http://ansinet.com/itj

ITJ

ISSN 1812-5638

# INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 9 (6): 1104-1114, 2010
ISSN 1812-5638


# An Earned-Value Approach to Assess and Monitor Software Project Uncertainty: A Case Study in Software Test Execution

Xiaochun Zhu and Bo Zhou
College of Computer Science, Zhejiang University, Hangzhou, 310027, China

**Abstract:** In this study, we proposed an earned-value approach to assess and monitor software uncertainty. It combines the value-at-risk method in financial field and the earned-value-feedback process of earned value project management and proposes a framework which contains value-at-uncertainty system, consumed-value-feedback system and experienced base to support in-process uncertainty measurement and contingency buffer management. Value-at-uncertainty system is used to assess the uncertainty in progress of a project and consumed-value-feedback system is applied to provide the decision support of how to deal with the buffer at that specific time. A case study in software test execution of 24 projects is conducted to evaluate the approach. The study shows how our approach works to measure the uncertainty and manage the buffer size in different project shapes. With the approach, both the accuracy and the effectiveness of uncertainty assessment can be improved along with the test execution progress.

**Key words:** Uncertainty assessment, contingency buffer, value at risk, earned value feedback process

## INTRODUCTION

Although, there are kinds of software effort estimation approaches and techniques, including model-based, expertise-based, learning-oriented, regression-based and composite ones (Boehm and Abts, 2000), most software projects still suffer from cost overruns and schedule delays (Jenkins *et al*., 1984; Standish, 1994; Molokken *et al*., 2004). Different approaches do not differ much in estimation accuracy (Molokken and Jorgensen, 2003) even though the approaches take dozens of factors into consideration or introduce quite a few outstanding techniques. One threat to estimation accuracy is the uncertainty which is inherent in software engineering activities. There are so many uncertain factors which can impact the schedule and effort of software projects that almost no estimation approach can cover all of them to make sure the accuracy.

The uncertainty might be caused by known risks or resulted from unforeseen events (Jorgensen, 2005a). Everything which may impact the schedule or the budget but not covered in the plan can be an uncertainty factor.

Software uncertainty assessment methods are mainly categorized into two types:

- Human judgment-based uncertainty assessment
- Formal uncertainty assessment

Until now, the human judgment-based uncertainty assessment methods are typically unconscious without any explicit assessment process or enough systematic feedback (Jorgensen, 2004). However, expert judgment may be more efficient than the formal models in some cases (Jorgensen, 2005a).

Formal models mainly take care of uncertainty assessment by two different ways:

- Integrated models
- Independent models

Integrated models quantify the known risks and uncertainty factors and usually integrate uncertainty assessment into estimation models. The typical model is the Risk Exposure (RE) model (Boehm, 1991) which multiplies the loss by its probability. Based on RE model, Madachy (1994) extended COCOMO (Boehm, 1981) by incorporating risk assessment into cost drivers to estimate effort. Känsälä (1997) proposed a checklist-based cost estimation approach to generate both cost and risk estimates. Braz and Vergilio (2006) introduced fuzzy use case size points to deal with the uncertainty of requirements. Suri and Soni (2007) proposes to simulation techniques to measure the risk when doing project scheduling.

Independent models measure uncertainty directly. Instead of providing a determinate cost on uncertainty, they usually derive prediction intervals or set

**Corresponding Author:** Bo Zhou, College of Computer Science, Zhejiang University, Hangzhou, 310027, China

a probability level to an assessment. There are several ways to estimate the prediction intervals, like analogy based methods (Angelis and Stamelos, 2000; Li *et al*., 2008) empirical and parametric distributions based on previous estimation accuracy (Jorgensen and Sjoberg, 2003), variance model to calculate the maximum expected cost and minimum expected cost (Yang *et al*., 2008) etc. In financial field, risk and uncertainty assessment is also a hot topic and one popular tool is Value at Risk (VaR) system, which has similar essential core with Jorgensen's method. VaR measures the worst expected loss under normal market conditions over a specific time interval at a given confidence level (Jorion, 2001).

A good practice is to combine integrated and independent ways together. However, both of them have limitations to decide how much contingency buffer for uncertainty is enough. Integrated models bring a part of buffer into the whole cost estimation to cover a set of key known risks, but ignore other unforeseen events. Independent models assess the uncertainty directly, but a challenge is how to manage a proper confidence level over the prediction interval so that the interval is not so wide.

As a remedy on uncertainty, to assign a contingency buffer based on the uncertainty and risk assessment is a common technique in project management. However, it is a challenge to decide how much buffer is enough to overcome potential cost of all uncertainties. From the perspective of schedule, it is safe to have a large buffer to keep a project within schedule. But from budget perspective, it is critical to control the buffer size since more buffers mean more cost.

Once a buffer is approved, another challenge is how to monitor the contingency buffer as a project progresses. Along with the engineering activities of a project, the impact of uncertainty will change accordingly too. So, the uncertainty has to be re-assessed again since we have more information and some assumptions maybe proved to be unpractical already. The uncertainty assessment should not be only done once at the beginning but should be monitored through the whole project life cycle. It is valuable to know whether the remaining buffer is large enough to cover the deviations of the remaining project tasks during the project progress so that project manager can ask for more buffer or decide to release some buffer accordingly at earlier stages. However, above approaches mostly focus on uncertainty estimation at the early stage, but none of them aims to monitor and re-assess uncertainty through the whole software project life cycle.

