

# IMPLEMENTATION OF P2M FOR SOFTWARE DEVELOPMENT TO MULTI-PROJECTS ENVIRONMENT

-Managing critical chain and human factor for mission achievement-

Yuji Kishira (Director Being Corporation, Director Japan TOC Advancement Committee)

e-mail: [kishira@beingcorp.co.jp](mailto:kishira@beingcorp.co.jp)

Shigenobu Ohara<sup>1</sup>(Nihon Institute of Technology)

Keywords: Project management, Program management, P2M, Software development, Integration management, TOC, CCPM, Critical chain

## Introduction

A recent survey by Nikkei Business reported that more than half of all software development projects were missing project due dates, more than half completed over the budget, and one-fourth were forced to reduce originally intended functions during the development. This paper details the findings of a case study for the implementation of a P2M (Project & Program Management for Enterprise Innovation) framework for software development in a multi-project environment. In real life, projects are not carried out in a single project environment. A real challenge which every corporate management team faces everyday is how to deal with multi-projects. The case study demonstrates not only the elimination of problems such as missing due dates, man-power problems, and too much overtime work, but also the shortening of lead time to 1/2-1/4 of what it used to be without increasing resources.

## P2M framework

P2M is the standard guidebook of project and program management in Japan, which provides a framework of broader knowledge for practitioners with a mindset of mission achievement. In the arena of public works, “value for money” is a key mission today. The mission reflects the demand for the best quality of service value, but the smallest possible lifecycle cost is budgeted for construction contractors to manage project risk. For many years, the goals of cost, time, and quality have been decided in advance without the knowledge and wisdom of all team members, who accordingly tend to think these three goals to be mutual trade offs. Given the mission, the approach is initiated to spur debate about the essence of projects and subsequently to find hurdles and solutions for overcoming them by all team members. This process is hard, but is indispensable in achieving the mission by identifying reality for targeting ideality. P2M framework focuses on the human aspects to identify the hurdles and solutions by management from multiple angles.

## Analysis of Current Situation

In P2M methodology, gap analysis is stressed for describing reality and ideality in comparative form. To comply with this thinking approach, a “Questioning Session” was conducted by the initiative of the author. In a questionnaire, software programmers were asked to write three responses to a question, “What is preventing you from making profits?” The session leader then repeated “Why? Why? Why?” to each problem given, until the problems noted were drilled down to the root-cause issue to be solved.

The “Questioning Session” is usually conducted in two days. In this session, attendees can share their problems, realize that what they feel strange is similar to what others feel strange, or find that other programmers have solutions to their problems. Attendees structure the “current situation”, the “ideal situation”, and the gap between them. At the conclusion of the “Questioning Session”, all the attendees decided on the priority of problems and summarized them. In the discussion, almost 80% of critical problems are related to project management. Since the session leader does not impose any opinions, and only attendees’ words are written and summarized in agreement, the participants are generally motivated to solve their own problems immediately after the session.

In summary, the following problems became clear:

- **Ever-changing specifications**
- **Scope creep**
- **Lack of time**
- **Chronically delayed due dates**
- **Shortage of human resources (especially capable persons)**
- **Ever-increasing number of product lines**

---

<sup>1</sup> Professor of Graduate School of Nihon Institute of Technology Japan, Director of Project Management Certification Center, Adjunct Professor of University of Technology, Sydney

- **Round-the-clock development**
- **Yelling managers**

The above situation continued for a long time, and it is no exaggeration to say that they have been developing products in a tunnel with a feeling of no way out. The six months before this project started were especially terrible; programmers worked everyday almost without a day off, many of them worked overnights, and some even fell sick. We decided to launch the project to achieve the following goals:

- Keep the due date with original scope
- No overtime or holiday work
- Human resource development in multi-project Environment

### **Implementation of Critical Chain Project Management**

Critical Chain Project Management (CCPM), a methodology of TOC (Theory of Constraints), was used to cope with these issues. This methodology focuses on human problematic behaviors in performing tasks in a project and implements schedule planning and schedule management based on the idea of correcting such behaviors naturally. In CCPM, human problematic behaviors are:

