

# NPD Flow: Getting the Most from Product Development Capacity

*Under pressure to develop more products while holding their budgets constant, product development organizations are operating under severe resource constraints. Little margins for errors remain in allocating and controlling resources under these conditions.*

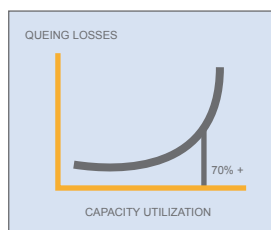
*By the same token, developing more products faster can be the key to competitiveness and higher profits. Businesses that can get the most out of product development capacity will reap substantial rewards.*

Over the last many years, product development organizations have had to increase the rate at which they develop new products, while holding resources at the same levels. As a result, product development capacity has become the major limitation today for many businesses. These capacity limitations affect all aspects of product development – from deciding which projects to undertake, creating technology roadmaps, to coordination and control in execution:

**I. Portfolio Selection:** At the portfolio level, it is no longer possible to accept a project on its own merits. The question one has to answer is: which current or new projects will have to be sacrificed in order to free capacity up for the project being considered, and what is the net effect on business goals of sacrificing those projects?

In addition, lead times for developing new technologies are affected by capacity. Often the best option might be to use an off-the-shelf technology or extend an old platform<sup>1</sup>, than wait for capacity to be freed up for leading edge technologies and new platforms.

**II. Pipeline Planning:** Projects-based operations experience substantial queuing losses (contention for resources, wait times, multitasking etc.) beyond a certain level of capacity utilization. Overload resources by even 10% and the entire pipeline gets clogged ... start 10% less projects than what development capacity allows and you sacrifice substantial opportunities. Pipelines have to be carefully loaded and projects properly sequenced to maximize the flow.



**III. Pipeline Execution:** Most common complaint of project managers during execution is that they do not get resources when needed (even if promised during planning) – somehow required people are always working on other projects. Resources on the other hand complain about conflicting priorities, and about being forced to multitask.

Determining good priorities for resources, and assigning them to the right tasks at the right time can dramatically shorten lead times and increase project completion rates.

While it is clear that little margins for errors remain today in managing product development capacity, it has hitherto been impossible. The next section explains why.

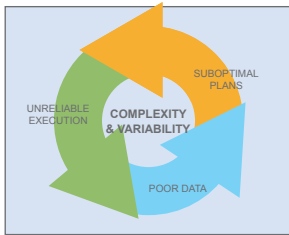
## Obstacles to managing product development capacity

While capacity management has been successfully applied and integrated into manufacturing management, following factors block us from managing product development capacity:

**I. Complex linkages:** developing products involves hundreds to thousands of interrelated activities. Linkages exist not just within a project, but also across projects: e.g.: many product projects depend on platform projects, which in turn are fed by technology projects. With such complexity, balancing resources becomes virtually impossible.

**II. High uncertainties:** development tasks cannot be perfectly estimated and delays are inevitable. These delays quickly multiply through activity dependencies and shared resources, preventing management from establishing stable plans and priorities.

**III. Poor data:** Effects of uncertainties are compounded by poor data. Not having managed their product development capacity in a systematic manner, most organizations have poor data about it. Time and effort required for collecting and refining data becomes an additional obstacle to undertaking capacity management.



Result is a vicious cycle that organizations find difficult to escape from. We need a pragmatic but powerful solution for tackling these obstacles.

## Judging product development performance

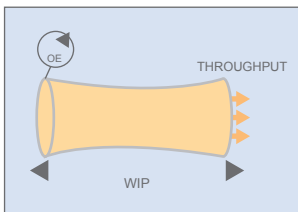
In considering a solution, it is important to keep in mind the results we want:

**I. Increase Throughput (T):** Throughput is the rate at which projects are completed. These projects could be: a) products that are ready to be manufactured, b) technologies that can be licensed to other companies, c) experiments that generate useful information for subsequent steps in the development process, d) designs that can be incorporated into products, or e) platforms off which new generation products can be developed.

**II. Reduce Operating Expense (OE):** OE is the rate at which money is spent to operate the pipeline capacity. It includes all departmental and overhead expenses, and excludes project-specific expenses.

**III. Reduce Work-in-Progress (WIP):** WIP is projects that have entered the NPD pipeline, but have not yet finished:

- The longer a project stays in the pipeline, the less useful it becomes because of market changes and new technologies being created.
- WIP interrupts project flow. Cutting WIP cuts lead times.
- WIP creates confusion, causing delays and “multitasking”. Cutting WIP increases effective capacity and throughput.



