PERGAMON

International Journal of Project Management 21 (2003) 9–18

INTERNATIONAL JOURNAL OF PROJECT MANAGEMENT

www.elsevier.com/locate/ijproman

Managing multi-project environments through constant work-in-process

S. Anavi-Isakow, B. Golany*

Faculty of Industrial Engineering and Management, The Technion—Israel Institute of Technology, Technion City, Haifa 32000, Israel

Received 27 April 2001; received in revised form 29 June 2001; accepted 8 August 2001

Abstract

This paper proposes new project control mechanisms that limit the number of active projects in multi-project environments. Incoming projects first enter a backlog list. Then, they are staggered into a network of inter-related resources in a way that maintains stable load on the system. The proposed mechanisms adapt the concept of constant work-in-process (CONWIP) that was proposed earlier in the context of production management. We report on an extensive set of simulations that were conducted with several types of projects in dynamic settings. We discuss possible preference rules for the backlog list and the resource queues within the system and demonstrate the superiority of a new rule whose objective is to achieve a dynamic balance of the loads of the various resources.

Keywords: Multi-project systems; Constant work-in process

1. Introduction

In a recent consulting engagement the authors were introduced to a large infrastructure firm that designs and builds switching nodes for its own network. The design and construction of each node is a project that may last several years and cost millions of dollars. At any given time in recent years, the firm has been handling between 40 and 80 such projects. The need for network modifications arises from population growth and migration or from changes in the intensity of industrial activity in certain areas. Thus, the "arrival" of new projects can be modeled as a random variable with values ranging between 6 and 12 new projects a year. The projects are characterized according to their location and technical characteristics. However, their structures (in terms of activity networks) are rather similar. So, for the most part, they all have to go through similar activities that require joint resources. These resources include (1) engineering department whose task is to prepare the detailed design specification (2) real-estate department whose task is to locate alternative sites and make the necessary preparations to purchase them (3) legal department whose task is to handle the legal issues related to the purchase and obtaining the appropriate authorizations from national and municipal agencies (4) construction department whose task is to prepare the detailed construction plan (5) contractors department who makes the appropriate outsourcing decisions and contracts. Other smaller units (e.g. supervisors) are also involved. As the coordination among the departments and projects is rather complex, the firm created a specially designed organizational unit to meet the challenging task of controlling and managing this multi-project environment.

Traditionally, as soon as the need for a new project is realized, it is routed to the appropriate resource(s) where it joins a queue of other projects that entered the system at earlier times. For the most part, these queues are managed according to the first-come-first-serve rule where the exceptions are those projects that were tagged as important (or urgent) to stay in line with the network's standards of performance. Such projects are expedited through the system, often at the expense of other projects whose tasks are pre-empted.

Thus, it is quite common to find resources (e.g. engineers in the engineering design department) that face long queues of projects. These situations have negative effects on the productivity of these resources. Due to internal pressures, the resources find themselves

* Corresponding author. Tel.: +972-4-829-4512; fax: +972-4-823-5194.

E-mail address: golany@ie.technion.ac.il (B. Golany).

0263-7863/02/$22.00
PII: S0263-7863(01)00058-8

working in a "time-sharing" manner, allocating ever-decreasing slices of time to the different projects, in an attempt to satisfy the constant demand of the project managers to see progress in their individual projects. Every time they switch from one project to another, some "setup" time is lost.

Goldratt [1] describes at length these "multi-tasking" phenomena and their negative effects. Goldratt's book focuses on single projects but makes it clear that the multi-tasking problems are only getting worse in multi-project environments. Indeed, in a more recent publication [2], the president of BAE Systems Flight Simulation and Training reports on his experience in applying Goldratt's "critical chain" methodology to the multi-project environment that characterizes his company. An important element in this application is the staggering of arriving projects so as to prevent the creation of large queues in the system. Fricke and Shenhar [3] analyzed multiple engineering projects in a manufacturing support environment and report that: "Most managers agreed, however, that two to three "major" projects at one time was an effective maximization of an engineers productivity. In this way, engineers were not left with idle time when they reached slow points on a project, nor were they overburdened with two many competing responsibilities". Their analysis confirms an earlier finding reported by Adler et al. [4] who recommended that organizations take on fewer projects at a time and by Knolmayer [5] who found out that the more projects carried out simultaneously, the longer the average duration of the individual project, and the higher the coordination expenditures. However, all of these studies treated the issue in a qualitative manner and did not provide quantitative analysis to support their findings.

The purpose of this paper is to provide the missing quantitative analysis that will justify a new control mechanism that would limit the number of active projects in multi-project environments such as the ones described earlier in ways that will alleviate some of the difficulties arising from the multi-tasking phenomena.