- Parkinson's Law (use all the time and budget given)
- Murphy (anticipating unexpected problems and estimating longer schedules for tasks)
- Student Syndrome (start slowly and work overnight before the due date)
- Unreported Early Finishes (finish early but use all the time for elaboration)
- Multi-tasking (give top priority to all tasks and start all as early as possible)

CCPM confirms these human problematic behaviors hidden in each task and implements schedule planning and schedule management to eliminate such behaviors naturally. To minimize these problems, CCPM recommends schedule planning considering the following:

- Aggressive task duration estimates (eliminate buffer in each task and schedule each task duration with 50% probability of success)
- Reporting early finishes (guarantee that the next task duration will not be shortened)
- Relay-runner work ethic (start immediately after receiving the baton. Once started, finish as soon as possible and pass the baton to the next process)
- No penalty for delay (half of the tasks are expected to be delayed) -> Manage the whole process of the project, not individual tasks
- Aggregating safety (manage how many more days are required) -> Manage the whole process of the project, not individual tasks, and share the aggregated safety
- No multi-tasking (focus on one task at a time)

At first there was some hesitation to use this method because there were not many implementation cases in Japan. However, we realized this method was almost the same in the way of thinking as with the method of some of the most excellent project managers. Let me quote an episode of one programmer's question & answer session after 5 days of CCPM training:

*Programmer: I can understand the implementation of Critical Chain after these five days, and I am interested in trying it. But, it might fail. Do you know any unsuccessful cases?*

*Consultant: That's a good question. We have helped the implementation of Critical Chain in many software companies, but we have no unsuccessful case. I also heard from fellow consultants that there was no such case in software companies, and we discussed it might be because software programmers are a group of the smartest people on earth.*

*Programmer: Really? Then, we have to work hard not to be the first unsuccessful case on earth!*

### **ODSC and Priority Definition**

ODSC (Objectives, Deliverables, Success Criteria) were defined for each of 12 projects and carefully reviewed by involving company management. ODSC was very effective not only for project members but also for company management in having shared goals, because it allowed an opportunity to redefine project objectives in many ways such as customer benefits, company benefit, future business benefit, human resource development and etc. Especially in the success criteria discussion, team members needed to define a clear "target number" to achieve, which leads all project members to make the commitment to make it successful, and also for company management to understand what they need to do to support this success. Though ODSC definition itself was not so difficult, priority definition required a lot of time and effort to be agreed on. However, we felt that this process became **essential to have a strong tie between company management goals and each project's goals**: Programmers felt they could contribute to achieve the goals of company while management understood how to support them.

## Project Network

After project priorities were fixed as a company decision, four development sites<sup>2</sup> managers were requested to make aggressive but possible schedules by removing safety from each task in every project. An unexpected surprise was that everybody could easily build their project networks without pain. We realized that the excellent project managers were doing exactly the same things to make profitable project plans: they understood that in each task, people tend to have safety which we call it as “SABA” in Japanese. By removing “SABA”, they are making an aggressive but possible schedule to limit undesired human behaviors. For easier understanding for all programmers, we decided to call it “Scientific SABA Removing Preparation Plan” (in Japanese, “Kagakuteki SABA TORI DANDORI”). We loved this expression and it was quickly spreading among all programmers and managers who are implementing it. We learned that sometimes an easy expression can be a powerful tool to drive people in the desired management direction. Finally, we placed a time buffer in the project plan called the “project buffer”. The project buffer is 50% of the duration for each aggressive task duration estimate, aggregated and added to the end of the project as a buffer, which can be used to protect the due date from variations in the project.

In TOC theory, if traditionally progress is managed by each task, people become so sensitive to local delay that unnecessary intervention confuses the project, which promotes longer duration estimates and hidden safety in each task. CCPM eliminates human problematic behaviors such as Student Syndrome and Parkinson’s Law as much as possible by removing the safety in each task, and enables simple and easy-to-understand progress management by monitoring the penetration of “project buffer.”