This study introduces an earned value approach to assess the uncertainty and monitor the contingency buffer along with the progress of a project. The approach combines the value at risk concept (Jorion, 2001) in financial field and the earned value feedback process (Fleming and Koppelman, 2005) together to support uncertainty assessment and management.

## VALUE AT UNCERTAINTY

### Value at uncertainty concepts

**Definition 1: Deviation Rate (DR):** The DR indicates the degree of deviations due to risks and unforeseen events against the original plan and it is calculated as:

$$DR = Effort\_Deviation/Planned\_Effort \quad (1)$$

where, Effort_Deviation means the effort deviations and Planned_Effort stands for the estimated effort for a project or a task.

**Definition 2: Buffer Rate (BR):** Buffer rate means the degree of the buffer size against the original plan and it is calculated as:

$$BR = Buffer\_Size/Planned\_Effort \quad (2)$$

Obviously, if the buffer rate of a project is larger than the deviation rate at its closure time, the project does not overrun its schedule and budget. To make DR less than BR is one important criterion of a success project in many cases. However, if the buffer size is too large, it is a waste of project resources and it might reduce the business value of the project. So the goal is to make BR larger than DR, but the gap between BR and DR as small as possible.

**Definition 3: Value at Uncertainty (VaU):** VaU($\alpha$) measures the worst expected effort deviation rate under certain internal and external environment over a project or a task at a given confidence level $\alpha$. VaU customizes itself from VaR as below:

- Take original planned man-effort as the asset
- Use DR as the indicator of loss from uncertainty
- Take certain internal and external environment (stable or similar software development environment) as the normal market
- Use a project or a task as a measure unit

The empirical deviation rates information of previous projects is the base of VaU system and the VaU system provides references to BR. For example:

- If we make VaU(α) as our buffer rate, it means we have α % probability to make the deviation less than the buffer size
- If a buffer has been allocated already, we can calculate the BR and then find an equivalent DR as the given BR in the VaU system to derive the confidence level. We can then know the probability that the buffer size is large enough in normal case

## VAU CALCULATIONS

From mathematics perspective, VaU at the confidence level α is given by the smallest dr such that the probability that the DR exceeds dr is not larger than (1-α), as shown below:

$$VaU(\alpha) = \inf\{dr \in R: P(DR>dr) \leq 1-\alpha\} \quad (3)$$

where, inf means the lower bound of the set.

Assuming $F_{DR}(dr)$ is the Cumulative Distribution Function (CDF) of dr, VaU(α) can be expressed as:

$$VaR(\alpha) = \inf\{dr \in R: F_{DR}(dr) \geq \alpha\} \quad (4)$$

In order to set up VaU systems, the most important thing is to figure out the distribution of deviation rates. There are mainly three ways to establish VaU models (Damodaran, 2007).

Assuming ($dr_1$, $dr_2$, …, $dr_n$) is a set of deviation rates from n previous projects, we can establish VaU models as:

**Historical distribution:** We can historical distribution based on empirical deviation rates directly to establish the VaU model.

A simple way is that we make ascending sort on the set ($dr_1$, $dr_2$, …, $dr_n$) and the sequence list will be ($dr_1'$, $dr_2'$, …, $dr_n'$). So, at confidence level α, we give the percentile (α) as:

$$VaU(\alpha) = dr_k', \text{ for } n\times\alpha \leq k < (n\times\alpha+1) \quad (5)$$

**Parametric distribution:** If we can derive the probability distribution of deviation rates, it should be relatively simple to calculate the VaU. For example, if deviation rates are normally distributed, it is easy to get the estimators of the mean (μ) and the variance (standard deviation squared, $\sigma^2$) from the observations ($dr_1$, $dr_2$, …, $dr_n$). Using maximum likelihood estimation, we have:

$$\hat{\mu} = \bar{d} = \sum_{i=1}^{n} d_i / n \quad (6)$$

$$\hat{\sigma}^2 = \sum_{i=1}^{n} (d_i - \hat{\mu})^2 / n \quad (7)$$

Then we can have the value at uncertainty at the confidence level α as below:

$$VaU(\alpha) = \hat{\sigma} \times \Phi'(\alpha) + \hat{\mu} \quad (8)$$

where, Φ' is the percentile function in a standard normal distribution.

In some cases, deviation rates may follow as the Gamma distribution, the Log-Normal distribution, or some discrete probability distribution, so different parametric VaU models should be established accordingly. More parametric models can be found from Jorgensen's method (Jorgensen and Sjoberg, 2003) or VaR method (Damodaran, 2007). The main difference is that Jorgensen calculates two sides of a prediction interval, while VaR only assesses one bound.

**Simulation approach:** The idea of simulation approach is to simulate DR movements and generate the DR distribution. Since, it is always hard to have enough similar previous projects in a set, simulation technique can be very useful.

One popular simulation method is Monte Carlo simulation (Jorion, 2001). It simulates the random process that how risks and unforeseen events impact the software effort. A random process model such as the Geometric Brownian Motion (GBM) model is defined to help generating thousands of DRs based on the given sample set. And then formula (5) can be applied to derive the VaU(α).

## AN EARNED VALUE APPROACH TO ACCESS AND MONITOR UNCERTAINTY

**Consumed value feedback process:** Once a buffer is allocated, regardless that it is estimated by VaU method or it is given by a fixed value, it is important to monitor and control it through the whole project life cycle.