Ideally WIP should be measured in time units, i.e., how many weeks or months of potential throughput it roughly represents. Stated another way, if no new projects were started, how much time would pass before the pipeline dries up.

In the following sections we describe how focusing on NPD flow allows managers to maximize throughput, and minimize work-in-process and operating expense.

## Product Development Planning: portfolio selection and project sequencing

*Its most limiting resources govern flow of projects through the product development pipeline.*

No matter how many activities need to be completed, how complex their interrelationship is, or how many different resources are required, flow of projects through the NPD pipeline is governed by its bottlenecks, i.e., resources that have least capacity compared to the demand placed on them. As corollaries:

- Throughput of the pipeline equals throughput at the bottlenecks.
- Releasing projects in violation of bottlenecks' capacity creates unnecessary WIP.
- Capacity (and operating expense) should support throughput at the bottlenecks.

Following is how we use these inescapable facts to optimize resource allocation during planning:

**I. Portfolio Selection:** In an unconstrained environment, the rule for selecting projects is very clear. The correct decision is to take a project on if it has positive NPV. But let us assume that there is a finite capacity constraint imposed on the business<sup>2</sup>. How should the NPV rule be modified to consider projects?

As an illustration, Table 1 shows five projects. With no capacity constraints we would accept all five because they all have positive NPV. Now suppose there is a test lab whose capacity is required by these projects for a total of 85 weeks, but only 50 weeks of capacity is actually available.

If we prioritize projects according to their individual NPV, we would accept project 1, skip project 2 (because it needs more than remaining capacity), and accept project 3. Portfolio throughput would be \$75,000,000.

But we can increase total throughput to \$105,000,000 by accepting projects 5, 2, 4 and 3. These projects have the greatest combined NPV among those combinations of projects that use no more than 50 weeks of test lab capacity.

The logic leading to correct decision is formalized by prioritizing projects on NPV per unit of constraint's capacity required for each project (constraint-indexed NPV).<sup>3</sup> Using constraint-indexed NPV leads us to first accept project 5, then 2, 4 and finally 3.

Table 1: Illustration of constraints-based portfolio optimization

| PROJECT                      | NET PRESENT VALUE (RISK-ADJUSTED) | CAPACITY REQUIRED AT TEST LAB | DECISION WITH SIMPLE NPD | NPV PER UNIT OF TEST LAB CAPACITY | DECISION WITH CONSTRAINT-INDEX NPV |
|------------------------------|-----------------------------------|-------------------------------|--------------------------|-----------------------------------|------------------------------------|
| 1                            | \$50,000,000                      | 35 WEEKS                      | SELECT                   | \$1.43 M/ WEEK                    | DISCARD                            |
| 2                            | \$45,000,000                      | 20 WEEKS                      | DISCARD                  | \$2.25 M/ WEEK                    | 2ND CHOICE                         |
| 3                            | \$25,000,000                      | 15 WEEKS                      | SELECT                   | \$1.67 M/ WEEK                    | 4TH CHOICE                         |
| 4                            | \$20,000,000                      | 10 WEEKS                      | DISCARD                  | \$2.00 M/ WEEK                    | 3RD CHOICE                         |
| 5                            | \$15,000,000                      | 5 WEEKS                       | DISCARD                  | \$3.00 M/ WEEK                    | 1ST CHOICE                         |
| <b>PORTFOLIO THROUGHPUT:</b> |                                   |                               | <b>\$75,000,000</b>      |                                   | <b>\$105,000,000</b>               |

<sup>2</sup> These are also called constrained capital budgeting problems (reference: "Financial Theory and Corporate Policy", 3rd edition, by Copeland and Weston, Page 55). The constraint could be availability of budgets, or capacity, for example.

