INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'07
28 - 31 AUGUST 2007, CITE DES SCIENCES ET DE L'INDUSTRIE, PARIS, FRANCE

PRE-EMPTIVE CONCURRENT DESIGN PLANNING AND SCHEDULING

Peter Matthews[1] and Graham Coates[1]
[1]School of Engineering, University of Durham

ABSTRACT
This paper introduces a significant revision to the Concurrent Engineering (CE) methodology that enables a shortened project completion time. Under the CE methodology, sequential tasks can only be performed as such. We introduce a method for starting sequential tasks concurrently using a pre-emptive approach. Where there are a, suitably small, finite number of possible alternative subsequent tasks, we propose that a more agile approach is to begin work on these alternative subsequent tasks concurrently to the preceding task, sharing the resource needed for the subsequent task amongst the different alternatives. Further, where the probability for each alternative task is known, we demonstrate that by setting the resource allocation equal to the probabilities of each outcome, it is possible allocate resources to minimise the expected completion of the overall project. A simple two task case study is developed and analysed to illustrate this method. The paper concludes by revisiting the original assumptions and discussing how resource efficiency is traded off for minimising project completion time.

Keywords: Agility, Process Simulation, Concurrent Engineering, Resource Allocation

1 INTRODUCTION
A key competitive edge can be gained from being able to complete design and manufacture projects more rapidly than the competition. With this principle in mind, this paper introduces a methodology to support this goal and analyses the cost of achieving this aim. This paper considers how a sequential process-based system can allocate resources to decrease total overall process time. This is achieved through an 'agile' methodology that starts pre-emptively processing in parallel a set of alternative potential outcomes before the actual outcome is known. The methodology proposed in this paper builds on the Set Based Concurrent Engineering (SBCE) [Terwiesch et al, 2002; Nahm & Ishikawa, 2006], which describes the nature and quality of information that must be passed between tasks and augments it to determine the optimal resource allocation to multiple alternative outcomes to minimise the expected overall process completion time.

Fundamentally, this approach revises the Concurrent Engineering (CE) methodology. Where CE has the distinct concepts of concurrent tasks, which can be performed in parallel, and sequential tasks, which can only be performed in sequence, the pre-emptive resource allocation approach allows for sequential tasks to be performed concurrently. A number of fields already informally adopt this principle, for example when a news editor is reporting on an election story, the journalists will be requested to submit both a "President Re-elected" and "President Defeated" version of the story *before* the election result is known. When the election outcome is officially declared, the editor selects the correct version of the story and updates any details, for example the exact poll result. This provides agility to the editor, enabling a more rapid news production process. The cost of this is efficiency: in this case the effort of writing the unprinted story is wasted.

The remainder of this paper is structured as follows: Section 2 provides the key background material that forms the basis of this work. Section 3 introduces the theoretical arguments and the problem representation. This is followed in Section 4 by a detailed case study, using simulation software to illustrate the agile pre-emptive resource allocation methodology. Finally, Sections 5 and 6 respectively discuss the case study results and conclude the paper.

2 BACKGROUND

There are three main components supporting this work: Concurrent Engineering, Agile methodologies and Resource Allocation. Concurrent Engineering provides the basis of this work and Agile ideas are used to add significant revision to the CE and resource allocation methodologies.

Concurrent engineering is the distribution of the design, and potentially manufacture, work between a number of agents [Carter & Baker, 1991]. An agent in this case will be a design team, manufacturing shop, assembly facility, or some other related facility. These agents can then either recursively apply concurrent engineering again, or follow the basic linear design process if they are a 'terminal' agent. For example, the design team might sub-contract some of the design work to another specialist design group. The agents are selected according to their known expertise [Armoutis & Bal, 2003]. These agents are networked through a virtual enterprise while the project is underway and combining the work towards the end of the project. Through this network, agents communicate as necessary. This concurrent engineering approach provides a means for rapidly creating enterprises with high degrees of competency without the need to support these competencies during projects which do not require the same competency profile. Thus, the virtual enterprise has the benefits of a large, well found enterprise, without having to pay for the maintenance overhead of resources that are not required for other projects.

