Coping with the Cone of Uncertainty: An Empirical Study of the SAIV Process Model*

Da Yang[1,3], Barry Boehm[2], Ye Yang[2], Qing Wang[1], and Mingshu Li[1]

[1] Laboratory for Internet Software Technologies, Institute of Software, Chinese Academy of Sciences, Beijing 100080, China
{yangda,wq,mingshu}@itechs.iscas.ac.cn
[2] University of Southern California, 941 w. 37th Place Los Angeles, CA 90089-0781
{boehm, yangy}@sunset.usc.edu
[3] Graduate University of Chinese Academy of Sciences, Beijing 100039, China

Abstract. There is large uncertainty with the software cost in the early stages of software development due to requirement volatility, incomplete understanding of product domain, reuse opportunities, market change, etc. This makes it an increasingly challenging issue to deliver software on time, within budget, and with satisfactory quality in the IT field. In this paper, we introduce the Schedule as Independent Variable (SAIV) approach, and present the empirical study of how it is used to cope with the uncertainty of cost, and deliver customer satisfactory products in 8 USC (University of Southern California) projects. We also investigate the success factors and best practices in managing the uncertainty of cost.

Keywords: process model, SAIV, cost estimation, cone of uncertainty.

1 Introduction

Cost estimation is the basis for bidding, budgeting, and planning. It may come from expert intuition, analogy with historical projects, formal parametric cost estimation models, etc [1]. However, because of the incomplete information about the system scale or cost drivers, the learning process of the project stakeholders, the requirement volatility, the market change, etc [2, 3, 4], it is difficult to get accurate cost estimation at the early stages of a software project. And the uncertainty of cost is a threat to ensuring the on-time and within-budget delivery of software.

It is illustrated in [5] that the uncertainty ranges of cost estimations present a decreasing trend as the software development lifecycle proceeds. This phenomenon is named as the cone of uncertainty [6, 4, 7].

Kitchenham states in [8] that the uncertainty is an inherent character of cost estimation, the managers do not understand how to use estimates correctly and, in particular, they usually do not handle properly the uncertainty and risks inherent in estimates.

* This work is supported by the National Natural Science Foundation of China under grant Nos. 60573082 and 60473060; the National Hi-Tech Research and Development Plan of China under Grant No. 2006AA01Z185; the National Key Technologies R&D Program under Grant No. 2005BA113A01.

Q. Wang, D. Pfahl, and D.M. Raffo (Eds.): ICSP 2007, LNCS 4470, pp. 37–48, 2007.

Cantor [9] also proposes that the variances in the estimate of schedule and budget are quite high, and that the reason many projects fail to meet stakeholders' needs is that they are managed as if these variances do not exist.

Despite the awareness that coping with the uncertainty of cost is important, there is a lack of empirical study in the current literature. Here we studied several instrumented e-services software projects performed at USC (University of Southern California) to address how the practitioners can effectively make cost estimation and handle the uncertainty of cost.

As Brooks states in [10], there is no silver bullet to the success of software project. We think it is the same with the issue of coping with the cone of uncertainty, and cost estimation techniques alone can't solve the problem. Empirical studies on how practitioners handle the uncertainty of cost can give us insights on resolving this problem.

Since 1996, there are about 10 real-client projects every year accomplished by the students enrolled in the software engineering class at the University of Southern California. These projects span across broad areas like digital library, e-business, credit card theft monitoring, etc. The main challenges for these projects are that the development teams are required to deliver customer satisfactory products within 24 weeks. The Schedule as Independent Variable (SAIV) process model [11], an architecture-based extension of timeboxing, is adopted by these teams, and guides them to consistently deliver on-time, within-budget, and stakeholder satisfactory software products.

In this paper, we will discuss how the project teams make cost estimates, assess the uncertainty of cost, make project plan, allocate resources and ensure the delivery of stakeholder-satisfactory products. We use the empirical data to analyze the uncertainty of cost estimations and their influence over the projects. We'll also discuss the critical success factors and best practices of these projects.

