INTERNATIONAL DESIGN CONFERENCE - DESIGN 2010
Dubrovnik - Croatia, May 17 - 20, 2010.

DESIGN 2010

RE-DESIGNING PD PROCESS ARCHITECTURE BY TRANSFORMING TASK NETWORK MODELS INTO SYSTEM DYNAMICS MODELS

H. N. Le, D. C. Wynn and P. J. Clarkson

Keywords: process architecture, model transformation, applied signposting model (ASM), system dynamics

1. Introduction

Planning and management plays an essential part in product development (PD). One of the criteria to judge the success of product development is its lead time. Thus, process managers are interested in exploring means to reduce the lead time, while making adequate allowance for other objectives and constraints, such as quality and resource availability. Project lead time can be influenced by characteristics of the process architecture. For instance, consider the degree of task overlapping: on the one hand, increasing overlapping can help to compress lead time; on the other hand, it is likely to create additional rework which might outweigh the positive impact [Krishnan et al. 1997]. Another example of the influence of process architecture is the frequency of design reviews. [Ha and Porteus 1995] show that it is more beneficial to conduct frequent reviews if parallel development can be facilitated or if the probability of design flaws is high. They also point out that both infrequent and too-frequent design reviews will result in negative impacts on the total development time.

Apart from process architecture-related influences, project lead time can be affected by management policies and process-inherent feedback control [Lyneis and Ford 2007]. For instance, when the project is behind schedule, process managers may take actions to get the project back on track. These actions may include hiring additional workforce or scheduling additional work to be performed in overtime. This can lead to an increase in work productivity. However, these actions entail unintended side effects that create resistance to the policy, and the intended main effect may not ultimately occur as expected [Lyneis and Ford 2007]. Due to the large number of factors which can influence process performance, and the interactions between them, it is often difficult to determine what actions should be taken to improve a complex process which can involve many hundreds of activities and actors. Managers could therefore benefit from support to evaluate the impact of different 'levers' which they can manipulate to improve process performance.

One way of achieving this is through simulation modelling, which facilitates the construction of a virtual environment or 'performance playground', allowing managers to quickly see the overall impact of different changes they could make. Various process modelling and simulation frameworks have been proposed to support this, including task network models and System Dynamics (SD) models. Each model type has its own characteristic strengths and weaknesses; for instance, task network models such as PERT/DSM are suitable for studying the architecture of a specific process, while SD models can help to evaluate the impact of policies upon process performance.

This paper presents initial results from research to combine the benefits of task network and SD models. In particular, we introduce an approach to transforming a task network model of a design process automatically into an SD model with similar behaviour. We show how this allows structural characteristics of the specific process under consideration to be represented as continuous variables

whose influence on performance can be easily studied and, potentially, optimised. The remaining of this paper proceeds as follows. Section 2 reviews task network and SD modelling frameworks, and points out their view on processes. In Section 3 we outline an integrated simulation and analysis approach which aims to incorporate some of the strengths from both frameworks. A method for model transformation, which is the first part of the integrated framework, is then presented in Section 4. Through an illustrative example, Section 5 demonstrates the feasibility and usefulness of our approach. Section 6 indicates directions for future work and Section 7 concludes.

2. Models of different abstraction levels

As outlined in the introduction, factors influencing process behaviour, and thus product development lead time, can emerge from different process context levels. Researchers have developed various simulation modelling frameworks viewing the product development process from different abstraction levels to investigate these factors. This section will discuss two such modelling frameworks: task network models and SD models.

2.1 Task network simulation models

Task network models are flow-chart style representations aiming to capture precedence/dependency relationships between tasks in a process. One such simulation framework, on which this paper is based, is the Applied Signposting Model (ASM) (see Figure 1). The ASM provides a comprehensive framework to capture different sources of uncertainties and constraints, as well as different task types, which allow explicit modelling of route selection and iteration discovery. In the ASM modelling framework the following elements are relevant to this paper [Wynn 2007]:

1. **Precedence relationships** represent the interaction between tasks and are the foundation of many task network models. They specify the order of task execution and thereby constrain which tasks can be executed at a given time.
2. **Task types and definitions** provide information on each specific task in the model. In the ASM, there are three classes of tasks: (1) simple tasks, (2) compound tasks and (3) iteration constructs. Compound tasks can have different user-defined output scenarios, determining which tasks can be executed next. Iteration constructs are a specific form of compound task and have exactly two output scenarios; they can be viewed as "backward branch" tasks, ensuring all subsequent rework is executed correctly when design errors are discovered.
3. **Task durations** are essential to determine the process lead time. They may be specified as probability density functions to express uncertainties in time requirement for task completion.
4. **Resource demands and pools** provide a detailed representation of resource limitations by allowing specific tasks to require specific resource types. These specifications co-determine the order of tasks being executed during a simulation.
5. **Task selection and resource allocation policies** influence the course of the process by specifically selecting tasks and assigning resources needed for their execution.