The paper is organized as follows: Section 2 provides a brief discussion of topics relevant to this research and describes the way they were treated in the literature to-date. Section 3 develops two model variants and discusses some of their characteristics. In Section 4, we describe the experimental design we used to evaluate the performance of the new models. In Section 5, we analyze the results of the simulation experiments in several dimensions including total flow time and the number of projects in the system. In Section 6, we investigate various queue management policies and point out a preferred rule for the environment we model. In Section 7, we extend the analysis to cases where penalty (in terms of time) is incurred on tasks or projects that have been waiting in queues for too long and in Section 8 we summarize our findings.

2. Background

Most of the literature on project management has been dedicated to single projects. Nevertheless, in recent years there is a growing interest in problems related to control mechanisms for multi-project environments. A number of articles focus on *procedural and organizational control*. For example, Hendriks et al. [6] tackle the issue of allocating human resources in a multi-project R&D environment. They define two new indicators (project scatter factor and resource dedication profile) to aid in the allocation process. Payne and Turner [7] develop reporting procedures for systems with different types of projects and discuss consistent (or standard) procedures vs. procedures that are tailored for each type of project. Merwe [8] deals with organizational issues in managing multi-project environment and stresses the benefits of work breakdown structure (WBS) control using responsibility charts, time control matrices and other procedures and report on a successful implementation in South Africa.

Another line of research dealt with *scheduling of activities* in multi-project environments. As the underlining scheduling problem is NP-hard, many of the articles focused on the development of scheduling and dispatching heuristics for *static* environments—see, e.g. [9–13]. Another approach to the scheduling problem can be found in [14–16]. The latter articles offer hierarchical procedures where, in the first stage, resources are allocated to projects and then each project schedules its activities independently with its own resources.

However, most multi-project environments are characterized by *dynamic* (and stochastic) arrival of projects into the system [17–19]. Unfortunately, the research in this area did not converge to one solution nor could it offer a scheduling rule that is robust enough to hold in the general case. This outcome stems from the fact that each article had its own set of assumptions and was using different objective functions (e.g. average project duration, lateness or tardiness, etc.). As a result, the existing literature on multi-project management does not point unequivocally to an optimal scheduling rule.

The approaches discussed thus far follow the general practice in multi-project management that enables the beginning of work on an in-coming project immediately upon its arrival (provided the first relevant resource is available). This practice is equivalent to the "*push*" *principle* in production management where there is no control over the number of products (or work-in-process—WIP) in the system and the flow along the production routes is determined by the production rates of the various machines. Push control mechanisms do not consider the overall load on the production system when new work is introduced. Rather, they operate according to pre-determined schedules. In contrast, "*pull*" *control mechanisms* allow new work to enter only when the production system signals that its ready to

accept it. This fundamental difference has a profound effect on the performance of the different systems. Hopp and Spearman [20] analyzed these differences and proposed a new control mechanism based on constant-work-in-process (CONWIP). This technique was further developed in [21] and [22].

This paper explores the possibility of adapting the CONWIP concept to multi-project settings. It presents two variants of this control mechanism—one limiting the number of projects and the other limiting the total work (e.g. in hours) in the system. In both cases we employ a "backlog list"—a pool of projects waiting to be entered into the system. Different controls are used to determine when the system is ready to accept an additional project and which of the waiting projects should be entered.

The development of the new models is partially based on the *process management approach* to project management as given in [4]. While traditional project management literature focuses on the network of activities and their characteristics, the process management approach focuses on a network of resources where activities flow among the resources and compete on obtaining services from these resources. The concept of backlogging incoming projects was already discussed in [23] in the context of screening project proposals and further developed in [1]. The latter work, although anecdotal and intuitive in nature, contains excellent insights and solution directions to the problem at hand.

3. Model development

3.1. The backlog list

In coming projects may find the system or resources that needs to serve them in one of two states. Either it is loaded up to a pre-defined level of activity—in which case they have to wait—or it loaded below the threshold level—in which case they may enter the system without delay. Projects arriving at times when the system is unable to accept them enter an external queue where they wait until the load on the system has fallen under the threshold. The backlog list serves several useful purposes. First, it staggers the release of projects into the system in a way that reduces over- and under-loading of resources and helps in smoothing their workload. Second, by observing the size of the list and its composition and comparing it with various statistics that describe the status of the projects in the system we may rearrange the order of the projects in the list in order to improve the overall performance of the system. Third, the monitoring of the backlog list serves as an early warning control to the management of the multi-project organization indicating, for example, the need to shift resources from one place to another in order to meet a surge in demand for a particular operation. It can also help in negotiating with the entities that create the projects (e.g. external clients, other departments in the same organization, etc.) as to the sensibility in generating demand for additional projects. Fourth, the costs associated with projects that are temporarily held in the backlog list are expected to be lower than those that are already in operation. In particular, overhead expenses that are accumulated for each day a project is in the system are not charged when the project is still in the backlog list.