The concept of agility in a manufacturing context has recently emerged [Lau et al, 2003; Jiang & Fung, 2003]. Most authors agree that 'agility' is the ability to rapidly respond to some external and unexpected event. The argument promoting agility is that it enables better survival in turbulent market conditions. However, most agile responses are tailored to changes in product demand, either in the form of production levels or in alternate design. The solutions to these tend to fall in line with traditional manufacture theory (for example, by applying Just-in-Time methods) or design modification (such as mass customisation applied after the initial product launch). Thus, enterprises use the agile methods to enable them to respond to the market, based on a given design.

One of the key ideas is taken from the agile software development community [Cockburn, 2005]. Where in CE a set of tasks in sequence cannot begin until the precedent tasks have been completed, the software development approach is to pre-emptively begin work on the subsequent tasks earlier. As more information becomes available to the subsequent tasks from the preceding tasks, rework must be done. However, provided the total amount of rework plus the remaining work on the subsequent tasks does not exceed the total time it would have originally taken for the subsequent tasks, a time saving is made. This time saving results in a more rapid deployment of the finished goods. Agile methods are required for this approach, as rework introduced while the process is ongoing represents the need to be able to change the tasks with minimal impact to the overall project.

Resource allocation is an area of key significance in manufacturing [Tharumarajah, 2001; Maropoulos et al., 2003; Wallace, 2003; Wu et al., 2005], engineering/project planning [Alcaraz & Maroto, 2001; Kara et al., 2001; Leus & Herroelen, 2004; Ursu et al., 2005; Guikema, 2006] and agent systems [Chen, 2004; Galstyan et al., 2005; Li and Soh, 2005; Manvi et al., 2005; Raftopoulou et al. 2005]. In the context of manufacturing, Tharumarajah (2001) presents a survey of resource allocation methods for distributed manufacturing systems. Further, a number of issues are highlighted that are involved in distributing tasks such as how problems are decomposed, how tasks are assigned to who, and communication between resources and tasks. Similarly, Wu et al. (2005) indicate that the inherent decentralised nature of many organisations leads to investigations focussing on distributed, rather than centralised, methods for the resource allocation problem. On the theme of distributed resource allocation problem in manufacturing systems, Wallace (2003) presents a method utilising agents to manage the sequential allocation of resources. The key finding of this work was that agent-based allocation of resources utilising resource, group and actor agents has the potential to manage sequential resource allocation without encountered deadlocks. Maropoulos et al. (2003) introduce a resource model that has been developed to aid the dynamic planning of manufacturing operations within production networks.

With regard to project planning, Leus and Herroelen (2004) assert that projects involving scheduling resources often assume that perfect information is available. As this is rarely the case, a branch-and-bound algorithm is presented aimed at solving the resource allocation problem in settings where uncertainty exists and variability in task durations. On the same theme, Alcaraz and Maroto (2001) present a genetic algorithm aimed at optimising the resource allocation problem in project scheduling. Ursu at al. (2005) also present an optimisation algorithm to solve the workforce allocation problem using a distributed system of agents. The SignPosting methodology [Clarkson and Hamilton; 2000] uses the design information quality level (eg, rough estimate, first order calculation, fine grain calculation) as a means to suggest possible subsequent design tasks to be undertaken.

An approach to allocating resources to members of a concurrent design team has been presented by Guikema (2006). In the domain of concurrent engineering (CE), Kara et al. (2001) recognises that a complexity of CE in that it involves executing tasks with incomplete information. To address this complexity, a multi-project scheduling heuristic is presented, which is reported as minimising project completion time through concurrency and optimising the utilisation of resources. The Set Based Concurrent Engineering paradigm [Nahm & Ishikawa, 2006] represents fuzzy estimates of required final information as a means to commence work pre-emptively. By considering the intersection of parameter values from multiple alternatives, it aims to identify feasible regions in which to invest pre-emptive work. However, SBCE does not consider likelihood of final parameter outcomes and hence does not bias the future estimates or work resource allocation towards more likely outcomes.

