Using the Team Software Process in an Outsourcing Environment

Dr. Miguel A. Serrano and Dr. Carlos Montes de Oca
CIMAT

This article describes an experience using the Team Software Process℠ (TSP℠) in an outsourcing software project. One characteristic of this type of project is that the company offering the service might have to face an economic penalization if the project is not delivered on time. This article describes how TSP techniques were used to quote the project, how TSP helped to control the project during development, and the lessons learned from this experience.

Small companies make up the majority of software development organizations; they face many challenges to maintain their business and to survive [1, 2]. In particular for outsourcing software development companies, fulfilling commitments is everything. Finishing projects on time is critical. Moreover, finishing projects within budget and with the expected quality is the whole business focus because delivering quality software on time means maintenance costs will be minimal. To be able to meet these constraints, outsourcing companies must excel in estimation, planning, project management, and quality assurance.

The Team Software Process℠ (TSP℠) and the Personal Software Process℠ (PSP℠) are well defined processes for software development teams and for software engineers, respectively [3, 4, 5, 6]. The TSP and PSP are designed to help teams and engineers improve their performance and to produce quality products on time and within budget.

This article describes the experience of QuarkSoft, a small, outsourcing, software development start-up company that used the TSP to run an outsourcing project. Specifically, this article describes how TSP techniques were used to quote the project before signing a contract, including estimating project cost and running and keeping the project under control. This article includes a description of the main problems and results of using the TSP, as well as a discussion of the lessons learned from this experience.

QuarkSoft offers consulting and outsourcing software development services. It was conceived as a company where quality software development is one of the main distinguishing characteristics. Thus, the company is committed to developing quality software on time and within budget. The strategy to meet these business goals includes following the Software Engineering Institute's (SEI) Capability Maturity Model® (CMM®) and its implementation with the TSP and the PSP [7]. In other words, the TSP and the PSP are used as the baseline processes in all software development projects.

The TSP provides an operational process to help software engineers do quality work. It also provides the mechanisms to maintain an effective team working environment [8, 9, 10, 11, 12]. The TSP provides team members with the forms, instructions, standards, processes, and scripts to do disciplined and effective teamwork. Teams working under the TSP start with a project launch in which the goals, strategy, risks, plan, and schedule for the entire project are addressed. Normally, the plan is decomposed into several cycles, and during the launch a detailed plan for the first cycle is defined. The project launch takes three to four days. Once the project launch is finished, the team executes the plan. The TSP teams have periodic status meetings during each cycle. At the end of the main phases and cycles, postmortem analyses are conducted. In addition, each new cycle starts with a project relaunch in which the detailed plan for the cycle is built [8].

> *"TSP techniques were used to quote the project before signing a contract, including estimating project cost and ... keeping the project under control."*

The Project

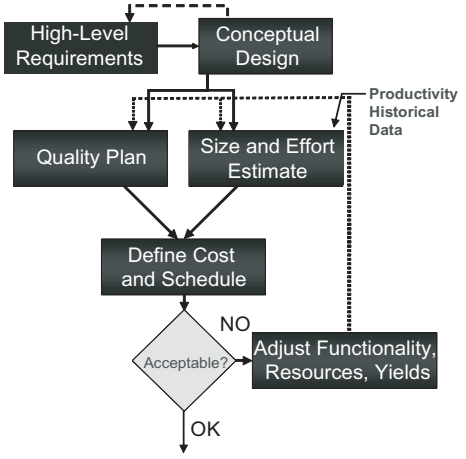
The QuarkSoft project was a major upgrade of a legacy system. It involved reengineering and a great deal of new functionality dealing with databases and compiler techniques over a distributed environment. The core part of the legacy system was the query engine (QE). The old version of the QE was developed for the DOS-Intel platform using indexed files. The client, a large international information and ratings service company, required development in less than nine months.

Getting and Negotiating the Project

The first important issue in this project was defining the development cost and schedule as well as specifying the functionality and quality that the final product would have. On one hand, it was required to quote the project before signing the development contract. On the other hand, since the TSP is used in all projects, it was necessary to perform a TSP launch to have a reasonable quote. In other words, the information to produce a quote is an outcome of a TSP launch. Unfortunately, the client did not want to start the project without the quote, and QuarkSoft could not start without a contract.

