Development of Enterprise Business Application Software by Introducing Toyota Production System

● Tsutomu Sekimura ● Tomiko Maruyama

(Manuscript received March 28, 2006)

Fujitsu Applications, Ltd. (FAP) was established in April 2004 as part of the Fujitsu Group to focus on developing business application software in Japan. To develop business application software (the basic task of FAP based on SDAS development standards), reducing the development period and achieving high quality are essential. In December 2004, we began introducing the Toyota Production System (TPS) for developing a full-scale application software production phase, while studying and attempting to apply TPS for developing business application software. This paper describes how TPS was concretely introduced (for relatively short-term, large-scale projects involving business application software based on SDAS), and how daily *Kaizen* activities are linked to achieve expected results. Also described are the issues to address for achieving high productivity and high quality, as clarified through case examples.

1. Introduction

Fujitsu Applications, Ltd. (FAP) is a company that specializes in developing enterprise business application software in Java and develops enterprise business application software of high quality on a short-term basis.

To achieve its goals, FAP is making the utmost efforts to innovate in the implementation processes for developing enterprise business application software.

To develop enterprise business application software in a short term, it is necessary to "establish the specifications quickly," "reduce unnecessary development," and "make efficient arrangements."

At the implementation sites, not only the period but also the cost of development must be reduced. There are two points to consider in achieving both goals. First, a development standard and as well as a project management standard proposed by SDAS must be established for each implementation site. Secondly, *Kaizen* activities that solve certain problems relatively quickly must be practiced on a daily basis.

By taking advantage of the opportunity to participate in the "Toyota Production System (TPS) field guidance meeting" sponsored by a hardware department of Fujitsu in November 2004, FAP felt that TPS could be introduced in its software department to develop enterprise business application software and decided to do so in the field of short-term, large-scale enterprise business application software.

FAP paid special attention to the following two points during the course of introducing TPS. The first point is that "*Gemba* nurtures workers so that they can exhibit their abilities." *Gemba* is a Japanese word meaning "on the spot" or "actual scene." The second point is that "*Muda* must be thoroughly excluded." *Muda* is the Japanese word for anything that is wasteful and doesn't add value. This second point is also a key component of TPS.

This paper describes how TPS has been

introduced by combining it with SDAS based on these concepts, and cites an example of large-scale implementation (in 1.1 Mega steps) and short-term (2.5 months) development. This project was undertaken by FAP in cooperation with Fujitsu (Xi'an) System Engineering Co., Ltd. (FXS), a subsidiary in China commissioned to handle development.

The names of phases used in this paper comply with Solution-oriented system Development Engineering Methodology (SDEM), which is Fujitsu's system development standard and is described elsewhere in this special issue.

2. TPS practice (nurturing workers with *Gemba* so they can exhibit their abilities)

FAP addressed the four issues described below so that workers engaged in actual implementation processes could exhibit their abilities and achieve competency.

1) The importance of workers establishing and improving standard work through *Gemba*

Based on SDAS development standards, the *Gemba* leader must define each work item in detail, issue work orders, and estimate the time needed to complete each work item according to actual *Gemba* conditions. Under the *Gemba* concept, workers submit the actual time spent on each work item in a predetermined format daily to determine any differences between the estimated time and actual time spent on each work item. In case of a difference, the reasons are examined and reported to the leader, and if improvements are required, appropriate action must be taken immediately. **Figure 1** shows an example of the definition of detailed work items.

Since this project involved short-term, large-scale development, work was divided as shown in **Figure 2** and some work items that could be developed in parallel by several persons were developed simultaneously to reduce the implementation period, while other work items ranging from the program structure design (PS) phase to the program test (PT) phase were developed conventionally by one person.