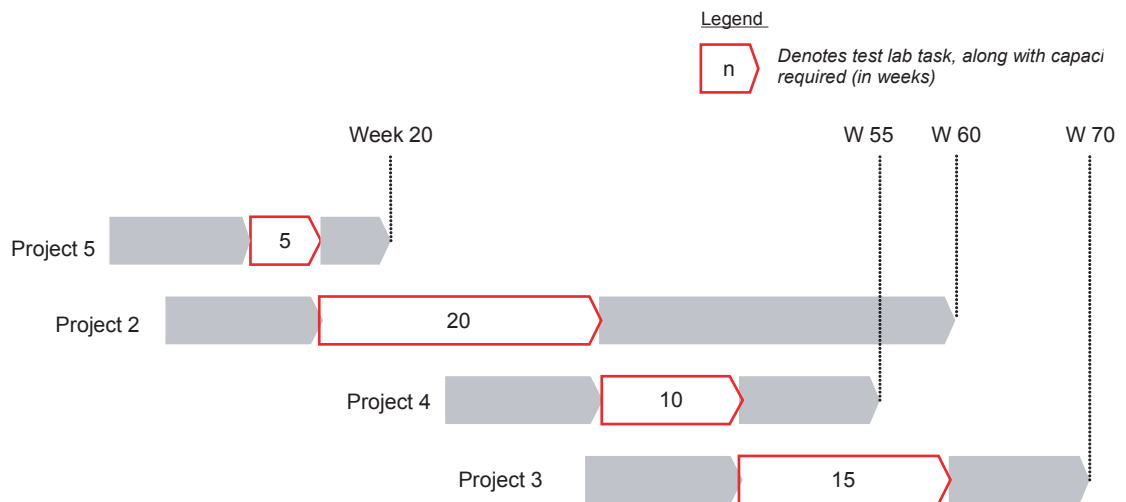
<sup>3</sup> Use of constraint-indexed NPV yields the same results as linear programming optimization. Therefore, not only is using constraint-indexed NPV simple, but also optimal.

**II. Pipeline Planning:** in an unconstrained environment its critical path dictates lead-time of a project. However in capacity-constrained cases, project schedules and lead times depend on when capacity is available at the constraints.

Continuing with our previous example, once we select Projects 5, 2, 4 and 3, what should be the due-dates of those projects. For simplicity assume that testing lies on the critical path of each project. Table 2 and the accompanying figure show how the test lab is loaded and, then, how project due-dates are established.<sup>4</sup>

Table 2: Illustration of constraints-based due-date quotation

| PROJECT | PROJECT LENGTH PRIOR TO TEST | TESTING CAPACITY REQUIRED | TESTING SCHEDULE | PROJECT LENGTH AFTER TEST | PROJECT LEAD TIME |
|---------|------------------------------|---------------------------|------------------|---------------------------|-------------------|
| 5       | 10 WEEKS                     | 5 WEEKS                   | WEEK 11 - 15     | 5 WEEKS                   | 20 WEEKS          |
| 2       | 10 WEEKS                     | 20 WEEKS                  | WEEK 16 - 35     | 25 WEEKS                  | 60 WEEKS          |
| 4       | 10 WEEKS                     | 10 WEEKS                  | WEEK 36 - 45     | 10 WEEKS                  | 55 WEEKS          |
| 3       | 10 WEEKS                     | 15 WEEKS                  | WEEK 46 - 60     | 10 WEEKS                  | 70 WEEKS          |



If projects are released earlier than when test lab is available, that will only disrupt the existing flow of projects – causing loss of throughput and due-date performance.

**Increasing constraint's capacity cuts lead times.** In our example, if the company doubled the capacity of the test lab, they can finish projects 4 and 3 in almost half the time!<sup>5</sup> Thus, an understanding of the constraints and their capacity also forms the basis of a uniquely valuable piece of information: tradeoff between lead times and capacity.

<sup>4</sup> At the planning stage it is not necessary to create detailed project schedules. Due to high variability and strong inter-dependencies in product development, schedules will change anyway on a daily basis.

<sup>5</sup> Assuming no other resources emerge as new bottlenecks.

## Product Development Execution: coordinating work and resources

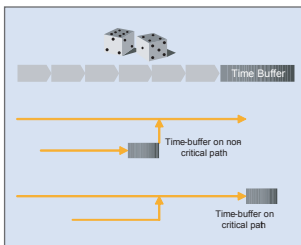
*Uncertainties limit us from achieving the full throughput potential of constraints.*

Uncertainties make local delays inevitable, and due to strong dependencies among tasks those delays tend to propagate rapidly. When delays mount, even non-constraints start experiencing persistent peak loads that interrupt overall pipeline flow. As a result:

- WIP goes up and project due-dates start slipping, jeopardizing throughput.
- Priorities become unstable – people are shuffled randomly between tasks, reducing their productivity by 20 to 80 percent. (Knowledge workers are not machines that can be switched on, and made to produce full-stream instantly. And if switched off, half-finished work decays rapidly).