We introduce the earned value feedback process into our approach to access and monitor uncertainty in process. Since a buffer is not earned but consumed, we call our process as Consumed Value Feedback Process (CVFP).

Figure 1 shows how CVFP works for buffer monitoring and re-assessment and we will introduce the process in detail.

**Set BR referring to VaU(α):** There are different ways to determine the buffer size. A simple one is to using VaU(50%) to set a mean BR, which is equal to a P50 estimate (Jorgensen, 2005b). In case there are a few known risks with high severity, it is common to use a high

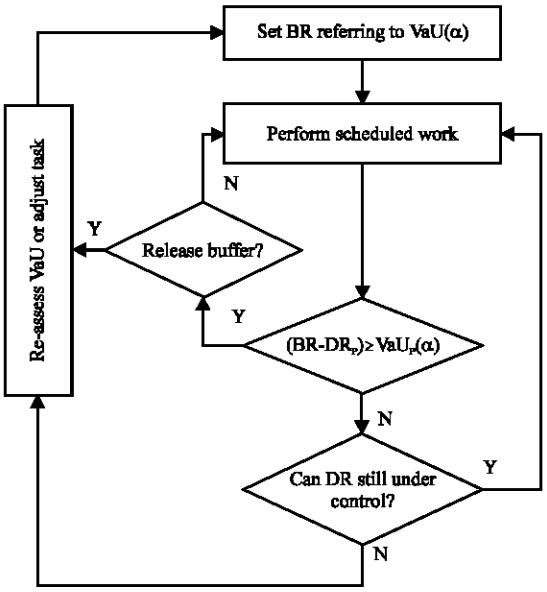


Fig. 1: Consumed value feedback process

confidence level, like 80%. If a project has a fixed delivery date, the buffer size will be derived once the original plan is settled, by calculating the gap between planned date and fixed date. BR can be then calculated and the confidence level α can be estimated via the VaU system.

**Perform the scheduled work:** We define the original planned effort as Total Budgeted Cost of Work Scheduled (TBCWS).

As a project progresses, we derive some quantities at specific time T:

- **The Budgeted Cost of Worked Performed ($BCWP_T$):** The sum of the earned values of all tasks actually completed by time T (Boehm and Huang, 2003)
- **Percentage of $BCWP_T$ ($PoBCWP_T$):**

$$PoBCWP_T = BCWP_T / TBCWS \quad (9)$$

It measures the percentage of work performed against total work scheduled.

We use $PoBCWP_T$ to track the progress of a project. Different projects with different TBCWSs can compare deviation rates with each other at the equivalent degree of progress. Let, us use an example to illustrate why we introduce $PoBCWP_T$.

We have below scenarios:

- Project X has a 150 man-days TBCWS and project Y has a 100 man-days TBCWS
- They start at the same time $T_0$

Table 1: An example of deviations

| Day | Deviation | P (%) | Effort_Deviation$_P$ | DR$_P$ |
|---|---|---|---|---|
| 5 | 1 | 25 | 1 | 0.05 |
| 11 | 4 | 50 | 5 | 0.25 |
| 25 | 1 | 100 | 6 | 0.30 |

- At time T, both project X and Y have the $BCWP_T$ 60 man-days and both of them delay 30 days

Then which project has the better status at time T? They have same $BCWP_T$ and deviations, but the deviation rate of X is smaller than that of Y. It seems X is in better shape than Y. However, given 60% tasks of Y have been completed but only 40% tasks have been done in X, it is hard to make such judgment. We cannot compare between two projects at two different stages. If we use $PoBCWP_T$ instead of the absolute time T, we can compare projects at equivalent progress. Thus we can observe the deviation of Y at 40% $PoBCWP_T$ at time T and then compare with X at time T. We make $PoBCWP_T$ a measure of project progress.

**Monitor the DR in process:** At any time T, when a delay happens, we should assess whether current total deviations exceeds the worst loss at the specific progress P and whether remainder buffer size is large enough to cover future uncertainties, at confidence level α. We introduce several elements to support the monitoring.

**Definition 4: Deviation Rate at progress P ($DR_P$):** Deviation rate at progress P indicates the degree of total deviations due to risks and unforeseen events against the original plan at a specific progress P and it is calculated as:

$$DR_P = Effort\_Deviation_P / Planned\_Effort \quad (10)$$

where, $Effort\_Deviation_P$ means total deviations until progress P and Planned_Effort stands for the total estimated effort for the whole project or task. For a specific project, the Planned_Effort will be fixed once the original plan is set up and agreed by the project team, but the $Effort\_Deviation_P$ will be accumulated as progress moves. When a project is closed, we have the final effort deviation against the original plan and then we can derive our final deviation rate $DR_F$.

Let, us use an example to illustrate the deviation rate system with progress stamp. We have a small project which has an original plan with 20 days and Table 1 shows the deviations through the project life cycle.

At the 5th day, an unexpected issue happens, so we use one day to solve the issue. It means we use 6 days to complete tasks for 5 days. It means we have one day

deviation and the updated project plan would include 21 days. We can calculate the percentage of BCWP at day 6 after the issue is fixed: P=PoBCWP$_6$=5/20=25%. Then, we calculate Effort_Deviation$_{25\%}$ = 1 and the DR$_{25\%}$ = 0.05. We still use original plan 20 days as our basis to evaluate deviation rate, but not the updated plan with 21 days.