2 Related Work

Lederer [2] found that requirement changes, users' lack of understanding of their requirements, lack of adequate estimation methodology or guidelines, lack of historical data, etc. can all contribute to the inaccuracy or uncertainty of estimates. Todd Little presented in [4] that according to the Landmark Inc. data, the uncertainty of cost estimation remains high until late in project development. He observed a pipe rather than a cone of uncertainty. As a reply to Little, Gryphon proposed that the Cone of Uncertainty doesn't reduce itself, and it may be reduced by the improved estimation methods that become available as the project progresses [7].

Jørgensen asserted that reflecting the underlying uncertainty of cost estimation would improve the budgeting and planning process. He also proposed several guidelines for the assessment of software development cost uncertainty [3]. The COCOMO II [5] cost estimation model can make the optimistic and pessimistic estimations, which form the interval of cost and schedule with 90% confidence. Other models such as SLIM [12], SEER[13], and Knowledge PLAN [14] provide similar capabilities. In recent years, several probabilistic cost estimation methods also try to assess the uncertainty of cost estimation and use probability distributions to reflect the uncertainty [15, 16, 17].

Though many researchers have addressed the issue of software development cost uncertainty, there is still lack of empirical research on how practitioners properly handle the uncertainty of cost.

In this paper, we will investigate 8 USC software projects, analyze how the uncertainty of cost is handled, and identify the critical success factors. We use the same uncertainty terminology as described in [3]. The uncertainty is defined in terms of probability, i.e., the degree of uncertainty of an event is described by the probability that the event will happen.

3 Backgrounds

This empirical study is based on 8 USC real-client projects, which began in Fall 2005 and completed at the end of Spring 2006 semester. These projects have real world clients from business companies, governmental organizations, and academic organizations. The software products include: Online Reading Assessment, PubMed Data Mining, Football Recruiting Database, Code Generator, XML Editing Tool, EBay Notification System, Rule-based Editor, and Code Count. These projects followed the SAIV process model, and used the MBASE approach and the Lean-MBASE development guideline [19].

The MBASE Model-Based Architecting and Software Engineering [18, 19] is a process framework and also a process model generator. It uses a set of common anchor point milestones [20, 21]: key life-cycle decision points at which a project verifies that it has feasible objectives (LCO: Life Cycle Objectives); a feasible life-cycle architecture and plan (LCA: Life Cycle Architecture); and a product ready for operational use (IOC: Initial Operating Capability).

In the USC real-client projects, the top constraint of the success model is that the teams have to develop the LCA packages in 12 weeks during the Fall semester, and to develop and transition the IOC packages in 12 weeks during the Spring semester. As a result of this success model constraint, the SAIV [18] process model is generated from the MBASE. In SAIV, the schedule becomes the independent variable, and the lower-priority features become the dependent variable. The SAIV is defined by explicitly enacting the following six process elements [11]:

- Shared vision and expectations management
- Feature prioritization
- Cost estimation
- Architecture and core capabilities determination
- Incremental development
- Change and progress monitoring and control

4 Empirical Analysis of the SAIV Development Process

In the 8 projects under investigation, the students made cost estimations and detailed development plans at the LCA milestone. We find the uncertainties of the cost estimates are high. The students, however, didn't go on making more accurate cost estimations during the later construction phase as suggested in [7]. They have used the SAIV process to cope with the uncertainties.

We identified, in this empirical study, four success critical factors of coping with the cone of uncertainty as: estimate cost and its uncertainty, create the opportunities to handle uncertainty, enable flexible process to cope with uncertainty, and risk driven strategies for uncertainty mitigation. Fig. 1 shows the four success factors (in rectangle) and the six related SAIV process elements (in ovals).

In this section, we will discuss the four success factors in subsections 4.1-4.4, and evaluate the project performance in subsection 4.5. In each subsection we will present the related SAIV process elements.

