Research in Engineering Design manuscript No.
(will be inserted by the editor)

Improving the engineering design process by simulating iteration impact with ASM2.0

David C. Wynn · P. John Clarkson

Received: date / Accepted: date

Abstract Engineering design processes that are dependent on Computer-Aided Engineering (CAE) tools commonly involve complex patterns of tasks, including concurrency and intertwined iterations. There are often inefficiencies and improvement opportunities in such processes, but it can be difficult to identify and evaluate them—partly because of the complex iteration patterns, and partly because process knowledge is usually tacit and often distributed among process participants so that it is difficult to appreciate the causes and effects of iteration in an integrated way. The Applied Signposting Model (ASM) was developed to address these issues during more than a decade of research and case studies in the aerospace sector. This article describes an evolved version of the ASM, called ASM2.0, alongside a detailed case to demonstrate its application to improve CAE-driven, iteration-intensive engineering design processes. A review of applications is also provided. A differentiating feature of ASM2.0 is that it integrates visually-familiar flowchart mapping with quantitative analysis of process performance, to enable the modelling of complex iteration patterns that are typical of engineering design practice.

Keywords Iteration · Engineering design · Process mapping · Discrete-event simulation · Applied Signposting Model 2.0 · ASM2.0

David C. Wynn
Department of Mechanical Engineering, University of Auckland, 20 Symonds Street, Auckland 1010, New Zealand.
Tel.: +64 9 923 8178 Fax: +64 9 373 7479
E-mail: d.wynn@auckland.ac.nz

P. John Clarkson
Department of Engineering, University of Cambridge, Trumpington Street, Cambridge, CB2 1PZ, UK.

1 Introduction

Inefficiencies are common in engineering design processes due to their iterative and concurrent structures. The iteration is largely inevitable (Wynn and Eckert, 2017), but its adverse impact on process duration and effort can be mitigated—or amplified—by the process configuration. In particular, inappropriate arrangements of tasks can amplify the effects of design rework (Eppinger et al, 1994), can cause people to work with information that will later be updated (Krishnan et al, 1997), and can cause bottlenecks in which some participants are overwhelmed with urgent tasks while others find themselves starved of information, with little useful work to do (Wynn and Clarkson, 2018). Addressing process configurations that create such inefficiencies is one route to reducing development time and cost, and to releasing resource for improvement of design performance and quality. However, such process improvements require an in-depth appreciation of how iterative task structures impact the design process. This appreciation can be difficult to develop.

This article introduces a pragmatic approach to improve an engineering design process by modelling and quantitative evaluation of iterative task structures within it. The approach can help to develop insight into the drivers of design process performance and into the impact of proposed process changes. A range of applications indicate that it can be of practical value to companies wishing to develop products faster and more efficiently. A key research contribution is to show how a process modelling approach can account for the intertwined progressive and corrective iterations and partial concurrency that are typical of engineering design, providing quantitative analysis alongside a visually-familiar

and scaleable notation suitable for application to improve design processes in engineering practice.

1.1 Challenges to be addressed

The focus of the article is set on improving CAE-driven engineering design processes that are well-established, such that participants have experience of completing similar processes in the past, e.g. for the design of other, similar products, perhaps in the same product family. Because of this experience the tasks and information flows are usually known in principle—even so, such processes have characteristics that make their improvement challenging:

- *The design process involves multiple iteration modes.* It is well-appreciated in literature that engineering design is highly iterative and that the iterations are an important driver of design time and effort (Eppinger et al, 1994). Some iterations will be anticipated and may be planned, e.g. allowance might be made to repeat a definition-analysis cycle several times to improve the design. Other iterations will be unplanned and undesired—although in some cases their possibility might be foreseen. For instance, it is clear that design rework is possible following a planned design evaluation. The interplay between different iteration modes, that have different characteristics, can be difficult to appreciate in practice (Wynn and Eckert, 2017).
- *The design process involves complicating interactions.* Engineering design processes on a larger scale are complex due to the intertwined information flows, iterations, partial concurrency and multiple task outcomes that lead to many possible process routes. Even where these mechanisms are individually well-understood, the interactions mean it can be difficult to appreciate the overall significance of individual influencing factors on process performance. This raises challenges in identifying inefficiencies and judging the overall impact of proposed process changes (Eckert and Clarkson, 2005).
- *Knowledge about the design process is usually tacit and may be distributed.* Even experts who perform the tasks regularly may find it difficult to identify opportunities for improvement in their own design processes. But engineering design processes on a larger scale also typically involve tasks that are infrequently encountered and involve specialists from different disciplines, departments and even different organisations. In such contexts, although a structured process will usually exist and knowledge about that process will be available, the knowledge is tacit and distributed among a number of individuals. Consequently, it is often the case that no individual will possess both an overview of the process alongside a detailed understanding of all the tasks and information flows within it. This lack of integrated, explicit process knowledge creates further challenges to identify and evaluate improvement opportunities in the engineering design process.

These charactistics and challenges are all found in the archetypical situation that motivated this research, which is the engineering design process for a jet engine component. The design of such components is highly constrained by the demands of engineering physics and so the design processes rely on Computer-Aided Engineering (CAE) analysis such as FEA and CFD even in the early conceptual stages—as the designs progress, increasing confidence justifies the application of higher-fidelity, but more time-consuming analyses (Ahmed et al, 2003). The design objectives are typically well understood and the design considerations that must be accounted for are well known from the outset. In consequence these CAE-driven engineering design processes have established architectures of tasks and information flows, but are sufficiently complex that practitioners struggle to perceive that architecture, to appreciate where the inefficiencies are, and to prioritise process improvements (Pepe et al, 2011). This is especially the case where the processes cross disciplinary or organisational boundaries.

1.2 Premise and research questions

The work reported in this article builds on the premise that process mapping can help to generate an overview of engineering design processes of the type described above, while at the same time, the iterative behaviour is sufficiently complicated that quantitative evaluation is needed to adequately assess opportunities for process improvement. The following research questions are addressed:

- RQ1. What requirements does the context of CAE-driven engineering design place on an approach for mapping and quantitative evaluation of design processes?
- RQ2. How can these requirements be realised in a pragmatic approach?
- RQ3. How can the approach be applied to support improvement of engineering design processes in practice?

1.3 Research method

The questions stated above were considered during a series of research projects spanning more than a decade. Initially, an empirical study of jet engine component design was undertaken, which led to development of a modelling and simulation approach during the first author's PhD research. The approach was then progressively improved and expanded between about 2003 and 2013. During this period a software tool was also developed to implement the approach. The tool and approach were deployed to a limited number of users in an aerospace company, where they were used to model and improve a number of engineering design processes. They were also deployed to support a number of research projects in the university context, mainly at the University of Cambridge but also at some other institutions. These applications in industry and research generated feedback and helped to refine the approach over the years, and, as will be discussed, provide a measure of validation for the ideas.

1.4 Overview of this article

The article proceeds as follows. Section 2 sets the context, reviewing existing approaches, pinpointing the research gap and describing how an empirical study established requirements for the approach described in this paper. Section 3 details the features of the Applied Signposting Model 2.0 and explains how they meet the requirements. Section 4 discusses a deliberately-simplified but realistic case to show how ASM2.0 can be used to model and improve a CAE-intensive engineering design process, while further clarifying how the general approach to modelling and simulation differs from that in many other models. Section 5 complements this simplified example by discussing some cases where the approach has been applied to larger-scale process improvement projects in conjunction with a UK aerospace manufacturer. Section 6 reviews contributions of this article, provides some critical reflection and suggests areas for further work. Section 7 concludes.

2 Background

This section first provides a brief overview of literature on engineering design process mapping, modelling and quantitative analysis. Then, an empirical study is discussed and used to identify requirements for process modelling in industry practice. Finally, the reviewed models are evaluated against these requirements and the research gap addressed in this article is pinpointed.

2.1 Overview of related work

2.1.1 Diagrammatic task-based models of engineering design

Wynn and Clarkson (2018) write that the design process modelling approaches most commonly used in practice are based on flowchart diagramming, typically including activities connected by information flows and logic gates that determine the sequences in which tasks are attempted. One popular approach of this type is the Business Process Modelling Notation (BPMN). Kreimeyer and Lindemann (2011) describe application of the similar Event-driven Process Chain (EPC) notation to map the body-in-white process of a German automotive manufacturer. Advantages of these notations include the easy-to-interpret diagrams and the availability of robust software tools allowing large-scale models to be created. However, the need to explicitly model logic gates makes models relatively verbose even for relatively simple processes. Some commercial software tools offer discrete-event simulation based on BPMN, EPC and similar notations, but the models are not designed with engineering design in mind—in consequence they have very limited capabilities to handle the complex iteration scenarios that are common in practice.

Another established approach is IDEF0 (USAF, 1981). Applied to engineering design, a process is represented as a set of interlinked diagrams, each comprising a set of functions interconnected by labelled arrows that indicate each function's inputs and outputs, as well as controls influencing the function and mechanisms that denote resources required for execution (USAF, 1981). In the context of engineering design, Kusiak et al (1994) argues that the notation can help with perceiving constraints and considering how they might be relaxed, while Romero et al (2008) introduce additional types of information flow related to coordination and cooperation among functions in a design context. Although powerful, there are also various limitations to the IDEF0 approach with respect to this article. Because it emphasises constraints, IDEF0 does not express a sequence for attempting the tasks and consequently, does not capture how iterations unfold. IDEF0 models are also extremely verbose. To manage the visual complexity that arises from depicting four or more distinct types of information flow, the notation stipulates each diagram must be limited to seven functions. In consequence a large set of diagrams must be created to represent a process of any significant scale (Colquhoun et al, 1993).

Object-Process Methodology (OPM) (Dori, 2002) differs from the approaches discussed above in visually representing both processes (e.g. design tasks) and

the objects (e.g. design information) that they operate on, as well as various types of interrelationships within and across these two domains. The approach is based mainly on a graphical notation in which processes, objects and all the relations are drawn on the same diagram, that can be hierarchically composed. An advantage of this approach is its explicit capture of the purpose of the activities, which could, for instance, assist with ensuring process work packages are appropriately aligned to the structure of information that needs to be produced (Sharon et al, 2013). On the other hand, due to the emphasis on representing processes and objects together, mapping a given situation requires more boxes and arrows than an equivalent model in a purely process-focused approach. In common with the approaches mentioned above, OPM also does not emphasise the complex interplay between different iteration scenarios in engineering design.

Finally in this subsection, Park and Cutkosky (1999) develop the Design Roadmap (DR) approach for modelling mature engineering design processes that comprise large numbers of tasks, organised hierarchically. DR defines tasks in terms of explicit input and output information, because in comparison to representing this implicitly using arrows between tasks, "the description is more complete, the boundaries between tasks are defined, and a basis for developing interfaces between tasks is established" (Park and Cutkosky, 1999). DR, like IDEF0, also distinguishes between various types of relationships between tasks: relations that strictly constraint task sequences, relations that indicate information flow that might or might not occur, relationships that indicate other possibilities for tasks and information to affect each other. This approach offers insight into task relationships that are specific to engineering design, however, in common with other approaches reviewed in this section the focus is placed on a graphical notation and quantitative analysis capabilities are not offered.

2.1.2 Computable task-based models of engineering design

Complementing the diagrammatic approaches discussed above, researchers have also developed computable modelling approaches to provide quantitative insights into the behaviour of engineering design processes. For example, Program Evaluation and Review Technique (PERT) (Pocock, 1962) and Graphical Evaluation and Review Technique (GERT) (Pritsker, 1966; Nelson et al, 2016) and its variant Q-GERT (Taylor and Moore, 1980) have been used to analyse engineering design processes under the assumption that tasks have variable duration, and may trigger iteration probabilistically. Other researchers have applied variants of the Petri net (Van der Aalst, 1998), noting that this approach allows dynamic behaviours of serial, parallel and iterative task patterns to be modelled explicitly (McMahon and Xianyi, 1996). A shortcoming of the Petri net is that execution issues such as deadlocks and tasks being triggered out-of-sequence can appear if the net is not appropriately structured, which becomes more difficult to achieve as the process involves more iteration possibilities that are more intertwined (Wynn and Clarkson, 2018). Ha and Suh (2008) addressed this by developing template Petri Net fragments to represent specific patterns of task interactions. Larger models can then be assembled from these templates, but the modeller is required to recognise the appropriate pattern and organise their model in those terms. Also considering the correctness issue, Karniel and Reich (2011) develop an approach to create a Petri net from a Task DSM, ensuring a properly-structured net. An advantage of these approaches is the formal correctness offered by the Petri Net, however, the detail of flow logic must be directly reflected in the network structure. As a result, the graphical structure of the models can be visually complicated and more difficult to interpret than flowchart-style approaches.

Other process models are based on a Task DSM, a square matrix in which a mark in a cell indicates that the task in the row receives information from the task in the column (Eppinger et al, 1994). The Task DSM provides a different balance of capabilities than diagram-based modelling techniques. A key strength is in providing a concise visualisation of dependency structure in moderately-sized but strongly connected process decompositions, for example, in which a number of tasks run concurrently while feeding information to each other. It is also useful to concisely visualise the properties of different dependencies, if meaningful symbols and/or numbers are placed in each cell (Browning, 2016). On the other hand DSMs are less suitable if a large number of tasks need to be modelled and/or the tasks are quite sparsely connected. Especially in respect to this article, sequential and parallel flow structures are difficult to visualise (Park and Cutkosky, 1999) partly because there is no equivalent of swimlanes and partly because tasks must be implicitly placed in a sequence. The Task DSM has provided a basis for a number of design process simulation models used to examine rework and its relation to process configuration. For instance, the model of Browning and Eppinger (2002) considers the rework possibilities created when a task must be started in advance of information required from a downstream one, and shows how appropriate task sequencing can reduce the risk of ex-

tended process duration. More applications of the Task DSM are reviewed by Browning (2016).

The Signposting approach initially developed by Clarkson and Hamilton (2000) represents design tasks in terms of the confidence in design information that would be sufficient to trigger them, as well as the increased confidence the designer would be expected to gain through performing the task. The aim is to show how different routes emerge as designers structure their process in response to their changing confidence levels. O'Donovan et al (2004) use this approach as the basis of a discrete-event simulation to investigate how different process routes emerge from an ensemble of tasks that are not explicitly connected into a network. One advantage of this approach is that it accounts for the situated nature of design task sequences. Another is that it captures to some extent the distinction between progressive iterations (as increases in confidence) and corrective iterations (resulting from decreases in confidence). On the other hand, Signposting is not well suited to visualising the overall process flow and it is difficult to verify that the multitude of routes allowed in the model reflect the possibilities of the true process.

2.1.3 Other models of the engineering design process

In addition to the main families of task-based models, which are discussed above, other perspectives have also been taken on modelling and simulating the engineering design process. These include the agent based perspective that focuses on individual actors, their capabilities, knowledge and appropriate assignment to tasks (e.g. Crowder et al, 2012; Perišić et al, 2018; Hassannezhad et al, 2019); the system dynamics perspective, that emphasises influences on the development process, such as management levers and policies (e.g. Lyneis and Ford, 2007; Le et al, 2012); and the coordination perspective, which emphasises project structures and mechanisms for managing dependencies among activities, such as ensuring appropriate response to exceptions from standard work (e.g. Levitt et al, 1999; Rajapaksha et al, 2017). Finally, network-oriented models of design processes been used to study the impact of task network structural properties on certain aspects of design process performance, such as the rate of design convergence (e.g. Braha and Bar-Yam, 2007). While yielding important insights, these models do not focus on the specific sequences of tasks undertaken in practice as the current article does. For further information on other perspectives and approaches to improving the engineering design process the reader is referred to Wynn and Clarkson (2018).

2.1.4 Summary

Overall, a variety of approaches have been developed to model and analyse the engineering design process with a view towards process improvement. The approaches differ quite substantially, in part due to the different application contexts and different challenges they are intended to address.

An appreciation of process modelling challenges faced by practitioners in jet engine design was used to assess the advantages and limitations of each approach with respect to CAE-driven, iteration-intensive engineering design. This process is described in the next subsections.