	Programming (PG)	
	Screen application Target users: developers (location: FXS)	
Input	JSP DataBean class (Java) Message class (Java) Online definition manual Screen element control table Screen items definition manual BusinessID_Screen application quality check sheet	
Job contents ↓ Job time ↓ Detailed job procedure manual	Asset allocation ← PG job procedures	6
	JSP coding (specific parts) and correction ← JSP correction procedure manual	18
	Screen application coding (specific parts) ← Screen application development description procedure manual	102
	Screen application self-review	18
	Screen application face-to-face review	120
		264
Output	JSP (implemented) Screen application class (implemented in Java)	

Unit: ms

Figure 1
Example of detailed work items.

2) Establishment of system that can easily detect abnormalities

Mieruka is an important factor for introducing TPS. *Mieruka* is a Japanese word that can be interpreted to mean "visual management." Thus, workers can detect abnormalities quickly and take immediate action under the concept of *Mieruka*. A typical example of *Mieruka* would be the sharing of information by illustrating progresses, describing quality, and listing problems on a white board.

This method is effective when applied to only one workplace, but if parallel development is performed at several locations including offshore sites, there must be some mechanism that can detect abnormalities at all related sites by using information technology. At the same time, it is necessary to minimize the load on development personnel, who must strictly follow just three rules: 1) submit work time every day; 2) save

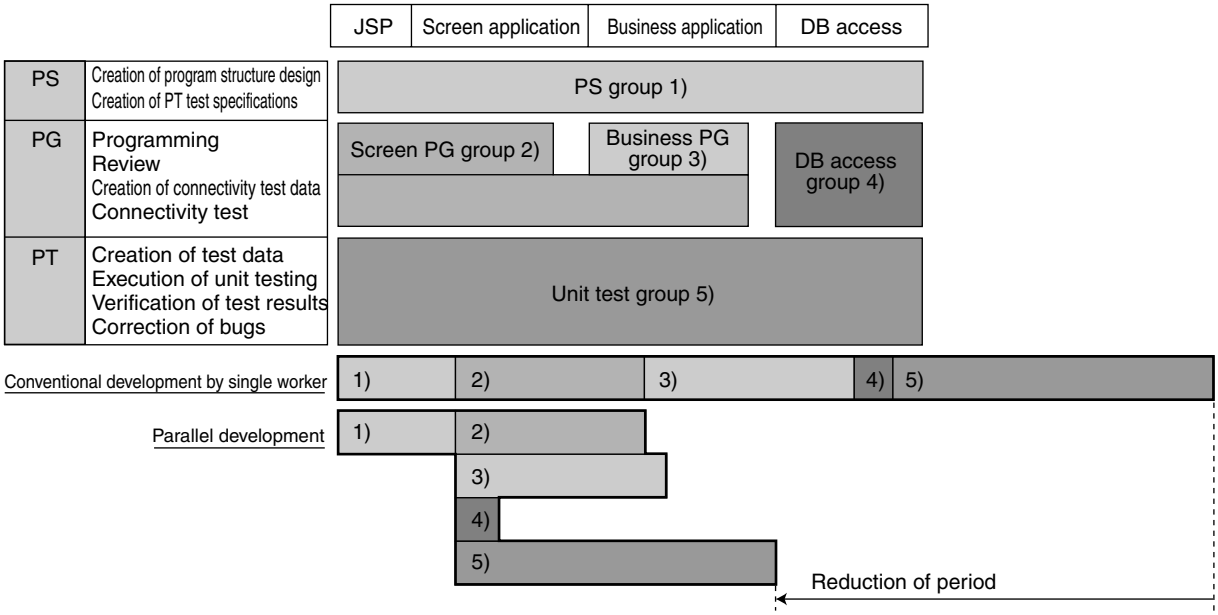


Figure 2
Division of work and image of parallel development.

information in specified folders on a shared server; and 3) correctly describe information on specified work sheets.

By following these simple rules, the collection of information is simplified and any abnormalities during development are detected as shown in **Figure 3**.

3) Utilization of tools

In short-term development, the automatic generation of program source code is technically essential to reduce the amount of work for developers.