To address this deadlock situation, a TSP-based quoting process as depicted in Figure 1 was defined. This process is based on techniques used during a TSP launch. The advantage of this approach is that it requires fewer resources while keeping some of the TSP estimation practices. Clearly, some of the major disadvantages of the approach are the lack of a deep analysis of the problem and the lack

Figure 1: *The Quoting Process*



℠ Team Software Process, Personal Software Process, TSP, and PSP are service marks of Carnegie Mellon University.

of a deep risk analysis.

First, a small team of two engineers was put together. One of these engineers was knowledgeable to some extent of the domain because he developed version one of the QE several years before. The other engineer was knowledgeable about technical issues such as the new platform required for the project.

Second, a conceptual design was developed; TSP estimation is based on product size. Then, using historical productivity data from previous projects, the estimate of time, effort, and quality is done. The TSP bases the initial size estimation on a conceptual design that is done during the project launch [8]. Thus, the TSP-based quoting process started by creating a conceptual design. The conceptual design was developed using the experience of the first engineer and a high-level requirements document provided by the client. Then, client domain and technical experts validated the conceptual design.

Once the conceptual design was agreed on, the estimating process started. First, the size of the product was estimated in source lines of code (SLOC). Each of the components defined in the conceptual design was estimated. The team started with the QE component. Then, other components were estimated using historical data from the client. Finally, best-guess estimation was done in the totally new components such as data base access, graphical user interface, reports, logs and security, and integration with other systems. A combination of the Wideband-Delphi [3, 13] and Standard-Component [14] estimation methods was used to get these estimates and to derive the size estimate of the whole system.

Next, the effort was estimated using estimated productivity data from QE version one and from a previous TSP project. Then, a quality proposal was developed. The quality proposal defines a target for defects/thousand SLOC during the product life. According to this objective, it is possible to define targets for the yield (i.e., percentage of defects in the program that are removed in a particular phase or group of phases [3]) of each of the phases of the development process (e.g., design and code reviews, inspections, compilations, testing). Having the yield for each of the phases, it is possible to estimate the amount of defects that will be in the product during integration test, system test, and product delivery. Thus, the team defined a target yield for each phase and estimated the defects in system and integration test. Using the TSP Quality Guidelines rates for defect removal on those phases, the team estimated the amount of time testing would take. This effort was added to the estimate.

The team came back to the client with the estimate. To meet the project time and budget restrictions, the quoting team and the client worked out a final proposal: Both sides agreed to cut about 30 percent functionality and decided that the development team would include one engineer from the client's staff. Finally, it was agreed that the project would be a nine-month effort, utilizing four engineers.

The Team

The team was assembled with the two senior engineers that made the quote, one junior engineer, and another junior engineer from the client's staff. It was agreed that the project would be developed in QuarkSoft's offices, which provided a TSP-friendly environment.

It was the first time the team members worked together. Also, this was the first TSP project for all team members except for one of the senior engineers who had participated in a one-year TSP project previously. In addition, everyone but the engineer from the client staff had been trained in the PSP. Since the TSP teams require PSP-trained team members, this engineer had to be trained before starting the project.

One practice QuarkSoft promotes is that every team/project has a distinctive name and logo, both selected by team members. The name is in the spirit of the NASA mission names. All documents are tagged with the team name and logo. The team decided to be called Maximus and designed their logo. This practice is aligned to one of the TSP objectives, that is, to form jelled teams [8]. It has been observed that the name and logo help to build team identity (and team pride). In this regard, having an engineer from another company in the team was seen as a risk in terms of forming a jelled team. However at the beginning, this risk was not considered to have high impact.

One deviation from the TSP guidelines had to be done. The team coach and the team leader roles were assigned to the same person. The reason was that the senior engineer that played both roles was the only trained TSP Launch Coach and he was the engineer who developed version one of the QE component.

Launch

After the team was formed and the PSP training was finished, the next step was to perform a TSP launch. Strictly speaking, at this point the TSP starts. The launch consists of nine meetings and lasts three to four days. The launch is the planning phase for the whole project. During the launch, the team follows strategic management principles, together with risk analysis and quality planning to produce a sound plan to develop the project.