As a task-based network model, the ASM captures knowledge about how a process may be carried out and offers a common base for visualisation and vocabularies for discussion. It can also form the basis for process planning and management, helping to schedule activities and derive resource and time requirements. The ASM has been applied to simulate processes both in an academic context and in industry. For instance, [Kerley *et al.* 2008] describe how it has been used to support the integration of life-cycle engineering activities into an existing PD process at a UK aerospace manufacturer.

2.2 System Dynamics models of PD processes

SD models of PD processes are mostly based on variants of a generic structure which captures *flows* of activities between *stocks* that represent their current state of execution [Ford and Sterman 1998]. SD ignores the task-specific details that are included in task network models. The main generic feature of SD models in the product development literature is the rework cycle which was first introduced by [Cooper 1980]. Subsequent modellers have developed variants of this rework cycle, such as [Ford and Sterman 1998] (see Figure 1).

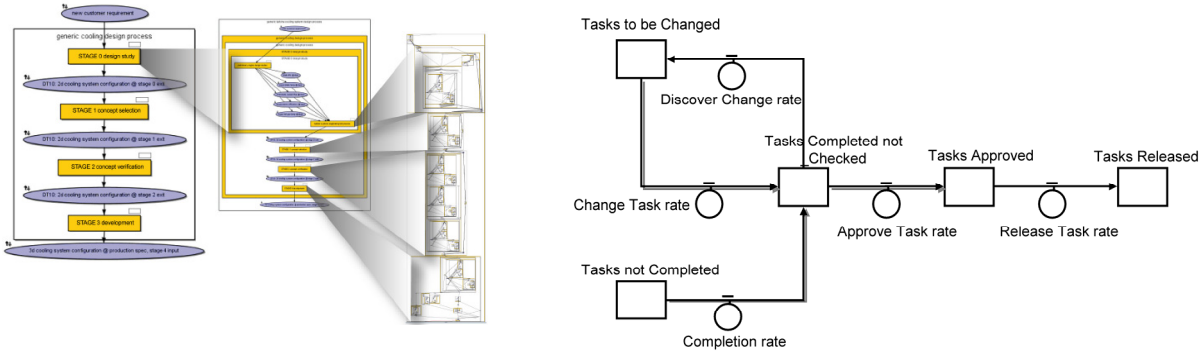

Figure 1. ASM model (left, [Bell et al. 2007]) and SD model (right, [Ford and Sterman 1998])

In overview, the rework cycle operates as follows. At the beginning, all tasks are in the stock "Tasks not Completed". They are processed through "Tasks Completed not Checked" and "Tasks Approved" until they reach "Tasks Released", where all tasks need to arrive to complete the process. However, not all tasks are processed flawlessly and thus need to be reworked once these errors are discovered. Yet rework can generate more rework and so on because errors in design are often detected some time after their generation [Lyneis and Ford 2007]. Hence, when they are amended knock-on rework may be revealed. These effects can lead to significant distortion of project schedules.

The rework cycle comprises various feedback loops which govern the rates at which activities flow between stocks. For instance, an increase in workload and overtime working can have a negative impact on the productivity of work force due to increased work intensity and limited resource. This shows one way in which policy resistance may evolve as management policies are initiated to influence the course of the project, i.e. to keep the project on track. Rework, hence, can have great influences on process behaviour and development lead time.

According to [Ford and Sterman 1998], the relevant elements required for constructing a basic SD model of an iterative PD process are:

1. **Process structure** includes *development activities* and *phase dependencies*. The three process driving development activities are (1) completion, (2) quality assurance and (3) change of tasks. The *process concurrence relationship* represents the degree of interdependency between tasks within a phase by limiting the number of activities which are available for execution at a progress state. Similarly, *phase dependencies* represent information flow constraints between a cascade of overlapping development phases, where each phase is modelled as a rework cycle which could otherwise be independently parameterised.
2. **Resources** are characterised by their *quantity, allocation* and *effectiveness*. The quantity parameter indicates the level of available resources, and the allocation policy regulates the assignment of resources to each development activity. The effectiveness parameter describes the rate at which tasks can be processed at each development activity.
3. **Scope** represents the *original scope* of work – described as the total number of tasks to be completed – and its changes over time in response to schedule, cost and quality influences. *Rework* represents the number of tasks which need rework due to error generation in the development activities.
4. **Targets** are composed of *deadline, quality goal* and *budget* specifications. These specifications are related to the overall project goals as well as phase-level targets and influence the feedback loops in the model.

Through explicitly modelling the influence of rework and policies, such as how to respond to targets and delays, the benefit of an SD model lies in its capability to capture the dynamics and complexity of real systems (for example, software [Abdel-Hamid 1996] and mid-size chips [Ford & Sterman 1998]). This modelling framework can improve high-level understanding of process behaviour and its impact on process performance.

2.3 Summary

A summary and comparison of the essential characteristics of the two modelling approaches can be found in Table 1. (summarising [Browning *et al.* 2005] and [Wynn 2007]).