In tool development, three prerequisites were determined according to the TPS concept. The first prerequisite is not to aim at automation (the development of tools). In other words, do not increase the load on workers by introducing tools, and do not seek to only reinforce the functions of tools alone. The second prerequisite is incorporating a mechanism in the tools that prevents simple human mistakes, such as those that may occur in simple transfer and repetitive work. The third prerequisite is generating determinate program descriptions according to standardization rules.

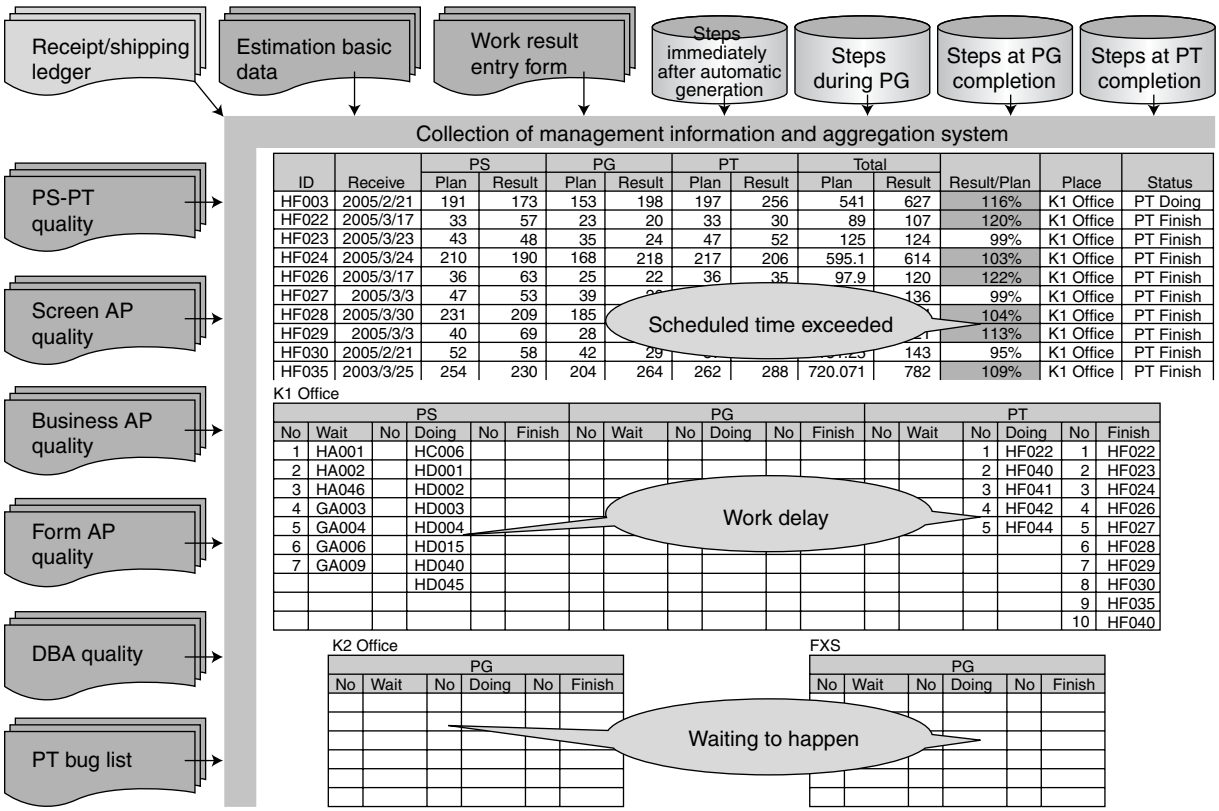
With these three prerequisites in mind, more than 80% of the actual source code for applications was automatically generated in this case.