3 THEORETICAL ARGUMENT

To construct the argument for this work, it is sufficient to consider a project consisting of two sequential tasks, Task 1 followed by Task 2. This is a problem structure that the CE approach cannot address or improve upon as it requires the tasks to be performed in sequence. Under the CE methodology, the project will first complete Task 1 which provides the full information for Task 2 to be undertaken, and only then does Task 2 begin. The 'pre-emptive' approach will be to start Task 2 with incomplete information. More specifically, provided there are a sufficiently small number of distinct possible alternatives of Task 2 depending on the outcome of Task 1, the pre-emptive approach will begin work on all these alternatives concurrently. The question remains is how to optimally allocate resources to each of the Task 2 alternatives.

A number of assumptions are made regarding the tasks. Firstly, there are a finite number of alternative versions of Task 2. Realistically, it will be necessary that there are a reasonably small number of alternatives, otherwise the resources available for Task 2 are spread too thinly and insufficient work can be completed on this task. Secondly, it is assumed that at least an approximate probability distribution function (PDF) is known for the various outcomes. A naïve approximation would be to set all probabilities equal. Finally, it is assumed that the resources required for Task 1 are distinct from those required for Task 2. For example, a project might require process time on two different computers with different software packages. Effectively, it assumes that the resources of Task 2 are idle while Task 1 is underway.

3.1 Problem representation

At this point, this work only considers the time aspect of a project. The time required for Task i is denoted $t(i)$, and for simplicity the time will be represented in the reference frame of each task. Therefore, the start of each task is at time 0 with respect to that task. Clearly, the aim is to minimise the time in terms of the global reference frame. This is achieved by referencing to the earliest task, in our case Task 1.

The time where Task 1 has sufficient information to start a sufficiently small set of alternative versions of Task 2 is denoted by t_0. The probability of each alternative being the 'correct' outcome at the end of Task 1 is denoted p_i. Each alternative task is allocated a proportion of the total available resource for Task 2, and this is denoted by α_i.

The aim is to determine the optimal resource allocation for the given PDF in terms of expected time to complete Task 2. Once Task 1 has terminated, the 'correct' alternative will be known. Therefore, all of Task 2 resources can be committed to the correct, partially completed, alternative. All other alternatives are terminated at this point.

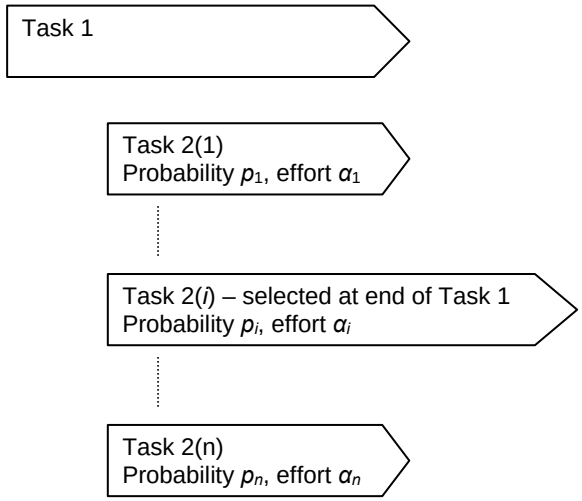

Figure 1. Temporal representation of an Agile Pre-emptive task processing for two sequential tasks

3.2 Equations: equal length alternatives

The basic case is where all alternative versions of Task 2 require the same amount of processing time, for example the same calculation is performed using different starting conditions. Figure 1 illustrates the schema of the project execution. A critical value here is the amount of time where all the alternatives are being processed, given by $t^{(1)} - t_0$, ie the time between the pre-emptive start of Task 2 and the end of Task 1. Therefore, the amount of work done on each alternative i of Task 2 is given by:

$$\alpha_i \frac{(t^{(1)} - t_0)}{t^{(2)}} \tag{1}$$

Hence, at the end of Task 1, the time remaining for alternative i is given by:

$$\begin{aligned} \Delta t_i &= t^{(2)}\left(1 - \alpha_i \frac{(t^{(1)} - t_0)}{t^{(2)}}\right) \\ &= t^{(2)} - \alpha_i (t^{(1)} - t_0) \end{aligned} \tag{2}$$