At the 11th day, the environment crashes. It takes another 4 days to fix the problem. In day 15, we complete tasks for 10 days in original plan. Now the project delays totally 5 days. The Progress P is 0.50, Effort_Deviation$_{50\%}$ is 5 and DR$_{50\%}$ is 0.25.

At the 25th day, it is supposed that the project should be completed as per the plan which is updated in day 15. However, the project team needs one more day to finish the last task because it is underestimated when doing original plan. Then in day 26, the project is finally completed, so the PoBCWP$_{26}$ is 100%. The total effort deviations are 6 days. We can then have Effort_Deviation$_{100\%}$ = 6 and DR$_F$ = DR$_{100\%}$ =0.30.

**Definition 5: VaU at progress P (VaU$_P$):** The VaU$_P$ measures the VaU from progress P to the end of a project. At any time T, we can calculate the PoBCWP$_T$ to indicate current progress P and then we use DR$_P$ and DR$_F$ of previous projects to build the VaU$_P$ system. The process is described as:

**Step 1 :** Calculate the BCWP$_T$
**Step 2 :** Calculate the PoBCWP$_T$ to indicate progress P
**Step 3 :** Retrieve DR$_P$ data of similar projects/tasks from historical data at equivalent progress P
**Step 4 :** Calculate (DR$_F$ – DR$_P$) for each project/task
**Step 5 :** Build VaU$_P$ system based on the data set of (DR$_F$ – DR$_P$) calculated from step 4

In the definition of VaU$_P$, the (DR$_F$-DR$_P$) means the remainder deviation rate from progress P to the end of a project. So, the VaU$_P$($\alpha$) measures the worst loss of a project from progress P to the end of a project at confidence level $\alpha$. The VaU($\alpha$) we mentioned in previous sections means the VaU$_0$($\alpha$): the VaU at the start point of a project. Since DR$_F$-DR$_0$ = DR$_F$, we use the final DR from empirical data to build the VaU at the first beginning of a project.

Once we get the VaU$_P$($\alpha$), we can then use it to verify whether remainder buffer can still be enough to deal with the uncertainties from remainder BCWS at confidence level $\alpha$. Is (BR-DR$_P$) =VaU$_P$ ($\alpha$)?

If (BR-DR$_P$)=VaU$_P$($\alpha$), it means the remainder buffer can cover the worst loss at confidence level $\alpha$ based on previous projects data at equivalent progress P. It means the buffer rate is still a safe one at the probability $\alpha$. Then can we reduce the buffer in this case? Usually, project managers will say no at early project stages. However, if a project goes very well and has not used any buffer when it arrives at a late stage, it is worth to think about to release some buffer. For example, at time T, PoBCWP$_T$ of a project with a P50 estimate is 80% with DR$_{80\%}$ is 0, so it means the BR keeps as 0.50. It implies there is a great chance that the project will not use out the allocated buffer. So to reduce some buffer for other projects, to add additional tasks, or to plan an early delivery to market can benefit both the product and the organization.

If (BR-DR$_P$)<VaU$_P$($\alpha$), it means the remainder buffer from time T will be probably not enough to remedy the worst loss at confidence level $\alpha$. It does not mean the project will definitely fail to complete within allocated buffer, but implies the probability of failure is high. So, it does not mean we should take actions immediately, but more investigations are required. If the gap between (BR-DR$_P$) and VaU$_P$($\alpha$) is small and known risks can be well controlled as per the planned strategies, we can still continue to perform the scheduled work without adjusting the buffer, while just keeping an eye on the progress and the deviation. But if the gap is big or even the DR$_P$ is large than BR, re-assessment on VaU and correction actions are required.

**Re-assess the VaU and re-set the buffer:** Once the decision is to do correction on BR, re-assessment is taking place. The basic rule is to make the buffer enough for the consumption of uncertainty and in the same time to control the buffer size so as to control total cost.

When a delay happens and the deviation exceeds the worst loss at the original confidence level, we need to increase the buffer rate. A simple strategy is to sum the DR$_P$ and VaU$_P$($\alpha$) as the new BR:

$$BR' = DR_P + VaU_P(\alpha) \quad (11)$$

Using the strategy, we make the assumption that from progress P, our project can work similarly with other previous projects for the remainder BCWS.

Of course, the best way is to use the risk management procedures and techniques to figure out whether the project is at a high risk level or a low risk level and then adjust the $\alpha$ to have a more accurate estimate.

If we want to release some buffer, the same strategy is applicable too. Since, we have a small DR at a late phase, the DR$_P$ will be much less than (BR-VaU$_P$($\alpha$)), so the new buffer rate BR derived from Eq. 11 can be serving to direct the revision of the buffer too.

For some projects, the delivery date is fixed and it is almost impossible to add any buffer. The solution is to cut

some low priority tasks, for example, narrowing regression testing scope. From the view of our value-based system, cutting tasks means reduction on TBCWS and it triggers the change of $PoBCWP_T$ too. Correspondingly, the $VaU_P$ system is also updated because we will use the distribution of DR data from our data set at a later progress P'. The ($BR\text{-}DR_{P'}$) can be probably less than the new $VaU_{P'}(\alpha)$ after the reduction of work scheduled even we use the same confidence level.