Table 1. Essential characteristics of task network and SD models

	Task network models (e.g. ASM)	SD rework cycle
Viewpoint on Process	• A set of activities with pre-determined precedence relationship	• A few major phases that create amounts of work and rework for each other
Typical Purposes	• Design process analysis • Task sequencing • Graphical process network visualisation	• Process behavioural analysis • Iteration management policy definition
Key Variables/ Attributes	• Network structure • Iteration (represented using decision points and cyclic dependencies)	• Project environment • Feedback loops • Rework cycle
Strengths	• Graphical overview of process architecture • Capturing process architecture • Process scheduling	• Process behaviour analysis • Modelling influence of rework cycle and dynamic feedback structure on process behaviour
Weaknesses	• Overwhelming data volume • Little traceability of process behaviour • Difficult to study process concurrency and overlapping degree	• Little reference to actual process architecture

The strengths of task network models, such as ASM, and SD models, such as the rework cycle, lie in their capability to investigate the influence of process architecture and high-level project issues, respectively, on process behaviour. In the remainder of this paper we set out to show that combining these two modelling approaches can yield new insights into factors influencing process performance.

3. An integrated simulation and analysis framework

3.1 Overview

By combining the two modelling approaches both high-level project issues of SD models and the process details of task network models can be captured. Figure 2 outlines the main information flows in the integrated simulation and analysis framework. In this framework a *task network model*, such as an ASM model, serves as the starting point. This model is then transformed into the rework cycle of an SD model. The *process architecture* – which is made up by elements of a process (i.e. tasks) and their pattern of interaction – is the core of the transformation owing to the fact that: (1) process architecture, although represented in different forms, is the main component of any process model; and (2) process architecture is one of the main components of process planning. Once within the *System Dynamics* modelling framework, the rework cycle can be complemented with SD-typical influencing variables and feedback structures that make up the *project environment*. *Management policies* can then be specified to complete the SD model. Together with the process architecture and management policies, the project environment exerts strong influences on *process behaviour*. Modifying these factors and analysing the resulting process behaviour will lead to a better understanding of how and to what extent each factor contributes to that behaviour.

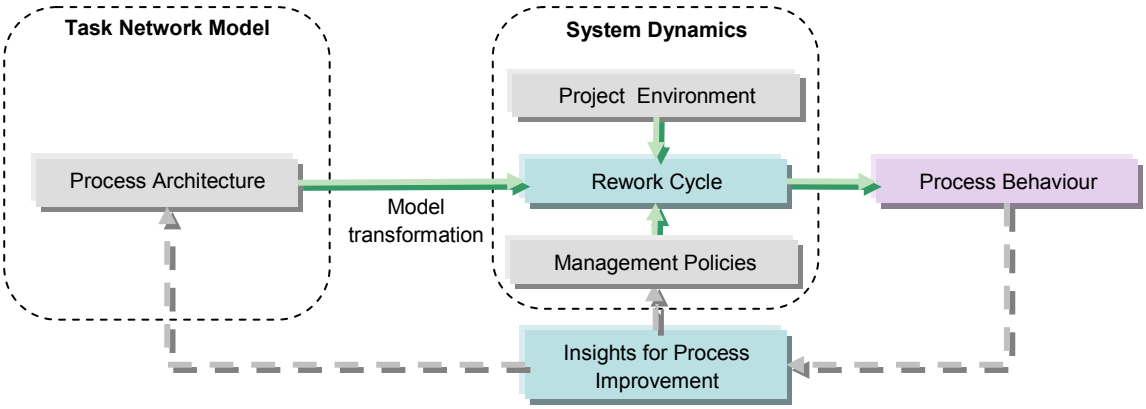

Figure 2. Integrated simulation and analysis framework

3.2 Model transformation issues

In task-based network models and SD models the process architecture evolves from the precedence and process concurrence relationships, respectively. The review of elements of both modelling approaches (as introduced in [Ford and Sterman 1998, Wynn 2007]) reveals some common interfaces which can facilitate the transformation of an ASM task network model into a high-level SD model. However, when transforming an ASM model into a SD model, neither all of the model elements nor their level of detail of each parameter can be considered. Table 2 summarises the matching of ASM framework elements with the ones in SD, and points out issues regarding the level of detail.

The course of a simulation of a process modelled using ASM depends significantly on the precedence relationships between tasks. These relationships describe interdependencies and interactions between tasks in a model. In an ASM model tasks are distinct and, hence, may have interdependencies between each other. The resulting precedence relationship regulates the availability of a task for execution during the project and thus contributes to task schedule generation. In SD, tasks are not distinct and only percentage process concurrence relationships can be used to describe the availability of tasks for execution at a given progress state. Based on the task schedule of an ASM, the process concurrence relationship of an SD model can be calculated by determining (1) the individual and (2) the cumulative percentage contributions to the overall completion scope achieved in each time step (see Section 4.2).