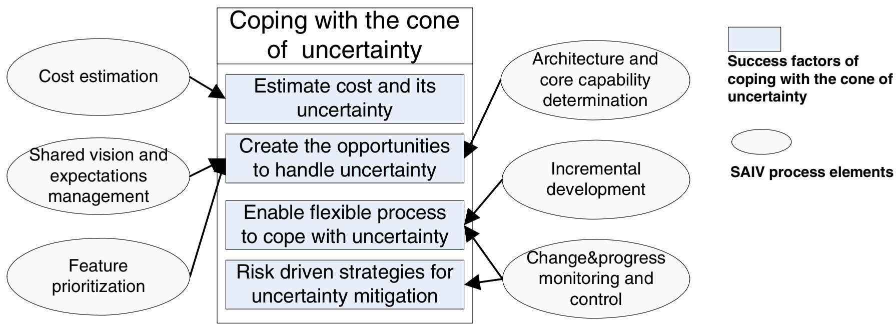

Fig. 1. SAIV elements & Coping with the cone of uncertainty

4.1 Estimate Cost and Its Uncertainty

4.1.1 Cost Estimation

All the 8 projects used COCOMO II for the cost and schedule estimation. As the students' projects are smaller and have a shorter schedule than the projects used in COCOMO II calibration, the students are provided with a new usage guideline based on past development experiences of USC projects.

The students disregarded COCOMO II schedule estimates and used COCOMO II effort estimates to determine how large a team is needed for a 12-week fixed schedule. The estimations are based on the following assumptions:

- Assume 12 hours/week of dedicated effort per person
- Assume 10 of the 12 weeks fill COCOMO II Construction phase (72% of total effort estimate); the final 2 weeks are for product transition into operations.
- Assume 100 hours/person-month for COCOMO estimates

According to the above assumptions, we can derive the following results for the construction phase of the students' projects:

- The estimated effort is "COCOMO II person months"*100*0.72.
- The assumed available construction effort is "number of team members"*12*10.
- Number of team members be larger than "COCOMOII person months"/1.67

Table 1 shows the three COCOMO II effort estimations for the construction phase (Optimistic, Most Likely, and Pessimistic), the number of developers of each team, and the assumed available construction effort (persons * 12 hoursPerWeek * 10 weeks). The column "Most Likely Effort vs. Available Effort" in Table 1 measures how much the estimated most likely effort deviates from the assumed available effort:

Most Likely Effort vs. Available Effort = (Most Likely–Available) /Available (1)

Table 1. Cost Estimation

Team	COCOMO II effort Estimations			Persons	Available Effort	Most Likely Effort vs. Available Effort	Actual Effort	Relative Error (RE)
	Optimistic	Most Likely	Pessimistic					
1	475.2	590.4	741.6	5	600	-0.02	1131.50	-0.478
2	540	669.6	842.4	6	720	-0.07	998.42	-0.329
3	403.2	504	633.6	5	600	-0.16	960.83	-0.475
4	576	712.8	892.8	6	720	-0.01	669.67	0.064
5	597.6	741.6	928.8	5	600	0.24	739.17	0.003
6	518.4	640.8	806.4	5	600	0.07	661.67	-0.032
7	554.4	691.2	864	5	600	0.15	467.08	0.480
8	532.8	662.4	835.2	5	600	0.10	607.67	0.090

We can see that the "Most Likely Effort vs. Available Effort" values are small and below 24%, which means that, according to the COCOMO II cost estimation, in most cases the students can finish the construction phase on time with around 12 hours/week dedicated effort per person. The stakeholders thus considered the current system architecture feasible with respect to the schedule and requirements, and committed to project development.

The cost estimation provides a useful basis to form the shared stakeholder vision on how many features can be delivered within schedule. The expectation management, feature prioritization, and cost estimation can be concurrently conducted.