3.2. Constant number of projects in process (CONPIP)

This method limits the number of projects that are allowed in the system to a fixed number that we denote as Maximal Number of Projects—M_{NP}. If a project arrives when the system is serving M_{NP} projects it will wait in the backlog list. When the next project is completed, the system picks up the first project in the backlog list and activates it. Otherwise, if there are less than M_{NP} projects in the system, the backlog list is empty and incoming projects are immediately accepted until M_{NP} is reached. Once activated, the project is broken down to its individual tasks. To be processed, a task has to be ready (i.e. all its predecessor tasks have been completed) and the relevant resource has to be ready (i.e. it is not occupied with another tasks). Thus, the completion time of an activated project is dependent on the status of the system the project meets upon its activation.

3.3. Constant time in process (CONTIP)

This method controls the total processing time required by all the projects that are active in the system. The control mechanism is somewhat more complicated than its CONPIP counterpart. Each time a task of an active project is completed by one of the resources in the system, the remaining processing time requirement (for all the activated projects) is updated. When it drops below a certain threshold (denoted here as the Maximal Processing Time—M_{PT}) a new project is allowed to move out of the backlog list and into the active system. As soon as the new project enters the system its requirements are added to the total requirements of the system. Since the actual processing time of each task in each project is assumed to be a random variable, we use the mean processing time to calculate the total requirements. As in the CONPIP, once a project is activated its completion time is dependent upon the status of the system.

CONTIP may have an advantage especially when we handle a system that manages several types of projects. As the arrival of the projects is stochastic, the mix of

projects in the system and in the backlog list may change quite dramatically over time. Since different types of projects may have very different processing requirements, relying on M_{NP} to control the system may become insufficient.

There is an analogy between work-in-process in production management and time-in-system in project management. Goldratt [1] discussed this analogy and stressed the need to create time buffers in the multi-project system to protect the set of the most important resources (which he denotes as the "critical chain") and ensure that it does not become idle. Buffer management is highlighted as an essential control over such systems. A similar analogy can be found in [20], pp. 442–447. There, the authors suggest the implementation of the CONWIP method to situations where several products of different types are produced on the same shop floor by measuring WIP in terms of the total processing time rather than by product units.

4. Experimental design

The problem we analyze is a rather complex one. We are seeking to find optimal M_{NP} or M_{PT} thresholds that will improve the system performance over the conventional push control. At the same time we seek to identify the best rules to order the projects in the backlog list and, possibly, to determine optimal queuing policies for the resources in the system. This problem cannot be solved analytically and we rely on simulation models to provide insights and solution directions.

4.1. The simulation model

Resources—Following Adler et al. [4] we defined seven work centers (resources), each containing a single server with no breakdowns (i.e. there are no periods in which the server is unavailable due to external reasons) and unlimited queue capacity in front of each work center. The server serves the tasks of the waiting projects according to a selected queue policy. Two work centers (out of the seven) perform re-work with given probabilities.

Projects—three types of projects move through the system. They differ in their resource requirements, their precedence networks and their arrival rates into the system. Due dates were randomly selected for each project independent of its type.

Fig. 1 details the routing of the three project types through the seven work centers. In the first type, five out of the seven tasks follow a serial pattern and only tasks 3 and 4 are done in parallel. The second type allows more parallel activity (four out of seven are done in parallel) and the third type opens a parallel "branch" with two activities (Nos. 2 and 4) in series.

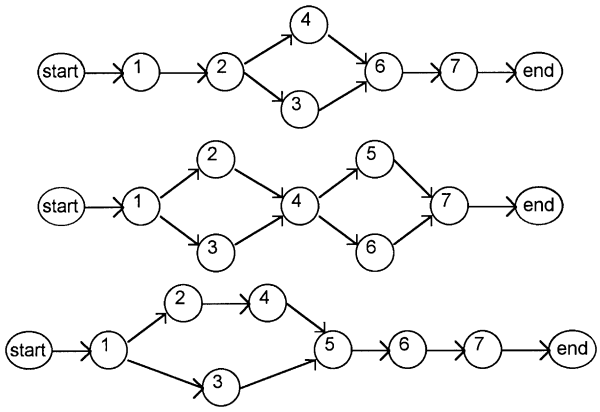

Fig. 1. Precedence relations for the three project types.

Service times—both Normal and Exponential distributions were used to describe the task durations. For the Normal distribution we tested several coefficients of variation to see what effect does increase in variance have over the performance measures.

Arrival rates—projects arrive according to the Poisson distribution (each of the three types has a different mean). Five multipliers were used to change the arrival rates in order to test the performance of the system under various loads.