2.2 Preliminary case study

An initial appreciation of industry requirements for a process mapping and evaluation approach tailored to CAE-driven engineering design was developed during a case study in which the first author spent 8 months on-site at a large UK aerospace manufacturer (Wynn et al, 2005). The overall objective of the study was to develop a method to support planning of the design process for major jet engine components. The intention was to model these design processes and computationally evaluate how they would unfold over time.

Despite the fact that the design objectives for the components were well understood, and the individual tasks being performed and design information being produced could be relatively easily identified, the design process as a whole was found challenging to model during the case study. This was because the process modelling techniques available, as reviewed in the previous subsection, were not ideally suited to describe the iterative behaviour of the process. On the one hand, the process was characterised by progressive iterations in which designers used CAE tools to trial design options and converge on solutions by iteratively adjusting a set of interlinked models of the design. On the other hand, it was possible for some tasks to reveal design problems that would lead to rework. The combination of these progressive and corrective iteration patterns could not be accurately represented in any existing approach.

Existing process maps were collected and studied to appreciate how the practitioners described their design process. These maps were constructed using office productivity tools and each map focused mainly on a single one of the engineering disciplines involved. They indicated the software packages used at different process steps, and data types produced. Most of the maps collected were organised as flowcharts, and indicated the inherently iterative nature of the process. The maps

were each targeted to a very small number of specialists and each used a different notation and nomenclature, with few explicit connections between the maps. The company also maintained a more integrated picture of the overall processes, but this was on a higher level of abstraction. The lack of a suitable modelling approach that could be used to describe the technical details of the design processes including planned and unplanned iterations, and that was scaleable, was thought to be a limiting factor in generating comprehensive and accurate process descriptions (Wynn et al, 2005).

During the case study, a process modelling approach was developed with the intention to integrate the process maps generated by different engineering disciplines into a single, coherent process model that would be recognisable to all. The process modelling approach allowed a process to be modelled hierarchically, with all required information being entered into a spreadsheet. The spreadsheet allowed a graphical process map to be automatically generated from the tabular data, and a discrete event simulation was programmed to evaluate the duration of the process (Wynn, 2007). Concurrent with the modelling approach development, an integrated process model of the component design process was developed comprising approximately 100 activities decomposed into 3 levels of subprocesses. This was a time-consuming activity because the design process involved multiple disciplines and specialists who were very knowledgeable about their own tasks, but could not fully describe each others' contributions to designing the component. Many discussions were needed with the engineers involved to synthesise a single perspective that was acceptable to everyone. This experience led to a focus on ensuring the process models created with the approach were easy to visualise and manipulate.

The approach developed and validated during this study became the basis of the Applied Signposting Model described by Wynn et al (2006). Over the following years many improvements and extensions were subsequently made to create ASM2.0, that is reported in full for the first time in the current article.

2.3 Requirements for a process modelling approach to support design process improvement

The preliminary case study discussed in Section 2.2 led to an appreciation of the requirements for a modelling approach to support the improvement of CAE-driven, iteration-intensive engineering design processes. The insights were evolved and expanded significantly through a series of applications over the following years. Here, the insights are summarised as eight modelling requirements (MRs):

- *MR1: Be visually familiar and flexible enough to replace informal design process flowcharts created using general-purpose diagramming software.* This requirement arose from the observation during the case study that the model needed to reflect the process as technical specialists saw it, in order to engage them with the modelling process and with insights obtained through analysis of the model. Experience creating the process model during the study suggested that a modelling approach should be as visually simple and familiar as possible to ensure that all process participants could interpret it quickly. This facilitates the modelling process by minimising the need to learn a new notation and hence, allowing domain experts to more easily appreciate the content of a model and provide feedback on that model more effectively.
- *MR2: Be scaleable with the aim to minimally restrict the size of models that can be created.* This arose from the observation during the case study that existing informal process maps had been structured to fit on one page in order to work around the limitations of productivity software used to create them. This required compromising on the breadth and/or depth of the process map—which is undesirable.
- *MR3: Allow quantitative evaluation of the process behaviours that arise from modelled tasks, information flows and resourcing.* Because a process model typically involves numerous tasks and information flows that interact in complex ways, the method must provide a means to determine the effects of all modelled factors on the overall performance of the process. Because it is not possible to precisely determine in advance some properties like task duration and whether or not corrective iteration will be needed, the modeller should be able to enter uncertain values where necessary. These should be propagated to indicate the uncertainty in evaluation results.
- *MR4: Provide precise control over the behaviour of the evaluation model.* This arose from the observation that applying quantitative analysis to improve a design process is procedurally very similar to the CAE work the approach is developed to represent. In particular, the day-to-day work of a CAE practitioner typically involves creating simplified models of complex situations and refining those models until they provide an appropriate level of fidelity for the task at hand. For such practitioners to be convinced by a quantitative analysis of their design process, it is necessary to provide a high degree of control so that the process model can be refined as

far as they find necessary to reflect essential features of their processes.

- *MR5: Develop an evaluation model by progressively adding depth to a descriptive model.* This requirement arose from the observation that, while the initial descriptive model in the case study was being evolved into an evaluation model, it was essential that the stakeholders could still recognise how their knowledge was represented within the model such that they could continue to offer critique and propose improvements. It was therefore determined that it should be possible to develop a computable evaluation model by adding depth to a visually-familiar descriptive model, requiring a minimum of structural change and without requiring the modeller to abstract away from the process map except where they decided to do so.
- *MR6: Facilitate modelling of progressive iteration.* Design processes in which CAE plays a significant role are characterised by progressive iterations—for instance, designers repeatedly adjust geometry in response to insights gained through analysis. As increasing confidence in the solution is gained, more time is typically dedicated to analysis tasks, more design issues may be considered and more sophisticated tools may be brought into play (Ahmed et al, 2003). To represent such processes, a means is needed to model anticipated progressive iterations in which the design is further developed on each cycle.
- *MR7: Facilitate modelling of corrective iteration.* In this context, corrective iteration refers to the situation in which flaws are discovered in the design, and some previously-attempted tasks must be reworked to correct those flaws. The effects of design rework on the process are difficult to predict because the rework often propagates, such that tasks downstream of the erroneous information must potentially all be revisited. To ensure accurate representation of corrective iteration in a quantitative process model, it is particularly important to resolve the rework properly in cases where multiple streams of work are executed concurrently and where rework loop-backs are not nested hierarchically, but can interact in complex ways depending on how and when they are triggered. This important point will be illustrated by a detailed example in forthcoming sections.
- *MR8: Facilitate appreciation of how process model features determine evaluation outcomes.* This arose again from the observation that modelling and quantitative analysis of design processes is very similar to the CAE work being modelled. Consider for example the application of Finite Element Analysis to a machine part—if the FEA produces erroneous results, diagnosing the problem requires appreciating how specific modelling decisions, such as geometric simplifications and mesh sizing, affect the result. Similar reasoning should be possible when doing quantitative analysis of a design process. For instance, if the duration of the process as revealed by the analysis is greater than the time available to complete the work, the modeller should be able to identify which tasks would need to be accelerated and/or which potential iterations eliminated to reduce the overall time.

2.4 Need for a new approach

We now move on to illustrate the need for a new process modelling approach to address the challenges of CAE-intensive engineering design. The established approaches reviewed in Section 2.1 were considered in context of the requirements MR1-MR8 stated above. Against each requirement, the approaches were each assigned a score of High, Medium, Low or None to indicate our thinking about their advantages and limitations. The result, shown in Table 1, is not intended to be comprehensive but to articulate the research gap addressed by this article.

In particular, Table 1 reveals two categories of approaches. Diagram-oriented approaches such as IDEF0, OPM and BPMN perform well against MR1, MR2 and MR5—but either do not provide quantitative analysis at all, or have severe limitations with regards to representing the iterative behaviour of engineering design processes. Hence, they do not perform well against MR3. On the other hand, analysis-oriented approaches such as the Task DSM, Petri Net and Extended Signposting provide for quantitative evaluation of iterative processes (MR3), but do not offer the flexible, scaleable diagramming needed to elicit and integrate process knowledge from process participants (MR1, MR2 and MR5). Additionally, most approaches perform poorly on MR4 (provide precise control over behaviour of the evaluation model), MR6 and MR7 (modelling the specific characteristics of both progressive and corrective iteration) and MR8 (facilitating appreciation of the relation between process structure and evaluation outcomes). These observations establish the research gap to which this article contributes.

3 Applied Signposting Model 2.0

A process modelling and analysis approach was developed to address the requirements and research gap stated

Table 1 Suitability of selected approaches to meet requirements MR1-MR8.

Approach	MR1	MR2	MR3	MR4	MR5	MR6	MR7	MR8	$\sum$
Petri Net (Karniel and Reich, 2011)	L	L	M	M	H	M	L	M	14
Extended Signposting (O'Donovan et al, 2004)	-	L	H	L	H	M	M	L	13
Task DSM (Browning and Eppinger, 2002)	L	M	H	L	H	-	M	L	13
BPMN	H	H	L	-	M	-	-	H	12
IDEF0 (USAF, 1981)	M	M	-	-	-	-	-	-	4
OPM (Dori, 2002)	M	M	-	-	-	-	-	-	4
Design Roadmap (Park and Cutkosky, 1999)	-	L	-	-	-	-	-	-	1

Key: – = requirement not supported (0 points). L = low suitability vs. requirement (1 point), M = medium suitability (2 points), H = high suitability (3 points).

above. In overview, the model is based on familiar flowchart diagramming, while providing a notation tailored to describe engineering design processes and solving commonly-encoutered scaleability issues to avoid constraining the scale and connectedness of models. To represent process behaviour and predict the impact of proposed process improvements, the approach allows modellers to closely control the behaviours of individual tasks and how those behaviours would change in different iteration scenarios, then computationally determines task sequences including the effects of corrective iteration and partial concurrency.

The approach as described here is called the Applied Signposting Model 2.0 (ASM2.0), reflecting substantial development of the model's constructs and the principles for its application since the early version that was reported some years ago in a conference paper (Wynn et al, 2006). The following sections describe ASM2.0 in full for the first time. Differences from the original ASM are discussed in Section 6.2.

3.1 Modelling notation

Prior to analysing the impact of iteration on a specific engineering design process, the ASM2.0 approach requires that the process is modelled. This involves elicitation and integration of process knowledge by developing a process map capturing iteration possibilities. Typically the map will be improved through several cycles of refinement, often involving discussion and feedback from multiple domain experts, until it is agreed to be suitable for purpose.

3.1.1 Visually familiar notation

To address MR1 (be visually familiar and flexible enough to replace informal process flowcharts), the approach is based on a deliberately-simple graphical flowchart notation comprising five types of node (Figure 1):

1. *Deliverable*, representing one or more *items of information* that are created, updated and/or considered by tasks during the process.
2. *Simple task*, representing a task that transforms a set of inputs into a set of outputs.
3. *Iteration construct*, representing a task that determines whether iteration occurs, or not. An iteration construct has several outcomes, one of which is selected when the task is completed. Forward branch outcomes trigger the next task(s) in the process. Backward branch outcomes indicate potential for iteration and, when triggered, lead to one or more previously-created information items being updated. Each of these outcomes can connect to multiple nodes in the process map, if required.
4. *Compound task*, representing a task that can have multiple possible forward branch outcomes. As shown in Figure 3, each outcome can involve a set of nodes, and is indicated as a label on the corresponding edges. Each outcome of a compound task leads forward in the process (as opposed to an iteration construct, where one or more outcomes are backward branches).
5. *Milestone*, representing a point of special interest in the process. A milestone can be included where the modeller wishes to visually denote the point of interest and/or investigate the time and state of the

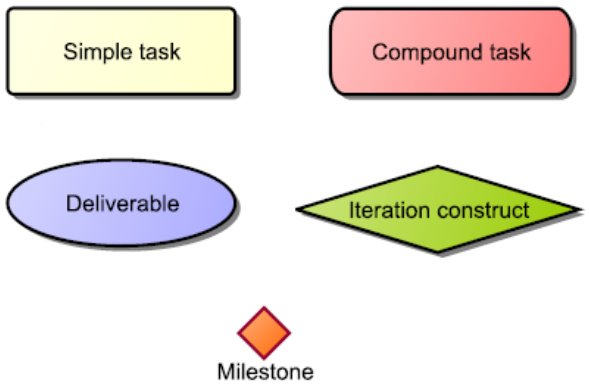

Fig. 1 Five types of node can be used in an ASM2.0 diagram.

process it which it is achieved. For instance, a milestone may be used to represent successful outcome of a design review.

Any node may be connected to any other node in an ASM2.0 model, which allows for flexibility in diagramming as required by MR1. The main constraint, required for the analysis algorithm described in forthcoming sections to operate correctly, is that the model must be acyclic if all backward branches emerging from iteration constructs are removed.

Another element of the model, that does not appear as a node on the process flow diagram but is often useful to model the engineering design context, is the *item of information*. In ASM2.0 items of information typically represent aspects of the design that are developed during the process. As already noted, each deliverable in an ASM2.0 model is specified in terms of one or more information items. Optionally, a qualifier may be added to each information item in the context of a deliverable. A qualifier is a free text label typically used to indicate how the maturity of an item of information changes at different points in the process. For instance, a modeller might use the qualifier to indicate that a certain item of information has 'low maturity' where it appears in deliverables towards the start of the process, and 'high maturity' where it appears in deliverables later in the process. Qualifiers thus allow the modeller to express the progressive development of information during design. Examples are provided in Section 4.3.

3.1.2 Scaleability considerations

The notation described above allows a process to be described in familiar diagrammatic format. However, it also suffers from scaleability issues in common with other flowchart-based approaches. As the numbers of nodes and edges in a model increase, it becomes more difficult to handle flows that cross a long distance of the model, more difficult to adjust the layout to accomodate new nodes or reorganise existing nodes, and increasingly challenging to communicate the model content given the constraints of a typical computer display. Additional model features were therefore introduced to improve scaleability of the approach, allowing development and manipulation of very large process models and thereby addressing MR2. To illustrate, the industry applications discussed in Section 6 involved typically several hundred diagram nodes, in one case approximately 1300 diagram nodes.

Early applications of ASM2.0 indicated that although many engineering design tasks involve multiple input deliverables, only some of those deliverables are typically responsible for triggering the task. For example, some of the information may have been created much earlier in the process such that, in practice, it is always available when the task would be considered for execution. Therefore, to avoid complicating the process flowchart with arrows that are redundant to the task sequence, two types of input to tasks are allowed in an ASM2.0 process diagram (Figure 2):

1. *Flow connection*, visualised as a solid arrow: Indicates that the production of the input information may be important to trigger the start of the task. Flow connections are visualised as connections between the task that consumes the information and predecessor(s) that produce that information.
2. *Data connection,* visualised as a dashed arrow: Indicates that the input information is needed for the task, but in practice is always available at the time at which the task will be executed. A data connection is visualised as an input from a 'floating' deliverable that is only connected to the task that consumes the information. Even if that same information is produced or consumed by other tasks in the process, a connection will not be shown on the process map and the connection will not be considered by the simulation algorithm set out in forthcoming sections.

Also to assist with handling larger models, ASM2.0 allows for hierarchical decomposition of process maps. In particular a process map constructed using the ASM2.0 notation is drawn on a worksheet, which may optionally be decomposed into subsheets. Subsheets allow for decomposition of a large model as required by MR2, and do not have a specific meaning in terms of the process logic. The modeller can choose whether and how to use them. In some of the applications described later, subsheets are used to decompose the process into logical sections, or as a purely graphical aid to organise large models for easier navigation.

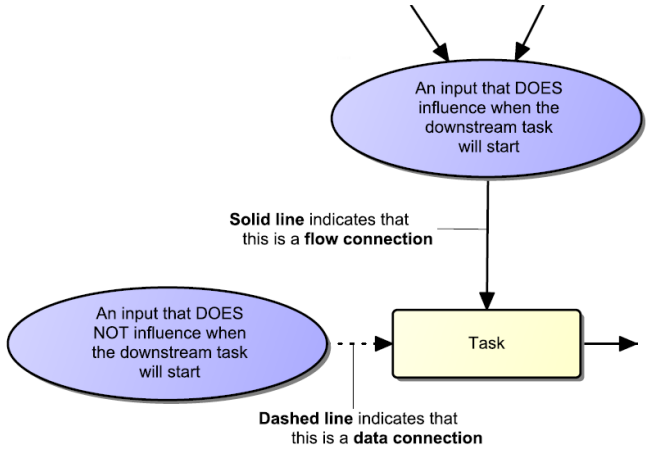

Fig. 2 Nodes in an ASM2.0 diagram can be connected using data connections, that do not influence the task sequences, or flow connections that do influence sequences.