4) Daily improvement cycle

Kaizen activities were practiced in Japan and at the FXS offshore site. The range of assignment in this case was limited to the programming (PG) phase, with Japanese leaders stationed at the offshore site to establish the implementation processes associated with the introduction of TPS, instead of dispatching a liaison SE to convey specifications. As a result, the earnest improvements listed in **Table 1** were made daily at the offshore site. These improvements were also applied to offshore development for other projects as well as this case. In particular, significant results were achieved for the rule regarding the description of specifications when the PG phase was requested.

3. TPS practice (thoroughly excluding *Muda*)

The seven types of *Muda*[1)] defined and described in TPS were referenced as a way to



ID	Receive	PS Plan	PS Result	PG Plan	PG Result	PT Plan	PT Result	Total Plan	Total Result	Result/Plan	Place	Status
HF003	2005/2/21	191	173	153	198	197	256	541	627	116%	K1 Office	PT Doing
HF022	2005/3/17	33	57	23	20	33	30	89	107	120%	K1 Office	PT Finish
HF023	2005/3/23	43	48	35	24	47	52	125	124	99%	K1 Office	PT Finish
HF024	2005/3/24	210	190	168	218	217	206	595.1	614	103%	K1 Office	PT Finish
HF026	2005/3/17	36	63	25	22	36	35	97.9	120	122%	K1 Office	PT Finish
HF027	2005/3/3	47	53	39					136	99%	K1 Office	PT Finish
HF028	2005/3/30	231	209	185						104%	K1 Office	PT Finish
HF029	2005/3/3	40	69	28						113%	K1 Office	PT Finish
HF030	2005/2/21	52	58	42	29				143	95%	K1 Office	PT Finish
HF035	2003/3/25	254	230	204	264	262	288	720.071	782	109%	K1 Office	PT Finish

K1 Office

PS No	PS Wait	PS No	PS Doing	PS No	PS Finish	PG No	PG Wait	PG No	PG Doing	PG No	PG Finish	PT No	PT Wait	PT No	PT Doing	PT No	PT Finish
1	HA001		HC006											1	HF022	1	HF022
2	HA002		HD001											2	HF040	2	HF023
3	HA046		HD002											3	HF041	3	HF024
4	GA003		HD003											4	HF042	4	HF026
5	GA004		HD004											5	HF044	5	HF027
6	GA006		HD015													6	HF028
7	GA009		HD040													7	HF029
			HD045													8	HF030
																9	HF035
																10	HF040

K2 Office

PG No	PG Wait	PG No	PG Doing	PG No	PG Finish

FXS

PG No	PG Wait	PG No	PG Doing	PG No	PG Finish

Figure 3
Collection of project management information.

recognize *Muda* in the actual processes of implementation enterprise business application software.

1) *Muda* in overproduction

This means creating any functions that are not specified in the system structure design (the SS phase).

2) *Muda* in waiting

This applies when workers on standby cannot initiate an implementation process due to delays in providing specifications. Waiting for tests due to trouble that occurs during program testing is also a form of *Muda* in waiting.

3) *Muda* in transportation

This refers to wasted work associated with the movement of assets between processes. Checking the receipt of specification sheets and the shipment check made at delivery appear to be

Table 1
Example of daily improvements made at offshore site.

Data	No.	Problems and actions
	⋮	
2005/01/17 ~ 2005/01/21	4	Each developer extracts at least five problems or tasks found in OJT, and then conducts an interview.
	5	Japanese text in specifications is not clear (e.g., low-order 2 digits, 4 or less, A and B). Create Excel documents in “clearly understandable Japanese.” Continue updating in the future.
	6	Wait time →Repeat exercises prepared for initial education.
	7	Find common mistakes through face-to-face reviews. →Reflect extracted items on a self-review check sheet.
	⋮	

waste-free tasks, but both the supplier and receiver make these checks, resulting in the wasted action of making a duplicate check.

4) *Muda* in processing itself

This applies to the failure of work items to function properly at the development site.

5) *Muda* in inventory

This means a condition where programming work does not proceed due to the wrong specifications being described and a delay in confirming the content of specifications.