We decided to call the “project buffer” the “OYAKATA BUFFER”, or “Boss buffer” because we found that an excellent project manager (generally called ‘OYAKATA’) was doing exactly the same thing as project buffer: removing safety from contractors in charge of each task, but on the other hand, devising ways to protect the due date from unexpected troubles or events by having a buffer for the overall project. The project managers can judge the health of project progress by checking the consumption of OYAKATA buffer. This prevents confusion of the process from unnecessary intervention. As previously stated, while CCPM plans an aggressive schedule by removing safety from each task, the relay-runner ethic enables sharing and utilizing of the common project buffer without worrying about each task-based process.

When programmers participated in a seminar, they found it interesting that an American instructor asked “Goju Goju Desuka (Fifty-fifty)?” in Japanese with an English accent when he checked whether the project schedule was aggressive but possible. They imitated him during “SABA” removing at each site, and “Goju Goju Desuka?” became a buzzword in the company. I would like to quote this episode from internal e-mail newsletter.

---

<sup>2</sup> Being Corporation has four development sites in Japan

*Hello. This is Yamao in Technology Strategy Section in Development Division.  
Development Leader Meeting was held at headquarters from Monday to Tuesday this week.*

*In this meeting, regarding the development schedule for 2<sup>nd</sup> half of 2004:*

- *Confirm and brush up ODSC of each project*
- *Set priority on each project schedule*
- *Generate the project network*

*We tentatively finalized the development plan for 2<sup>nd</sup> half of this term.*

*Practice a project network generation*

*As previously informed in this newsletter, to create a project network, we ask, “How can we achieve this?” from ODSC in reverse order. In fact, we eventually practiced three multi-projects schedule. After creating the project network (task order), we estimated task duration. Catchphrase here was...*

*“Goju Goju Desuka?” (You must pronounce it like a weird foreigner(一一;))  
Translation: Is this fifty-fifty chance duration?*

*If we estimate in the traditional way, we put “SABA (=safety)” in each small task, and the total schedule turns out to be very long. To prevent this, it is an important point in TOC to estimate “aggressive but possible schedule” for each task.*

*Project leader: Well, this task takes...4, no, 5 days?*

*Chairman: Goju Goju Desuka?*

*Project leader: Well, if you say so, it can be done in 3 days?*

*Chairman: In TOC, 3-day task will have 1.5-day project buffer, so it will be about 5 days.  
Buffer protects the task.*

*Project leader: Well, then it can be 3 days...O.K., 3 days!*

*In this way, we remove safety...Really ( ° ▽ ° ;)*

*Now, if you see the above, the facilitator always says,*

*“Goju Goju Desuka?”*

*“It has a project buffer. Buffer protects the task,”*

*and it can be shortened. Well, can it be done by everyone?*

*Therefore, I passed the facilitator's baton to Mr. Iseki in the middle of the meeting. Then he repeated “Goju Goju Desuka?” As an excellent facilitator(?), he could make the schedule in CCPM. (This is true!)*

*Now Mr. Iseki, who mastered how to facilitate CCPM scheduling, might come to your site to take the chair.*

*So, catchphrase is “Goju Goju Desuka?”*

*( ↑ Somehow I want to make it popular.)*

### **Resource Allocation in Multi-Project**

There were 12 projects planned at 4 sites across the country. We consolidated those projects into one project platform data<sup>3</sup> set in order to assign programmers from one company resource pool, and allocated programmers across all projects with respect to the priority which was decided as a company decision. Very interesting things happened at this time. **Resource allocation for each task can be easily done if the content and work of each**

---

<sup>3</sup> BeingProject-CCPM software was used for the case study. BeingProject was originally developed for the construction industry where people need to face ever-changing project realities. It was polished by those engineering needs with simple and easy use yet very deep functionality to meet the demand of the most complex construction projects. BeingProject-CCPM was used by this case study for meeting original ODSC's with original schedules intact.

**task are clear.** It turned out that many of the tasks in the busiest projects can be assigned to different programmers who have been working on other projects. Also we found **similar tasks<sup>4</sup> across the projects could be consolidated into fewer tasks among multi-project networks.** On the other hand, we discovered that some of the tasks were still requiring specific knowledge and skills, and we understood that we need to **develop a plan to educate programmers to increase the resource pool for the future project resource allocation flexibility.** And more, when we started the resource management across the projects, we unexpectedly found out that we allocated many resources for less important projects. **One integrated project network with one resource pool across the company taught us a lot about “MANAGEMENT” of projects.** It was an excellent learning experience for all programmers, managers and management.