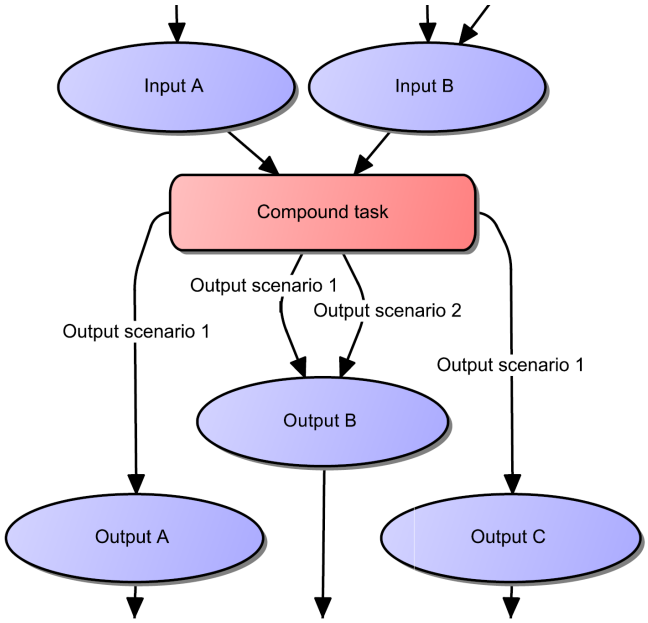

Fig. 3 A compound task has several outcomes, each representing progression of the process along a different route. A single outcome is selected when the task is completed, and each outcome can be mapped onto multiple flow connections. To illustrate, this case depicts a compound task with two possible outcomes. Either Scenario 1 or Scenario 2 occurs when the task is completed. If Scenario 1 occurs, Outputs A, B and C will all be updated. If Scenario 2 occurs, only Output B will be updated.

Finally, the following implementation-related features were developed to improve scaleability as the approach was implemented in the Cambridge Advanced Modeller (CAM) software:

- *Expanding the worksheet.* The user may expand the sheet by adding either horizontal or vertical space at any point on the diagram. This was essential because it allowed the emerging process map to be adjusted relatively easily without repositioning all nodes.
- *Hierarchical modelling.* Features allowing users to easily connect and move elements across levels of the subsheet hierarchy are provided.
- *Hyperlinking.* Replicating the tunnel link functionality introduced by the Design Rationale Editor (DREd) of Bracewell et al (2009), pairs of nodes on the process map can be connected using bidirectional hyperlinks instead of arrows, thereby avoiding the need for an edge to be visually routed in every case. As in DREd, double-clicking on one end of the hyperlink transports the view to the other end, at whatever location and level of the worksheet hierarchy it is located. This feature is especially important to avoid the model becoming unwieldy and difficult to read when nodes are connected across long distances of the model or across levels of the hierarchy. It is also useful in models where many intertwined iteration loops are required, which would otherwise require many crossing edges in the process map.
- *Adding annotations including swimlanes, boxes and text labels* to create process maps that can be more easily communicated and navigated. Examples are provided later.
- *Focusing the display.* The implementation allows the modeller to create classification categories, and assign tasks and deliverables to several of those categories. For instance, a category *Task type* could be created, with subcategories such as *Analysis*, *Synthesis* and *Evaluation*. Another category *Discipline* might have subcategories such as *Mechanical*, *Aero*, etc. Each tasks could then be assigned to any combination of categories using a painter tool. The focusing functionality allows the modeller to request that nodes in a certain logical combination of categories are highlighted, or hidden.

3.2 Quantitative evaluation

Once a process has been mapped with information flows and iteration loops identified, the model can be used to examine the impact of those iteration loops and investigate opportunities for process improvement. Due to the complicated interactions typically found in engineering design processes, MR3 required that the model should be evaluated quantitatively to help modellers investigate the relationship between the configuration of a process and its performance in terms such as total duration and effort, and to allow the impact of potential changes to the process to be investigated.

In ASM2.0 this is achieved by discrete-event simulation, which places minimal constraints on the pro-

cess models that can be evaluated while also propagating uncertainty in task properties to indicate the uncertainty in evaluation results, and therefore indicating risk associated with the process. The next subsections detail the simulation algorithm, the ASM2.0 features for configuring tasks to control the algorithm, and the approaches that have been developed for visualising and analysing the results.

3.2.1 Simulating information flow

The information flow during a modelled design process is simulated by attaching a representation of status to each flow connection that is input to each task in a model. During simulation, each such connection takes one of three values:

- *Unavailable.* Indicates that the information that flows along the connection has not yet been created.
- *Available.* Indicates that the information that flows along the connection has been created and already consumed by the downstream task.
- *Updated.* Indicates that the information that flows along the connection has been created and has not yet been consumed by the downstream task.

To initialise the simulation, all such connections are set to Unavailable, except for inputs where there is no upstream task, which are set to Updated.

In the default configuration, a task will become available to start when none of its inputs are Unavailable (except inputs that can only be created by an Iterate Again outcome) and at least one of its inputs are Updated. When a task is started, all Updated inputs are set to Available, which simulates taking the information into account. When the task is completed, the output connections to be activated are determined, traced downstream until the next task(s) are reached (if any) and the values of the corresponding connections at the input to those tasks are set to Updated. This may allow those tasks to begin in turn.

3.2.2 Process variables

Requirement MR4, provide precise control over the behaviour of the evaluation model, is addressed in ASM2.0 by the inclusion of *process variables*. The modeller can create process variables to represent values that can change during the design process, that can be influenced by task execution and/or that can influence the behaviour of tasks. Commonly, process variables are used as counters to precisely control task and iteration behaviour, or used to describe the design information maturity as it evolves through iterations. Detailed examples of how process variables are used to configure ASM2.0 model behaviour are provided in Section 3.2.4.

To assist with managing the large numbers of process variables typically required to configure larger-scale models, the modeller may associate process variables with different objects in the model. A task may only operate on process variables that are visible to it, i.e. those that are defined within the task itself, those that are defined within information that is used as input or output to the task, and those that are defined within the worksheet the task is placed on (or a worksheet higher in the hierarchy).

3.2.3 Configuring tasks

To recap, MR5 requires that a modeller is able to progressively develop a descriptive model into a simulation model. The network structure of an ASM2.0 diagram contains the information needed to determine the task sequences that are possible. In addition, the way the process unfolds also depends on the properties of each task—the details of when it can be started, how long it takes and what resource it depends on, as well as what happens when the task is completed. To satisfy MR5 ASM2.0 hence allows the modeller to configure each task on the process map in detail, specifying each stage of the task's lifecycle:

- *Preconditions* determine when a task will be available to start (Figure 4). As discussed above and depicted in the figure, preconditions are expressed in terms of the status of information the task requires as input. The modeller may optionally configure the start of the task to be dependent on a function of process variables, which may be helpful if additional constraints on task execution are required.
- *Resources* that are required during execution of the task. Resources are created in the model at the worksheet level and are intended to represent an individual or design team required to execute the task. If that resource is occupied with another task, the task must wait before it can begin. Configuring resources therefore allows concurrency of the simulated process to be controlled. Optionally, the priority of the task may be specified as either a numeric value or as a function of process variables. In a case where several tasks meet their start conditions and compete for the same resource, the one with the highest priority will begin first. If several tasks that compete for the same resource have the same (highest) priority value, one of them is selected randomly to begin first.

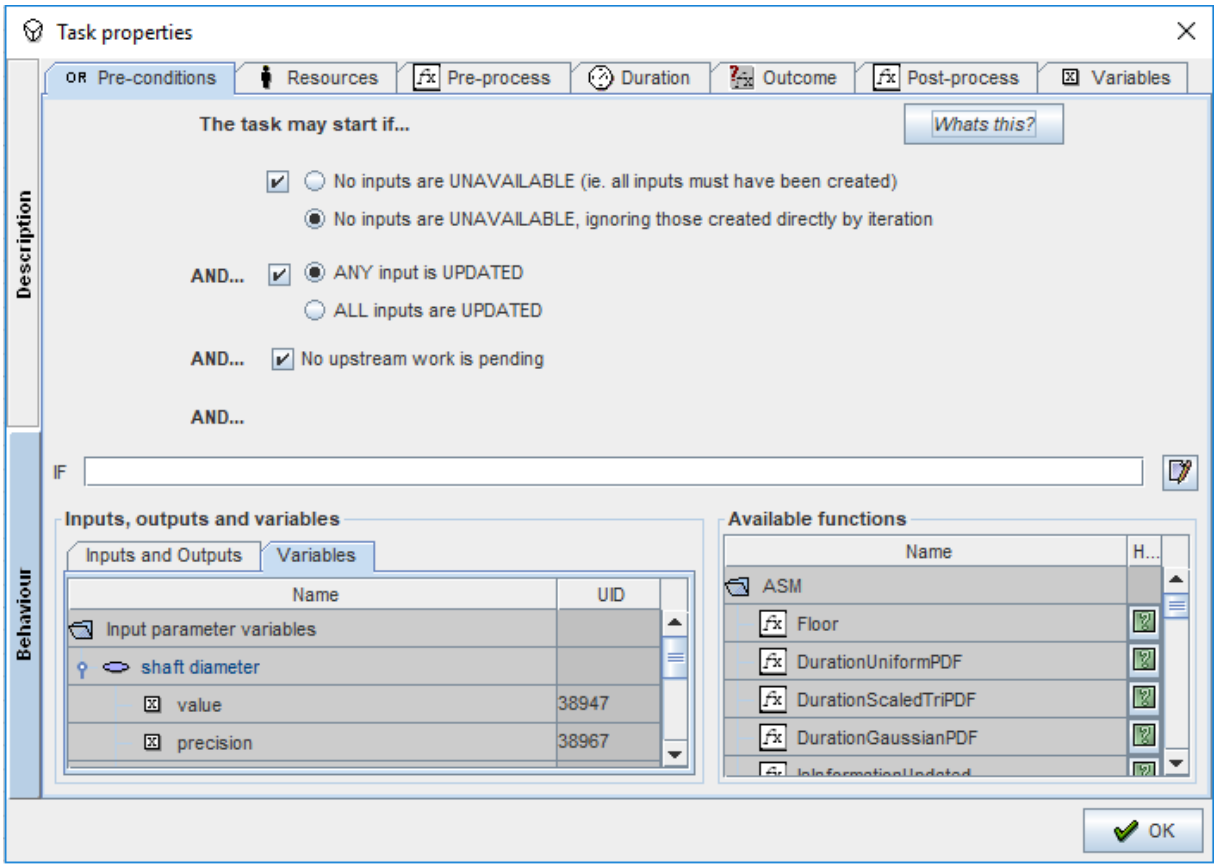

Fig. 4 Configuring preconditions for a task. The checkbox and radio buttons selected in the top part of the dialog are the standard conditions when a task is created. The field labelled *IF* allows the modeller to enter an additional condition that must be satisfied for the task to start, written as a function of the process variables. A function can be constructed using boolean logic and the inbuilt functions shown on the bottom-right, that can operate on the process variables that are in-scope for the task (which are shown on the bottom-left).

- *Preprocessing* determines what happens in the simulation once a task has been selected to begin, and resource has been seized, but before the work starts. Firstly, input deliverables that have status *updated* are downgraded to status *available*. This simulates the updated information being taken into account by the task. It also prevents the task from being considered for execution again unless one or more of the inputs are *updated* in future. Additionally, the modeller may specify that changes to the process variables are to be applied prior to beginning the task. These configuration options are shown in Figure 5.
- *Task duration* can be specified as a probability density function of various kinds or, alternatively, by writing a function of the process variables. The latter can be used, for example, to implement learning curves or to make the task duration depend on the variables associated with input parameters.
- *Outcome selection*, which is available only for a compound task or iteration construct, is evaluated on task completion to determine which of the multiple defined outcomes will occur. The modeller may specify stochastic selection by entering probabilities of each outcome taking place, or may specify the selection as a function of process variables as shown in Figure 6.
- *Postprocessing* configures the actions taken to update the model state to simulate completion of the task, and thereby trigger the start of the next task (if any). Completion of a task always involves setting the state of all output connections in the selected outcome to be Updated—simulating release of updates to the task's output information upon its completion. Additionally, process variables may be updated by writing appropriate functions, depending on the selected outcome.

When a task is first created, all these options are set to default values that allow a simulation to be run. For many models, it is initially sufficient just to enter the duration of each task in the model. The modeller can progressively refine the model as far as necessary

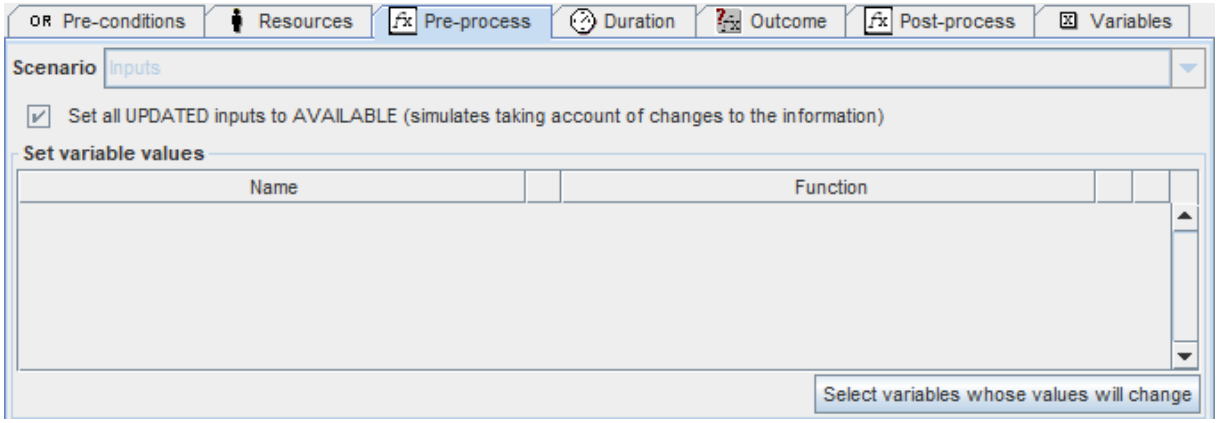

Fig. 5 Configuring preprocessing for a task.

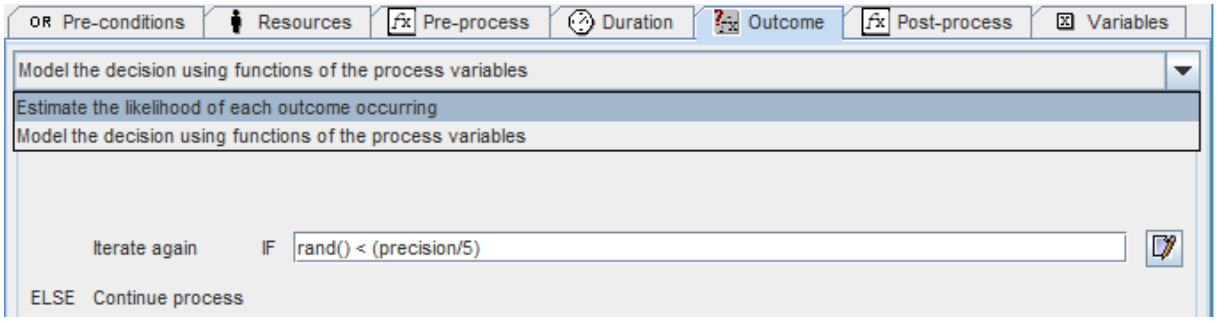

Fig. 6 Configuring outcome selection for a task.

by adjusting the options within individual tasks. This is demonstrated by example in Section 4.