**Framework of the earned value approach:** Figure 2 shows the framework of our earned value approach. It consists of three major components: the Consumed Value Feedback System (CVFS), the VaU system and the Experience Base (EB).

The CVFS monitors the whole process and provide decision support, the VaU system provides value of uncertainty and confidence level information and the EB contains the empirical information, including DR data and progress information of previous projects, best practices and rules, profiles and detail information of each project, etc.

It is necessary to have an experience base in our approach. The minimum EB should contain the deviation information and the PoBCWP when each delay happens. More data in EB can enhance CVFS for decision and support VaU models. Besides, the data generated in current CVFP can be stored back into the EB as the experience data for future use.

**Assumptions of the approach:** Since, VaR models assumes financial activities are conducted in normal market, VaU models which are established based VaR theory are only capable to assess uncertainties of a project based on data from a number of similar projects. If there is no similar project in EB, VaU models can't work for target project, even we have many data of projects conduced in very different external and internal environments.

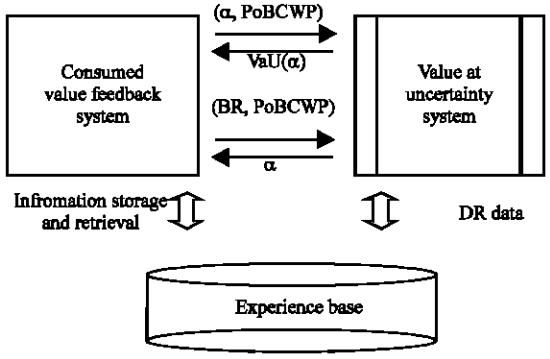


Fig. 2: Framework of the earned value approach

Since, earned value system is a task-based system, the CFVP can only work at a task-driven environment. The project plan should be specified at task level and the tasks should be subdivided at same granularity level.

## THE CASE STUDY

**Target projects:** We conduct a case study on a set of projects form a technology center which serves for financial software company.

Since, all projects in the set are from the same technology center during past 4 years, the business domain, the software development process and standards, the technologies as well as test strategies are very similar. The maturity of software teams on different projects is equivalent and teams keep stable. The similarity on project profiles makes sense for us to use previous project data to direct current project.

We focused on the test execution parts of 24 similar projects in the set. Since, software testing becomes more and more important in software development life cycle, test execution effort now should be well estimated and scheduled (Aranha and Borba, 2007; Silva and Abreu, 2009; Benton, 2009). In our case, independent test teams are assigned to conduct test activities. In each project, there are 3 or 4 testers to run the tests. Test execution activities are estimated and scheduled carefully via same expert-judgment approach. All tasks in the plan are subdivided at one-day granularity level: most tasks are with the budgeted cost of one day and the rest are assigned by 0.5 day. Test plans are well tracked through the same process. Although test execution is much simpler than other software activities, it still suffers from deviations caused by different uncertainties, such as test environment issues, human resources issues, unexpected showstopper defects, etc. We make daily records on deviations which delay the project more than 0.5 day.

Our data in EB contain historical information of each project, including original planned effort for test execution, work tasks and their budgeted/actual cost and information of every delay through the whole test execution progress. The data collection process is based on corporate standard. We derive the data directly from the data repository but not by survey, or other indirect ways so that data quality can be ensured.

Above assumptions can reduce the effort to analyze impact from those external factors.

Since, deviations happen randomly through a project, it is hard to list all the moments in a table. We just pick 10 checkpoints in our case studies as shown in Table 2 to have a simple presentation. It doesn't mean we have to

Table 2: The data set of projects with in-process deviation rates

| | | Deviation rates at different PoBCWPs (%) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Project No. | Scheduled days | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| 1 | 12.5 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 2 | 10.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 3 | 19.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.05 |
| 4 | 16.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.13 |
| 5 | 20.0 | 0.00 | 0.00 | 0.16 | 0.16 | 0.16 | 0.16 | 0.21 | 0.21 | 0.21 | 0.21 |
| 6 | 23.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.22 |
| 7 | 23.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.22 | 0.22 | 0.22 | 0.22 |
| 8 | 29.0 | 0.07 | 0.07 | 0.07 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.24 |
| 9 | 16.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.03 | 0.03 | 0.03 | 0.34 |
| 10 | 12.5 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.36 | 0.36 | 0.36 | 0.36 | 0.40 |
| 11 | 17.0 | 0.18 | 0.18 | 0.18 | 0.18 | 0.18 | 0.42 | 0.42 | 0.42 | 0.42 | 0.42 |
| 12 | 21.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.10 | 0.10 | 0.43 |
| 13 | 24.0 | 0.00 | 0.13 | 0.13 | 0.13 | 0.25 | 0.25 | 0.25 | 0.25 | 0.25 | 0.46 |
| 14 | 21.0 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.24 | 0.24 | 0.24 | 0.48 |
| 15 | 33.0 | 0.16 | 0.16 | 0.16 | 0.16 | 0.16 | 0.16 | 0.21 | 0.21 | 0.21 | 0.58 |
| 16 | 18.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.61 | 0.61 | 0.61 |
| 17 | 13.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.15 | 0.15 | 0.69 |
| 18 | 14.0 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.71 |
| 19 | 13.0 | 0.08 | 0.08 | 0.08 | 0.23 | 0.23 | 0.23 | 0.23 | 0.23 | 0.23 | 0.85 |
| 20 | 16.0 | 0.13 | 0.13 | 0.13 | 0.13 | 0.13 | 0.13 | 0.13 | 0.13 | 0.32 | 0.95 |
| 21 | 13.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.46 | 0.46 | 0.46 | 1.00 |
| 22 | 20.0 | 0.00 | 0.00 | 0.00 | 0.95 | 0.95 | 1.05 | 1.05 | 1.05 | 1.05 | 1.05 |
| 23 | 31.0 | 0.19 | 0.19 | 0.67 | 0.67 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 1.23 |
| 24 | 15.0 | 0.60 | 0.60 | 0.60 | 0.60 | 0.60 | 1.60 | 1.73 | 1.73 | 1.73 | 1.73 |