4. Model transformation method

The model transformation used in this paper proceeds in the following steps (see Figure 3):

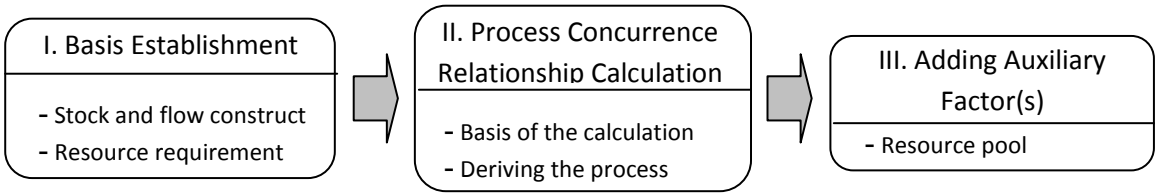

Figure 3. Steps of model transformation

4.1 Basis establishment

The basic component of both ASM and SD frameworks are tasks. However, tasks as represented in ASM and SD differ significantly from each other:

- **Semantics of tasks in each model** – Tasks in ASM have a specific description of their contribution to the design and development of a product. On the other hand, tasks in SD are small indistinguishable units of deliverables (hereafter: work packages for a clear distinction), which contribute to the project completion in terms of percentage.
- **Time and resource requirements** – Tasks in ASM are distinct and, therefore, can have individual specifications regarding time and resource requirements for execution. Work

packages in SD are fungible – they are modelled to be the same size in order to have the same average duration, though they may be processed at different rates. There is also no specification regarding resource requirements for individual work packages.

Table 2. Matching ASM and SD modelling elements

ASM elements	SD elements	Matching interpretation	Level of detail issues
Precedence relationships	Phase dependencies (Process concurrence relationship)	The precedence relationships specify which tasks can be executed at a given time and thus can serve as the basis for calculating the phase dependencies	The specific indication of what tasks can be executed next in ASM will resolve in percentages when being transformed to the associated element in SD
Task types and definition	Development activities	Unique tasks in ASM can be transformed into work packages which flow through different development activities	The unique identification of each task in ASM cannot be reflected in the SD framework. The different output scenarios of a compound task cannot be reflected in SD
	Project scope	The overall amount of effort in ASM can be used for calculating the original project scope of each phase and the whole project	-
	Rework	Information from compound tasks and iteration constructs regarding tasks to execute following an unsatisfactory outcome can be used to calculate potential rework scope	The probability of an output scenario as well as potential second-order impacts on other parts of the process in an ASM model cannot be reflected in SD
Task duration	Resource effectiveness	The uncertainty in task duration in ASM can be used to calculate the average upper and lower limit of the resource effectiveness in SD	The unique uncertainty in duration of each task cannot be reflected in the SD framework
Resource demand and pool	Resource quantity	The resource demands of tasks in ASM can be used to calculate the total required effort which is an indicator for resource demand	The distinct demands of different resource types of each task in ASM cannot be reflected in the SD framework
Task selection and resource allocation policies	Resource allocation	No transformation possible	The distinct selection of and resource allocation to tasks in ASM cannot be reflected in SD frame-work; only development activity priority policy possible in SD

Taking these descriptive differences into account, the model transformation proceeds as follows:

1. **Creation of stock and flow constructs** – Tasks and their contributions to the overall development in ASM are assigned to phases (the meaning of phase assignment is discussed in more detail through the illustrative example in Section 5). Each phase is modelled as a generic SD stock-and-flow construct, encompassing the rework cycle and SD modelling elements. The basic SD stock-and-flow construct of a phase is adapted from [Cooper 1980]. However, the Cooper rework cycle is extended with the "Rework rate" flow for rework processing since it is arguably justifiable to specify a different rate for rework compared to the initial completion rate (see Figure 4). Once the stock-and-flow constructs are created, they are individually parameterised as described below.
2. **Resource requirement** – The different types of resources in the ASM framework should be reduced to one common type for simplification. Tasks that require more than one resource unit should be converted into several tasks, equalling the original number of required

resource units but needing only one resource. These tasks are to be executed in parallel and have the same input and output, and are of the same duration as the original tasks.

3. **Determining task/work package duration** – Tasks with different durations in ASM need to be reduced to a common denominator since all work packages in SD require the same average duration. The average duration to process a task in SD is set to one time unit. The number of work packages in the SD model is calculated by setting it equal to the sum of individual task durations in the ASM model, thus ASM tasks are broken down into small units of work packages. This enables more precise calculation of the numerical process concurrence relationship (see Section 4.2) and thereby allows the SD model to have a closer imitation of the ASM model's behaviour. Furthermore, it provides the basis for a simple way to cope with tasks requiring more than one resource unit.

4.2 Determining the process concurrence relationships for each phase

The course of a modelled process in ASM depends significantly on the precedence relationship between tasks, which evolves from information flows between them. In SD, work packages are not distinct and, therefore, only a percentage process concurrence relationship can be modelled. This relationship can be seen as a 'look-up table' which regulates the percentage of work packages available for completion at a progress state.