4.2. Performance measures

The statistics collected during the simulation runs included total flow time (from arrival until the project is done), active flow time (as above but excluding time in the backlog list), throughput (number of projects completed during the simulation period), lateness with respect to due dates, the utilization of the seven resources and the profile of their queues.

4.3. Resource queues' policies

Since the literature does not point out a single optimal queuing rule for our problem settings, we tested a number of well-known rules. These were:

First come, first served (FCFS)—this rule was found to be efficient by some of the previous researchers and it is also the easiest to implement. Therefore, it was also the default rule.

Shortest operation first (SOF)—this rule prioritize tasks in a decreasing order of their required duration. Thus, the first task is the shortest one, the second is the second shortest one, etc. This rule was not proven to be efficient in static multi-project settings but has not been tested in dynamic systems. But, since its production management equivalent (Shortest Processing Time—SPT) is a common scheduling rule it was also tested here.

Shortest project first (SPF)—projects are ordered by an increasing order of their critical path length. Since we had three project types where the mean values of their respective critical paths were sufficiently different it actually meant imposing strict priority ordering among the three types.

Shortest activity from shortest project (SASP)—this rule, which was found efficient in some articles, combines information used by the two rules above it.

Earliest due date (EDD)—this rule aims to increase the number of projects that are finished either before their due date or as close to it as possible.

4.4. Technical details

Simulating the performance of a multi-project system with the attributes listed earlier is quite difficult. In particular, such systems do not have a "natural" start and end points in time and therefore, the simulation must relate to the operations of the system under stable conditions. To this end, we have set certain parameters as listed below.

Warm-up time—to avoid a potential bias from starting conditions we have employed a long warm-up time. When the warm-up time ended, the various simulation statistics were initialized so that their final values reflected the performance of the system after it has reached stability.

Run time and the number of replications—each of our experiments was run at least 20 times where a different seed for the random number generator characterized each replication. This ensured that the system was tested under a large enough variety of scenarios. The length of each run was set so as to enable the system to handle several tens of thousands of projects. Confidence intervals for the mean flow time were used to test the validity of the simulation outcomes. Our guideline for setting the number of replications was to reach confidence intervals that were smaller than 10% of the mean flow time with probability of at least 95%.

5. Simulation results—mean flow times

This section reports on the performance of the CONPIP and CONTIP models (in which the system capacity is limited) and compares them with the push control (where the system is uncapacitated). The following diagram shows the mean total flow time as a function of M_{NP}.

As indicated in Fig. 2, M_{NP} values smaller than 20 cause large flow times (time-in-system values of 246 to 110). These cases are characterized by relatively low utilization of the resources and large backlog lists. As we increase M_{NP} beyond 21 we see a negligible effect on the total flow time in the system. Similar result was obtained with the CONTIP model—beyond a certain level of M_{PT} there was no improvement in the total flow time. These results indicate that the two models (CONPIP and CONTIP) offer some advantages over the conventional push control. By holding projects in the backlog list when the system has reached its capacity we do not lose flow time performance while obtaining the benefits mentioned earlier (e.g. no accumulation of overhead costs).

These experiments were repeated with different values of mean service times, mean arrival rates and several coefficients of variation. The total flow time reached a stable minimum with M_{NP} values that ranged between 10 and 20 and M_{PT} values ranging between 90 and 160. This observation can be explained by referring to the system parameters as follows. The average service time of a project by one resource was three times units (i.e. 21 time units across all seven resources). Since at any given time some projects are near completion while others have just started, the average time requirement of an active project is about 9–10 time units. Therefore, the outcome of the CONTIP model (M_{PT} values in the

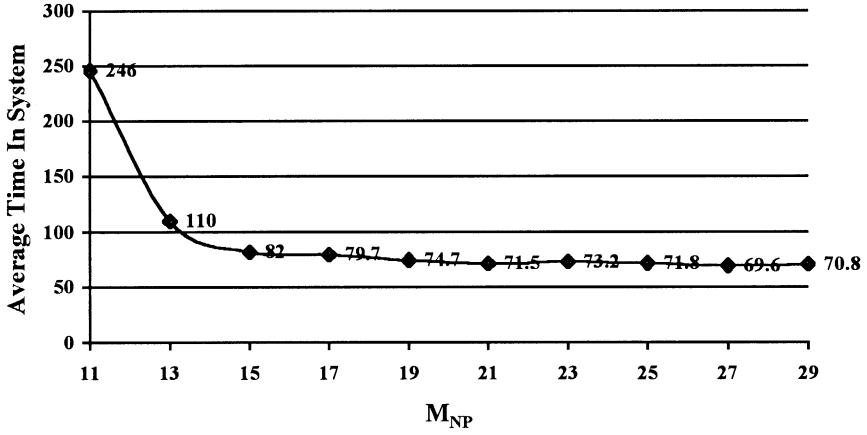

Fig. 2. Flow time performance in the CONPIP model.

range 90–160) can be translated to M_{NP} values in the range of 10–16.