**Progress Management and Human Resource Development**

Since we had a multi-project environment in which 12 large projects going on simultaneously, progress management was also important. We uploaded the progress management onto the server once a week and checked the buffer progress of the integrated projects. In this way the progress management got very simple, easy and visualized to every body: Everyone only checks the buffer consumption.

At development sites, progress meetings were conducted as usual. However, site managers reported that the content and density of discussion changed. As people reported “how many more days to complete”, they reported the progress in consideration of task completion, and since they could see the project buffer (OYAKATA buffer) of their own projects and the other projects, if other tasks were delayed and they had spare time, they naturally helped each other on a daily basis. **With only the magic words, “how many more days to complete,” they knew the progress by the remaining duration for completion and could help each other.** A manager told me that this method was the same as teaching management to programmers. On the one hand, **in fact management became suddenly busy.** Since programmers in charge are working on the aggressive schedule, they report problems and issues on site immediately after they occur. In the past, they were not necessarily reported to management because they could be absorbed by safety hidden in each task. Now, management needs to judge problems “with clear priority.” We cannot delay the judgment which may lead to project delay. The ambiguous judgment would make programmers on site confused. On the other hand, however, **as programmers understood ODSC, they made positive suggestions such as “in view of ODSC, I think this has priority.”** - We were very pleased to see that programmers and project managers started to see all projects and tasks from the aspect of the management view, which is undoubtedly good for human resource development in the multi-project management environment.

**Dramatic Before and After**

I would like to introduce the dramatic before and after aspects of this implementation.

Before: Almost all projects were missing due date (several months delay was common).

After: Almost all projects kept the due date. The longest delay was 8 days, compared to several months delay before.

Before: Competitors were always ahead of us in releasing new products or new functions.

After: We are always ahead of competitors to release new products or new functions by at least several months

Before: Scope creep continually

After: When problems occur, programmers immediately report to management with recommendation based on ODSC

Before: Round-the-clock development

After: Almost no overtime or holiday work

Before: It was not clear who is working on what or when they would be available



**Each project manager starts to care the other projects with management view!**

<sup>4</sup> It seems natural that we have similarities in customers’ requests in every region and market because we live in the same age.

After: Resource management is much more flexible because each task is clear

Before: Continuous multi-tasking (Dish-spinning trick: Broken dishes. Fire fighting management)

After: Programmers focus on single task. Even with unexpected trouble, management sets priority for programmers to focus on single task

Before: Incomprehensive vague discussion

After: Since definition of task is clear, we can discuss in ordinary language to reduce task lead time

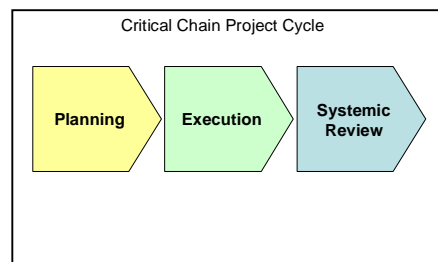
Before: Programmers are always busy for some reason

After: It turned out that programmers sometimes generated tasks by themselves. They meant good for company and customers, but from management view point, sometimes the priority was wrong

The above are our actual experiences and findings from this case study. I think the significant result was that we could **realize that project management, understanding that most companies exist in a multi-project environment, is corporate management itself** – manage projects portfolio with company priority, resource management across the company along with that decision, human resource development for current and future projects.