1. **Basis of the process concurrence relationship calculation** – The order of tasks in ASM reflects the precedence relationship among tasks and can, therefore, be used to calculate process concurrence relationships in SD. Using this approach an SD model can imitate the behaviour of an ASM model, in the case where there is no uncertainty included in the ASM model that cannot be modelled in SD.
2. **Deriving the process concurrence relationship** – The precedence relationship of an ASM model can be converted into an SD process concurrence relationship by using project schedules (visualised as Gantt charts) resulting from simulations to determine (1) the individual and (2) the cumulative percentage contributions to the overall completion scope achieved in each time step. To illustrate, consider the ASM simulation output shown in Figure 4. Assuming that the project starts on Monday 12 Oct and the overall completion scope is 100%, then in this case the individual percentage contribution of Friday 16 Oct will be 3%, as outlined in the solid box (see Figure 4) and the cumulative percentage to date will be 7%. On Saturday 17 Oct another 3% of the total work packages will be available for completion, as outlined in dotted box (see Figure 4). Considering the entire schedule of tasks which results from ASM process simulation in this way allows the process concurrence relationship – which equals the ASM project progress profile – in SD to be derived automatically from an ASM simulation model.

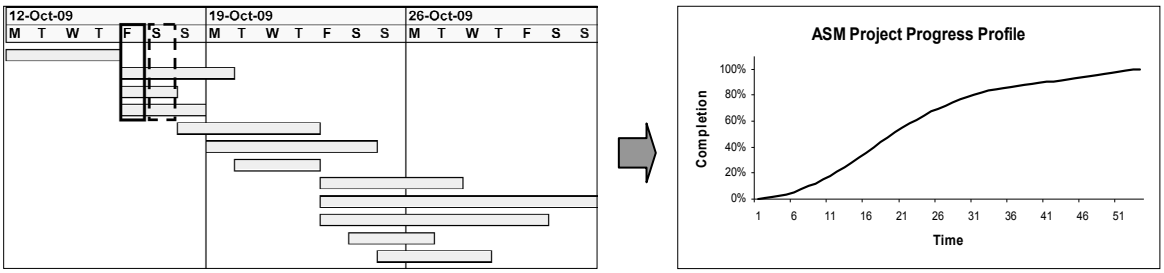

Figure 4. Calculating the project progress profile from a Gantt chart

4.3 Adding auxiliary factor(s)

1. **Resource pool** – It is assumed that only "Progress rate" and "Rework rate" need resource in order to process work packages; however, they may have different rates of processing work packages. The SD framework has to adapt the available resource amount from the ASM framework in an aggregated form. Yet the allocation policy from the ASM framework cannot be used in the SD framework.

To summarise, by generating a simplified SD model from a potentially complex task network model, the transformation method provides a way to study and explore the impact of management actions and policy changes on the dynamic behaviour of the system. Once potential improvements have been identified, by referring back to the original ASM model the modeller can consider in detail whether these actions and changes are practical and how they could be achieved.

5. Case study

In this section, an existing process model is used to illustrate the model transformation approach and highlight some of the benefits.

5.1 Building the model

The case study is based on an ASM process model originally created by researchers in the Cambridge EDC to capture the engine development process at Rolls-Royce. To create the model, 32 interviews were conducted with 27 designers, resulting in a task network of around 430 activities and the dependencies between them (for more information please see [Wynn 2007]). For the purposes of this paper, a more compact version of the model was created, containing only 177 tasks (see Figure 5) by considering only the top three hierarchical levels of the model. It was further simplified such that tasks had identical durations of one time unit. This kept the effort for the to-date manual transformation within a reasonable limit.

The model transformation method outlined in Section 4 was applied and the calculated process concurrence relationships were incorporated into: (1) a single-phase SD model construct as shown in Figure 5; and (2) a four-sequential-phase SD model, where each phase contained the tasks from consecutive sub-processes in the task network models. Vensim® was used to model and simulate the SD model.