Recall, the aim is to minimise the expected total time taken to complete Task 2. Therefore, we aim to minimise:

$$\begin{aligned} \min \mathbf{E}(\Delta t) &= \sum p_i \Delta t_i \\ &= \sum p_i t^{(2)} - \sum p_i \alpha_i (t^{(1)} - t_0) \\ &= t^{(2)} - (t^{(1)} - t_0) \sum p_i \alpha_i \end{aligned} \tag{3}$$

From the above equation, it can be seen that the completion time is minimised when the second term (the remaining summation) is maximised. This summation is effectively the 'dot product' between the PDF vector and the resource allocation vector, and is maximised when the two vectors are parallel. As both vectors must also sum to unity, the project completion time is minimised when the resource distribution is set to the probability distribution.

3.3 Equations: varying length alternatives

The more general case is where each alternative version of Task 2 requires a different amount of processing time, namely where Task 1 determines which of a set of very different subsequent tasks

would be performed. For example, a car design project where Task 1 determines what type of propulsion to use (for example either: petrochemical, electric or hybrid) and Task 2 is the design of this propulsion system. As each propulsion system in significantly different, each will require a different amount of time to complete.

The time required to complete each alternative version of Task 2 will now be denoted as $t_i^{(2)}$. Equation 3, representing the expected remaining completion time of Task 2 once Task 1 has completed is now given by:

$$\begin{aligned} \min \mathbf{E}(\Delta t) &= \sum p_i \Delta t_i \\ &= \sum p_i t_i^{(2)} - \sum p_i \alpha_i (t^{(1)} - t_0) \\ &= \sum p_i t_i^{(2)} - (t^{(1)} - t_0) \sum p_i \alpha_i \end{aligned} \tag{4}$$

This equation has the same structure as Equation 3, namely a constant term less a term dependent on the resource allocation vector. Again, the overall expression is minimised when the resource allocation distribution is equal to the PDF.

4. CASE STUDY AND SIMULATION

As an example, the conceptual design of an aircraft fuselage requires a number of analyses to be performed to make an initial assessment of the structure's geometry and configuration. Based on the structural engineer's experience, judgement and knowledge of the loading cases involved, an initial idealisation can be proposed. This idealisation involves defining the fuselage diameter, number of booms representing stringers and longerons. In addition, the location and area of these booms is required. Based on bending and shear flow analysis, direct and shear stresses can be determined that inform the next idealisation of the structure.

In the context of this work, Task 1 would be the initial representational of the fuselage (i.e. structural idealization). Task 2 would then take, say, five possible directions, depending on the outcome of Task 1. By assessing Task 1 during its execution a PDF is defined for Task 2.

4.1 Computer-based simulation

The theoretical benefits of pre-emptive approach as described in the previous section were illustrated using a computer-based simulation. To do this Simul8 was used to model 2 tasks traditionally carried out in series. Simul8 is a desktop PC simulation environment which has been in use for around 10 years. It is traditionally used for simulating the manufacturing environment, making use of Work Cells, Work Items, Resources, Conveyors and more. However it is possible with some manipulation, as will be demonstrated, to use this software for the simulation of less physical and more theoretical scenarios such as design tasks. Results can be recorded for the majority of variables in the model, such as average time for a product to be in the system, Work Cell waiting time, working time and blocked time etc.

4.2 Implementation of the simulation

The scenario of the aircraft fuselage is considered, with two traditionally serial tasks – the initial representation followed by the detailed definition of the booms, longerons and their positions. This can be seen in the control part of the simulation (Figure 2).

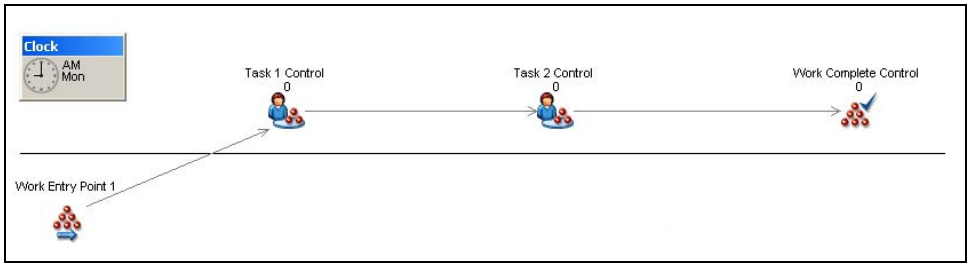

Figure 2. Simul8 graphical representation of control experiment

Each of the 2 tasks was assigned an Operating Time of 100 time units, and the travel times between tasks were set to zero, as they were not relevant at this stage of the research. This means that the control experiment would take a total of 200 time units to complete the 2 serial tasks.