4.1.2 The Uncertainty of Cost Estimation

The actual effort for construction phase in Table 1 is collected from the students' daily effort report. The accuracy of cost estimation is measured with relative error:

RE(effortEst) = (Most Likely Effort - Actual Effort) / Actual Effort (2)

While the accuracy of an individual cost estimate can be assessed by comparing it to actual effort, the individual cost uncertainty assessment has no obvious corresponding actual values. To assess the uncertainty of a series of estimates, however, we can compare the percentage of confidence level to the proportion of correct assessments ("Hit rate") [3]. The following definition of "Hit rate" is based on uncertainty assessments on the cost prediction interval format, e.g., that it is believed to be "90 percent probable that the actual cost is in the interval [Optimistic cost; Pessimistic cost]".

$$HitRate = \frac{1}{n}\sum_i h_i \,,\; h_i = \begin{cases} 1, Optimistic_i \le Act_i \le Pessimistic_i \\ 0, Act_i > Pessimitic_i \vee Act_i < Optimistic_i \end{cases} \quad (3)$$

We find the actual effort is within the optimistic-pessimistic estimation interval in 4 out of the 8 projects and the HitRate is 50%, lower than the COCOMO II 90%

confidence level. That means the actual uncertainty of cost estimation is even higher than the assessed. Causes for the high uncertainty can be the lack of experience in cost estimation, uncertainties about COTS or open-source component capabilities, the learning process of the students, etc. We will discuss in the following sections how these projects effectively handle the uncertainties.

4.2 Create the Opportunities to Handle Uncertainty

In the SAIV process, the success factor of creating the opportunities to handle uncertainty is related to the process elements: "shared vision and expectations management", "feature prioritization", and "architecture and core capability determination".

4.2.1 Shared Vision and Expectations Management

Expectation management holds the key to providing win-win solutions to the stakeholder negotiation [22]. As described in [23], many software projects lose the opportunity to assure on-time delivery by inflating client expectations and over promising on delivered capabilities. The first element in the SAIV process model is to avoid this by obtaining stakeholder agreement that delivering the system's Initial Operational Capability (IOC) is the most critical objective, and that the other objectives such as the IOC feature content can be variable. The expectation management also provides a basis for effective system feature prioritization.

4.2.2 Feature Prioritization

For each project, the stakeholders used the USC/GroupSystem.com EasyWinWin requirements negotiation tool to converge on a mutual satisfactory set of project requirements. In the negotiation results, there are four categories of requirement priority as "**Must** have", "**Should** have", "**Could** have", and "**Want** to have".

Table 2. Feature prioritization and Core capaility

Team	Capability Requirements (CR)					Percentage of top priority	Core Capabilities (CC)					CC Total /CR Total
	Must	Should	Could	Want	Total		Must	Should	Could	Want	Total	
1	20	14	4	6	44	0.45	20	0	0	0	20	0.45
2	9	3	4	1	17	0.53	8	2	1	1	12	0.71
3	6	2	2	2	12	0.50	5	1	0	0	6	0.50
4	6	2	1	0	9	0.67	4	0	0	0	4	0.44
5	5	2	2	0	9	0.56	3	0	0	0	3	0.33
6	12	0	0	1	13	0.92	12	0	0	0	12	0.92
7	13	0	1	3	17	0.76	13	0	0	0	13	0.76
8	5	5	0	2	12	0.42	5	5	0	0	10	0.83

Table 2 shows the distribution of the capability/functional requirements among four priority levels. Column "Percentage of top priority" measures the percentage of the top priority features marked with "Must", and the average percentage is 60%.

The feature prioritization is vital to be able to establish the core capabilities, which should be delivered on time even under pessimistic cases.

4.2.3 Architecture and Core Capability Determination

The core capability requirements must be selected so that its features add up to a coherent and workable end-to-end operational capability. The core capability should have at least 90% assurance of being completed in 24 weeks, which means even under pessimistic COCOMO II estimation the core capability can be completed. The architecture must also take into account the remaining lower-priority requirements, and make it easy to add or drop borderline features.