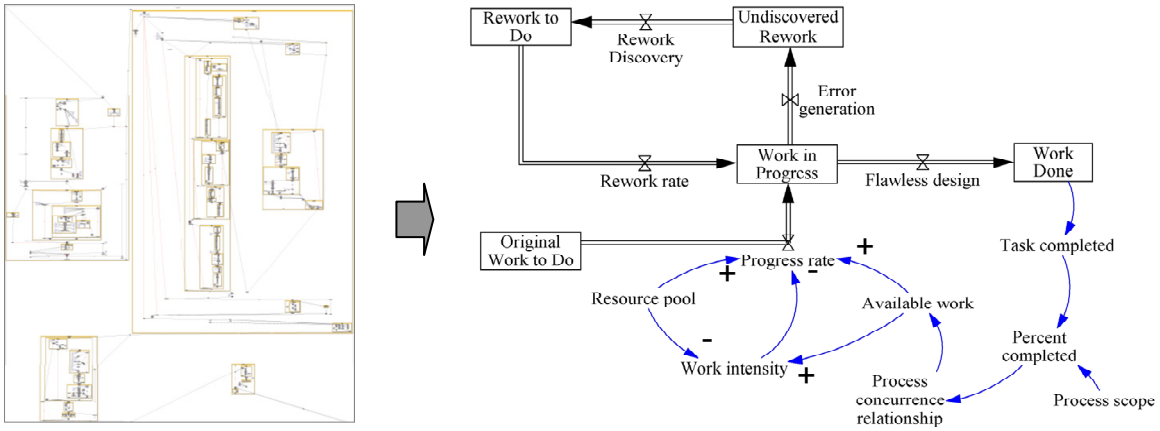

Figure 5. ASM basis model (left) and constructed SD (right)

Figure 6 depicts the project progress profile of both ASM and SD models, showing that the transformed SD models deliver a similar course of progress to the original ASM model. This verifies that the transformation is appropriate, especially for the case where the ASM model is divided into four phases and is transformed into an SD model with four rework cycles. The profile generated through ASM simulation (solid line) is nearly identical to the profile generated from the SD model with four phases (dash-dot line in Figure 6). In this first simulation run no uncertainties (such as duration uncertainty) or process constraints (such as resource limitations) are considered. By

transforming a model, the variety of different possible process schedules – and their impact on process performance – of an ASM models are lost (see Table 2). However, the transformation facilitates process behaviour related analyses which were not possible in ASM. Furthermore, the process concurrence relationship itself can be modified to some extent in order to explore ways of improving process lead time.

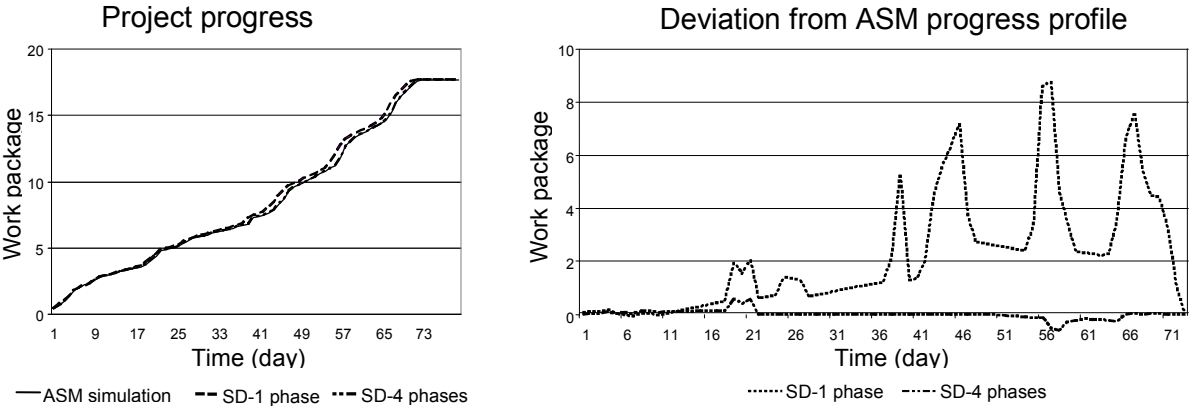

Figure 6. Comparing the accuracy of model transformations

5.2 Analysis

To illustrate the potential of the transformed model to analyse the process in terms not possible using the original task network, the one-phase SD model was modified to explore the effect of task overlapping. In task network terms, this means that all tasks are allowed to start before the final completion of their predecessors, with the assumption that this overlapping creates potential for 'churn' iteration as activities start based on early release of information from their predecessors, necessitating rework within the downstream task. The main assumptions in the SD model which represent this situation are:

1. The rework amount resulting from task overlapping is expressed by Equation 1:

   $$rework\% = c * (2overlap\% - 1) \quad (1)$$

   Where*rework%*: average percentage of rework of total task duration
   c: $0 \leq c \leq 1$, downstream task sensitivity constant
   overlap%: average percentage of task overlapping
   An exponential function implies that the later the overlapping starts, the lower the proportional rework amount will be. "2" is chosen as the base of the exponential because *rework%* would equal 100% (i.e. one) for the case where c is one and *overlap%* is 100% (i.e. one). A base of "2" thus represents the upper limit where the rework percentage does not exceed the overlap percentage and, therefore, overlapping may still be reasonable.
2. Tasks are executed in one continuous stream, i.e. there is no break and no release of resource between the first attempt and rework.
3. The starting point of a downstream task depends on the degree of overlapping and the time of completion of its upstream task (including rework time).