The conclusions we draw from these experiments are:

1. The total flow time reaches a stable minimum in the ranges mentioned earlier. This threshold level varies with respect to the system parameters that affect the overall load. The threshold is a non-decreasing function of the system's load.
2. The lower bound on the number of active projects in the system in which a stable minimum of flow time is reached converges to the number of work centers (seven in our case). This result coincides with the finding reported in [21] on optimal WIP levels in a CONWIP-controlled production system.
3. The upper bound on the number of projects in the system is equal to the number of resources (7) times the number of project types (3)—that is 21. As the system is dynamic and stochastic the project mix in the system varies over time and with it we find random shifts in the location of bottlenecks in the system. Golany et al. [22] found that when there are various types of products the required WIP levels in a CONWIP-controlled production system increases to two times or more than the number of resources. The result in the present study concurs with the earlier result.

Fig. 3 shows the relations between the size of the backlog list (no. of projects in system−no. of projects in resources) and the number of active projects as a function of M_{PT}. For large M_{PT} values (i.e. near "push" conditions), the backlog list ceases to serve its purpose and almost all projects are active. For $M_{PT} = 140$ about a half of the projects in the system are always found in the backlog list while at $M_{PT} = 160$ about one third are in the list. As stabilization of overall flow time occurred for this range of M_{PT} values, it suggests that a proportion in the range (0.33–0.5) for the size of the backlog list might be an appropriate control tool in the multi-project environment. When the backlog list is smaller, it becomes ineffective as most projects are pushed into the system as soon as they arrive. On the other hand, when the list is too long, it creates unnecessary delays in the flow of projects through the system.

6. Simulation results—resource queues policies

To test the effectiveness of the various queue management policies that were discussed earlier we maintained the FCFS policy for the backlog list and ran the simulator with various service times, coefficients of variation and multipliers for the arrival rates. Under each setting, we first identified the M_{NP} value that caused the system to achieve the best performance when all queues follow the FCFS policy. Then, we used the same M_{NP} value to evaluate the performance of the system under different queue policies. Note that different M_{NP} values might have resulted in better performance under queue policies different than FCFS. However, we needed to fix one of the 'degrees of freedom' here to enable a meaningful comparison.

By determining M_{NP} in that manner we actually favored the FCFS rule over other rules, as it was the natural candidate to become the leading queue management policy. Our results, however, indicate that the SOF policy outperformed all other rules. In particular, it achieved an average improvement of 30% over the FCFS rule with respect to total flow time. It was also effective with respect to the other performance measures (e.g. proportion of projects finished on or before their due dates).

The EDD rule has generally outperformed the FCFS rule but by a very small average margin (2%). The other rules exhibited mixed performance. In particular, the SPF policy resulted in considerable deterioration of performance in cases when each of the three types of projects had its own clear bottleneck resource. In our

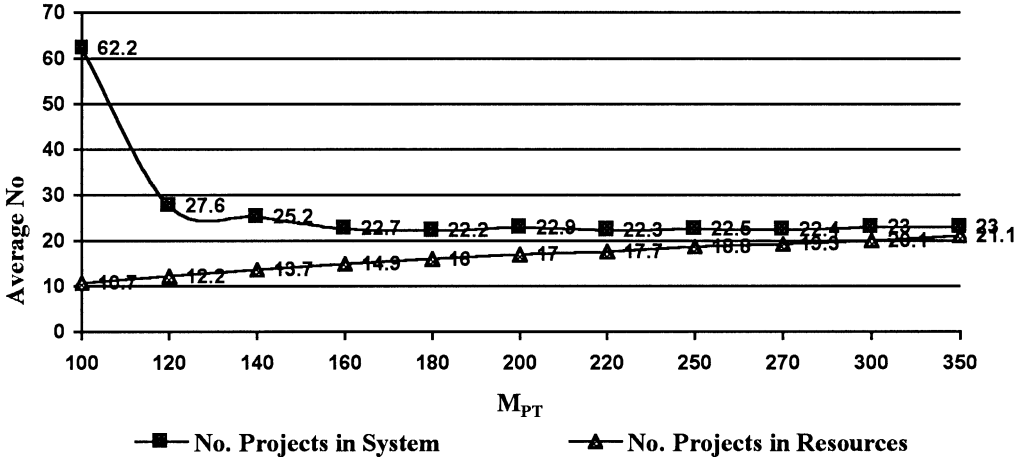

Fig. 3. Number of project in the system (active and inactive).

simulations, the SPF rule assigns the highest priority to projects of type 1 (with the shortest mean critical path time) and the least priority to projects of type 3. Consequently, the latter projects are always pushed to the bottom of the queue in the backlog list and projects of type 1 and 2 enter the system, create large queues in front of 3–4 resources while the other resources are idle most of the time.