3.2.4 Modelling progressive iteration

MR6 requires that the approach allows a modeller to represent progressive iteration scenarios in which aspects of the design are gradually matured as tasks are revisited multiple times. This is achieved through the explicit modelling of the state of design progression, how each task contributes to progression, and the conditions under which the iteration is deemed complete, using process variables and by configuring tasks that participate in the iteration process. A detailed explanation by example is provided in Section 4.

3.2.5 Modelling rework propagation

MR7 requires that the approach facilitate modelling of corrective iteration including rework propagation. The main challenge here is that the simulation should accurately reflect the logic by which a multitude of routes may emerge depending on the conditions under which rework is triggered and on the patterns of information dependencies in a process. In doing so it is important to avoid logical problems such as deadlocks, in which some anticipated tasks in the simulation are not attempted during the simulation, and livelocks, in which the simulation does not terminate. Additionally, it is necessary to ensure that the task sequences that emerge during simulation match the modeller's expectations as reflected in the information dependencies shown in the ASM2.0 process map.

With consideration to MR5 it was determined that a process' structure should be possible to model as an intuitive flowchart, including interacting feedback loops where necessary, and the simulation algorithm should be able to resolve the outcome of the feedback loops in a sensible way which would hold up against close inspection of the generated routes for any combination in which they might occur. This is addressed in two parts:

1. The first part addresses the observation that the state the network should attain after such an iteration depends on the state before the iteration occurred. In ASM2.0, when a failure occurs, all dependencies downstream of the failure point are traced using a depth-first-search approach. When a currently-executing task is encountered at any branch of the search tree, it is interrupted immediately. This reflects the fact that one of that task's indirect predecessors had been invalidated and hence the ongoing

work would be based on invalid assumptions, and hence should not be continued.
2. The second part is to ensure a task may not start if any task is possible or executing anywhere upstream of it. This means that no task will ever start if it can later be interrupted by another—unless the root cause of interruption is an iteration.

These features allow realistic process models to be simulated, ensuring that all tasks are attempted in appropriate sequence. In keeping with the philosophy that the tool should be configurable, these conditions can be disabled or combined with any other preconditions explained previously.

3.2.6 Investigating impact of process model features on evaluation outcomes

MR8 states that the modeller must be able to appreciate the provenance of simulation output in terms of how it arises from the configuration of the model. To achieve this, process traces generated by the simulation can be visualised as a Gantt chart for inspection, where they can be easily traced back to features of the model. This is demonstrated by example in the next section.

4 Improving the design process by evaluating iteration impact

Having completed description of the features of the ASM2.0 modelling notation, this section moves on to the second aspect of the approach—how ASM2.0 is applied to develop an understanding of an engineering design process and to generate improved process configurations. This is done through in-depth discussion of an application case. The case has been constructed by the authors for the purposes of this article and is deliberately simple to facilitate detailed discussion, while being similar in important respects to the CAE-based engineering design processes observed during our case studies over the years. The advantage of a synthesised case is that it allows the application of the approach to be described in full depth. To complement this example, applications of ASM2.0 to more complex, larger-scale engineering design processes are briefly discussed in Section 5.

The application case concerns the mechanical design of a lightweight polycarbonate machine part. It is based on a student project at the University of Auckland, in which the first author has been closely involved for a number of years. In the project, student designers work in pairs to design a part to be supported in a single plane through four pins. The part must be designed for specified stiffnesses under several load cases in which three pins are held in position while the fourth is subject to small displacements, while minimising weight and avoiding failure by yielding or buckling. Additionally, the design must allow for some pipes to pass through the design region without interference with the part. The pipes are flexible such that their positions can be adjusted within certain ranges. These constraints are carefully selected so that the problem cannot be solved computationally with the tools available to the students, but requires iteration to learn about the feasible design space and to refine the design to achieve the best possible performance. The problem allows for solutions with different topologies, as long as all four pins are solidly supported.

4.1 CAE-based design process

In line with best practice, participants in this design process are advised to approach the problem by first generating alternative concepts for topologies and using rapid, low-fidelity FEA to compare them before selecting one to take forward. The selected topology is then optimised through weight reduction iterations using higher-fidelity but more time-consuming FEA, considering the various constraints until satisfactory performance is reached. The design objectives are coupled and very sensitive to small changes in the geometry, so many iterations are typically required to build understanding and converge on a good solution. Project teams have been observed to sometimes abandon a selected topology if they find it does not allow for a high performance solution, in which case they start again with a new concept.

Once the geometry has been finalised, a tool path must be created so that the design can be milled in a CNC machining centre. In some cases this leads a project team to realise that their design contains features that cannot be machined, e.g. sharp internal corners, in which case minor adjustments to the geometry may be needed. In some cases a project team may find that their design takes too long to machine, usually because it has too many internal cutouts, in which case they may decide to substantially rework the entire concept.

The process as described above is summarised in Figure 7. Although simple, this process encompasses all key elements of CAE-intensive engineering design practice that were observed in our case studies and that are addressed by ASM2.0, including the dependence on particular software packages used in cycles of progressive iteration, the possibility of corrective iteration, and the need to efficiently divide work among team members.

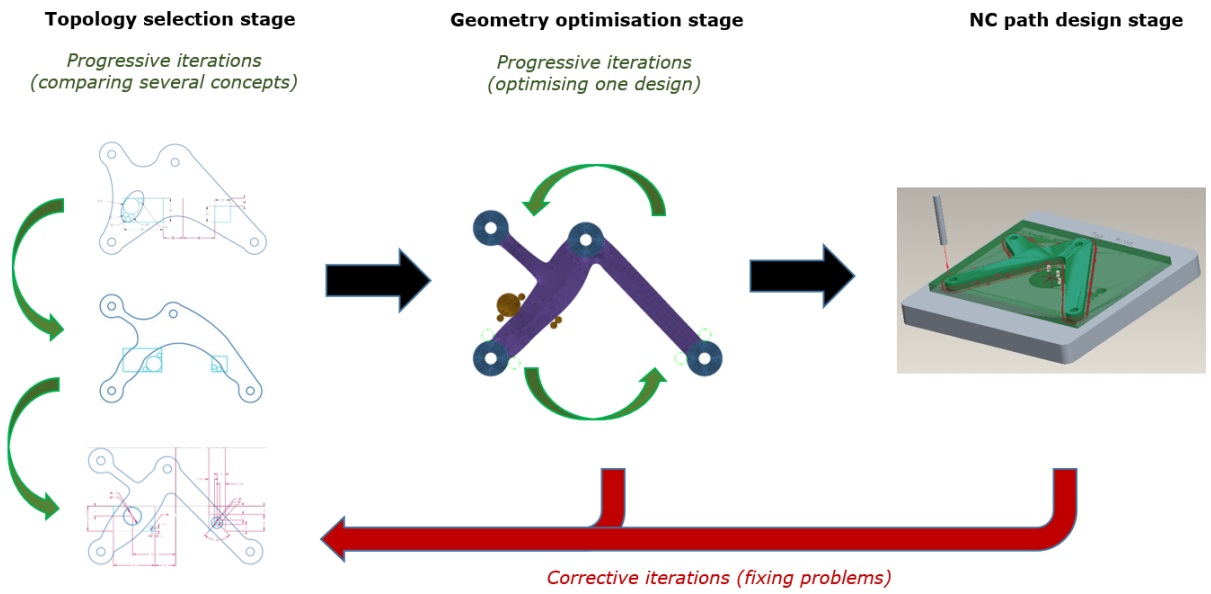

Fig. 7 Overview of the design process for the lightweight polycarbonate machine part.

4.2 Process configuration questions

The process has been completed a number of times by different teams, and several questions have arisen regarding whether improvements could be made.

Firstly, it has been noted that it is necessary to transfer the design from a CAD package (used to model geometry and needed to design the NC path for manufacture) to the FEA software (used to analyse performance) every time a geometry adjustment is made. The data transfer is a repetitive and non-value-adding manual processing task. This leads to the first question for the worked example:

– Question 1: *How much could the process be improved by completing all design iterations using the design modeller built into the FEA software, eliminating the data transfer task?*

Secondly, the project is performed in teams of two. Team members typically work together on the initial concepts while also getting to grips with the software and the design problem. Then, they typically choose to divide their efforts, so that one student focuses on fine adjustment and optimisation of the geometry, while the second works concurrently on the NC path design. This is presumably thought to be efficient because it allows each person to concentrate on a subset of the overall problem, but leads to the second question for the worked example:

– Question 2: *Is it effective to divide work by function, i.e. into (Designer A) geometry optimisation, vs. (Designer B) manufacturing design?*

4.3 Addressing the questions using ASM2.0

4.3.1 Mapping the design process

The first step in applying ASM2.0 to address these questions is to map the process in order to identify the tasks and information flows involved. Different representations might be possible; for this example the map shown in Figure 8 was constructed. The process is depicted in three distinct stages reflecting the description above, namely the concept design, geometry optimisation and manufacturing process design. Because the first question to be addressed relates to data transfer tasks, the process is decomposed to the level where these tasks are made explicit. The evaluations that potentially lead to iterations are also incorporated. Some of these iterations (i.e. within the concept and detail phases) are planned and expected as part of the design progression, while others, across the phases, are in essence undesirable.

4.3.2 Configuring the model for simulation

The next step is to configure the process model for simulation, with special consideration to iteration behaviours. This is described in depth because the ability to precisely model design iteration is one of the main distinguishing features of the ASM2.0 approach.

First, the duration of each task was estimated. This was possible from experience, noting that due to the iteration-intensive nature of the process, individual tasks are performed many times in a typical process, and several processes have been observed over the years.

Table 2 Configuration of tasks in the process model

Task	Configuration of task
Generate topology concept	*Duration:* 10-20 min
Data transfer	*Duration:* 1-2 min *Post-process:* DTTime =DTTime + ASMTaskDurationCompleted("DAY")
Low-fidelity FEA	*Duration:* 5-10 min
Concept acceptable?	*Pre-process:* conceptstried = conceptstried+1 *Iterate again IF:* (conceptsoptimised==0 && conceptstried<4+(6*rand())) || (conceptsoptimised> 0 && conceptstried< $1 + (3 * rand())$) *Post-process [Continue Process]:* conceptstried= 0
Detail / refine geometry	*Duration:* 1-2 min
High-fidelity FEA	*Duration:* 20-30 min
Performance plateau reached?	*Pre-process:* optimisationsthisconcept=optimisationsthisconcept+1 *Iterate again IF:* manfcorrectionsneeded==0 && ((conceptsoptimised==0 && optimisationsthisconcept<10 + (11*rand())) || (conceptsoptimised>0 && optimisationsthisconcept<5+(10*rand()))) *Post-process [Continue Process]:* optimisationsthisconcept= 0
Performance acceptable?	*Pre-process:* conceptsoptimised = conceptsoptimised+1 *Iterate again IF:* manfcorrectionsneeded==0 && rand()<0.3
Tool path design/simulation	*Duration:* 0.5 * manfcorrectionsneeded + 2* (1-manfcorrectionsneeded)
Machining possible?	*Iterate again if:* manfcorrectionsneeded==0 && rand()<0.25 *Post-process [Iterate again]:* manfcorrectionsneeded=1
Machine time acceptable?	*Iterate again:* probability = 0.1

The next step was to consider the iterations, the sitations in which they would occur and how task durations might change on subsequent iterations. In this case, it may be observed that (in common with the industry examples discussed earlier in this article) the design process involves an interplay between progressive and corrective iterations. The five iteration loops are known from experience to have different characteristics:

- For the progressive iterations in the concept stage, it is known that between 4 and 10 concepts are typically studied. On each iteration, a different concept is modelled and analysed.
- For the progressive iterations in the detail stage it is known that between 20-30 iterations are performed. On each iteration, the design is adjusted slightly and the FEA is re-run, until the designer is not able to achieve further improvements in performance by small modifications to the selected topology.
- If the result is deemed insufficient at the *performance acceptable?* decision, new topologies will be considered. If this occurs, only 1 or 2 new concepts will be considered and experience shows that no more than 10-15 detail iterations will be required to optimise the selected new concept. This is because the student designer will have gained experience with the problem and will have a better idea of what changes will be needed.
- If machining is found to be impossible, only 1 round of further detail refinement will be needed to correct the issue. This is because the issues are typically very simple, such as adding appropriate rounds to the geometry.
- If machining time is found to be unacceptable, the concept and detail stages of the design process will need to be revisited. This is because iterations in this case involve significant changes to the design, such as reducing the number of internal cutouts.

Having decided on the behaviour of each individual iteration loop as discussed above, the model was configured to exhibit these behaviours using process variables. To achieve this the loops were worked through one-at-a-time to configure their effects, in the sequence from start to finish of the process. As each additional loop was considered, the expressions defining impacted task behaviours were progressively expanded to take the loops into account.

To illustrate, consider the topology selection stage (the first stage) of Figure 8. Working from top to bottom, the iteration construct *Concept acceptable?* was encountered first. The desired behaviour was to select the *Iterate again* outcome between 4 and 10 times as explained above. To achieve this a process variable named *conceptstried* was created within the task. The task preprocessing was configured to increment the variable, i.e. using the function $conceptstried = conceptstried +$

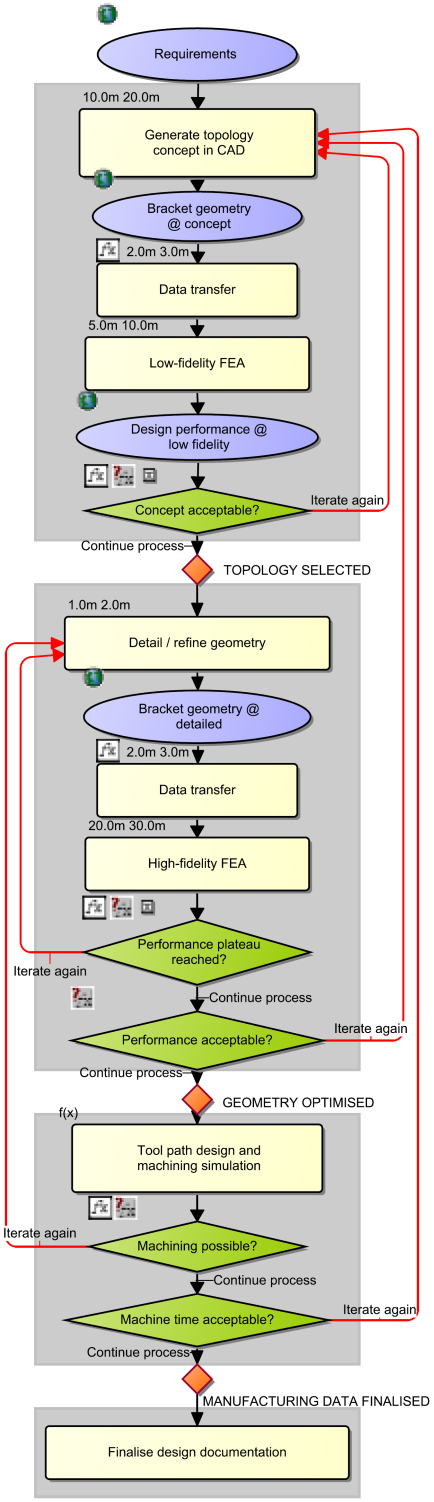

Fig. 8 ASM2.0 model of the design process for the machine part shown in Figure 7. Note the use of milestones and background annotations (grey boxes) to improve readability of the model. Note also the icons shown above each task—each icon indicates a situation where the modeller has adjusted the task configuration from its default setting. In the software implementation, the user can hover over each icon to view the corresponding configuration option.

1. The effect is to create a counter that increments each time the task is encountered. Then, the task outcome was configured to depend on a function of the process variable. The following function was entered into the *Iterate again IF* field for the task:

$$\text{Iterate again IF: conceptstried} < 4 + 6 \times \text{rand}() \qquad (1)$$

In ASM2.0, *rand*() is a built-in function that returns a value in the range 0.0 to 1.0. Hence, this function configures the task and the iteration loop to have the desired behaviour. Additionally, it was noted that the topology selection stage might be visited several times. For this reason, the counter should be reset when the *Continue process* outcome of the task is reached. This is achieved by adding a suitable function to the task postprocessing, i.e. $conceptstried = 0$. Note that because the process variable *conceptstried* was defined within the task, it can only be considered or altered by that task.