The launch was a typical TSP launch. In meeting No. 1, client representatives gave an overview of the project and talked about its importance for the company as well as the impact on sales and customer satisfaction.

In meeting No. 2, Maximus set the project and team goals and assigned TSP roles to team members. Since the first cycle was planned to write the Software Requirements Specification (SRS), only five of the eight roles that TSP defines were assigned. The planning and quality manager roles were assigned to a junior engineer, the support manager role to one senior engineer, the process manager role to the other junior engineer, and the interface manager role to the other senior engineer. Design, implementation, and test manager roles were not assigned.

In meeting No. 3, the conceptual design developed to quote the project was used as a starting point to develop the project strategy. It was decided to implement the project in seven cycles. The first cycle would be devoted to writing the SRS and the Statement of Work (SOW). In the second cycle, a prototype would be developed to define the technology to build the project. In the third cycle, Maximus would build the High Level Design. The rest of the cycles would be for detailed design, implementation, and system integration and system test.

The rest of the launch (i.e., meetings four through nine) included the creation of the general plan for the seven cycles and the detailed planning of the first cycle (four weeks), the quality plan, and a risk analysis.

The detailed plan for the first cycle included activities to create the SRS and the SOW. To estimate the time required for developing these products, the team used historical data from a small TSP project that was finished by another team.

Two important deviations from a normal TSP launch occurred. On one side, the conceptual design, which is normally built in launch meeting No. 3, was already done while quoting the project. Nevertheless, during meeting No. 3, Maximus revisited the conceptual design and made some adjustments. On the other side, the schedule was committed before the launch, so the planning had to be done and adjusted accordingly.

During meeting No. 8 the team prepared the launch report and the presentation for the client. Finally, in launch meeting No. 9, the overall plan was presented to the client. The client was informed of the strategy to build the system, the overall plan, the deliverables, the schedule, the quality plan, and the risks. Moreover, during this meeting several issues were detected that were not considered by the client at first. These findings eased the requirements elicitation.

Running the Project

After the launch, Maximus started executing the plan for the first cycle. Maximus held weekly status meetings, and at the end of the cycle a postmortem meeting was held. Once the first cycle was finished, the rest of cycles followed the same general TSP process: a relaunch, weekly status meetings, and a postmortem [8]. During relaunches, the general plan is reevaluated using the historical data of previous cycles and the detailed plan for the new cycle is built.

Role assignment was the same during the first three cycles. On the fourth cycle, team members started to switch roles. Maximus' members explained that during the first cycles it was not easy to perform the roles. Since they were new to TSP, they required some experience before switching roles. In addition, the first cycles were devoted to building the SRS, documents, and the design. Some roles such as implementation manager do not play an important role in those cycles.

Some of the findings that Maximus got from the weekly meetings and postmortems were the following:

- During the first three cycles, Maximus had time estimation errors up to 300 percent. The most important factors for these estimation errors included lack of historical data, lack of experience in the development platform, and underestimation of the learning curve for both process and technology (e.g., Component Object Model, Distributed Component Object Model, parallel processing).
- By the end of the second month, the project was four weeks behind schedule according to the earned value prediction. This problem was discussed with the client.
- After some weeks into the project, data showed that the engineer member of the client staff was not performing well. His weekly data showed that his earned value was falling behind. Later in the project, postmortem data analysis of the fifth cycle showed that this engineer was still not working as expected and was putting the whole project in jeopardy. It was then decided to take this engineer out of the Maximus team. The decision was supported with data from the first five cycles. It was estimated that keeping the engineer working at his historical earned value rate would make it impossible to finish the project on time. In addition, it was estimated that in order to meet the delivery date, it was necessary to add two more engineers.

Consequently, Maximus had some reorganization. The team leader and coach became only the coach of the team. The other senior engineer became team leader, and two new PSP-trained engineers were added to Maximus.

The relaunch of the sixth cycle lasted one day more than normal. Maximus invested one and one half days in performing a detailed reestimation of the project. The detailed plan for cycle six included time to account for the learning curve of the two new team members. In addition, a strategy to mitigate the impact of the learning curve was devised. The strategy consisted in having specialists for each of the PSP phases. That is, a specialist in design, a specialist in coding, and specialists in testing. As a result of this relaunch, the original strategy and schedule of builds were adjusted but the delivery date remained the same.