If organizations want high project due-date performance and highest possible productivity, they need to contain the effects of uncertainties. Good planning is not enough, and execution tools are required to assure highest level of performance:

**I. Time-buffers:** these are blocks of time with no scheduled work – typically placed at the end of a set of activities to absorb variability in those activities:



- On non-critical paths, time-buffers **protect integration points**, without increasing project length.
- On the critical path, time-buffers **protect project due-date**

**II. Priority setting:** even with adequate average capacity, task-time variability during execution can cause peak loads. These peak loads can cause queuing losses in the form of delayed projects and expediting costs. “Just-in-time” resource assignment can be used to contain those losses:

- Since it is difficult to predict actual timing of tasks, they are scheduled when they are actually available to be worked on.
- Critical Ratio is calculated for various tasks in the queue (= work remaining through to project completion / time to project completion.)
- Tasks with highest Critical Ratio are the ones most critical to due-dates of their respective projects, and get first priority. Table 3 and accompanying figure illustrate how to create a just-in-time schedule.

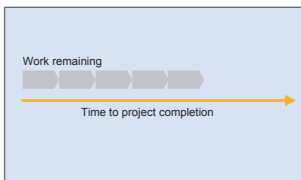
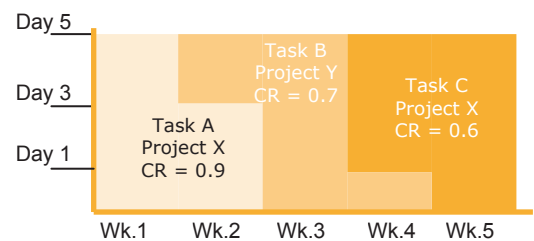


Table 3: Just-in-time queue control using Critical Ratio (CR)

| TASK (PROJECT) | WORK REMAINING | TIME TO COMPLETION | CRITICAL RATIO | CAPACITY NEEDED |
|----------------|----------------|--------------------|----------------|-----------------|
| A (X)          | 18 WEEKS       | 20 WEEKS           | 0.9            | 8 DAYS          |
| B (Y)          | 14 WEEKS       | 20 WEEKS           | 0.7            | 8 DAYS          |
| C (X)          | 9 WEEKS        | 15 WEEKS           | 0.6            | 9 DAYS          |



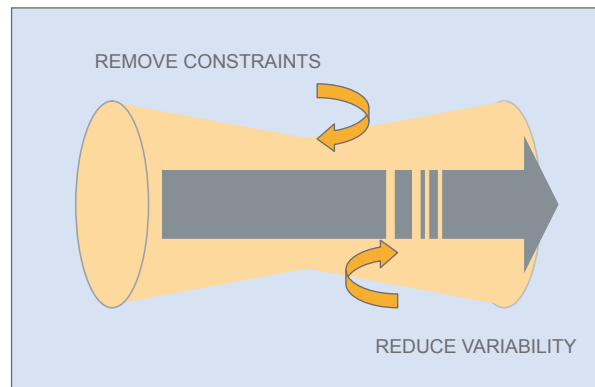
## Instituting a process of ongoing improvement

*Improvement initiatives should be targeted at the sources of biggest disruptions to flow.*

Product development organizations are being tasked not just to make sporadic improvements – but progressively and rapidly improve their performance. This is reflected in growing popularity of concepts such as 6-sigma that focus on process improvements.

Since both the time and resources available to make improvements are finite, managers need to identify areas where local improvement will yield immediate and substantial gains in overall performance. Question is, how to identify high leverage areas?

As discussed earlier, constraints establish the upper limit on throughput, and uncertainties limit an organization from achieving that full potential by creating interruptions to flow. Thus, improvement efforts can be directed at **removing constraints** and **reducing uncertainties**.

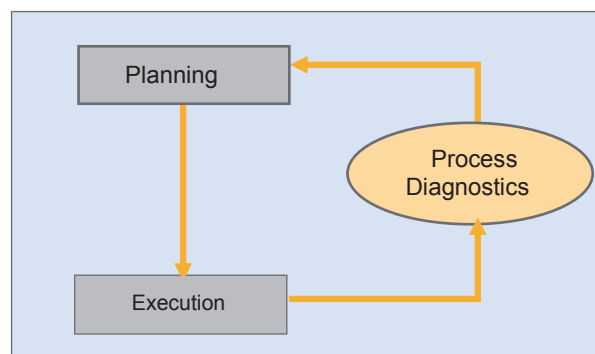


While constraints are easily identified, difficulty is in selecting areas for reducing uncertainties.

### Using buffer performance history to reduce uncertainties

When time-buffers are used to protect schedules, we could also keep a history of which activities actually use up that protection. If we classify those activities (by the type of resources required to perform those activities; type of work they represent; and type of projects they are in), we can have the data to find the biggest sources of interruptions to flow.