Table 2 shows the number of capability requirements (CR), the core capabilities (CC), and the percentage of core capability requirements (CC Total / CR Total). We can see that the core capability usually includes most of the capability requirements marked with "Must" and some of the capability requirements marked with "Should".

4.3 Enable Flexible Process to Cope with Uncertainty

This success factor of enabling flexible process is related to the process elements: "incremental development" and "change and progress monitoring and control".

4.3.1 Incremental Development

The project teams are required to prepare an incremental development plan at the LCA milestone. In their project incremental development plan, the construction is to be completed with two or more iterations. The first iteration will implement the core capability and the remaining iterations will add the lower-priority features. After the first iteration there will be a client-operated Core Capability Drive-Through (the core capability completion milestone).

Since the core capability has 90 percent assurance of being completed in 24 weeks, this means that under the pessimistic case of COCOMO II estimation, the core capability can still be completed within schedule, sometimes with some extra effort.

We compare the duration of the first iteration with that duration of the construction phase, and calculate the percentage as showed in Table 3. The planned first iteration will take 43%-72% of the construction-phase duration. To assess the first iteration duration under the pessimistic case, we use the rate of under-estimate to adjust the planned duration:

Table 3. Percentage of duration for iteration one

Team	Planned	Pessimistic	Actual	RE
1	0.72	0.91	0.71	0.01
2	0.43	0.54	0.54	-0.20
3	0.54	0.68	0.57	-0.06
4	0.64	0.80	0.76	-0.16
5	0.51	0.64	0.75	-0.33
6	0.70	0.88	0.75	-0.07
7	0.68	0.85	0.62	0.09
8	0.59	0.74	0.58	0.01

$$\text{pessimistic duration} = \text{planned duration} * (\text{pessimistic effort} / \text{most likely effort}) \quad (4)$$

The pessimistic percentage of the duration for core capability implementation is between 54% and 91%, that means even under pessimistic case the core capability can be achieved with 9%-46% construction phase time remaining.

In the most likely (planned) case, however, the project will achieve its core capability with about 28-57% of the schedule remaining as planned.

Table 3 also shows the actual duration of the first iteration. The relative error (RE) measures the uncertainty of planned duration for the core capability implementation:

$$RE(\text{scheduleIter1}) = (\text{Planned Duration} - \text{Actual Duration}) / \text{Actual Duration} \quad (5)$$

4.3.2 Change and Progress Monitoring and Control

There are several major sources of change that may require re-evaluation of the projects' plans, such as requirements change, technical difficulties, underestimate of effort, staffing difficulties, COTS changes, customer or supplier delay, etc. The core capability completion milestone is a client-operated Core Capability Drive-Through, which often leads the client to reprioritize the remaining planned features.

The project teams may take many options to accommodate to these challenges. They may drop or defer lower-priority features, dedicate more time each day in construction, reuse existing software, or add expert personnel. In some cases, the changes can be accommodated within the existing plans. If not, there is a need to rapidly renegotiate and restructure the plans.

4.4 Risk Driven Strategies for Uncertainty Mitigation

MBASE is a risk-driven process framework, and the SAIV is also a risk driven process model [18]. The projects' monitoring and control activities include:

- Development of a top-N project risk item list that is reviewed and updated weekly to track progress in managing risks (N is usually between 5 and 10)
- Inclusion of the top-N risk item list in the project's weekly status report

When the uncertainty is high, the risk management can help the students determine what to do next and how much is enough, e.g., prototyping, COTS evaluation, architecting, testing, and business case analysis. The risk management strategies include Buying Information, Risk Avoidance, Risk Transfers, Risk Reduction, and Risk Acceptance [24]. Take Team 1 as an example. The team members explained in their problem report "The lack of GUI prototypes may lead to significant rework at the end of the project in order to accommodate the clients GUI requirement changes". The students were suggested to adopt the Buying Information strategy and construct more GUI prototypes to mitigate the uncertainty of GUI requirements.