After implementing these changes, the last cycles of the project improved considerably. Table 1 shows some final data from the project.

Actual Size (in SLOC)	28,344
Size estimation error	2.26%
Effort estimation error	26.59%
Defect density (defects/KSLOC)	0.18
Productivity (SLOC/Hr)	6.14

Table 1: *Project Final Data*

Lessons Learned

Getting the Contract

Using the TSP approach for project estimation helped to have a more realistic estimation of size and effort. It gave solid arguments for negotiations. The TSP-based quoting process produced the information to convince the client that to meet the time and budget restrictions it was necessary to cut down some functionality and to add more resources.

The client technical experts were sensible to the estimation process and participated on validating the conceptual design and some of the size estimates (i.e., some of their data was used). Thus, they did not have much room for trying to cut costs. The client was also sensible to the fact that the estimation was error prone. It was difficult to negotiate a 25 percent estimation error but using the data at hand, the client had no arguments to go against the proposed estimation error. In summary, the TSP-based estimation process provided the data to elaborate, support, and defend the quote for the project. In addition, it gave the client a sense that the project was estimated professionally as opposed to being obscure nonsense estimations.

Project Launch

One of the advantages of a TSP launch is that everybody knows and agrees to the plan. This common knowledge facilitates the communication among stakeholders and gives a common vocabulary for such a communication. According to the team leader, a major advantage of using the TSP was the planning. He said, "From the beginning, all team members know the activities that each one would perform, the sequence and dependences of them, and the time they would take."

According to one of the engineers:

> Having this detailed planning, it is possible to have better estimates, to plan time for researching the best technology and the best approach for the development of the project. In addition, each team member has a clear idea of the whole project as opposed to other projects where I have participated in which we had no idea of the context. In those projects, the good ideas start coming at the coding or testing phase, when it is very difficult to implement them. With TSP, the good ideas start coming from the launch.

The client liked the last meeting in which Maximus presented the result of the launch. The client was impressed by the TSP methodology (e.g., the level of detail of the planning and all the work products that the launch produced), and considered the team very professional.

Some of the problems reported by Maximus' members were the following:

- A formal method for doing the initial conceptual design is needed, since it is the most important part for estimating the project.
- Risk identification was not easy. Although the TSP contains a risk

analysis, it is not clear how to identify the risks.

Running the Project

Relaunch

Relaunches helped to keep the project under control and allowed detailed planning for the cycle. Relaunches were the perfect time to revisit the strategy, objectives, and risks. In addition, relaunches provided the time to make adjustments to the team and to the original plan in an orderly way. One of our favorite comments stated by one of the engineers was: "Relaunches are a fundamental part of TSP. They allow us to do detailed planning for short periods as opposed to doing detailed planning for a whole nine-month project."

Weekly Status Meetings

Weekly meetings were the best thermometer of the project, during which the team evaluated the status of the project. Weekly meetings were perhaps the most important activity to keep the project under control and to foster communication among team members. They were the best moments to identify and solve problems. Especially, a weekly meeting was a great moment to resolve dependences, misinformation, lack of information, and sort out personal issues. However, it took time to get in that type of mood.

According to Maximus team members: "At the beginning, we wanted to rush the meeting, and we left many open issues unattended."

In another comment, the team leader said:

> We checked risks and objectives weekly, but until cycle six we realized that the whole point of this activity was to generate activities to mitigate risks and to meet objectives. So, a mechanism/process to produce activities to this end is needed.

Other Lessons

Metrics and Data Analysis

One of the main advantages of using the TSP is that the team produces a great deal of information that is available at the right time, which is fundamental for decision making. Several metrics and indicators can be produced. They are really helpful for early detection of problems and for adjusting planning if necessary. For example, without the TSP data and processes, it would have been difficult to detect that the project was behind schedule and to do the analysis to take corrective actions. In addition, as data is collected, more realistic plans are built since they are based on historical data from previous cycles.

In practice, Maximus did not have the time to perform all the data analysis that the team wanted. From the beginning, the project was estimated without considering time for the roles' activities (e.g., planning manager, quality manager). Therefore, no time for data analysis was planned. Also, Maximus realized that they would have needed to develop a tool to help them do the data analysis.