For the experimental half of the simulation shown in Figure 3, the task was configured in a similar way, but making use of the Visual Logic capabilities of Simul8 to assign Labels to Work Items and therefore manipulate the Operating Time of Task 2. This allowed the Operating Time of Task 2 to be reduced dynamically based on whichever of the Task 2 Options eventually proves to be the selected one.

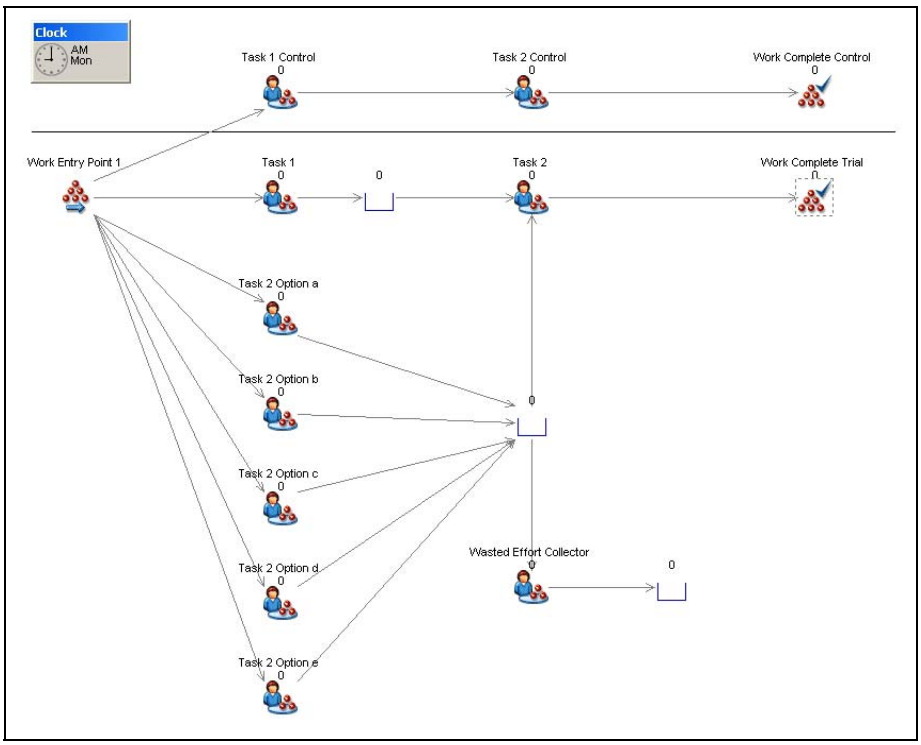

Figure 3. Simul8 Graphical Representation of scenario with 5 potential options for Task 2

The simulation is started when Work Entry Point 1 distributes a Work Item (for example a design brief) to Task 1 Control, Task 1, and each of the Task 2 Options. This allows the five Task 2 Options (a-e) to be started at the same time as Task 1 (a delay will be introduced later). Each of the items distributed has the same label attached to it (called Task 2 Option) stipulating which of the 5 Task 2 Options (a-e) will be the option to be used during Task 2. The value of the Task 2 Option label is not random, but is determined according to a Probability Distribution Function (Table 1).

Table 1. Probability Distribution Function for Task 2 Options

Task 2 Option	*Probability*
a	10%
b	15%
c	20%
d	25%
e	30%

Upon receiving the Work Item from Work Entry Point 1, the five Task 2 Options each change the Task 2 Option label to their own value, (a-e). Each of the Task 2 Options also has an additional label named Task 2 Work Remaining which holds the value for the amount of work that is said to have been completed during that Task 2 Option task. These values are manually set for each of the Task 2 options using Equation 4.