6) *Muda* in motion

Wasteful movement occurs when employees assigned to the same workplace perform their jobs at separate places.

7) *Muda* in defects

This refers to the occurrence of bugs in implementation processes.

4. Initial effects of introducing TPS

Figure 4 shows the initial effects of introducing TPS through the TPS practices described above. The productivity of implementation processes from primary to secondary development was improved by 46%.

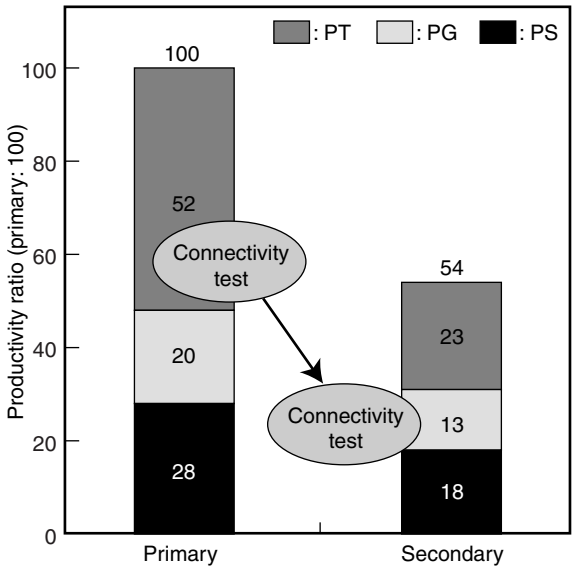


Figure 4
Initial effects of introducing TPS.

Since the unit of productivity varies according to the project and person in charge, a secondary ratio based on the entire primary development as 100 is compared with the work time ratio of each phase of the primary and secondary implementation processes. The effects of introduction are calculated as shown in Figure 4.

1) Productivity in the PS phase improved by 36%

In conventional implementation processes, one person mainly handled the tasks from the PS phase to the PT phase, and dealing with problems was postponed without clarifying processing procedures and program conditions in the PS phase until the PG or PT phase.

However, in this project, implementation processes were reviewed and implementation was not allowed unless the PS phase was properly carried out. For example, the implementation process for one program cannot be divided and created by more than one person without clearly defined specifications. Moreover, any ambiguity in the specifications is not permitted for the automatic generation of 80% or more of the program source code.

Although this implementation process represented just the first attempt, the primary productivity exceeded that of any other past projects conducted by FAP. The improvements made regarding the following two points affected secondary development and succeeded in improving efficiency by 36%. The first point is improving the productivity of relatively inexperienced persons through pair programming in the PS phase. Pair programming refers to a collaborative style of work involving a team of two programmers working together. The second point is eliminating inventory in the PS phase by reducing the time it takes to answer questions. The answer rate of 57.5% within 24 hours in primary development was increased to 77.5% in secondary development.

2) Productivity in the PG phase improved by 35%

The PG phase was handled offshore by FXS and effects were achieved through daily activities, as well as simple, continuous improvement activities. In particular, prior to secondary development, the detailed reasons behind the questions raised in primary development were analyzed and a standard method of describing PS phase specifications was determined.

As a result, the number of questions asked was reduced to about one-third. Moreover, a targeted reduction in the time needed to answer questions was set in the same way as for the PS phase, and an answer rate of 100% within 24 hours was achieved.

3) Productivity in the PT phase improved by 55%

It took 35% of the total PT phase time to conduct connectivity tests between the screen applications and business applications in primary development. This was attributed to the waste in processing caused by inappropriate work allotment for connectivity tests. The work items used for the connectivity test conducted in Japan were changed to those for the FXS PG phase.

To reduce production defects, bugs detected in the PT phase were analyzed in primary development, with corrective action taken within the organization.

It was proved that significant effects were achieved by defining implementation processes adapted to specific projects, and engaging in daily improvement activities (for detection and elimination) without applying special techniques or tools based on the results of improved productivity in the three phases above.