Furthermore, limited resource availability, which is specified to be 4 resource units, was assumed. As illustrated in the SD model in Figure 5 (right), it is assumed that "Work intensity" has a negative influence on the "Completion rate" (or work productivity). "Work intensity" is greater than "0" when "Available work" is greater than "Resource pool", meaning that there is more work available than the limited resource can process. It is further assumed that a greater "Work intensity" influences "Completion rate" more negatively. Based on this simple assumption set, simulation runs with the degree of overlapping from 15% to 50% were executed for two scenarios: (1) No resource constraints;

and (2) with resource constraints and work intensity. The four-phase SD model was also modified to explore the impact of phase overlapping. In ASM terms, this means that a number of activities within a downstream phase are allowed to start before the upstream phase is completed. The assumptions are identical to those for the single-phase model.

5.3 Results

The time-effort trade-offs for different overlapping degrees of both cases are compared to each other in Figure 7, where the percentage numbers in the graph indicate the overlapping degrees. The most obvious observation is that the resource constraint causes longer process duration. However, more interesting is the observation that in a project setting with resource constraints a higher degree of overlapping does not imply shorter process duration. In this example, process duration is reduced through higher overlapping degree until the overlapping degree reaches 25%. From 25% onwards, a higher overlapping degree brings about the opposite effect, i.e. it causes process duration to increase.
The curve of the simulation run with resource constraint and work intensity reveals some interesting insights. It shows that a higher degree of task overlapping can lead to more compressed lead time. However, this is only true until the amount of additional work – i.e. rework due to task overlapping – reaches a critical extent where the time saving benefit is outweighed by the negative impact. This negative impact results from decreasing productivity due to ever-increasing work intensity, which is caused by higher task overlapping degrees. In reality, there are other factors that have an equally relevant contribution to work productivity, either in a positive or negative way. For process managers, it is essential to understand these influences and to be able to estimate this threshold value. It can help them to derive robust project planning.
The result for the four-phase overlapping experiment is illustrated by the right-most curve in Figure 7 (45% and 50% were not considered). A similar trend to the middle curve can be observed, though the values are not identical. This may be explained by the same influences described for the middle curve.

5.4 Summary

Despite the simplicity of this example analysis, the case study shows how the integrated simulation and analysis framework can be useful for analysing multiple factors from different process abstraction levels. The example has demonstrated how process architecture characteristics – task and phase overlapping degrees – can influence process behaviour and, consequently, process lead time, and how the transformation approach allows these influences to be studied by abstracting data from a detailed model of a specific process. Furthermore, it has shown that environmental issues of the process can interact with its architecture to cause nonlinear behaviour. Capturing influencing factors from two different process abstraction levels in one single simulation and analysis framework can make use of knowledge about individual influencing factors and to provide practitioners with a more holistic exploration framework to evaluate process architecture and management policies.

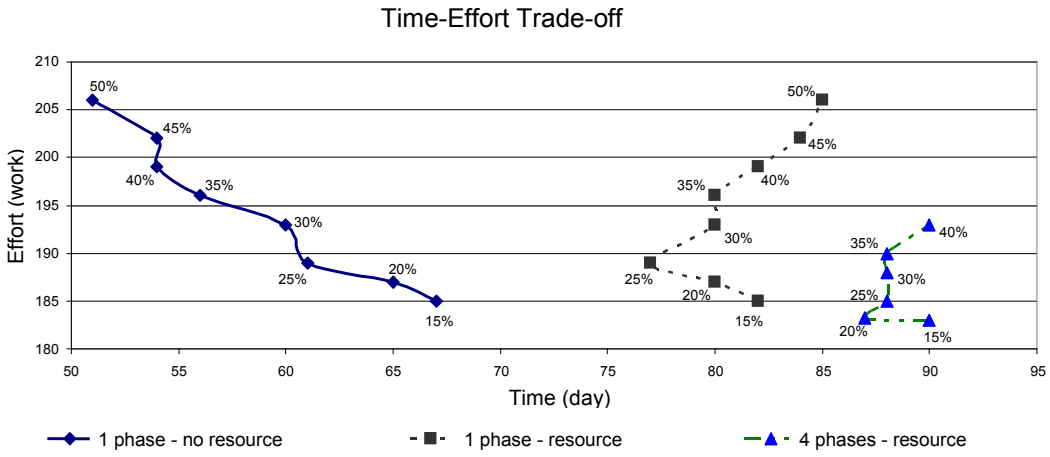

Figure 7. Comparing simulation results

6. Discussion and further work

Transforming the precedence relationships of a task network model of PD processes into the concurrence and overlapping relationships of an SD model of the same process has provided a means to analyse the influences of process architecture and management policy related influences on process behaviour. This proposal can be realised without too much effort and complexity. Furthermore, this way to transform a model does not compromise much on the traceability of the resulting model and can be used to derive improvement suggestions regarding certain characteristics of the process architecture.

However, some limitations are apparent. One of them is that only one of the many possible schedule alternatives of an ASM model is used for generating the process concurrence relationship of its SD counterpart. This may significantly limit the validity of the resulting process concurrence relationship. Further work needs to be undertaken to understand the extent of this limitation. Additional limitations of this transformation proposal are listed in Table 2 under "Level of detail issues".