Project Under Control

A major advantage of using the TSP was that the project was under control. Problems were identified on time and corrective actions were applied. Furthermore, corrective actions were backed up with historical data. For example, the decision to take one member out of the team and add two new members was backed up with data, and a strategy to do the switching was developed. The result was a relatively smooth transition. The new Maximus team worked very well from that change on. The team became more cohesive. Without the type of data and process that TSP provides, this kind of analysis and decisions would have been difficult to accomplish.

Another advantage of TSP is the visibility of the project. Every week all team members and other stakeholders know the project status, which includes aspects such as work done, problems detected, new issues, new action items, risks' status, schedule slippages, and goal accomplishment.

Although teams that follow the TSP produce a wealth of information, there is the need to improve the process and tools for issue tracking, action items tracking, and goals and risk tracking. For example, the team leader said that "although we agreed on many things during meetings, they are not done unless they are urgent."

Cultural Change

Another relevant aspect of the TSP is that it is an excellent medium to promote cultural change toward a disciplined process-oriented work method. As we mentioned before, the majority of Maximus' team members were TSP first-timers. Nevertheless, they willingly followed the process, collected data, and committed to finishing the project on time. They also committed to delivering a high quality product. They performed design reviews, code reviews, and inspections. None of them did that in the previous organizations they had worked for.

In addition, TSP promotes the formation of cohesive teams. This might be explained by the fact that they own the plan, and they own the process. While being external observers, we have seen how Maximus team members have developed a camaraderie that goes beyond the workplace. The team leader said he had noticed how the TSP fostered disciplined work in all areas.

Limitations

Maximus identified and documented several process improvement proposals (PIPs). However, Maximus members were unable to implement the PIPs due to the lack of a process to do so and a shortage of available time.

It was observed that not all of the day-to-day activities of the team members' roles were well defined. Thus, engineers had a difficult time performing effectively the roles assigned to them. The TSP provides general guidelines for each role. Also, some role activities are included in some of the TSP scripts. However, in the daily basis, the responsibilities of each role are not detailed.

The TSP does not include a deployment phase. This was a relevant issue for Maximus. After having guidelines and a process for all the development phases, going into deployment of the system without such support caused some project instability. In particular, the deployment phase was critical because the system was deployed in two different countries.

Conclusions and Recommendations

The TSP starts with size estimation. Then, the effort estimation is calculated using historical data such as productivity rates and defect insertion rates. It might be risky to share with the client the productivity rates used to estimate effort. If the client does not have a deep understanding of the TSP and the PSP estimation and planning processes, the client might assume that productivity rate is the only parameter used to estimate effort. With this belief, the client might push the estimation team to use a higher productivity rate that would lead to a reduction of effort estimation, which in turn derives in a reduction of the cost of the project. The client might argue that the productivity rates used are too conservative making it difficult to convince the client otherwise. This situation might lead to an unrealistic estimate.

If historical data is lacking, it is convenient to adjust the estimate to account for the estimation error. QuarkSoft did not

have a definite number on that, but a number between 20 percent to 30 percent estimation error had been used for first-time TSP projects during initial estimation. The estimation is adjusted during each relaunch with the data collected in previous cycles. Thus, the estimation error tends to decrease after some cycles.

One important aspect is the TSP tool support. The SEI's prototype TSP tool is pretty helpful, but it lacks functionality that would make some tasks much easier. To put it in perspective, without the SEI tool, it would be very challenging to have a TSP team collecting and analyzing all the data they produce. However, the functionality for data analysis and reporting provided by the tool is still not enough.

Regarding the effect of the TSP on the client, it was observed that at the beginning, the client was excited about the TSP. The client was well impressed with the TSP process. But at the end, the only thing that mattered to the client was to deliver what was wanted on time.

In addition, it is necessary to help the client in becoming PSP/TSP literate. The TSP produces and uses a great deal of information that can be easily misunderstood and used against the development team. For example, TSP team members might be logging 30 hours/week in planned tasks. The client might demand 40 hour/weeks. However, team members might be working more than 40 hours/week if the time invested in overhead activities is accounted for.

