflow end of the spectrum, the process is captured as a flow of data/information between worksteps. The classic way of handling this is with some sort of formalized network language, e.g. Petri-Nets. Apart from bringing a graphical presentation to the problem, with tools to capture and manipulate process models, the formal underpinning of a Petri-Net makes it possible to check for inconsistencies (such as deadlock, etc.) and to automate process control. To cope with real complexity such as is found within the building design process, various extensions to the existing process modelling languages are under active development, e.g. the development of "coloured" Petri-Nets in an attempt to represent concurrency.

However, even given a decent language, there are problems inherent in the above scenario:

- Firstly, it assumes that the process is static, i.e. that it can be pre-defined and does not change with time. (Of course, the process can handle changes, but the process itself does not change.) This allows it to be optimized for through-put, cycle time, accuracy and reuse. This is typically not the case in design.

- Secondly, as the process is purely "flow", it can only have limited "context dependency", e.g. what to do next can depend only on the data available in the network at that point. For example, after an analysis, which workstep (design function) to invoke next can only depend on data passed into the workstep. Of course, most actual systems make further "global" information available for the decision, thereby subverting the formal process description and any network validation.

- Thirdly, work-flow type processes are rigid, in that they fall apart if one component of the process unexpectedly fails (the key word here is "unexpectedly" - clearly process components can fail without the process failing if the specific failure was anticipated and a "failure path" specified). Building any reasonable component failure handling into a single network complicates the network to such an extent that it can become difficult to see the process for the error handlers!

- Finally, the (nasty) problems of concurrency (managing parallel threads that are accessing/updating the same data), consistency (checking worksteps have up-to-date information) and realism (ensuring that the model accurately reflects those parts of the process that are carried out manually) are simply avoided by assuming these are managed outside the work-flow system.

The second level of process support is provided by "process support" systems. These are generally based on some kind of event-handling metaphor and borrow heavily from the discrete event modelling systems, although they still need a formal "language" underpinning the process model. This approach gives some clear advantages over the work-flow methodology described above:

- Being reactive, they naturally facilitate "management of change", i.e. they make it easy to change the process as the environment changes - but note that the change must be managed - and permit the system to be incrementally developed. However due to the complexity of many interactions, it can

be extremely difficult to design large processes. Because of this it is essential to have a tool to assist the design process - hence the discrete event simulation background.

They are also more flexible and more sensitive to local conditions. In particular, rush jobs that occasionally (or frequently) occur can be handled within the system instead of having to be manually processed. This becomes much more important in the design process, where each design is subtly different from the one before and requires a slightly different process.

The flexibility also enables a departure from the "one way to do it, with exceptions for special circumstances" scenario to a "many ways to do it, some preferable to others" scenario. The result is to give as much autonomy as possible to the personnel actually doing the work, delegating responsibility etc. while maintaining overall management control.

- Finally, concurrency and cooperative working must be "handled", not made implicit. Of course, like any modelling system, there is no guarantee that they will be handled any better than they would be in a work-flow approach, but at least the handling is made explicit and support can be provided to encourage well designed processes.

The difference between these approaches therefore reduces to the classic "prescriptive" versus "descriptive" argument, as represented by the formal "procedural" languages (e.g. Petri-Nets) and the knowledge based "declarative" systems (e.g. cooperating intelligent agents). Clearly, what is needed is a system which enjoys the best of both worlds: the freedom and flexibility of knowledge based agents with the rigor and clarity of a formal process description language.

In the COMBINE system, there are two levels of process support. At the lower level, the process is represented using a Petri-Net, allowing simple control of the design process. In spite of the disadvantages mentioned above, this is a major step forward and permits design offices some measure of control over some parts of design process. In addition to this, part of the COMBINE project (Task 5) explored the use of a KB process support system. Starting with a Petri-Net based network, the activities (design functions) and flows (data exchange) are represented and controlled by Prolog knowledge bases. These knowledge bases control the rigidity/flexibility of the system process, the users relationship with the process and its design functions, the handling of parallelism/cooperative working, etc. At the moment, in keeping with most other process support tools, the infrastructure is more-or-less in place but there has been little experience in actually using it.

While COMBINE concentrates on the integration of the activities carried out during design, the PSS project [Platt 1994] used a process support system to look at quality management in civil engineering design. The management processes in a design office were examined and modelled using ProcessWise. These processes were found to be significantly different from those previously modelled in, for example, manufacturing or finance, as they were much more concerned with roles (defined as a set of responsibilities) and role interactions than with activities. It was found that the roles within the design office remained relatively stable from one design project to the next, whereas the specific activities carried out tended to vary quite widely in both content and sequence. Thus, by focusing on the people involved rather than the tools being used, it was possible to model the way the design office operated and provide the IT support necessary. This project, like COMBINE and many other "process oriented" projects, found that the underlying Process Support technology is still relatively immature.

CONCLUSIONS

While there has been a lot of attention paid to the Product Modelling aspects of integrated design support environments, it is only recently that the Process Modelling aspects have received appropriate consideration. This is reflected in the much more developed state-of -the-art in Product Modelling. Though there are still many problems to be tackled, a lot of the major issues are reasonably well understood and useful systems can be developed and deployed. Process Modelling, on the other hand, has really only been seriously considered by the groupware/ workflow communities and is still a fairly immature technology. Despite this, however, it is crucial for the uptake of simulation tools by the design profession that this Process aspect of design support environments is given the study it deserves.

REFERENCES

Scott Morton M. S. (1991). "The Corporation of the 1990's," Oxford University Press, New York. 1991

Augenbroe, G. (1994) "An overview of the COMBINE system," First ECPPM Conference, Dresden, Oct. 1994
[see also other COMBINE papers in that conference]

Platt, D. and D. Blockley (1994) "Process modelling in Civil Engineerind Design" Design Studies, vol15, 3, p317

ISO-TC184/SC4, "ISO-STEP 1992, Industrial Automation Systems, Product Data Representation and Exchange" Draft International Standard, 1992

IBM (1995) "An Introduction to FlowMark for OS/2" IBM Corp. Publication No.GH19-8215-01, 1995

ICL (1993) "ProcessWise, helping you manage change" ICL, 1993