Having developed an initial configuration for the topology iterations as described above, a similar process was followed for the next iteration construct, which governs the progressive iterations within geometry optimisation stage. Then, continuing to work through the model in sequence, the *Performance acceptable?* task was reached. With reference to the desired behaviour stated above it was determined that the *Iterate again* probability would be set to 0.3. Additionally it was noted that if the *Iterate again* outcome occurs and the topology concept is revisited, on the second and subsequent iterations, fewer passes of the topology stage would be required. This was configured by, first, creating a counter variable *conceptsoptimised* whose value is incremented in the preconditions of the *Performance acceptable?* task. The content of the Iterate again IF field for the task was then expanded from Equation 1 to account for the number of concepts that had been optimised at the time the task is attempted, as follows:

Iterate again IF: [conceptsoptimised== 0 && conceptstried $< 4 + 6 \times$rand()] || [conceptsoptimised> 0 && conceptstried $< 1 + 3 \times$rand()]

(2)

In this expanded equation, the term in the first set of square brackets defines how the decision will behave on the first pass of the topology development stage, while the term in the second set of square brackets determines what will happen on subsequent passes, if any.

By continuing this process, the configuration shown in Table 2 was ultimately developed. The example given above illustrates that, although the content of Table 2 might initially appear complex, it is arrived at through

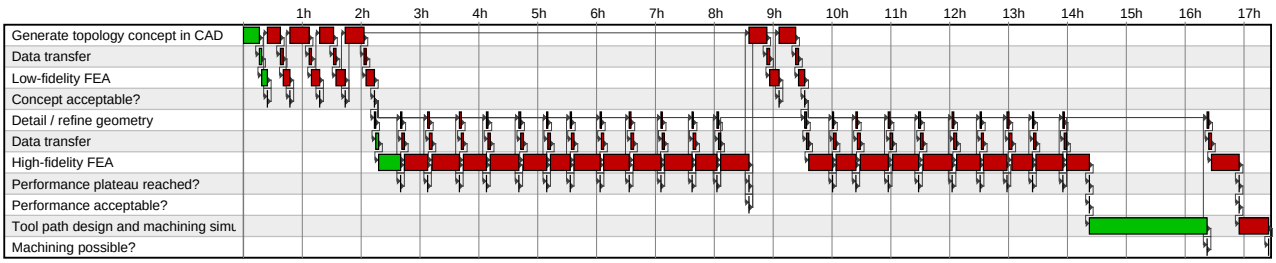

Fig. 9 A representative process trace for the sequential process. This trace represents a process in which 5 concepts were initially considered, 13 detail iterations followed by a second round in which 2 concepts led to 10 additional detail iterations. One revision of the final concept was required to correct machinability problems.

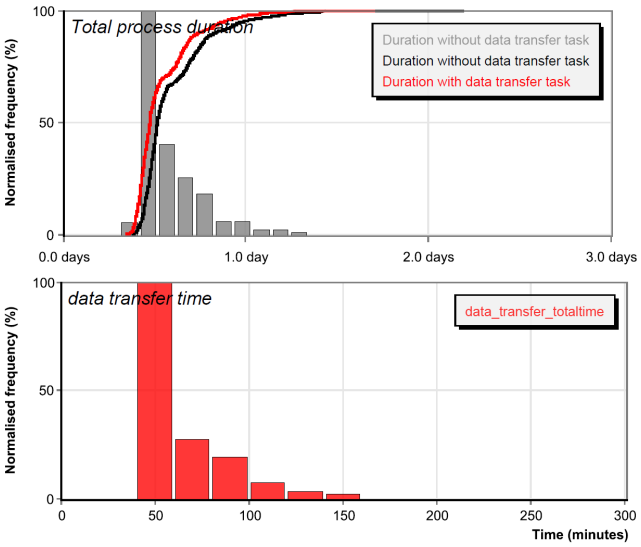

Fig. 10 Process simulation result showing total duration of the process with and without data transfer task (top) and the total time spent in data transfer task (bottom).

a series of simpler reasoning steps that progressively expand the modelled behaviours of tasks to account for different contexts in which those tasks might be executed. While this example was constructed by the authors to explain the approach, industry applications reported later in this article and elsewhere (See Table 4 for a summary) demonstrate that researchers and practitioners who are used to working with advanced engineering tools have been able to configure the ASM2.0 simulation for their own situations.

4.3.3 Verifying an ASM2.0 simulation model

While a model is being progressively configured, it should also be verified to check that the simulation exhibits the desired behaviour on a detailed level. This is an important step in the approach for several reasons: to ensure that the intended behaviours have been properly configured, to assess whether the task sequences that are generated appear realistic, and to build understanding of how the model configuration impacts its behaviour. The general procedure recommended in ASM2.0 is to run a small number of Monte-Carlo simulation runs and examine a number of randomly-selected process traces to verify that the model exhibits the expected behaviour in all the examined cases. This should be repeated throughout the configuration process to verify behaviours as they are progressively added to the model. In the case under discussion, for example, a verification check was completed after each iteration loop was configured. This progressive approach allows any configuration problems to be pinpointed and corrected as they emerge.

To illustrate a typical verification check, in the case under discussion and for the completed configuration, one such trace generated by simulation is shown in Figure 9. This example can be interpreted as the situation where a single round of concept revision to improve performance is conducted. In this case it can be verified from Figure 9 that the durations of tasks within the progressive iteration loops are as intended, that the sequences are correct, and that the number of iterations of the revised concept is reduced as intended. This trace also simulates the situation in which a manufacturability problem was corrected, which resulted (as intended) in a single additional pass of the detail design tasks. Several traces were generated and examined in this way, each involving a random selection of the events and outcomes that are allowed by the model, to ensure that all could be explained in terms of the model configuration.

Once a selection of traces have been verified for the completed configuration, the overall process duration predicted by the simulation should also be considered to ensure it is within the bounds expected, typically drawing on previous experience of the actual design process.

Adjustments to the model might be required. Overall, these activities build confidence that the behaviour of the simulation model reflects the modeller's intentions, and verify that it provides adequate basis for investigating the process configuration.

After verification was completed for the case under discussion, the questions stated earlier (related to potential process improvements) were considered in turn.

4.3.4 Question 1: Can data transfer effort be reduced?

This question was addressed by simulating the model in the above-described conditions, and comparing the result to an altered configuration in which the duration of the data transfer tasks in both design stages were set to zero. Results of these two configurations are compared in Figure 10. These results show that even considering the many iterations that take place, the total impact of the data transfer task is less than 1 hour in the majority of simulated cases. This can be verified by comparison to example Gantt charts such as Figure 9, where the typical number of iterations of the data transfer task can be counted and multiplied by the known duration of that task. Overall, the ASM2.0 modelling and analysis indicates that although the data transfer task is manual and repetitive, the time that could be saved by eliminating it is not significant.

4.3.5 Question 2: Is it effective to divide work by function in this process?

To address this question, the model discussed above was reconfigured to simulate the functionally-oriented division of work, in which the optimisation of geometry proceeds in parallel with the manufacturing-related design tasks, which are each done by an individual specialising in the respective tasks. The reconfiguration was achieved by adjusting the information flow into the start of the first manufacturing task, so that task can begin as soon as the detailed geometry deliverable is first available. A second flow was also added so that the design cannot be finalised until the newly-concurrent manufacturing and optimisation work streams are both complete. Apart from these changes, that are shown in Figure 11, the rest of the model configuration remains the same. The adjustment of information flows allows geometry optimisation and manufacturing design to work as partially concurrent streams in this reconfigured process.

The simulation behaviour of this reconfigured model was verified again by examination of randomly-selected process traces. One such trace is shown in Figure 12, which indicates how the ASM2.0 is able to correctly

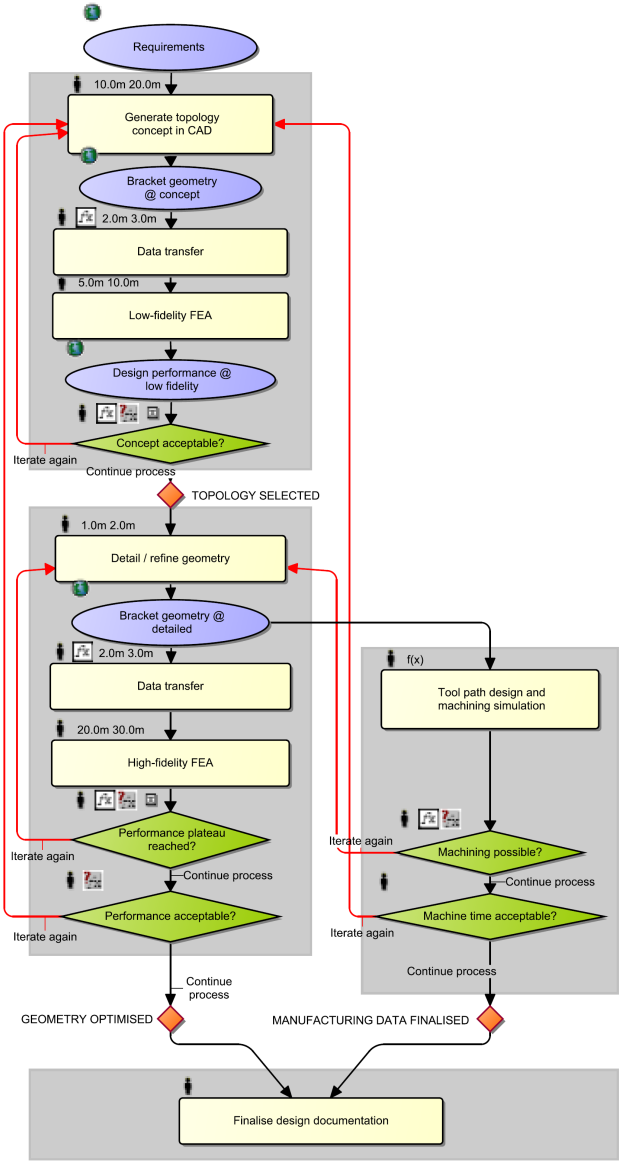

Fig. 11 The design process reconfigured to allow concurrent work on geometry optimisation and manufacturing model development.

identify the appropriate task sequence. In particular, although the new process configuration allows manufacturing tasks to start earlier, on each occasion that the geometry is adjusted by the optimisation specialist, the concurrently-executing manufacturing task that depends that geometry is interrupted and will begin again to take the updated information into account. This is determined automatically by the algorithm and does not require any special configuration of the tasks involved.

In terms of insights gained, Figure 12 indicates that because of the repeated interruptions, little time can be saved by attempting to execute the optimisation and manufacturing design work in parallel, while much of

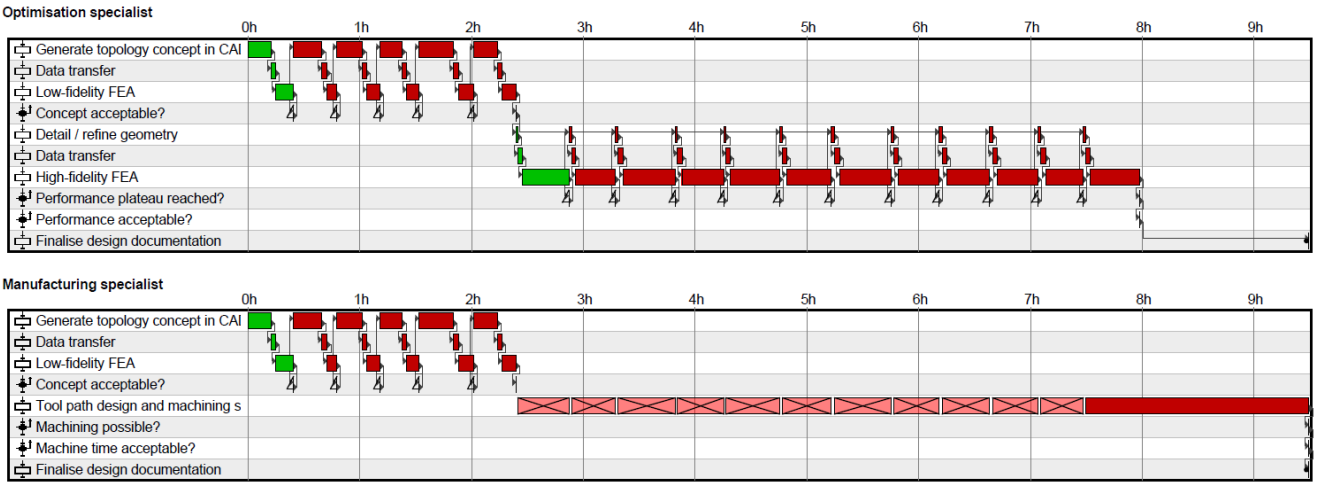

Fig. 12 Representative process trace for concurrrent configuration. The pink bars indicate interruption of the manufacturing design task, which occurs on each occasion the geometry is iterated.

the effort of the manufacturing specialist will be wasted as they repeatedly rework their model due to changes in the input to their task. Therefore, this is not an effective process configuration. This conclusion might perhaps seem obvious when considering the structure of the reconfigured model, but was not evident prior to the process modelling and simulation.

4.3.6 Insights for an improved process configuration

The understanding generated through the modelling discussed above yielded insight for a third, improved process configuration that had not initially been considered. In particular, it was noted that if two promising topologies were taken forward from the concept stage and then independently optimised by the two students, this would increase the probability of at least one of the concepts yielding acceptable performance and hence, on average, would reduce the number of iteration loops required before finalising the geometry. Considering the *Performance acceptable?* check was initially estimated to have probability 0.7 of achieving a succesful result on any given iteration, and assuming that the concepts being considered by the two students are independent and have independent and equal probabilities of success $P(S_1) = P(S_2) = 0.7$, the probability of achieving a successful result from at least one of the students for every pass of the concurrent process can be modelled as:

$$P(S_1 \cup S_2) = P(S_1) + P(S_2) - P(S_1)P(S_2) \qquad (3)$$

where $P(S1) = P(S2) = 0.7$

$P(S_1 \cup S_2) = 0.91$ which means that under these assumptions, the probability of rework for each pass of the optimisation process is reduced from 0.3 to 0.09 when the two students work in parallel.

To investigate the impact on overall lead time, the model was reconfigured again as shown in Figure 13. Here, apart from the restructured tasks and information flow, the only change to the tasks' configuration is to reduce the *Iterate again* probability of the *Performance acceptable?* task from 0.3 to 0.09 in line with the estimate above. It may be noted that during the geometry optimisation stage, because the numbers of iterations of the two concurrent streams are variable, in most cases one of the streams will finish before the other. The simulation algorithm automatically ensures that the *Performance acceptable?* task will not start until both streams are complete, due to the standard precondition that a task cannot start if any upstream work is pending (as shown in Figure 4). This illustrates again how the algorithm is able to correctly determine a process sequence in the majority of cases, without requiring task-by-task consideration of the logical conditions under which a task can be executed.

As with the other process configurations, simulation traces were reviewed to verify the model. Once verified, aggregate results from simulation were considered. Figure 14 confirms that this configuration does not significantly affect the best-case duration, as can be expected when comparing the process maps in Figure 10 and Figure 15—however, the risk of long-duration processes caused by iterations of the topology concept is substantially reduced. ASM2.0 reveals this is the most effective of the three configurations considered.