This method simulates the allocation of resources between the five Task 2 Options in accordance with the probability of each option occurring: more resources are allocated to work on an option which is more likely to occur.

Upon completion of Task 1, the Work Item passes into the storage buffer, as do the five Task 2 Options. Task 2 is then configured to select the single item from the Task 1 storage buffer and read its Task 2 Option label set by Work Entry Point 1 to a, b, c, d or e. Task 2 then matches that label to a Work Item from the Task 2 Options storage buffer and obtains that item for 'assembly'. In completing this process, the Operating Time for Task 2 is then updated to equal the value of the Task 2 Work Remaining label. The Task 2 Options which were not used for Task 2 are collected by the Wasted Effort Collector so that they cannot be considered for a future Task 2. Task 2 is then completed using the new Task 2 Operating Time and the Work Item passes out of the system to the Work Complete Trial area.

4.3 Results

The results for the initial simulation are presented in this section. The simulation was run 100 times to allow for the probability distribution function allocating the Task 2 Option value to be fully utilised. The results show that the average time benefit of starting the five options at the same time as Task 1, and allocating the resources according to the probability for each option, was a saving of 22.6 time units from Task 2, which has a Control time of 100 minutes. This is an 11.3% reduction in processing time for the overall project.

However this must be contrasted with the loss of 'efficiency', which is calculated as a percentage of resource use which contributes to the final process. For each process there are five options, one of which will be chosen, and four of which will consume resources before being discarded. The efficiency can therefore be calculated as follows:

$$Useful = \alpha_i \left(t^{(1)} - t_0 \right) \tag{5}$$

$$Waste = \sum_{j \neq i} \alpha_j \left(t^{(1)} - t_0 \right) = \left(1 - \alpha_i \right) \left(t^{(1)} - t_0 \right) \tag{6}$$

$$Total = t^{(1)} - t_0 + \Delta t_i \tag{7}$$

Therefore, in this first simulation the average efficiency would be:

$$Efficiency = \frac{Useful}{Total} = \frac{100}{177.4} = 56.3\% \tag{8}$$

Therefore, efficiency has dropped from 100% for the control experiment where each task is done in serial, to 56.3% where five options for the second task are each started at the same time as the first task. However this increase in efficiency leads to a 22.6 % reduction in Task 2 Operating Time.

4.4 Delayed start of Task 2 Options

The previous example made the assumption that each of the five options for Task 2 would begin at the same time as Task 1 and would continue for the full Task 1 Operation Time. However there may be a benefit in delaying starting the options for Task 2 until some later point during Task 1. Table 2 Shows the results of efficiency and reduction in total process for ten scenarios where the starting of the five Task 2 Options is delayed, and the results are illustrated in Figure 4.

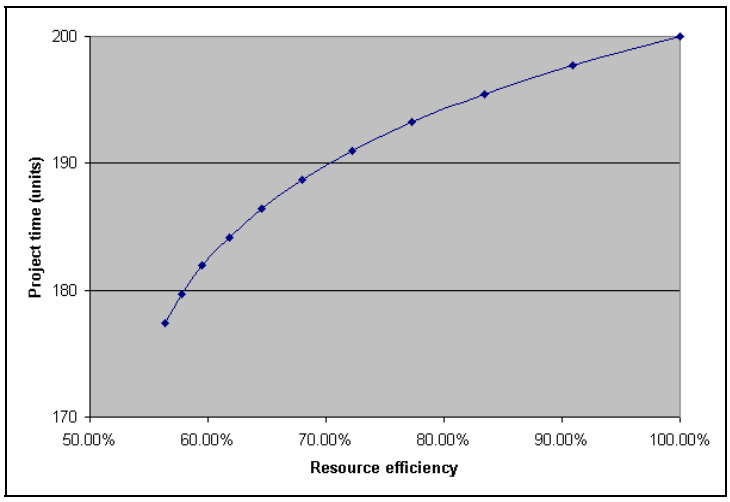

Figure 4. Resource efficiency plotted against Overall Project Time

4.5 PDF generating support tool

An important aspect of the pre-emptive theory lies within the ability of generating 'good' estimates of the PDF for the various task outcomes. The challenge herein is the general difficulty in estimating these probabilities where little prior evidence is available. Further, human decision makers in general find it difficult to place exact probability values on various outcomes. Hence, a support tool is needed to guide designers towards reasonable probability estimates.