Following the outcomes that were obtained with the CONPIP model, we ran the CONTIP model under similar settings using the FCFS and SOF as the rules for managing the queues. In each run we first found the M_{PT} value that resulted in the best outcome for each of the two rules. As before, the SOF rule outperformed the FCFS rule in a significant way. In the extreme cases (especially in experiments 3–5 in which we used unbalanced service time arrays), SOF improves the FCFS results by over 40%. Table 1 summarizes the results obtained in five experiments that were run for both CONPIP and CONTIP with different queue management policies.

7. Comparing push and pull controls

Real-world multi-project environments are characterized by an additional attribute that was not discussed till now—setup times. These time intervals are required for the resources to prepare for the next task, check and perhaps update the data associated with projects that have been waiting in line for a long time, etc. Also, in push systems where every incoming project is allowed to enter we might find very long queues in front of particular servers (resources). In these situations the server might be required to spend some time organizing the queue (perhaps changing the order of tasks according to some priorities) and select the next task to work on (see also [16] where similar setup times are considered).

In this section, we compare the performance of an ordinary push control system with the new CONPIP and CONTIP controls under two types of setup times.

Table 1
Average flow time for various queue policies[a]

Experiment no.	CONPIP					CONTIP	
	FCFS	SOF	SPF	SASP	EDD	FCFS	SOF
1	71.5	56.3	66.3	59.5	73.5	68.3	55.3
2	34.8	31.8	32.9	32.6	34.5	34.1	32.4
3	73.9	46.4	507	399	72.2	73.8	47.6
4	82.5	45.6	531	86.5	75.7	80.8	45.9
5	76.8	50.0	119	89.0	76.4	74.3	49.6

[a] CONPIP, constant number of projects in process; CONTIP, constant time in process; FCFS, first time, first served; SOF, shortest operation first; SPF, shortest project first; SASP, shortest activity from shortest project; EDD, earliest due date.

7.1. Penalty over long delays in resource queues

This scenario assumes that each time a task is about to be processed by some resource, the system checks how long it has waited for that resource. If the waiting time exceeded some threshold, the task is "penalized" by extending its service time by the relevant resource. This penalty reflects corrective or updating actions that must be executed before the task is done (e.g. in engineering projects it might be necessary to update or refresh the data associated with the task, to re-run certain tests as the validity of earlier tests has expired, etc.). We have tested various penalty functions including:

1. penalty that is proportional to the original service time regardless of the delay;
2. penalty that is proportional to the delay regardless of the service time; and
3. fixed penalty (regardless of both service time and the delay).

Simulations of push control with the first two types of penalties led disastrous results with queues that continue to increase throughout the simulation. In particular, two queues tend to "explode"—the one in front of the sixth resource (where re-work was simulated for 10% of the projects) and the one in front of the first resource (which serves as the "gate" to the system). The same simulations with the CONTIP and CONPIP models led to stable results.

Therefore, valid comparison was only available with the third kind of penalty that was experimented—fixed penalty. This penalty was implemented as a two-step function where no penalty is incurred when the waiting time is smaller than the first bound. A certain penalty is added if the waiting time is between the first and second bound and a larger penalty is added if the waiting time is larger than the second bound. The first bound was set to 50 times units (16 times larger than the average service time of tasks by individual resources) and the first penalty was set to 0.5 times units. The second bound was set to 80 units and the penalty to 1.5 units. The results are given in Figs. 4 and 5.

The introduction of a penalty function, even in its very modest form (as described earlier), badly affects the push system. The utilization of the first resource approaches 100% and, by Little's Law, its queue size grows to impossible levels. In the CONTIP model, the system starts with rather long flow time values at low M_{PT} levels. This is due to low resource utilization that leads to long backlog lists. At the other end, when the M_{PT} levels are high, the systems resembled push control and its performance indicators deteriorated. For certain intermediate levels, the system exhibited good performance. As seen in Fig. 5, similar results were obtained with the CONPIP model. As a matter of fact, the best results obtained with the CONPIP model were slightly