4.5 The Performance of SAIV Process

4.5.1 The Execution of Iteration Plan

Though the total project duration is an independent variable in the SAIV process, the duration of the first iteration can change to accommodate to the uncertainty of cost or other changes. When there is under-estimate of effort, the teams can extend the duration of the core capability development, delay some capabilities to future iterations, or drop more low-priority features.

The core capability features should be completed by the first iteration according to the iteration plan. We present in Table 4 the total number of core capabilities (CC Total), and how many of the core capabilities have been completed in iteration one as planned. Comparing the total number of completed capabilities and the core capabilities, we get the Completion Rate (Completed Total/CC Total) and Relative Error:

$$RE(capabilityIter1) = (CC\ Total - Completed\ Total) / Completed\ Total \quad (6)$$

Table 4. Core capability requirements completed in iteration one

Team	CC Total	Completed in Iteration One					Completion Rate (Completed Total / CC Total)	RE
		Must	Should	Could	Want	**Total**		
1	20	19	0	0	0	19	0.95	0.053
2	12	8	2	0	1	11	0.92	0.091
3	6	4	1	0	0	5	0.83	0.200
4	4	4	0	0	0	4	1.00	0.000
5	3	4	0	0	0	4	1.33	-0.250
6	12	7	0	0	0	7	0.58	0.714
7	13	13	0	0	0	13	1.00	0.000
8	10	4	5	0	0	9	0.90	0.111

Fig. 2 shows the uncertainties using boxplot [25], which simultaneously displays the median, the inter-quartile range, and the smallest and largest values for each group. We find the magnitude of relative error of iteration duration or core capability implementation is much smaller than that of effort estimation.

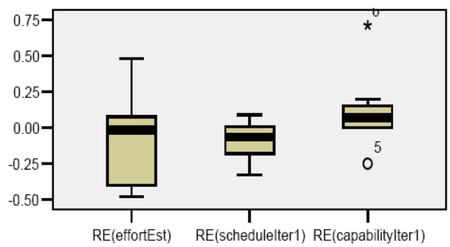

Fig. 2. The uncertainty of cost estimation, iteration1 duration, and iteration1 capability implementation

We use the Pearson's correlation analysis [25] to reflect the correlation between the inaccuracy of cost estimation and the deviations of iteration plan execution. Cohen and Holliday [26] suggest the following rule of thumb to interpret the Pearson's coefficient: 0.19 and below is very low correlation; 0.20 to 0.39 is low correlation; 0.40 to 0.69 is modest correlation; 0.70 to 0.89 is high correlation; and 0.90 to 1 is very high correlation.

Table 5. Correlations among the relative errors

Pearson's Coef.	RE (effortEst)	RE (scheduleIter1)	RE (capabilityIter1)
RE(effortEst)	1	.208	-.137
RE(scheduleIter1)	.208	1	.328
RE(capabilityIter1)	-.137	.328	1

Table 5. shows that there is no significant correlation among the inaccuracy of effort estimation and the deviations of iteration plan execution (change in duration or capabilities implemented).

We find that the completion rate of core capability in the first iteration has significant negative correlation with the percentage of core capability requirements.

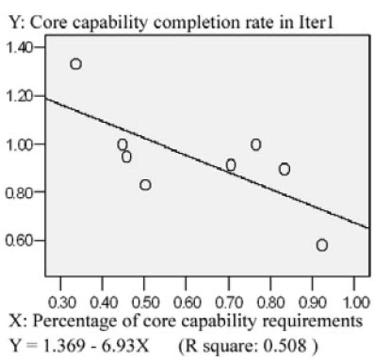

Fig. 3. Linear Regresssion

The linear regression in Fig. 3 graphically shows the correlation between the core capability completion rate in the first iteration and the percentage of core capability requirements. The relation shows, when high percentage of capability requirements are assigned as core capability, it may become more difficulty to complete the core capability in the first iteration as planned.