5. Future tasks

The terminology and expedient means peculiar to TPS are to be taken up emphatically. For example, the issue of “thoroughly excluded” or “lacking inventory” tends to become a focus of attention. The original form of an organization with regard to production and manner of operation, as well as the underlying basic concepts[2] are completely different from those employed conventionally. Therefore, it is no exaggeration to say that the attitude toward production reform has been recently called into question.

Introducing TPS at FAP is just the starting point.

The first step is establishing the start-up technology so that the persons in charge of *Gemba* can correctly understand and enforce defined implementation processes. When new standardization, techniques, and tools are applied to a project, start-up education must be provided to the persons in charge. In those cases, it is important to set up a curriculum with consideration given to the existing skill levels of the persons in charge. After providing said education, skill levels must be improved so as to achieve the productivity defined for the implementation processes of a given project. In doing so, a method may be presented whereby the persons in charge can identify and improve tasks, while at the same time fueling their motivation for making improvements. Productivity in software development differs between individuals and as far as the PG phase was concerned, there was a maximum difference of 5.5 times between individuals. As this gap suggests, measures must be taken with each person in mind rather than taking a uniform approach for all.

The initial education defined for the implementation processes of this project only accounted for 4.5% of the total person-hours. Therefore, productivity as well as the appropriate person-hours allotted for education will be accumulated statistically, and ways to more effectively use data will be studied.

Secondly, the TPS approach can be extended to the design and testing processes, without limiting the implementation processes. Especially in short-term development, reducing the number of questions asked about the description of specifications is essential, and the rule regarding the

description of specifications is determined at the *Gemba* level in cooperation with the designers. SDAS also provides a document standard as a development standard. However, this is only a general standard and the number of questions asked should be reduced at four stages, and with any ambiguity in the specifications eliminated to suit the circumstances of *Gemba*.

In the first stage, both the persons in charge of design and implementation must consider these needs. FAP provides specification information that is indispensable for implementation, and designers provide specification information required for customer approval. Both parties then thoroughly review and determine the description items.

In the second stage, the level (depth) of description items is agreed upon.

In the third stage, examples of specification descriptions determined based on evaluation in actual projects are then determined for use based on said prior evaluation.

In the fourth stage, both the persons in charge of design and implementation are educated with regard to the method of describing specifications.

6. Conclusion

There has been little experience with integrating the introduction of TPS and SDAS at FAP. Hereafter, both will become united on a full-scale basis at FAP. Since it is difficult to establish *Kaizen* activities, earnest improvement activities should be thoroughly undertaken based on a trial and error approach. Consequently, FAP plans to expand *Kaizen* from individual project activities to organizational activities.

Moreover, due to visualization, FAP considers the data and mechanisms that may contribute to rousing the desire for improvement through *Gemba*.

Finally, FAP will disclose useful information for making positive improvements.

References

1) T. Ohno: Toyota Production System: Beyond Large-Scale Production. Productivity Pr, 1988.
2) H. Iwaki: Practical Toyota Production System: Cost Reduction which makes the best use of personnel and the organization. (in Japanese), Nihon Keizai Shimbun, 2005.

Tsutomu Sekimura, *Fujitsu Applications, Ltd.*
Mr. Sekimura received the B.E. and M.S. degrees in Computer Science from the University of Yamanashi, Kofu, Japan in 1987 and 1989, respectively. In 1989 he joined Fujitsu Ltd., Tokyo, Japan. In 2004 he moved to Fujitsu Applications, Ltd., where he has been engaged in the development of enterprise business applications.

Tomiko Maruyama, *Fujitsu Applications, Ltd.*
Ms. Maruyama received the B.E. degree in Electronics Engineering from Ritsumeikan University, Kyoto, Japan in 1980. Later that year she joined Fujitsu Ltd., Tokyo, Japan. In 2004 she moved to Fujitsu Applications, Ltd., where she has been engaged in the development of enterprise business applications.