divide progress into 10 pieces of our approach. Actually, we can do monitoring at any time when a delay happens. For example, if a delay happens at 75% progress and causes a DR of 0.2, we will add this into $DR_{80\%}$ in this study. However, in practice, we can derive $VaU_{75\%}(\alpha)$ model to monitor the uncertainty. In Table 2, the deviation rate in each progress point is an accumulative value. We take project 11 as an example. It takes another 3 days to complete the first 10% tasks of the whole test execution. So, at 10% PoBCWP, the $DR_{10\%}$ is 0.18. There is no delay until progress 50%, so the $DR_{50\%}$ is still 0.18. Between progresses 50 and 60%, a 4-days delay happens. Now, we have delayed 7 days when completing 60% tasks, so the $DR_{60\%}$ can be calculated as 0.42. Since there is no any exception in the following progresses, the $DR_P$ doesn't increase any more.

In our approach, we try to use these data from a statistics view and the data set of similar projects should include all projects, including good projects without any deviation and projects with large deviations. From Table 2, we find it is a sparse matrix. Most projects only contain 2 or 3 deviations and some of them even have no deviation through the whole test execution life cycle. All the data reflect real cases of these projects: most projects will have deviations; however, a single project will only have 2 or 3 deviations with different sizes and the deviations are dispersed at different progress points. This is a typical scenario in uncertainty measurement.

Our approach uses the data in an accumulative way. It avoids the issue that the matrix is too sparse to provide meaningful data. It always assesses deviation rates from current progress point to the end and monitors buffer rates for remainder parts of a project. When a deviation happens, it deducts the deviation rate at current point from buffer rate. And then it assumes the project will go forward normally from a statistics view and measures the value at uncertainties in future progresses of the project with the accumulative deviation rates from current point to the end of the project.

**Assess and monitor uncertainties:** We take several projects to show how the value-based approach works to assess and monitor the contingency buffer for different scenarios. To be simple, we use the historical distribution approach to build VaU models in our analysis.

Firstly, we look at the scenario to increase the buffer of a project. We take project 11 as the target project for assessment and monitoring and leave other projects as the source of EB.

As shown in Table 3, we set confidence level at 50%. We can get the initial BR from $VaU_0(50\%)$. The $VaU_0(\alpha)$ is established from the data set of $DR_{100\%}$ based on the definition 5, so the BR is 0.46. After performing the first 10% budgeted work, the project has a deviation rate of 0.18, so the $(BR\text{-}DR_{10\%})$ is 0.28. The $VaU_{10\%}(50\%)$ at this time is 0.42 which is larger than $(BR\text{-}DR_{10\%})$. The $VaU_{50\%}(\alpha)$ is established based on the $(DR_{100\%} - DR_{10\%})$ data set of similar projects in EB. At this time, we need to think about adjusting the BR. Here we simply apply the strategy as per Eq. 11 and the new buffer rate is 0.60. A delay comes

Table 3: A sample to increase buffer

| | PoBCWP (%) | | | |
|---|---|---|---|---|
| Indicator | 0 | 10 | 60 | 100 |
| $DR_P$ | 0.00 | 0.18 | 0.42 | 0.42 |
| $VaU_P$ (50%) | 0.46 | 0.42 | 0.22 | 0.00 |
| BR' | 0.46 | 0.60 | 0.64 | 0.64 |

Table 4: A sample to keep buffer

| | PoBCWP (%) | | | |
|---|---|---|---|---|
| Indicator | 0 | 10 | 60 | 100 |
| $DR_P$ | 0.00 | 0.14 | 0.24 | 0.48 |
| $VaU_P$ (50%) | 0.46 | 0.42 | 0.21 | 0.00 |
| BR' | 0.46 | 0.56 | 0.56 | 0.56 |

between progresses 50 and 60%, so the accumulative deviation rate $DR_{60\%}$ is 0.42. We re-assess the VaU and find that we need a small update on the buffer again because the $VaU_{60\%}$ (50%) is 0.22, so we turn it to be 0.64. Of course, since the gap between 0.60 and 0.64 is small, if project manager has the confidence that project uncertainty can be well controlled from now on, the deviation rate can keep as 0.60 without any update. In our studies, we strictly follow the Eq. 12. Finally, the final BR is 0.64 while the actual $DR_F$ is 0.42.

Secondly, we look at the scenario to keep the buffer even a delay happens. We use project 14 as the target project and keep others in the EB.