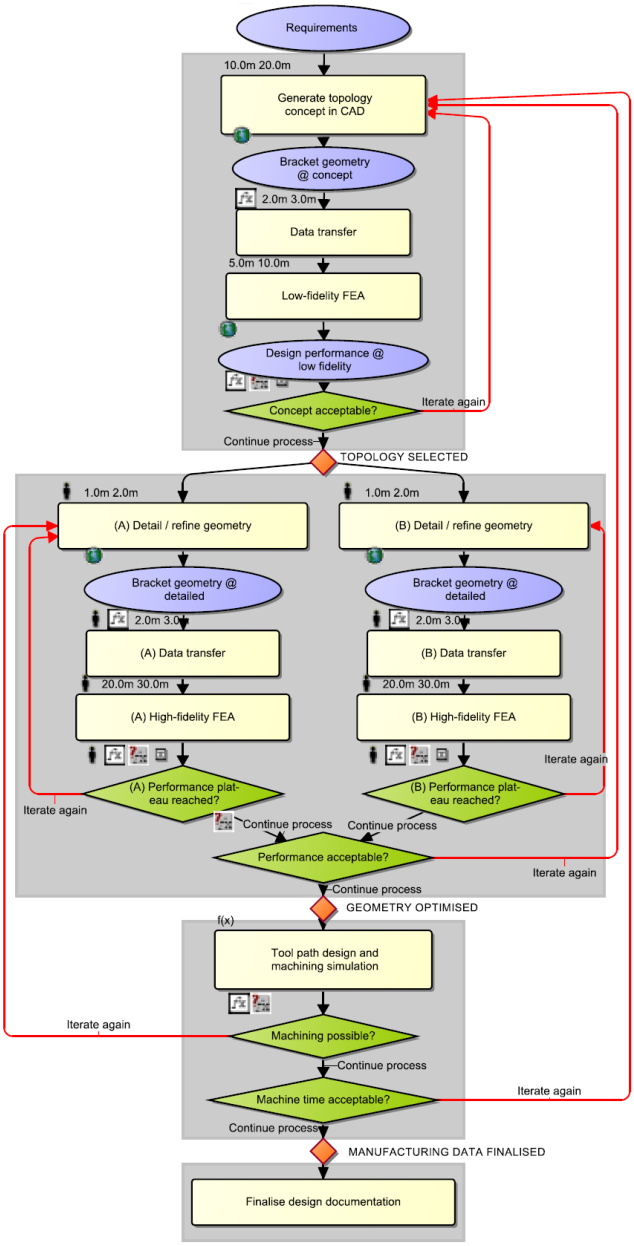

Fig. 13 The improved configuration.

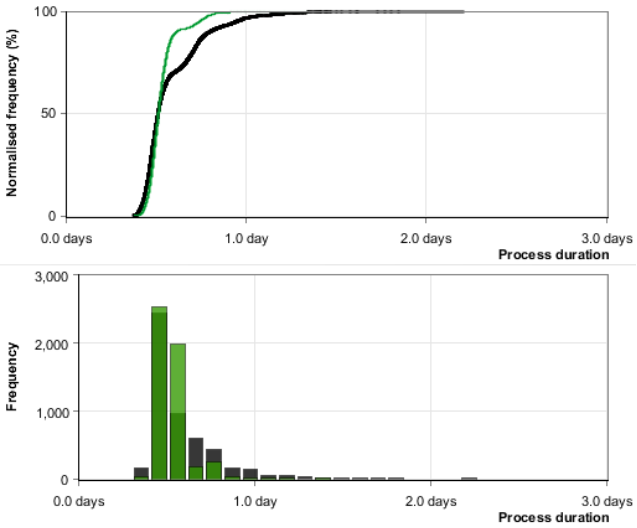

Fig. 14 Process duration for the improved configuration (green) vs. the original configuration (black). Note how the risk of higher process durations is significantly reduced.

4.4 Summary

The example discussed in this section illustrates the model features and also the general approach to modelling and simulation that underpins ASM2.0. Firstly, it shows how process mapping helps to make existing knowledge about the process steps explicit and prompts the modeller to integrate that knowledge into an appreciation of the structure of the overall process. Without that appreciation, inefficiencies are not easy to identify—even in this relatively simple case. Secondly, the example shows how the availability of process variables for configuring iteration loops prompts deep consideration of the behaviour of iteration loops and how they affect the tasks within them. Thirdly, it shows how verifying the emerging model by investigation of the Gantt charts as discussed in Section 4.3.3 further prompts the modeller to assess the realism of the resulting simulation and builds confidence in results. By the time a simulation is run to investigate proposed changes to the process configuration, in our experience the modeller will have developed sufficient understanding to explain the simulation results in terms of features of the process and to identify improvement opportunities.

Overall, the example illustrates how the underlying philosophy of ASM2.0 is to provide a relatively transparent approach that sets the focus of modelling on appreciating the behaviour of the process, which leads to insights for improvement. It also illustrates how opportunities for improvement arise from the iterative and concurrent dynamics of the process—a modelling approach that could not reflect these dynamics (for instance, if the iteration loops were all specified in terms of probabilities as is commonly done in research models) would not be suited to investigate the kind of reconfiguration questions and CAE-based design processes addressed in this example.

Certain features of ASM2.0 that are not offered in other models reviewed in Section 2 are essential to this approach. These features include the familiar graphical notation, the closely integrated process mapping and quantitative evaluation, the configurability of behaviour that is enabled by process variables, and in particular the simulation algorithm for handling corrective iteration with rework propagation, ensuring that tasks are started, reworked, and where necessary, interrupted

at the right time and in the right sequence. It can be noted that after the development of the initial model, the different configurations were modelled simply by re-organising the tasks on a graphical level without further adjustments to the process variables and task configurations. The algorithm is able to correctly resolve the modelled processes so that the iterations and interrupts are treated correctly. This automated resolution is essential as models increase in scale and complexity.

5 Larger-scale applications

The example discussed in the previous section demonstrates application of the approach in detail. However, due to its simplicity, the example does not demonstrate the ability of ASM2.0 to deal with larger-scale processes that are common in engineering practice. In such cases large numbers of diagram nodes are often required to adequately represent the processes. Also, large-scale processes cut across responsibilities and/or organisational boundaries, so that elicitation and integration of process knowledge from multiple participants are significant challenges for process modelling and improvement.

The approach has been succesfully applied to a number of cases of this scale, some of which are briefly discussed in this section. The cases have been previously published, so are not described in detail here. Also, it must be noted that the work discussed in this section took place at different stages of the ASM2.0 development and not all the features and insights described in this paper had been developed when each application took place.

In one very early application published soon after the original ASM, Bell et al (2008) describe how the emerging approach was used to map the process for conceptual design of turbine blade cooling systems. In this case, a key challenge was to generate overview and understanding of the process before potential improvements could be considered. 19 process participants were interviewed, leading to the creation of a process model capturing tasks, information flows and iterative cycles, facilitating a consensus view of the process. The modelling exercise revealed "many instances of nested iteration loops, task concurrency and branching between possible choices"' (Bell et al, 2007). The model itself comprised more than 1300 diagram nodes including 400 tasks, grouped into 50 subprocesses nested 11 levels deep (Bell et al, 2007) (Figure 15). Noting that this process is search-driven, multidisciplinary, and highly iterative, this descriptive model was used to gain overview of the process and to identify tasks that occurred earlier or later than the participants thought was ideal.

In another project, the emerging ASM2.0 was applied to map the jet engine conceptual design process with a view to incorporating additional lifecycle engineering considerations and tools Kerley et al (2008, 2011a). In this case, emphasis was placed on simulation to help assess the likely impact of including new tools and activities at different points in the design process—noting that the ultimate impact would depend on the iterative structure of the process. The model that was created and refined through a series of practitioner interviews, shown in Figure 16, was organised conceptually as a Vee model. The main design activities were depicted progressing down the left-hand-side of the figure, and followed by analysis and evaluation activities progressing up the right-hand-side. Red arrows flow from right to left and indicate the possibilities for corrective iteration following design evaluations. In this model, an emphasis was placed on an open-box approach in which company personnel participating in the modelling process were engaged, as far as possible, with the assumptions made and included in the model. Kerley et al (2011b) further expanded on this work applying ASM to predict the performance of a proposed automated design system under different application scenarios. The model shown in Figure 17 was created as part of this effort. In this case the model was structured as an iterative cycle, in which different paths were selected on each step to simulate the application of different tools as the design process moved forward. Detailed discussion of how the model was developed and applied to analyse different scenarios can be found in Kerley et al (2011b).

The final applications to be discussed in this section made extensive use of the configurable simulation functionality. Wynn et al (2011) apply ASM to model progressive iterations in a wing design process by using process variables to represent the unncertainty in items of information that are progressively developed during design. The model was configured such that uncertainty in the inputs to each task were used to determine the methods that are applied and the time required for that task—the assumption being that as design progresses, uncertainty levels determine which methods are selected for use at each step, and those levels are also iteratively reduced by completion of design and analysis tasks until the final design is reached. Wynn et al (2011) used this model to analyse the time at which high-fidelity design tools should be brought into play during the process to minimise process duration. Shapiro et al (2016) expanded this concept of using process variables for representing aspects of the design itself, creating the Design Process Change Method (DPCM) which is based on ASM modelling. DPCM can

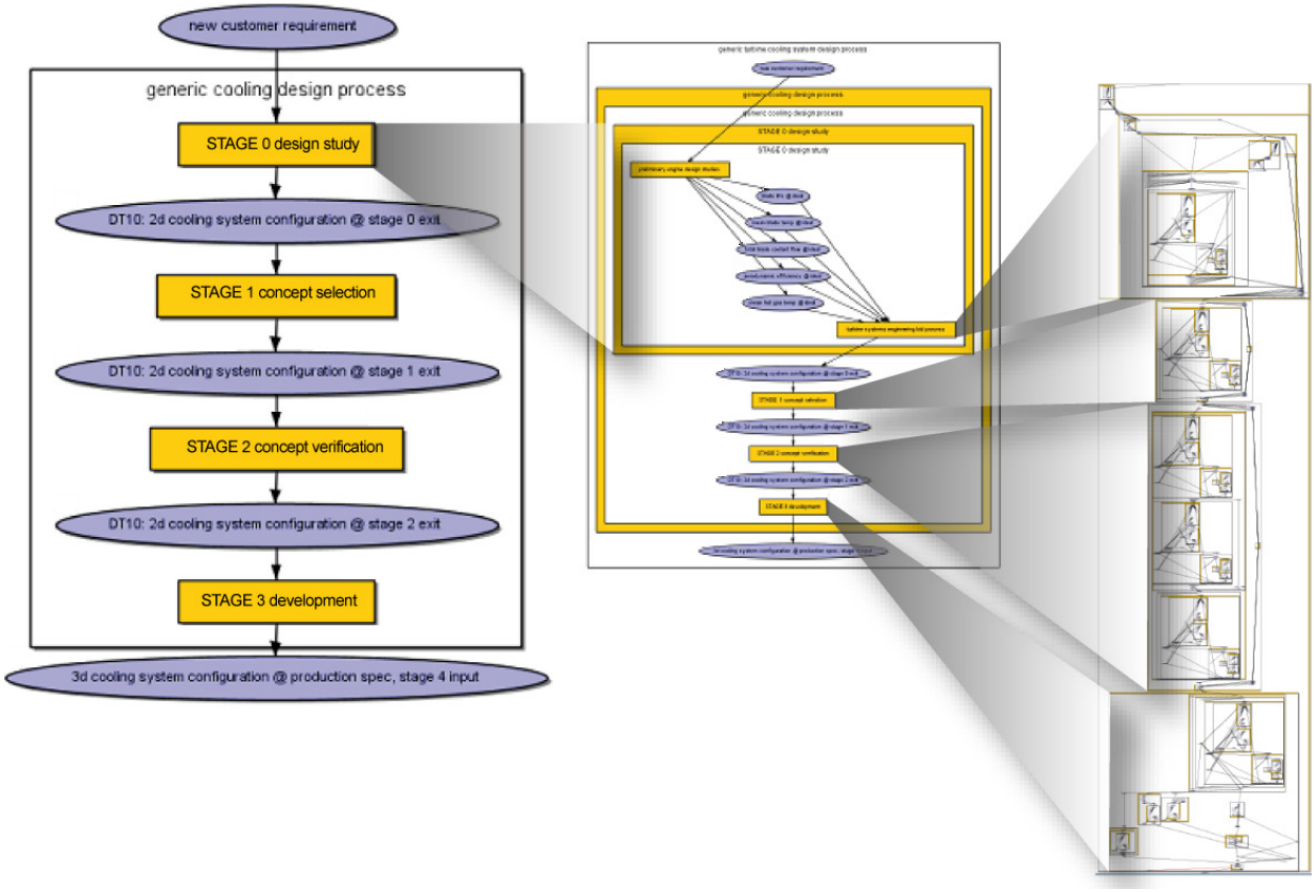

Fig. 15 Descriptive ASM model of turbine cooling system conceptual design process at a UK aerospace manufacturer. The orange boxes represent subsheets, which can be displayed either open or closed. Image adapted from Bell et al (2007) pending permission of the Design Society.

Table 3 Summary of selected applications. Models in the top part of the table were developed to represent real situations in an industry context and to generate specific insights to improve those situations. Models in the second part of the table were mainly developed in the research environment to address issues of research interest. For comparison, the model discussed in Section 4.3.6 of this article has 17 tasks.

Publication	Tasks	Objective of application
Wynn et al (2005)	73	Support process planning of fan blisk design
Wynn (2007)+	79	Develop comprehensive description of blisk design process
Wynn (2007)+	430	Identify provision of service information to tasks in jet engine design
Bell et al (2008)	400	Develop understanding of turbine blade cooling system design
Kerley et al (2011a)	75	Redesign a process to incorporate new design tools
Kerley et al (2011b)	51	Predict performance of a new integrated design system
Le et al (2012)	430*	Calibrate a system dynamics model of the design process
Zhang et al (2015)	150	Increase overlapping between two departments
Shapiro et al (2016)	35	Investigate impact of design changes on process duration
Wynn et al (2011)	13	Investigate impact of uncertainty levels during iterative design
Shapiro et al (2015)	17	Investigate impact of iteration likelihood on process duration
Xin Chen et al (2016)	10	Optimise resource allocation to tasks considering resource characteristics
Tahera et al (2017)	13	Investigate impact of task overlapping on duration

+ Application project is summarised in the mentioned publication but, as stated in that publication, was led by another person who did not publish the work. * This is the same model reported in Wynn (2007) but configured and analysed for a different purpose as stated.

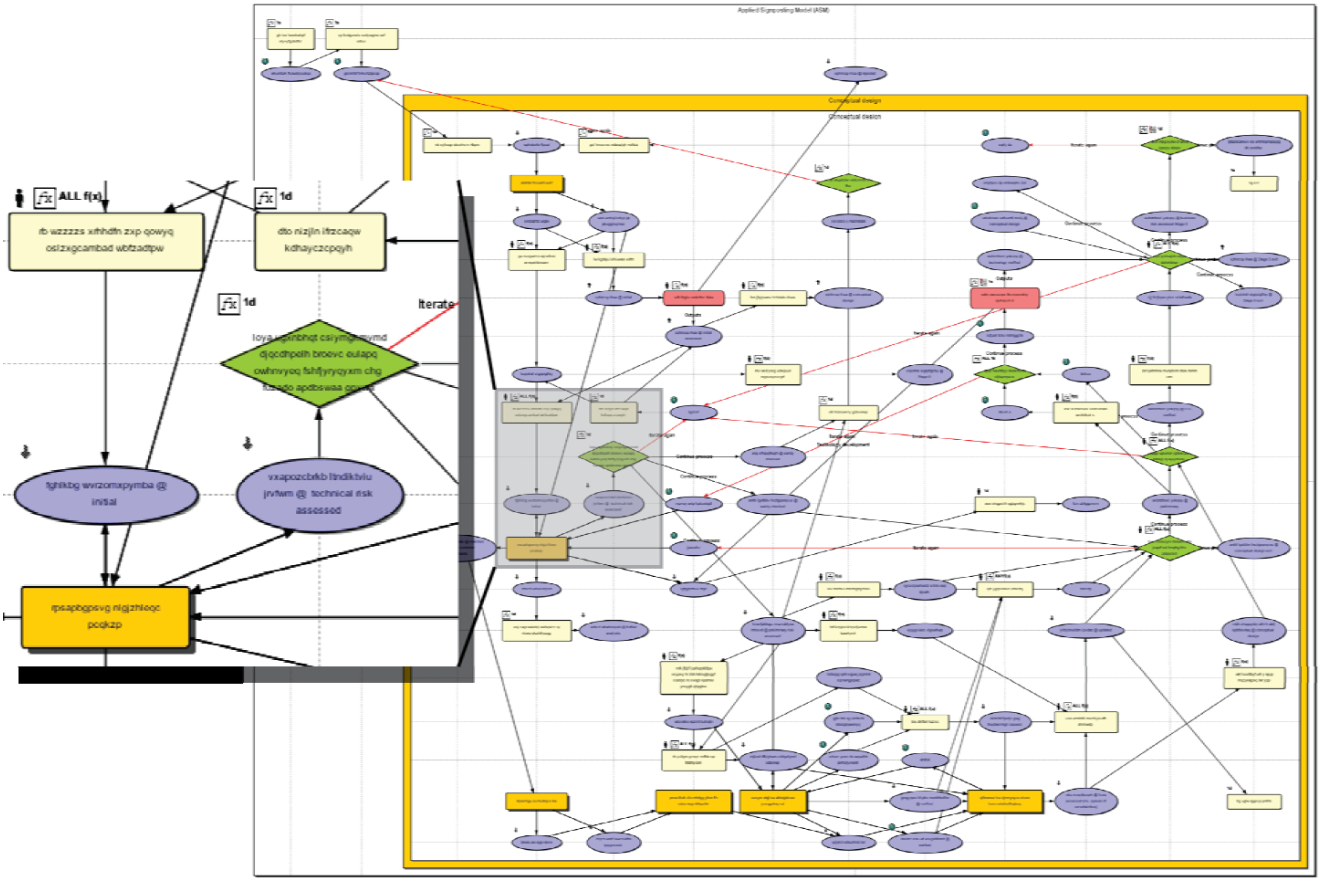

Fig. 16 Partial view of the ASM simulation model of the jet engine concept design process at a UK aerospace manufacturer, adjusted to account for proposed additional life-cycle engineering considerations. Reproduced from Kerley et al (2008) pending permission of the Design Society. Note that text is obscured due to commercial sensitivity.