An extremely high percentage of core capabilities also indicates that there may be some problems with the stakeholder expectations management or requirement prioritization, e.g., the high percentage of 92% in the case of Team 6. This team only completed 58% of the core capability requirements in the first iteration, which was the lowest completion rate among the 8 teams. The team members explained in the iteration review report that the clients required all the requirements to be implemented as core capability and it resulted in a lot of confusion amongst the team members. The students also proposed "This problem would not have arisen, if all the team members were more in touch with the distant client's need".

By analyzing how well the 8 projects executed their iteration plan, we find that:

- Even though the uncertainties of cost are high, the project plans have very small magnitude of uncertainty.
- There is no significant correlation between the inaccuracy of cost estimation and the error in iteration plan execution.
- The completion rate of core capability in the first iteration is well correlated to the percentage of core capabilities.
- The stakeholder expectations management and requirement prioritization are important for establishing a feasible project plan.

4.5.2 Clients' Evaluation of Projects

At the end of the project, the clients evaluated the delivered product with regards to five categories of criteria, which are documentation, team interaction, system preparation and testing, implementation, and overall value. The full grade is 20, and Table 6 shows the clients are satisfied with the development process and the delivered products. The product deliveries received high evaluations as every team finally implemented all the core capabilities and the planned lower-priority features.

Table 6. Client evaluation

Team	Evaluations
1	20
2	20
3	18
4	20
5	20
6	18
7	20
8	20

Table 7. Correlation analysis for client evaluation

Pearson's Coef.	Client Evaluation	RE of effort estimation	Actual Construction effort
Client Evaluation	1	.319	-.086
RE of effort estimation	.319	1	-.955
Actual Construction effort	-.086	-.955	1

The correlation analysis in Table. 7 shows that the stakeholder satisfaction correlates neither to the uncertainty of cost nor the dedicated effort by the team members. The customers' concern is not an accurate cost estimation or the dedicated development effort, but receiving their desired system capabilities within schedule.

5 Conclusions

We find in this empirical study that even though the uncertainty of cost is high, which may be due to the limited experience of cost estimation, the steep learning curve, reuse uncertainties, etc., the students can successfully deliver product on time with satisfactory quality. The project teams can accommodate to changes and complete the core capability iteration with 24%- 46% construction-phase time remaining. In addition, all the core capability features and planned lower-priority features are completed. The clients are satisfied with the development process and product delivery, and their satisfaction doesn't correlate to the uncertainty of cost or the dedicated development effort.

The SAIV process plays a critical role in coping with the cone of uncertainty by: estimating the cost and its uncertainty, creating opportunities to handle the uncertainty, enabling flexible process, and providing risk driven uncertainty mitigation strategies.

The critical practices for the 8 projects are:

- Win-Win stakeholders negotiation and effective expectation management
- Getting clients to develop and maintain prioritized requirements
- Establishing the core capability and architecting the system for ease of adding and dropping features
- Planning development increments and ensuring the on-time delivery of core capability even under pessimistic cases
- Risk driven progress monitoring and corrective action

To cope with uncertainty, agile methods also offer useful practices, e.g., embracing change, fast cycle / frequent delivery, simple design, and refactoring, and the plan-driven methods offer practices like requirements management, quantitative process management, project tracking and oversight [27]. The SAIV process is a balance of agility and discipline. Its usage on USC projects over the last 10 years and other research works [18, 28] indicate that the key practices introduced in this case study are applicable to a wide spectrum of software projects. Practitioners should choose appropriate practices to cope with the cone of uncertainty according to their development environment, and they can use risk, spiral model anchor point, Results Chain, etc. to balance the agility and discipline [27].

Managing the uncertainty of cost is an on going research, and our future work includes:

- Compare the current practices of coping with the cone of uncertainty, and provide more general guidelines.
- Investigate the sources of cost uncertainty and improve the current cost uncertainty assessment method.
- Provide tools to analyze the information of cost uncertainty, make feasibility analysis with given constraints or dependencies, and facilitate the stakeholder win-win negotiation and project planning.