As shown in Table 4, we again set the confidence level at 50%. In the progress interval (0, 10%), a delay with 3 days happens and we re-set buffer rate to 0.56 as per ($DR_{10\%}$+$VaU_{10\%}$(50%)). When completing 70% tasks, there is another small delay. After re-assessment, we find ($DR_{70\%}$+$VaU_{70\%}$(50%)) is less than 0.56. In this case, we will choose to keep the BR as it is. Finally, the BR is 0.56 while the $DR_F$ is 0.48.

Thirdly, we look at the scenario to release buffer. We take project 4 as the target project and put other projects in the EB.

There is no delay until 80% of tasks are completed. Since, it goes so well, we tend to release some buffer from it and plan to deliver it earlier. Then we do the uncertainty assessment and derive the $VaU_{80\%}$(50%) as 0.21. We assume the project can at least go well in average level and reduce the buffer rate from 0.46 to 0.21. The final deviation rate is 0.13 in real life of project 4.

**Evaluate the approach:** From above samples, we see the value-based approach can timely assess the buffer and adjust accordingly as a project progresses. However, not every adjustment is effective. In the sample presented in Table 3, the original buffer rate 0.46 is more accurate than the revised buffer rate 0.64. But in the sample presented in Table 5, the revised buffer rate is better than the original one.

Table 5: A sample to release buffer

| | PoBCWP (%) | | | |
|---|---|---|---|---|
| Indicator | 0 | 10 | 60 | 100 |
| $DR_P$ | 0.00 | 0.00 | 0.00 | 0.13 |
| $VaU_P$ (50%) | 0.46 | 0.21 | 0.21 | 0.00 |
| BR' | 0.46 | 0.46 | 0.21 | 0.21 |

In order to evaluate our approach, we introduce two metrics.

**Mean Absolute Error (MAE):**

$$MAE = \sum_{i=1}^{N} |BR\text{-}DR_F|_i / N \quad (12)$$

where N is No. of projects, BR is the current buffer rate and $DR_F$ is the final rate.

**Hit Rate (HR):**

$$HR = k/N \quad (13)$$

where, k is No. of projects which BR = $DR_F$ and N is No. of total projects in scope. Hit here means BR is no less than $DR_F$, while miss means BR is less than $DR_F$.

We use these two metrics to measure how our approach works during the test execution of all projects in our data set.

We do the cross validation (Mendes *et al*., 2008) against our date set and the algorithm is described as:

**Step 1:** Take out a project $P_i$ from EB and keep other projects in EB as the VaU support data
**Step 2:** Assess and monitor the uncertainties of $P_i$ via CVFP based on VaU models at each time
**Step 3:** Record the BR and Hit or Miss during CVFP
**Step 4:** Put $P_i$ back into EB
**Step 5:** Repeat step1~4 until all projects go through the CVFP.

Since, we are doing the evaluation on completed projects, it means we already know about actual results of the deviations of all projects. So, it seems not suitable for us to configure specific confidence level for each project based on more clear information, because known results may lead us to make inequitable decisions. Thus, we give a guideline which treats all projects in our studies equally. It sets the confidence level as 50% which is a common level in effort estimation, uses the strategy as Eq. 11 and releases buffer when a project does not have any delay until PoBCWP 80%.

The evaluation results are shown in Table 6. We review the MAE and HR at different progress points.

Table 6: Evaluation results

| | PoBCWP (%) | | | |
|---|---|---|---|---|
| Metrics | 0 | 30 | 50 | 80 |
| MAE | 0.335 | 0.285 | 0.264 | 0.185 |
| HR | 12/24 | 15/24 | 16/24 | 17/24 |

MAE means the accuracy of assessment, while HR stands for the effectiveness of assessment. The goal is to make buffer size larger than the total deviation and in the same time to be close enough to the deviation. It means a good result is to have a large HR and a small MAE.

We measure the MAE and HR at four progress points. At the first beginning, we just use $VaU_0$ (50%) to assess the uncertainty. The MAE is 0.335 while the HR is 50%. And then, we start to use the consumed value feedback system to monitor the buffer rate. Until PoBCWP 30%, the system makes 12 updates on buffer rates for all projects. The MAE and HR are then calculated from adjusted BR at PoBCWP 30% and $DR_F$. The MAE decreases to 0.285 while the HR increases to 62.5%. At PoBCWP 50%, the assessment is more accurate and the hit rate is higher after the system increases 2 buffer rates. At PoBCWP 80%, due to 12 buffer changes including 5 buffer releases, the MAE is greatly improved to be 0.185, while the HR still keeps high.

The MAE decreases from 0.335 to 0.185 as the project goes from the beginning to PoBCWP 80%. It means the accuracy of the buffer rate assessment is improved as the project moves forward. The HR increases from 12/24 to 17/24. It has a greater chance that revised buffer size generated from the earned value approach will be not used out by deviations. The evaluation results indicate both the accuracy and effectiveness of uncertainty assessment increase as a project goes on.

## DISCUSSION

The VaU model is introduced to assess the uncertainty from the statistics view. If we use the Risk Exposure (RE) model (Boehm, 1991) since, the projects in our case study are very similar, the RE values of each project should be very close too. However, the real deviation rates range from 0 to 1.73. In our approach, with our VaU model, we suggest a buffer rate with a given confidence level. This gives more information for managers to monitor the uncertainty.