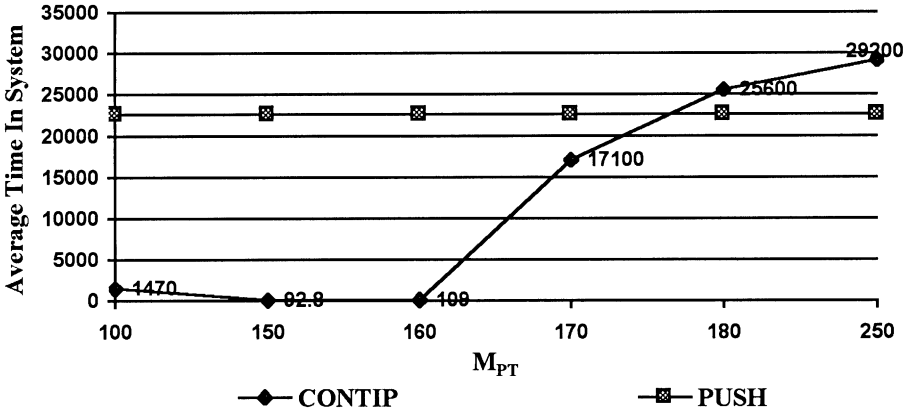

Fig. 4. Flow time for push and CONTIP models with fixed penalties.

better than those obtained with the CONTIP model. However, the average number of active projects in the system associated with the best CONTIP results was smaller than the same parameter for the CONPIP model. Another interesting phenomenon was observed in the build-up of queues. In the CONPIP model, large queues were built in front of resources one and six (as explained earlier) when M_{NP} levels were increased. In the CONTIP model, on the other hand, only the queue in front of resource six increased when M_{PT} levels were raised.

7.2. Penalties combined with queue management delays

When a resource is ready for processing another job it needs to select its next task from its relevant queue. When these queues are managed with the FCFS rule, it takes negligible time to select the next task. However, if other rules are used, one should account for the time required to select the next task. In general, this time requirement will be proportional to the length of the queue. We tested a linear penalty function that assigns penalties that grow in fixed rate as the queue's length increases. For example, we found that under the settings described earlier, a linear penalty function whose slope amounts to only 1% of the average processing time, causes a 22% deterioration in total flow time when we move from the two pull models (CONTIP and CONPIP models) to the push control. When the slope was increased to about 1.5% of the average processing time, the two pull models were hardly affected while push systems exploded.

8. Summary

This paper proposes a new approach to the management of dynamic, stochastic multi-project environments. We explore two variants of this approach—the CONTIP and CONPIP models—and demonstrate their advantages over the traditional push control. The new models are easy to implement and provide the management of this complex environment with information that is typically not available in todays systems. In particular, we show that in systems that are characterized by time penalties for projects or tasks that have been waiting in line for too long, the new models have a clear superiority over the push control. We also show that

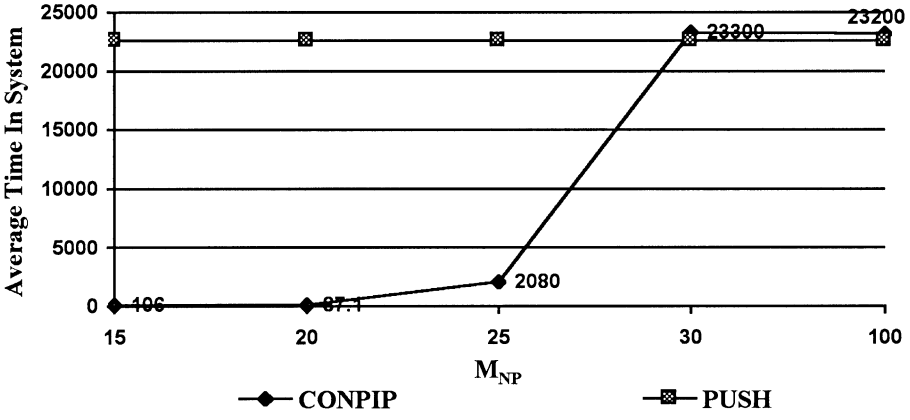

Fig. 5. Flow time for push and CONPIP models with fixed penalties.

efficient management of the individual resource queues using the SOF rule as substitute to the FCFS rule leads to significant improvements.

The introduction of the pull models we propose here may have a number of practical benefits to multi-project systems especially in engineering environments similar to the one we describe in section 1 or those referred to in [2], [3] and [6]. Although these benefits may be difficult to measure we are certain that practitioners will have no difficulty realizing their potential utility. These benefits include:

1. Easier monitoring of the projects in the system. In large multi-projects environments it is often rather difficult to obtain quick status or progress report on individual projects. Consequently, the organizations that operate such environments invest significant resources in building and maintaining sophisticated monitoring systems. When the number of active projects is limited, as we propose here, there is no need for such investments, as monitoring becomes a much easier task.
2. Easier forecasting of completion times. An important stumbling block that often spoils relations with customers is the inability to provide reliable estimates of completion times even when the project has been active for quite some time. The staggered controls we propose here eliminate much of the uncertainty in the system since many of the "synchronization" delays (a task is waiting for its predecessors to be completed but those cannot even start since the relevant resources are faced with queues of tasks from other active projects) disappear.
3. Positive effects on productivity. In conventional push controlled systems, the human resources in the system (mainly engineers in the systems we refer to), are sometimes faced with unbearable queues. Such situations are known to cause demoralization and loss of productivity. In staggered control, on the other hand, the stable workload creates a more relaxed working environment that is easier for the workers to cope with.
4. The backlog list adds an important dimension of flexibility to the management of the multi-project environment as it enables the re-ordering of projects before they enter the system according to various ordering criteria.