Apart from these limitations, the model transformation's result itself has to be verified by measuring the correctness of the simulated process behaviour resulting from the process concurrence relationship. The deviation between the process progress profiles of the ASM and SD models can be used as a parameter to evaluate the accuracy of the transformation result.

Further work will also explore the utility and practicality of the integrated simulation and analysis framework. It is necessary to investigate how this framework can capture relevant influencing factors and present them in a context that is useful for practitioners. In particular, factors related to iteration are of interest since iteration is observed to be an essential part of process behaviour [Browning 1998]. In terms of practicality it will be explored how a computer tool may be created based on the generated method to provide a practical means for modellers to explore the possibilities arising from model transformation. Furthermore, the approach needs to incorporate uncertainties – in activity duration, for instance – in order to explore the robustness of management actions, offering more insights for process managers.

7. Conclusion

Both process architecture and high-level project issues can have significant influence on process behaviour. Many simulation modelling frameworks have been developed to help investigate the impact of different influences described on one or other of these levels of abstraction. In this paper, an integrated simulation and analysis framework was proposed to combine the detailed process-architecture view of task network models with the high-level perspective of system dynamics simulation, and hence to incorporate the strengths from both modelling paradigms. An approach for model transformation, which is the first part of the integrated framework, was then presented.

The value of this integrated simulation and analysis framework lies in its potential to provide analysis opportunities to address new questions, or to investigate existing questions from a different angle. This can result in a better understanding of influences on process behaviour, enhancing efficient process management. In particular, it offers the opportunity to optimise management policies and process architecture. Finally, the framework provides a way to simplify large and complex task-based network models and to explore their underlying process behaviour.

The approach proposed in this paper differs from other integrated modelling frameworks, such as the one proposed by [Martin and Raffo 2001], in several respects: (1) it does not integrate the two modelling approaches in order to simulate the discrete activities within the context of an environment described by an SD model; (2) it includes the rework cycle to demonstrate the effect of iteration on process behaviour; (3) it provides a clear framework for analysing the influence of feedback control and management policies on process behaviour; and (4) it allows the modelling of task overlapping.

However, there is much work remaining to further develop the proposed approach and to show how it can be applied to address relevant problems in practice, as well as to evaluate its utility to support design management. Further work in this research project will also include the development and incorporation of a comprehensive framework of iteration causes and effects into a method, which will be implemented as a tool in Cambridge Advanced Modeller (formerly P3).

Acknowledgement

This research is funded by the Engineering and Physical Sciences Research Council and the Cambridge European Trust.

References

Abdel-Hamid, T.K. The Slippery Path to Productivity Improvement. Ieee Software. 13. 1996. 43-52.

Bell, C., Wynn, D.C., Dawes, W.N. and Clarkson, J.P. Using Meta-Data to Enhance Process Simulation and Identify Improvements. Proceedings of the International Conference on Engineering Design. Paris, France, 2007.

Browning, T.R. Modeling and Analyzing Cost, Schedule, and Performance in Complex System Product Development. PhD Thesis. Massachusetts Institute of Technology. 1998.

Browning, T.R., Fricke, E. and Negele, H. Key Concepts in Modeling Product development Processes. Systems Engineering. 9. 2005. 104-28.

Cooper, K.G. Naval Ship Production: A Claim Settled and a Framework Built. The Institute of Management Science. 10. 1980.

Ford, D.N. and Sterman, J.D. Dynamic Modeling of Product Development Processes. System Dynamics Review. 14. 1998. 31-68.

Ha, A.Y. and Porteus, E.L. Optimal Timing of Reviews in Concurrent Design for Manufacturability. Management Science. 41. 1995. 1431-1447.

Kerley, W.P., Wynn, D.C., Eckert, C.M. and Clarkson, J.P. Using Simulation to Support Integration of Life-Cycle Engineering Activities into an Existing Design Process: A Case Study. Proceedings of the 10th International Design Conference - DESIGN 2008, Dubrovnik, Croatia, 2008.

Krishnan, V., Eppinger, S.D. and Whitney, D.E. A Model-Based Framework to Overlap Product Development Activities. Management Science. 43. 1997. 437-451.

Lyneis, J.M. and Ford, D.N. System Dynamics Applied to Project Management: a Survey, Assessment, and Directions for Future Research. System Dynamics Review. 23. 2007. 157-189.

Martin, R. and Raffo, D. Application of a Hybrid Process Simulation Model to a Software Development Project. Journal of Systems and Software. 59. 2001. 237-246.

Wynn, D.C. Model-Based Approaches to Support Process Improvement in Complex Product Development.PhD Thesis. Engineering Department. University of Cambridge. 2007.

Hoang Nam Le
University of Cambridge, Department of Engineering
Trumpington Street, Cambridge, CB2 1PZ, UK
Email: hnl24@cam.ac.uk
URL: http://www-edc.eng.cam.ac.uk