Maximus' team members agree that the TSP is a powerful set of practices and processes. Many of the problems that they faced while using the TSP are covered by the TSP if applied completely. This suggests that team performance will improve with practice and that TSP coaching is particularly important in first-time TSP projects.

Finally, every project will eventually get into some sort of trouble. The important thing is to have the right information at the right time to make the appropriate decisions. This experience shows that the TSP can provide such information. Although TSP needs some adjustments for outsourcing projects, this experience suggests that the TSP is powerful and flexible enough to be used in outsourcing environments.◆

Acknowledgments

We want to thank Maximus' team members for sharing their experience and information. In particular, special thanks to Gerardo Borbolla (team leader), Ricardo Vazquez, and Ricardo Vidrio who provided most of the information to write this article.

References

1. Fayad, M., M. Laitinen, and R. Ward. "Thinking Objectively: Software Engineering in the Small." Communications of the ACM 43.3 (2000): 115-118.
2. Ward, R., M. Fayad, and M. Laitinen. "Thinking Objectively: Software Process Improvement in the Small." Communications of the ACM 43.4 (2000): 105-107.
3. Humphrey, W. A Discipline for Software Engineering. Addison-Wesley, 1995.
4. Humphrey, W. "Pathways to Process Maturity: The Personal Software Process and Team Software Process." SEI Interactive 2.2 (1999): 1-17.
5. Humphrey, W. "Three Dimensions of Process Improvement. Part I: Process Maturity." CROSSTALK 11.2 (1998): 14-17.
6. Kamatar, J., and W. Hayes. "An Experience Report on the Personal Software Process." IEEE Software 17.6 (2000): 85-89.
7. Davis, N. "Using the TSP to Implement the CMM." CROSSTALK 6.9 (2002): 30-38.
8. Humphrey, W. Introduction to the Team Software Process. Addison Wesley Longman, 2000.
9. Humphrey, W. "Three Dimensions of Process Improvement. Part III: The Team Process." CROSSTALK 11.4 (1998): 14-17.
10. Humphrey, W. "The Team Software Process (TSP)." Technical Report CMU/SEI-2000-TR-023. Pittsburgh, PA: Software Engineering Institute, 2000: 51.
11. McAndrews, D. "The Team Software Process (TSP): An Overview and Preliminary Results of Using Disciplined Practices." Technical Report CMU/SEI-2000-TR-015, ESC-TR-2000-015. Pittsburgh, PA: Software Engineering Institute, 2000: 53.
12. Webb, D., and W. Humphrey. "Using the TSP on the TaskView Project." CROSSTALK 12.2 (1999): 3-10.
13. Boehm, B. Software Engineering Economics. Englewoods Cliffs, NJ: Prentice Hall, 1981.
14. Putnam, L. Measures for Excellence: Reliable Software on Time, Within Budget. Englewoods Cliffs, NJ: Yourdon Press, 1992.

About the Authors

Miguel A. Serrano, Ph.D., is a researcher in the Department of Computer Science at the CIMAT, Mexico. He is a SEI-authorized Personal Software Process℠ Instructor and Software Engineering Institute-trained Team Software Process℠ Launch Coach. He is currently working on the Capability Maturity Model® Integration (CMMI®)/Standard CMMI Assessment Method for Process Improvement track. His current research interests include software process improvement, statistical process control, and software quality. He has a Master of Science in information systems and decision sciences, a Master of Science in system science, and a doctorate degree in computer science from Louisiana State University.

Apdo. Postal 402
Guanajuato, Gto., 36000
MEXICO
Phone: +52 (473) 732 7155 ext. 49544
E-mail: masv@cimat.mx

Carlos Montes de Oca, Ph.D. is a researcher in the Department of Computer Science at the CIMAT, Mexico. He is a SEI-authorized Personal Software Process℠ Instructor and Software Engineering Institute-trained Team Software Process℠ Launch Coach. Montes de Oca has over 10 years experience in software development and management. He is involved in several TSP℠ and PSP℠ projects in both academia and industry. His current research interests include software process improvement and software quality. Montes de Oca has a doctorate degree in computer science from Louisiana State University.

Apdo. Postal 402
Guanajuato, Gto., 36000
MEXICO
Phone: +52 (473) 732 7155 ext. 49577
E-mail: moca@cimat.mx