Whichever areas consistently consume the highest amount of buffers should be targeted for process improvement - tightening of technical processes, improvement in task estimates, deployment of computer-aided engineering tools, all can be prioritized.



## NPD at Project Flow 2004 Conference

Over 100 delegates at the forefront of multi-project management attended Project Flow 2004. From a wide variety of organizations, these delegates are breaking old rules of project management in order to realize dramatic gains in project speed and throughput.

Eight of the delegates - from the government as well as the private sector, from small and large organizations doing research and design - shared their implementation results and discoveries.

### Results reported

|  | BEFORE   | AFTER   |
|--|--|---|
| <b>New Product Development for Home Appliances</b><br>Hamilton Beach/ Proctor-Silex          | 34 new products per year.<br>74% projects on time.                               | Increased throughput to 52 new products in 1st year, and to 70+ in 2nd year, with no increase in headcount.<br>88% projects on time.    |
| <b>ASIC Design Technology Development</b><br>LSI Logic                                       | 74% projects on time for small projects; major tool releases were late.          | Due-date performance increased to 85% projects on time; major tools released on time for three years in a row.                          |
| <b>High Tech Medical Product Development</b><br>Medtronic                                    | 1 software release every 6-9 months. Predictability was poor on device programs. | 1 software release every two months. Substantial improvement in delivering device programs on time.                                     |
| <b>Warfighter Systems Testing</b><br>US Air Force Operational Test & Evaluation Center       | 18 projects in six months.<br>On time delivery unknown.                          | 26 projects in six months.<br>75% projects on time; 30% reduction in cycle time.  |
| <b>Telecommunications Network Design and Installation</b><br>eircom, Ireland                 | On-time delivery less than 75%.<br>Average cycle time was 70 days.               | Increased on-time delivery to 98+%.<br>Average cycle time dropped to 30 days.   |
| <b>Biotechnology Plant Engineering</b><br>Genencor   | 20% projects on time.  | 87% projects slated to complete on time, with approximately 15% immediate increase in throughput.                                       |
| <b>Design, Development and Upgrades of Telecommunication Switches</b><br>Lucent Technologies |  | 300 to 400 active projects with 30+ deliveries a month.<br>Cycle times are 10 to 25% shorter while throughput per person higher by 45%. |
| <b>Garment Design</b><br>Skye Group  | Product ranges were late to market.  | 100% due-date performance.<br>30% reduction in lead times and sampling costs.   |

## Getting started despite very poor data

*When data lies, intuition can be the guide.*

As discussed earlier, capacity and projects data is either poor or non-existent in most product development environments. However organizations do know which 20% of their resources are most overloaded (where there is perpetual need to hire more people or out-source work).

Such intuition can be used to focus data collection and cleanup efforts (capacity data for likely bottlenecks, and task-estimates for work performed by those resources), and quickly establish a good enough model to set project priorities and realistic due-dates.

Buffer performance data from execution can then be used to progressively make the model accurate, allowing managers to perform sophisticated analyses within a few months.

1. Use intuition to pinpoint probable constraints.
2. Clean up data on constraints to reflect your intuition.
3. Rationalize due-dates for projects flowing through those constraints.
4. Create time-buffers to protect rationalized due-dates.
5. Monitor buffer performance for a few weeks.
6. Analyze buffer history to refine the model.
7. Repeat steps 1 through 6 for a few cycles.

## Summary of benefits

Developing more products faster is the NPD key to increased competitiveness and greater profitability. Finite NPD capacity is what stands in the way. We have presented a pragmatic solution to getting the most from NPD capacity that provides the following benefits:

1. Boost performance through high leverage managerial decisions (not cultural change).
2. Allocate resources to the most profitable opportunities.
3. Achieve higher productivity by creating central resource pools<sup>6</sup>.
4. Determine tradeoff between project lead-time and global finite capacity.
5. Accurately estimate how much money to spend to achieve desired throughput.
6. Contain queuing losses while providing high levels of capacity utilization.
7. Quote feasible project due-dates.
8. Set stable priorities for all project participants – assuring high due-date performance.
9. Focus local improvement efforts on areas that cause biggest disruptions to throughput.
10. Break the vicious cycle of poor data, poor plans and unreliable execution.

---

**Please contact Realization Technologies for feedback or additional information.**

**Phone: 408.271.1711**

**e-mail: [info@realization.com](mailto:info@realization.com)**