References

1. Boehm B., Abts C., and Chulani S., Software Development Cost Estimation Approaches—A Survey, Annals of Software Engineering, Vol. 10, (2000) 177-205
2. Lederer A.L., and Prasad J., Nine Management Guidelines for Better Cost Estimating, Communications of the ACM, Vol. 35(2), Feb. (1992) 51 - 59
3. Jørgensen M., Evidence-Based Guidelines for Assessment of Software Development Cost Uncertainty, IEEE Transactions on Software Engineering, Vol. 31(11) (2005)
4. Little T., and Graphics L., Schedule Estimation and Uncertainty Surrounding the Cone of Uncertainty, IEEE Software, May/June (2006)
5. Boehm B., et al, Software Cost Estimation with COCOMO II, Prentice Hall (2000)
6. McConnell, S., Rapid Development: Taming Wild Software Schedules, Microsoft Press, (1996)
7. Gryphon S., Kruchten P., McConnell S., and Little T., Letters: The Cone of Uncertainty, IEEE Software, Vol. 23 (5), September/October (2006) 8-10
8. Kitchenham B., Linkman S., “Estimates, Uncertainty, and Risk”, Software, May (1997)
9. Cantor M., Estimation Variance and Governance, http://www-128.ibm.com/developerworks/rational/library/mar06/cantor/
10. Brooks F.P., The Mythical Man-Month, Addison-Wesley Publishing Co., (1995)
11. Boehm, B., and Brown W.. “Mastering Rapid Delivery and Change with the SAIV Process Model,” Proceedings, ESCOM2001, Apr. 2001.
12. Putnam, L., Software Life Cycle Model (SLIM), QSM, (2001) http://www.qsm.com
13. Galorath, D., SEER-SEM, Galorath, Inc., (2001) http://www.galorath.com
14. Jones, C., Knowledge PLAN, Artemis/SPR, (2001), http://www.spr.com
15. Briand L.C., Emam K., and Bomarius F., “COBRA: A Hybrid Method for Software Cost Estimation, Benchmarking, and Risk Assessment”, Proceedings of the 20th international conference on Software engineering, IEEE CS Press, Washington DC, (1998) 390-399
16. Pendharkar P.C., Subramanian G.H., and Rodger J.A., “A Probabilistic Model for predicting software development Effort”, IEEE Transactions on Software Engineering, IEEE Press, Piscataway, NJ, July 2005, pp. 615-624.
17. Yang D., et al, “COCOMO-U: An Extension of COCOMO II for Cost Estimation with Uncertainty”, Proceedings of International Software Process Workshop and International Workshop on Software Process Simulation and Modeling, 2006, pp.132-141.
18. Boehm B., Port D., Huang L.G., Brown W., Using the Spiral Model and MBASE to Generate New Acquisition Process Models, SAIV, CAIV, and SCQAIV, Cross Talk, Jan. (2002)
19. Boehm W., and et al, Guidelines for Lean Model-Based Architecting and Software Engineering (Lean MBASE), http://greenbay.usc.edu/csci577/spring2006/site/guidelines/LeanMBASE_Guidelines_V1.5.pdf
20. Boehm, B., Anchoring the Software Process, IEEE Software, Jul. (1996) 73-82
21. Royce W.E., Software Project Management: A Unified Framework, Addison-Wesley, (1998)
22. Boehm B., The Art of Expectations Management, Computer, Jan. (2000)
23. Yourdon E., Death March, Prentice Hall, (1997)
24. Boehm, B., Software Risk Management, IEEE Computer Society Press, 1989
25. Bryman A., and Cramer D., Quantitative Data Analysis with SPSS, Routledge, (2005)
26. Cohen L., and Holliday M., Statistics for Social Scientists, London:Harper & Row, (1982)
27. Boehm, B. and Turner, R., Balancing Agility and Discipline: A Guide for the Perplexed, Addison-Wesley Professional, (2003)
28. Boehm, B. Port, D. and Jain, A., Achieving CMMI Level 5 Improvements with MBASE and the CeBASE Method, Cross Talk, May. (2002)