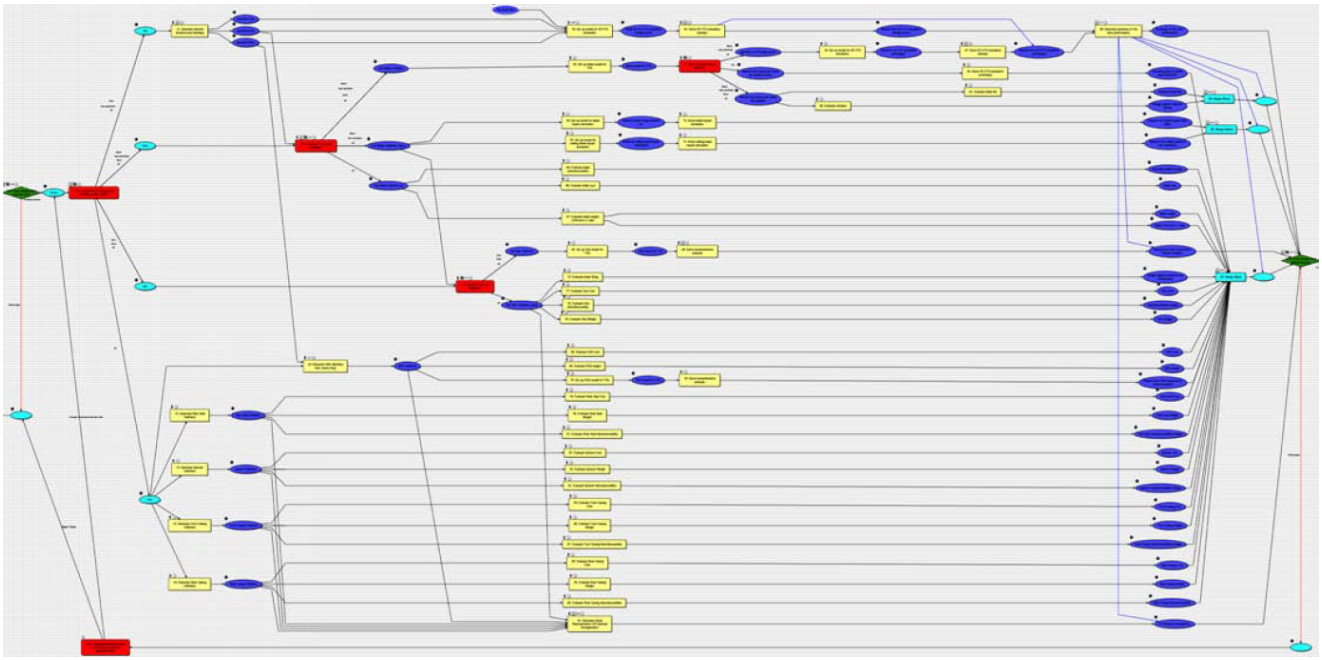

Fig. 17 ASM simulation model of fans design process at a UK aerospace manufacturer. Reproduced from Kerley et al (2011b) pending permission of the Design Society.

be used to evaluate the impact of proposed changes to design confidence levels (modelled using process variables) on the overall design process performance, and was developed and comprehensively evaluated through several applications in jet engine design (Shapiro et al, 2016).

A number of other applications of the approach have taken place over the years, of which some published applications are summarised in Table 3. Overall, this section is intended to illustrate that the ASM2.0 is applicable to larger-scale problems and has been successfully applied in a number of industry-focused projects, including several projects in which neither author was directly involved. We believe this provides a measure of confidence that the approach can assist with the challenges set out in the early sections of this paper and that it is useful to support design process improvement in practice.

6 Discussion

6.1 Recap of contributions

This article has addressed the following research questions:

- *What requirements does the context of CAE-driven engineering design place on an approach for mapping and quantitative evaluation of design processes?*

An appreciation of the requirements for such an approach was developed through a preliminary case study in a large aerospace company, as discussed in Section 2.2. Existing work in the literature addresses some of the identified requirements, that are summarised in Section 2.3—but except for ASM2.0, no approach can address all of them. In particular, no other approach offers a relatively seamless integration of graphical process mapping to assist the elicitation, expression and integration of process knowledge, with configurable quantitative analysis that can be used to develop insights for improvement of engineering design processes that involve partial concurrency and intertwined progressive and corrective iterations. This established the need for the new approach and pinpointed the research gap addressed by this article (Section 2.4).

- *How can these requirements be realised in a pragmatic approach?*

The realisation of each requirement in a new modelling approach, termed the ASM2.0, was described in Section 3 and is summarized in Table 4. As explained at the outset of this article, this approach has been developed and validated through an iterative process involving a series of applications over a number of years.

- *How can the approach be applied to support engineering practice?*

Section 4 explained how the ASM2.0 is applied by detailed discussion of a simplified, but realistic example that pinpoints novel features of the approach and demonstrates why they are necessary. Section 5 reviewed some applications of the approach to model and improve engineering design processes in practice, which requires larger-scale models. As well as answering the research question stated above, these cases provide a measure of validation of the approach. A summary of additional published applications was also provided.

6.2 From ASM to ASM2.0

ASM2.0 as reported for the first time in this article is a significant evolution of the original ASM reported by Wynn et al (2006), made more pragmatic through changes driven by reflection on application studies. Certain features were added, in particular, the inclusion of graphical markup and milestones into the process mapping notation, and perhaps most importantly, the inclusion of the rework propagation and interruption features. These latter features are essential to accurately simulating the corrective iterations common in real engineering design processes. Process variables had been included in the ASM as of Wynn et al (2006) but their application had not yet been explored. In developing ASM2.0 process variables were recognised to be essential to the realistic modelling of progressive iteration. Improvements to many aspects of the process variable functionality were developed, including their hierarchical structuring, their use in task preconditions, priority, and their use as measures of process performance. As well as setting out the features of ASM2.0 and the rationale behind these features in terms of requirements from an industry case study, another contribution of this paper has been the detailed discussion of an application case that sets out, in detail and for the first time, how the ASM2.0 is applied to generate insights for design process improvement by modelling and analysis of iteration impact in CAE-driven, iteration-intensive engineering design. Key elements of the application approach include the progressive configuration of a model's behaviour by adding process variables and the verification of simulation output by detailed examination of process traces. The example in Section 4 shows how this process prompts the modeller to develop insight for improved design process configurations.

Other features included in the original ASM were abandoned for pragmatic reasons that became apparent as the evolving approach was used in more applications.

Table 4 Summary of how ASM2.0 addresses the modelling requirements set out in Section 2.5

Requirement	Corresponding features of ASM2.0
MR1: Be visually familiar and flexible enough to replace informal design process flowcharts.	Deliberately simple notation. Minimally restrictive, allowing any nodes to be connected as long as network is acyclic if Iterate Again arrows are removed. Deliverables specify the information conveyed, and optionally its status at that point in the design process.
MR2: Be scaleable and minimally restrict the size of models that can be created.	Diagrams can be decomposed into subsheets. Flowchart notation designed to minimise verbosity. Information and parameters can be structured into hierarchies. Classification schemes allow display to be filtered to focus the view.
MR3: Evaluate the process arising from modelled tasks, information flows and resourcing.	Discrete event simulation generates a population of process outcomes and indicates the cumulative effects of uncertainties at the task level, upon the overall process.
MR4: Provide precise control over the behaviour of the evaluation model.	Duration, outcome and priority of tasks can be modelled using numerical values, independent probability density functions, or using functions of process variables whose values may be influenced by tasks previously executed.
MR5: Develop the evaluation model by progressively adding depth to a descriptive model.	Simulation algorithm places minimal constraints on diagram allowing detailing of a descriptive model towards simulation. Information required for simulation is configured within tasks to avoid adding diagram complexity.
MR6: Facilitate modelling of progressive iteration.	Progressive iteration can be modelled as tasks revisited several times within an iteration construct's loop. Process variables may be used to specify or influence the outcome of the iteration construct and to modify the properties of tasks on each cycle. No. of iterations and durations of tasks on each cycle can thereby be configured as required.
MR7: Facilitate modelling of corrective iteration.	Corrective iteration ensues when the 'iterate again' scenario of an iteration construct occurs. The algorithm automatically traces forward and interrupts any tasks that would be invalidated because they depend directly or indirectly on the updated information. The algorithm hence computes the emergent effects of rework automatically.
MR8: Facilitate appreciation of how process model features determine evaluation outcomes.	Process traces generated by simulation can be investigated individually, showing every task and the information flow that triggered it, along with iteration outcomes etc.

These are described below along with some reflections on whether the underlying ideas might deserve further research attention, in future.

6.2.1 Requirement for explicitly modelling deliverables

A deliberate feature of the original ASM reported by Wynn et al (2006) was that it did not allow tasks to be connected directly by information flows, but the detail of the information being transferred (i.e. the deliverable) was required to be specified. This arose from several observations in the context of descriptive process modelling in the engineering design process. Firstly, requiring modellers to specify information explicitly was found to lead to more rigorous modelling and in particular helped to avoid overlooking non-design tasks such as data file conversions. Secondly, considering the information required and produced by each task was found to help avoid overlooking information flows themselves—for instance, where a task depends on information produced by another task much earlier in the process, the dependency might be overlooked if visualising a process as a sequence of tasks. These dependencies might, however turn out to be important when considering a reconfiguration of the process.

Although these observations are still believed to be valid, in ASM2.0 the explicit modelling of all input and output deliverables is not strictly required. This is because for many applications, such as the case discussed in Section 4, including all information explicitly would lead to an overly verbose diagram that is difficult to manipulate. In ASM2.0, the decision on whether to strictly include all information required and produced by tasks is therefore left to the modeller.

6.2.2 Automatic graph layout

Early evolutions of the ASM incorporated the idea that a process flowchart could be automatically laid out from the underlying data in order to generate views customised for particular participants (Wynn et al, 2005), for instance showing only those tasks and deliverables relating to their disciplines. The potential benefits of automatically generating filtered views of engineering design or product development processes has been considered by other researchers in the engineering design community (e.g. Browning, 2009). This approach was implemented in early prototypes of the ASM based on a general purpose diagram layout algorithm, but several issues were encountered. One such issue was that

relatively minor changes to the structure of the model, e.g. adding an iteration loop, would sometimes lead to a significantly different diagram layout, disorienting the modeller. Correcting this would require a layout algorithm that was aware of the conventions for process flowchart layout and that was able to handle progressive alterations in a sensible way. Another issue was that modellers wanted freedom to arrange flowcharts in a way that conveyed the structure and flow of the process, and to supplement them with graphical markup such as labels, boxes and swimlanes. This was difficult to reconcile with automatic diagram layouts. Although some initial work was undertaken to address these visualisation problems (Wynn, 2007) it was decided instead to focus on the simulation aspects of the ASM2.0, leading to the approach reported in the present article. Nevertheless, further work to improve the visualisation and manipulation of design process models in specialised computer tools could be of significant value to ease large-scale process modelling in industry practice.

6.2.3 Process library

Another prominent concept in the original ASM was the process library. This was incorporated following the observation that in many engineering design processes, certain groups of tasks are repeated in multiple contexts. For instance, in the model of Bell et al (2007) that was discussed earlier, one such collection of subprocesses are the activities used to define CAD models representing 2D planar sections through the cooled blade, and then generating CAE models to predict thermal behaviour and operating life (Bell et al, 2007). The intention of the ASM process library was that these recurring subprocesses could be created in a library and then inserted as references in one or several process models (Wynn et al, 2006). The referenced subprocesses would be connected via defined ports. The advantage of this approach was thought to be that a library of subprocesses could be gradually developed and improved over time. The construction of future models could in principle be accelerated by assembling suitable subprocesses from the library, and because the system was based on references, improvements within the library (such as further detailing a particular subprocess) would automatically propagate to all the process models that used those processes (Wynn, 2007).

The concept described above was implemented in the original ASM but in doing so, challenges were uncovered. One issue was that although the structure of tasks might stay the same when a library process was used at different points in the model, often the behaviours of those tasks might be different. For example, the number of iterations within the subprocess or the durations of specific tasks might need to be dependent on the context. This led to a complex structure in which each library process would be embedded in a 'container' that would have the ability to override certain properties of the library process. The data structures and modelling operations required to embed a generic library process in the specific context of a process model required unexpected amount of effort to implement and debug, especially considering the multilevel hierarchical structure of a typical ASM model. Additionally, in the applications it was found that the additional complexities of the approach were not justified considering that the modeller could achieve a similar effect by simply copying and pasting groups of tasks throughout their model and then adjusting the configuration as needed. For these pragmatic reasons, the concept of the process library was abandoned in ASM2.0.

6.2.4 Learning factors and fixed numbers of iterations

The earliest iteration of the ASM reported by Wynn et al (2005) allowed for tasks to be configured with explicit learning factors, causing the duration of a task to reduce on consecutive iterations. Additionally, this early approach allowed for iteration constructs to be configured with fixed numbers of iterations, or iterations within a specified minimum and maximum number. However, these options were found to be rarely useful because of the need to configure iteration behaviours in a more flexible way when iteration cycles are nested, as is common in practice and discussed throughout the detailed example in Section 4. For these reasons, this functionality was removed from ASM2.0 and modellers are required to customise the exact behaviour they want using process variables. Potentially, a more sophisticated 'configuration wizard' could be of use to rapidly configure tasks for commonly-encountered situations.

6.3 Limitations of ASM2.0 and areas for future work

The patterns of tasks in CAE-driven, iteration-intensive engineering design processes are more complex than for most transactional process such as manufacturing or document processing. ASM2.0 has been developed with this in mind but even so, simplifications are inevitably needed to model the dynamic aspects of designing as a task network having fixed structure of information flows. Some of these simplifications are described in the next paragraphs.

Firstly, the ASM2.0 model assumes that tasks are not able to absorb information during their execution

without being interrupted and starting again, nor can they release preliminary information or otherwise impact process state during execution. This reflects the initial focus of the work on modelling the technical detail of CAE-intensive component design processes, however, limits application of the model on a higher level of abstraction. For instance, the model is not well suited to investigate end-to-end flows of work in very large-scale processes such as aircraft design that can involve 50+ teams working concurrently. Such processes are too complex to model at the high level of detail discussed in this article, and ASM2.0 does not allow for tasks to be modelled at a higher level of abstraction at which they are worked concurrently while exchanging information throughout. Another factor that limits application of the approach on a very large scale is the need for the modeller to collect, comprehend and integrate the domain knowledge from many process participants. This has also been observed as a bottleneck in business process modelling and management more generally (Dumas et al, 2018).