### Systemic Review

Critical Chain has three stages of project management: Planning Stage, Execution Stage, and Review Stage for Systemic Improvement. In the third stage, a discussion from review involved systemic improvement by using the **TOC Thinking Process**. There was an interesting episode which occurred in this discussion.



the

One of the undesirable effects (UDE) suggested was that it took too long to set priorities for all projects before starting projects (including go/no-go, budget, etc.) In fact, compared to the way we did things in the past (every project is top priority, start!, start!, start! now, or it will be late), project starts were considerably delayed because we took time to define ODSC's and coordinate them with management to set priority. The fact is that it was the first time and it took a few months to coordinate with several divisions and set priorities as company decision. However, we discussed this UDE to answer the session leader's common question "Is this a true statement?" In the Thinking Process, we ask ourselves, "Is this really a UDE? One of the managers said that "If we release the final product later than competitors, it is UDE, but we are much faster in releasing products with much better quality, and it is because we set priority using ODSC's as a company decision, which leads programmers focus on single task with much better efficiency." When we discussed it among the managers carefully, **defining ODSC properly, setting priority, and creating the environment where programmers could focus on one single task were actually the reasons why we could release a new product much faster than competitors**, and we confirmed that **taking time to define ODSC was the main reason this was actually creating a desirable effect**. It made us think how humans sometimes discuss things superficially.

### On-Going Challenge

We have achieved a dramatic effect beyond our expectations within the short period of several months. The pilot study was implemented only to the software development division but its success stimulated other divisions in our company: other division managers are now implementing it, such as human resource development program, sales activities, etc. This method is now spreading across the company for our company's "dream making" where everybody in our company is engaging in some projects to bring better products/service and more benefit to solve customers' problems while providing an opportunity for personal and professional growth.

### Conclusion

This success story is described to transfer experience in the form of practical messages and lessons as if readers could join in the team to share a feeling of participation, rather than simply reading an academic paper. It is significant to learn that the implementation is highlighted as a prototype model of software development in a context of value for all stakeholders. Key success factors are underlined in debating the process at initiation, sharing essential problems in planning, and managing human psychological factors in execution. Needless to reiterate, a unique software tool conforming to Critical Chain thinking and P2M framework for the visual guidance of members undoubtedly supported their performance. In brief, human-centric project management is

the key ingredient proven in this case lesson, which represents the essence of P2M.

### **Acknowledgment**

I can never give thanks enough for the friendship and valuable guidance on Critical Chain Project Management from Mr. David Updegrave, a principal of the Afinitus Group, and Satoru Murakami, president of Goal System Consulting Inc. I also appreciate the valuable advice from a logical standpoint of TOC from Mr. Ichigoro Kuroki, Development Manager of Being Corporation. Without the encouragement and valuable advice, not only from a professional standpoint but also a personal standpoint, from Dr. Adesh Jain, president of International Project Management Association, I could not have written this paper. Millions of thanks to everybody!

### **References**

- Kishira, Yuji and Ohara, Shigenobu *Implementation of P2M to Public Construction and analysis of the result*, International Symposium Project Management: Business. Ideas. Practice Saint-Petersburg May 2005
- Jain, Adesh, *Project thinking* International Symposium Project Management: Business. Ideas. Practice Saint-Petersburg May 2005
- David Pells, **Vladimir Mikheev** *"Third wave" - new paradigm of Project and Program Management*. International Symposium Project Management: Business. Ideas. Practice Saint-Petersburg May 2005
- Ohara Shigenobu "Project & Program Management for Enterprise Innovation"
- Kishira, Yuji "A Practical Guide to Marketing for Presidents" Chukei Publishing
- Gordratt, Eliyahu M. "Critical Chain" Trans. Ryo Sanbongi, Diamond, Inc.
- Murakami, Satoru and Ikawa, Shinji "Project Management for Fastest Development and Earliest Delivery Management Method of TOC 'Critical Chain' " Chukei Publishing
- Kishira, Yuji "Marketing Course for Construction Industry" Web Publication of KEN-Platz, Nikkei BP <http://kenplatz.nikkeibp.co.jp/cpd/0020/>
- Dettmer, William H. "Constraint Management" Quality America, 2000
- Leach, Lawrence P. "Putting Quality in Project Risk Management" PM Network, Feb. & Mar. 2001
- Guinta, Lynn "Building on Success" Constructech, May 2003
- Dettmer, William H. "Strategic Navigation: The Constraint Management Model" APICS, 2003
- Shragenheim, Elyakim M. and Daniel P. Walsh "A Combined Approach to Manufacturing and Project Management" Trans. Eizo Kobayashi, 2001