The PDF generating support tool initially uses the naïve estimate that all outcomes are equally likely. Next, the design team can rank the potential outcomes. Finally, a rough gradient needs to be placed on this ranking. This is achieved by providing relative outcomes between a subset of the alternatives (e.g. alternative 1 is twice as likely an outcome than alternative 4). Initially, a linear gradient is used between these estimated fixed points. This overall shape can then be adapted as the designer sees fit. Finally, as the PDF must sum to unity, the designer based profile can be transformed into an estimated PDF.

5. DISCUSSION

As a result of running the series of simulations, an interesting graph can be plotted (see Figure 4). The shape of this graph shows that the earlier Task 2 is pre-emptively started, the greater the proportion of waste. Further, this is not a linear relationship. This is to be expected, as clearly the later Task 2 is started the amount of work 'wasted' disappears to zero when Task 2 is started on completion of Task 1. At the other extreme, the earlier Task 2 is started will always result in some non-zero amount of Task 2 being usefully completed on completion of Task 1. As can be seen from the graph, the efficiency has an initial rapid rise, after which there is a diminishing rate of return for further delaying the pre-emptive start. However, this graph does not provide a clear point to an optimal time to start the pre-emptive work. Instead, under this case study, it remains a largely subjective decision as to how much wasted work is acceptable.

A further point to note with the presented case study regards the realism of being able to perfectly subdivide the resource for Task 2 arbitrarily, and with no context switching costs. This might be feasible where Task 2 is a strict computational job, and context switching is relatively inexpensive or where the resources available for Task 2 are in such abundance (e.g. a project team of 100 people) that this is possible. A more realistic scenario should also include overheads that are incurred with each alternative task.

Finally, this case study defined the PDF of all the alternative tasks and used this data to allocate resources. Within a real scenario this is an unlikely condition. This raises the question of how sensitive the pre-emptive system is to the accuracy of the estimated PDF.

6. CONCLUSIONS

The case study demonstrated that an overall reduction in project completion time can be achieved using pre-emptive resource allocation. This pre-emptive approach provides a level of agility to a project, however it comes at a cost of overall efficiency. Under this scheme, a proportion of resources are used to generate wasted work. Under the assumptions, this did not matter as the key objective was to minimise the overall project completion time. The approach taken in this paper could easily be adapted to consider more complex and complete objective functions. These objectives should include other costs incurred, and therefore partially wasted, by starting the subsequent task early.

Future work will also need to revisit the other assumptions. It was assumed that the PDF between the various alternative subsequent tasks remained static with time. This is unlikely, and a more typical scenario will be one where while the primary task completes a continuously improving image of the PDF becomes available, and therefore a dynamic PDF should be considered. As a result, what is the optimal resource allocation strategy under these dynamic conditions? Further links should be made to the occurrence of unexpected (external) events, thereby increasing the agility of this approach.

One potential means for addressing the dynamic nature of the PDF is to consider the iterative nature of the design process. This provides a means for updating the PDF with each iteration of the first task. Under the 'traditional' CE perspective, the subsequent task is only started once sufficient iterations of the first task have been undertaken to provide sufficiently high quality starting conditions for the subsequent task.

Finally, it was assumed that the PDF of the various alternatives was readily available. It will be necessary to conduct field research to measure how realistic an assumption this is, and to develop methods for approximating the shape of the PDF.

ACKNOWLEDGEMENTS

The authors wish to thank Chris Lomas for his supporting contributions in the implementation of the case study in Simul8.

REFERENCES

[1] Alcaraz, J. and Maroto, C. A Robust Algorithm for Resource Allocation in Project Scheduling. *Annals of Operations Research*, 102(1/4), 83-109, 2001.

[2] Armoutis, ND and J Bal. Building the Knowledge Economy: Issues, Applications, and Case Studies, chapter *E-Business through Competence Profiling*, pages 474-482. IOS Press, 2003.

[3] Carter, DE and B S Baker. Concurrent Engineering: The Product Development Environment for the 1990s. Addison-Wesley, 1991.