Additionally, while the model does allow for resource limitations, these are relatively simplistic and designed essentially as a way to control concurrency of the process. In reality, resource limitations are more complex. Individuals are able to contribute to multiple tasks with respect to their expertise and can divide their time among several tasks concurrently. Improving the ability of ASM2.0 to handle realistic resource limitations is another area for potential future work.

Another area for research is suggested by the rework propagation feature of the ASM2.0 simulation. To recap, this was designed to ensure appropriate task sequences by preventing tasks from continuing or beginning if any of the information those tasks depend on, directly or indirectly, is invalidated. Although this is perhaps the ideal case, it seems likely that process participants would not always be aware in practice that they should stop work because of failures in coordination between parts of the process. One opportunity for further work suggested by this observation is to investigate the effects of imperfect overview and imperfect coordination on the effective management of rework. This might yield generally-applicable insight for design process management.

7 Concluding remarks

Well-established engineering design processes often depend heavily on CAE, are highly iterative, and have stable architectures that remain similar as a company designs a range of similar products. However, the processes are complex and knowledge about them is typically tacit and distributed among numerous process participants, such that none of the participants are able to describe the entire process in depth. In consequence it is difficult to appreciate how the engineering design process could be improved or how it might respond to proposed changes, such as the addition of new tasks and analysis tools, increases in concurrency, or changes in resourcing levels. The research reported in this article is based on the premise that mapping and quantitative analysis of engineering design process, with a particular view to representing complex patterns of progressive and corrective iteration, can help practitioners to express and integrate their process knowledge and develop insight for improvement of their processes.

Engineering practitioners are often sophisticated users of computational modelling and their experience with CAE tools for modelling, simulating and designing physical products influences their expectations for tools for modelling and simulating—and designing—design processes. In this article we have drawn some parallels between common CAE practice and the application of process simulation to investigate and improve an engineering design process. These parallels informed our perspective on the features a design process simulation tool needs to offer and the way it should be applied to be most useful to engineering practitioners seeking to improve their processes.

In our approach the purpose of design process modelling is to express and integrate process knowledge that is normally tacit and may also be spread among multiple participants. This generates a commonly-agreed map of the tasks, information flows and iterations in that design process. The purpose of design process simulation is then to compute the combined effects of all modelled factors influencing process outcome, whose details are possible to comprehend and verify individually but which become too complicated in aggregate. For example, if the duration of a task is to increase, it can be expected that the duration of the process might increase. But the ultimate effect of the change on the overall process duration may be difficult to predict because it will depend on e.g. how the task is implicated in progressive and corrective iteration, the probabilities of those iterations occurring, and the availability of resource allowing concurrent execution. Simulation computes the effect of the change considering all modelled factors and their interactions. In our approach the modeller should examine the detail of the simulation trace to verify that the result is realistic. If not, the model should be adjusted and the verification should be repeated until the modeller is satisfied with the insight gained. Supplementing the examples in this article and the applications that are summarised in Table 5, a

Table 5 Some purposes for design process modelling and analysis that can be addressed using ASM2.0

Build consensus and understanding on what needs to be done to complete a design
Capture process knowledge to inform future similar projects
Appreciate what external information needs to be available and by when, to avoid unnecessary waiting
Identify information that is produced earlier than required, and could be deferred for more urgent tasks
Identify information that should be produced earlier to avoid unnecessary waiting by downstream or external tasks
Resequence tasks so that overlapping across concurrent work streams or departments can be increased
Determine the beneficial effect of eliminating or reducing a non-value-adding task, e.g. a data transfer task
Determine the effect of introducing a new task, e.g. to consider a new design objective, and where that task should be placed
Consider whether any process steps can be moved outside iteration loops to avoid unnecessary repetition
Consider whether any tasks can be subdivided so that part of the task is moved outside an iteration loop
Identify any bottlenecks caused by insufficient resources, and their impact on effort and duration
Identify unnecessary iterations caused by starting tasks before their input information is sufficiently mature

range of further design process improvement scenarios in which this approach could be of use are indicated in Table 5.

We have already discussed how ASM2.0 models can be verified by detailed examination of process traces. While this helps to avoid errors in model configuration, it does not establish that the numerical results of simulation are in line with reality. This is a particular challenge when modelling and simulating the engineering design process, because of the limited data available in practice to calibrate models. Noting that a single data point is of limited use to validate a statistical model, the ideal situation would be to compare data obtained through simulation with multiple records of actual processes, to establish that the computed distributions of process duration, effort etc. were in line with the modelled situation. Often however, engineering design processes are infrequently encountered such that many years would need to pass to collect even a few complete records of the process. In that time, the process would likely evolve so that collected numerical data might not be directly comparable. Validation of a simulation model might also be based on records of individual tasks or subprocesses, since these are in some cases encountered more frequently than entire processes, but our observations in a variety of engineering organisations suggests that such data are not usually available in practice and cannot be collected within the timeframe of a typical modelling project. This situation might change as engineering companies become increasingly aware of the value of data, but for now it is usually the reality that design process modellers do not have extensive historical data to calibrate their models.

But even though a design process model is difficult to validate quantitatively, modelling can still offer insight. Developing a model externalises knowledge of the design process, which help the modeller to identify gaps or inconsistencies in their understanding as well as to communicate that understanding to other stakeholders so that it can be critiqued and improved. Simulation helps to further ensure consistent understanding of the process. In particular it ensures that the expectations of different stakeholders regarding how the process will unfold are in accord with one another and satisfy the various constraints (i.e. information flow, iteration and resourcing) that are represented in the process model. For example, if simulation predicts an overall process duration which is not in line with expectations or experience, this implies inconsistency in the modeller's understanding of (1) the durations of the tasks; (2) the way they interact; and (3) the expectations regarding the overall process. In such a situation the modeller is prompted to reexamine their assumptions until the cause of the discrepancy is identified and the model corrected. This process of triangulation allows confidence in a model to be developed even in situations where it cannot be validated numerically.

Overall, the ASM2.0 approach emphasises development of understanding of the mechanisms that drive process performance, so that improvements can be proposed with confidence. This differs from some other design process modelling and simulation approaches in which the emphasis is placed on computing the emergence of unpredictable outcomes, without necessarily investigating the relationship between those outcomes and the modelling assumptions from which they are derived. It is hoped that the description and examples in this article will serve to convey the pragmatic emphasis of the ASM2.0 approach.

Acknowledgements The authors gratefully acknowledge past and present collaborators who used and provided feedback on the ASM2.0 during its development, as indicated in the

article text and bibliography. Particular acknowledgement is due to: Claudia M. Eckert, Warren P. Kerley, H. Nam Le, and Michael Moss. This work would not have been possible without the programming contribution of Seena Nair to the Cambridge Advanced Modeller software. The first author also thanks Xun Xu for discussions on teaching the bracket project over the years.

References

Van der Aalst WM (1998) The application of Petri nets to workflow management. Journal of Circuits, Systems, and Computers 8(1):21–66

Ahmed S, Wallace KM, Blessing LTM (2003) Understanding the differences between how novice and experienced designers approach design tasks. Research in Engineering Design 14(1):1–11

Bell C, Clarkson PJ, Dawes WN (2008) Improving the turbine cooling conceptual design process. In: ASME Turbo Expo 2008: Power for Land, Sea, and Air, American Society of Mechanical Engineers, pp 2631–2640

Bell CP, Wynn DC, Dawes WN, Clarkson P (2007) Using meta-data to enhance process simulation and identify improvements. International Conference on Engineering Design

Bracewell R, Wallace K, Moss M, Knott D (2009) Capturing design rationale. Computer-Aided Design 41(3):173 – 186

Braha D, Bar-Yam Y (2007) The statistical mechanics of complex product development: empirical and analytical results. Management Science 53(7):1127–1145

Browning TR (2009) The many views of a process: Toward a process architecture framework for product development processes. Systems Engineering 12(1):69–90

Browning TR (2016) Design structure matrix extensions and innovations: A survey and new opportunities. IEEE Transactions on Engineering Management 63(1):27–52

Browning TR, Eppinger SD (2002) Modeling impacts of process architecture on cost and schedule risk in product development. IEEE Transactions on Engineering Management 49(4):428–442

Clarkson PJ, Hamilton JR (2000) 'Signposting', a parameter-driven task-based model of the design process. Research in Engineering Design 12(1):18–38

Colquhoun GJ, Baines RW, Crossley R (1993) A state of the art review of IDEF0. International Journal of Computer Integrated Manufacturing 6(4):252–264

Crowder RM, Robinson M, Hughes HP, Sim YW (2012) The development of an agent-based modeling framework for simulating engineering team work. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 42(6):1425–1439

Dori D (2002) Object-Process Methodology: A holistic systems paradigm. Springer

Dumas M, La Rosa M, Mendling J, Reijers HA (2018) Fundamentals of business process management, 2nd edn. Springer-Verlag, Berlin

Eckert CM, Clarkson PJ (2005) The reality of design. In: Clarkson PJ, Eckert CM (eds) Design process improvement: A review of current practice, Springer, London, pp 1–29

Eppinger SD, Whitney DE, Smith RP, Gebala DA (1994) A model-based method for organizing tasks in product development. Research in Engineering Design 6(1):1–13

Ha S, Suh HW (2008) A timed colored Petri nets modeling for dynamic workflow in product development process. Computers in Industry 59(2):193–209

Hassannezhad M, Cantamessa M, Montagna F, Clarkson PJ (2019) Managing Sociotechnical Complexity in Engineering Design Projects. Journal of Mechanical Design 141(8), 081101

Karniel A, Reich Y (2011) Managing the dynamics of new product development processes: a new product lifecycle management paradigm. Springer Science & Business Media

Kerley W, Wynn DC, Eckert C, Clarkson PJ (2011a) Redesigning the design process through interactive simulation: a case study of life-cycle engineering in jet engine conceptual design. International Journal of Services and Operations Management 10(1):30–51

Kerley WP, Wynn D, Moss MA, Eckert C, Clarkson PJ (2008) Using simulation to support integration of lifecycle engineering activities into an existing design process: A case study. In: Marjanović D, Štorga M, Pavković N, Bojčetić N (eds) Proceedings of DESIGN 2008, the 10th International Design Conference, Dubrovnik, Croatia, Design Society, pp 1033–1042

Kerley WP, Armstrong G, Pepe C, Moss MA, Clarkson PJ (2011b) Using simulation to support process integration and automation of the early stages of aerospace design. In: Culley SJ, Hicks BJ, McAloone TC, Howard TJ, Clarkson PJ (eds) Proceedings of the 18th International Conference on Engineering Design (ICED 11), Lyngby/Copenhagen, Denmark, Aug 15-19, Design Society, vol 1, pp 134–146

Kreimeyer M, Lindemann U (2011) Complexity metrics in engineering design: Managing the structure of design processes. Springer

Krishnan V, Eppinger SD, Whitney DE (1997) A model-based framework to overlap product development activities. Management Science 43(4):437–451

Kusiak A, Larson TN, Wang J (1994) Reengineering of design and manufacturing processes. Computers and Industrial Engineering 26(3):521–536

Le HN, Wynn DC, Clarkson PJ (2012) Impacts of concurrency, iteration, design review, and problem complexity on design project lead time and error generation. Concurrent Engineering: Research and Applications 20(1):55–67

Levitt RE, Thomsen J, Christiansen TR, Kunz JC, Jin Y, Nass C (1999) Simulating project work processes and organizations: Toward a micro-contingency theory of organizational design. Management Science 45(11):1479–1495

Lyneis JM, Ford DN (2007) System dynamics applied to project management: a survey, assessment, and directions for future research. System Dynamics Review 23(2/3):157–189

McMahon CA, Xianyi M (1996) A network approach to parametric design integration. Research in Engineering Design 8(1):14–31

Nelson RG, Azaron A, Aref S (2016) The use of a GERT based method to model concurrent product development processes. European Journal of Operational Research 250(2):566 – 578

O'Donovan BD, Eckert CM, Clarkson PJ (2004) Simulating design processes to assist design process planning. In: ASME 2004 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Salt Lake City, Utah, Sep 28–Oct 2, American Society of Mechanical Engineers, vol 3a, pp 503–512

Park H, Cutkosky MR (1999) Framework for modeling dependencies in collaborative engineering processes. Research in Engineering Design 11(2):84–102

Pepe C, Whitney D, Henriques E, Farndon R, Moss M (2011) Development of a framework for improving engineering processes. In: Culley SJ, Hicks BJ, McAloone TC, Howard TJ, Clarkson PJ (eds) Proceedings of the 18th International Conference on Engineering Design (ICED 11), Lyngby/Copenhagen, Denmark, Aug 15-19, Design Society, vol 1, pp 417–428

Perišić MM, Štorga M, Podobnik V (2018) Agent-based modelling and simulation of product development teams. Tehnički vjesnik 25(Supplement 2):524–532

Pocock J (1962) PERT as an analytical aid for program planning—its payoff and problems. Operations Research 10(6):893–903

Pritsker A (1966) GERT: Graphical evaluation and review technique. Memorandum RM-4973-NASA April 1966

Rajapaksha J, Mirkovic K, Robinson D, Wynn D, et al (2017) Modelling and simulating the effect of coordination on pd performance while handling change. In: DS 87-2 Proceedings of the 21st International Conference on Engineering Design (ICED 17) Vol 2: Design Processes, Design Organisation and Management, Vancouver, Canada, 21-25.08. 2017, pp 179–188

Romero F, Company P, Agost MJ, Vila C (2008) Activity modelling in a collaborative ceramic tile design chain: an enhanced IDEF0 approach. Research in Engineering Design 19(1):1–20

Shapiro D, Hamraz B, Sommer AF, Clarkson PJ (2015) Investigating the impact of changes in iteration-likelihoods on design process performance. Concurrent Engineering 23(3):250–264

Shapiro D, Curren MD, Clarkson PJ (2016) DPCM: A method for modelling and analysing design process changes based on the Applied Signposting Model. Journal of Engineering Design 27(11):785–816

Sharon A, de Weck OL, Dori D (2013) Improving projectproduct lifecycle management with modelbased design structure matrix: A joint project management and systems engineering approach. Systems Engineering 16(4):413–426, DOI 10.1002/sys.21240

Tahera K, Earl C, Eckert C (2017) A method for improving overlapping of testing and design. IEEE Transactions on Engineering Management 64(2):179–192

Taylor BW, Moore LJ (1980) R&D project planning with Q-GERT network modeling and simulation. Management Science 26(1):44–59

USAF (1981) ICAM Architecture Part II—Volume IV—Function Modeling Manual (IDEF0), AFWAL-TR-81-4023. Tech. rep., Materials Laboratory, Air Force Wright Aeronautical Laboratories, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio, USA

Wynn DC (2007) Model-based approaches to support process improvement in complex product development. PhD thesis, University of Cambridge

Wynn DC, Clarkson PJ (2018) Process models in design and development. Research in Engineering Design 29(2):161–202

Wynn DC, Eckert CM (2017) Perspectives on iteration in design and development. Research in Engineering Design 28(2):153–184

Wynn DC, Eckert CM, Clarkson PJ (2005) A model-based approach to improve planning practice in collaborative aerospace design. In: ASME 2005 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Long Beach, California, USA, Sep 24-28, American Society of Mechanical Engineers, vol 5a, pp 537–548

Wynn DC, Eckert CM, Clarkson PJ (2006) Applied Signposting: a modeling framework to support design process improvement. In: ASME 2006 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Philadelphia, Pennsylvania, USA, Sep 10-13, American Society of Mechanical Engineers, vol 4a, pp 553–562

Wynn DC, Grebici K, Clarkson PJ (2011) Modelling the evolution of uncertainty levels during design. International Journal on Interactive Design and Manufacturing 5(3):187–202

Xin Chen HL, Moullec ML, Ball N, Clarkson PJ (2016) Improving design resource management using Bayesian network embedded in task network method. In: ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Charlotte, North Carolina, USA, Aug 21-24, American Society of Mechanical Engineers, vol 7, p V007T06A034: 15 pages

Zhang X, Hao Y, Thomson V (2015) Taking ideas from paper to practice: A case study of improving design processes through detailed modeling and systematic analysis. IFAC-PapersOnLine 48(3):1043–1048