We were unable to identify exact values for optimal M_{PT} and M_{NP} values. Rather, through our simulation we are able to point out a range—starting with the number of resources in the system and ending with the number of resources times the number of projects' types—as the interval where best results are obtained. Exact solutions depend on a slew of system parameters that determine the overall load in it. It is worthwhile noting that we have tested our models with various arrays of service times which were identical in the total time required by each resource for all the project types but differ in the variance of the service times in processing the projects by individual resources. As expected, our models achieved better results with rather low variance. Therefore, a designer of such multi-project environment should try to create a well-balanced system in which the variance in required service time is minimal for each project type. This outcome reaffirms similar results that were obtained in earlier studies that were done in the context of production management.

References

[1] Goldratt EM. Critical chain. USA: The North River Press, 1997.
[2] Pitts J. On time, on budget—the challenge? ETS-news (http://www.ets-news.com/) 2001.
[3] Fricke SE, Shenhar AJ. Managing multiple engineering projects in a manufacturing support environment. IIE Transactions on Engineering Management 2000;47(2):258–68.
[4] Adler PS, Mandelbaum A, Nguyen V, Schwerer E. From project to process management: an empirically-based framework for analyzing product development time. Management Science 2000; 41(3):458–84.
[5] Knolmayer G. Das Brookshe Gesetz. WiSt 1987;9:453–7.
[6] Hendriks MHA, Voeten B, Kroep L. Human resource allocation in a multi-project R&D environment. International Journal of Production Management 1999;17(2):181–8.
[7] Payne JH, Turner JR. Company wide project management: the planning and control of programmes of projects of different types. International Journal of Project Management 1999;17(1): 55–9.
[8] Merwe APVD. Multi-project management- organizational structure and control. International Journal of Project Management 1997;15(4):223–33.
[9] Abeyasinghe MCL, Greenwood DJ, Johansen DE. An efficient method for scheduling construction projects with resource constraints. International Journal of Project Management 2001; 19(1):29–45.
[10] Kurtulus I, Davis EW. Multi project scheduling: categorization of heuristic rules performance. Management Science 1982;28(2): 161–72.
[11] Kurtulus I, Narula SC. Multi project scheduling: analysis of project performance. IIE Transactions17(1) 1985;58-65.
[12] Lawrence SR, Morton TE. Resource constrained multi project scheduling with tardy costs: comparing myopic, bottleneck, and resource pricing heuristics. European Journal of Operation Research 1993;64(2):168–87.
[13] Tsubakitani S, Deckro RF. A heuristic for multi project scheduling with limited resources in the housing industry. European Journal of Operation Research 1990;49(1):80–91.
[14] Kim SY, Leachman RC. Multi-project scheduling with explicit lateness costs. IIE Transactions 1993;25(1):34–44.
[15] Speranza MG, Vercellis C. Hierarchical models for multi project planning and scheduling. European Journal of Operation Research 1993;64(2):312–25.
[16] Yang KK, Sum CC. An evaluation of due date, resource allocation, project release, and activity scheduling rules in a multi-project environment. European Journal of Operational Research 1997;103(1):139–54.
[17] Bock DB, Patterson JH. A comparison of due-date setting resource assignment and job preemption heuristics for the multi-

project scheduling problem. Decision Sciences 1990;21(2):387–402.

[18] Dumond EJ, Dumond J. An examination of resourcing policies for the multi-resource problem. International Journal of Production Management 1993;13(1):54–76.

[19] Dumond EJ, Mabert VA. Evaluating project scheduling and due-date assignment procedures: an experimental analysis. Management Science 1988;34(1):101–18.

[20] Hopp WJ, Spearman ML. Factory physics: foundations of manufacturing management. USA: Irwin, 1996.

[21] Dar-El EM, Herer Y, Masin M. CONWIP based production lines with multiple bottlenecks, performance and design implications. IIE Transactions 1999;31(2):99–111.

[22] Golany B, Dar-El EM, Zeev N. Controlling shop floor operations in a multi-family, multi-cell manufacturing environment through constant work in process. IIE Transactions 1999;31(8):771–82.

[23] Shtub A, Bard JF, Globerson S. Project management: engineering, technology and implementation. USA: Prentice-Hall, 1994.