[4] Chen, Z-L. Simultaneous Job Scheduling and Resource Allocation on Parallel Machines. *Annals of Operations Research*, 129(1-4), 135-153, 2004.

[5] Clarkson, P.J. and Hamilton, J.R. Signposting: a parameter-driven task-based model of the design process. *Research in Engineering Design*, 12(1), 18-38, 2000.

[6] Cockburn, A. Two case studies motivating efficiency as a "spendable" quantity, *Proceedings of the International Conference on Agility* (ICAM2005), 1-6 2005.

[7] Galstyan, A., Czajkowski, K. and Lerman, K. Resource Allocation in the Grid with Learning Agents. *Journal of Grid Computing*, 3(1-2), 91-100, 2005.

[8] Guikema, S. Incentive compatible resource allocation in concurrent design. *Engineering Optimization*, 38(2), 209-226, 2006.

[9] Jiang, Z and R Y K Fung. An adaptive agile manufacturing control infrastructure based on TOPNs-CS modelling. *International Journal of Advanced Manufacturing Technology*, 22:191-215, 2003.

[10] Kara, S. Kayis, B. and Kaebernick, H. Concurrent Resource Allocation (CRA): A Heuristic for Multi-Project Scheduling with Resource Constraints in Concurrent Engineering. *International Journal of Concurrent Engineering: Research and Applications*, 9(1), 64-73, 2001.

[11] Lau, HCW, C W Y Wong, K F Pun, and K S Chin. Virtual agent modelling of an agile supply chain infrastructure. *Management Decision*, 41(7):625-634, 2003.

[12] Li, X. and Soh, L-K. Hybrid negotiation for resource coordination in multiagent systems. *Web Intelligence and Agent Systems: An International Journal*, 3(4), 231-259, 2005.
[13] Lues, R. and Herroelen, W. Stability and resource allocation in project planning. *IIE Transactions*, 36(7), 667-682, 2004.
[14] Manvi, S.S., Birje, M.N. and Prasad, B. An Agent-based Resource Allocation Model for Computational Grids. *Multiagent and Grid Systems*, 1(1), 17-27, 2005.
[15] Maropoulos, P.G., Bramall, D.G., McKay, K.R., Rogers, B. and Chapman, P. An aggregate resource model for the provision of dynamic resource-aware planning. *Proc. IMechE Part B Journal of Engineering Manufacture*, 217(10), 1471-1480, 2003.
[16] Nahm, Y.-E. and Ishikawa, H. Novel space-based design methodology for preliminary engineering design, *International Journal of Advanced Manufacturing Technology*, 28(11-12), 1056-1070, 2006.
[17] Raftopoulou, P., Koubarakis, M., Stergiou, K. and Triantafillou, P. Fair Resource Allocation in a Simple Multi-Agent Setting. *International Journal of Artificial Intelligence Tools*, 14(6), 887-899, 2005
[18] Terwiesch, C., Loch, C.H. and De Meyer, A. Exchanging Preliminary Information in Concurrent Engineering: Alternative Coordination Strategies. *Organization Science*, 13(4), 402-419, 2002
[19] Tharumarajah, A. Survey of resource allocation methods for distributed manufacturing systems. *Production Planning and Control*, 12(1), 58-68, 2001.
[20] Ursu, M.F., Virginas, B., Owusu, G. and Voudouris, C. Distributed resource allocation via local choices: A case study of workforce allocation. *International Journal of Knowledge-based and Intelligent Engineering Systems*, 9(4), 293-301, 2005.
[21] Wallace, A. Sequential resource allocation utilizing agents. *International Journal of Production Research*, 41(11), 2481-2499, 2003.
[22] Wu, T. Ye, N. and Zhang, D. Comparison of distributed methods for resource allocation. *International Journal of Production Research*, 43(3), 515-536, 2005.

Contact: Dr Peter Matthews
School of Engineering
University of Durham
Science Laboratories, South Road
Durham DH1 3LE
United Kingdom
Phone: +44 191 334 2538
Fax: +44 191 334 2407
e-mail: p.c.matthews@durham.ac.uk
http://www.durham.ac.uk/engineering/