Although Jorgensen's distribution interval model (Jorgensen and Sjoberg, 2003) also contains the probability information along with an interval, it is mainly for estimation at the first beginning. It means with distribution interval model with confidence level 50%, we can have the MAE 0.335 and HR 50% in our case too. However, since our approach applies the VaU models within the CVFS, we can assess the uncertainty at each point. Based on the case study, when a test project moves forward, the uncertainty assessment can be more accurate and effective at a later stage as per the MAE and HR indicators (Table 6). Since, we can have more information as a project moves forward, we should be in a better and better status to assess the uncertainty. More information can make it more efficient. In case studies, we mainly use the deviation rate and value at uncertainty at certain progress points to adjust the buffer rate. However, if experts can have more effective feedback on project estimation information (Jorgensen and Gruschke, 2009), they can adjust the buffer rate by other strategies.

Comparing with traditional earned value feedback process (Fleming and Koppelman, 2005) our approach provide decision support on contingency buffer monitoring. It provides the prediction on future uncertainty before buffer is used out. The VaU assessment also provides the information on how much buffer should be added or can be reduced at a confidence level.

From the source data in Table 2, the deviations are not distributed into each progress interval equally. A project with a good start might have a bad end. The deviations in test execution approximately follow the 80-20 rules. About half deviations happen during the last progress interval (90, 100%). It doesn't follow the cone of uncertainty concept (McConnell, 1996) that uncertainty decreases throughout the project. Also, the Landmark's data show that the uncertainty range was nearly identical (Little, 2006). It means the uncertainties in different software organizations don't have identical trend. It is hard to have a universal distribution to assess the uncertainty. However, since our approach is based on the historical data, it doesn't make any assumption about the uncertainty trends. The experience data which used to build VaU model can reflect the deviation distributions.

When looking at our data set, another catch is that there is no negative deviation rate. A negative deviation rate means we do things so good that we are in advance of our plan. One reason is that our original plans are always made in an aggressive mode and another reason is that testers tend to use out planned time on some invisible tasks even they completed scheduled tasks ahead of schedule. This complies with Parkinson's Law (Parkinson, 1955). For example, some exploratory testing will be done after test case based testing is completed, but the exploratory testing is actually not stated in our plan. However, even if there are negative deviations, our approach can still work in theory.

## CONCLUSIONS AND FUTURE WORK

We propose an earned value approach to assess and monitor uncertainties timely along with the progress of a software project.

Firstly, the concepts of value-at-risk are introduced to measure the value at uncertainties. VaU models can be easily built based on historical data at different progress points.

Secondly, the earned value feedback process is adapted to contingency buffer assessment and management. The consumed value feedback process is running to assess and control the buffer rate at each progress point.

Thirdly, a new metric PoBCWP is used to measure the progress of a project. It is the base of our in-process assessment and monitoring approach.

The results of case studies indicate that our approach can well support in-process uncertainty assessment and monitoring in test execution activities, including increasing buffer size, keeping buffer size and decreasing buffer size. The accuracy and effectiveness indicators, MAE and HR, also show positive results.

However, there are still some challenges in our approach.

Firstly, we just validate our approach in test execution activities, but not the whole Software Development Life Cycle (SDLC). The performance of the approach on other software activities is still unknown. Since the whole SDLC is more complex than test execution, our approach may face new challenges. One future work is to apply our approach on other software activities or even the whole SDLC.

Secondly, the confidence level setting is a challenge. 50% seems a safe choice in case we do not have much information, but if our knowledge about a project is good enough, we can definitely provide a more reasonable $\alpha$. One future work is to develop guidelines for configuring a good confidence level so that we can have a good start in uncertainty assessment. The information of empirical database will be valuable to set a confidence level.

Thirdly, the deviation records at different progress points of previous projects are probably not available in quite a few organizations. The necessity of such data is another challenge. To build a framework to support data collection on deviation and progress data is another future work. To introduce simulation techniques, e.g., Monte Carlo simulation, is one solution on data limitation.

Fourthly, about the value of uncertainty, we make the assumption that all efforts at different project stages are equivalent towards their value to a project. However, it is probably not true. The deviations at an early stage may just lead to a scope change, while delays in a late stage probably cause a fatal damage to a project. To realize real values in different stages will be a good enhancement to our approach.

## REFERENCES

Angelis, L. and I. Stamelos, 2000. A simulation tool for efficient analogy based cost estimation. J. Empirical Software Eng., 5: 35-68.

Aranha, E. and P. Borba, 2007. An estimation model for test execution effort. Proceedings of 1st International Symposium on Empirical Software Engineering and Measurement, Sept. 20-21, Madrid, pp: 107-116.

Benton, B., 2009. Model-based time and cost estimation in a software testing environment. Proceedings of 6th International Conference on Information Technology: New Generations, April 27-29, IEEE Computer Society,Washington DC, USA., pp: 801-806.

Boehm, B.W. and C. Abts, 2000. Software development cost estimation approaches: A survey. Ann. Software Eng., 10: 177-205.

Boehm, B.W. and L. Huang, 2003. Value-based software engineering: Reinventing earned value monitoring and control. ACM SIGSOFT Software Eng. Notes, 28: 1-7.

Boehm, B.W., 1991. Software risk management: Principles and practices. IEEE Software, 8: 32-41.

Braz, M.R. and S.R. Vergilio, 2006. Software effort estimation based on use cases. Proc. 30th Annu. Int. Comput. Software Appl. Conf., 1: 221-228.

Damodaran, A., 2007. Value at Risk, Strategic Risk Taking: A Framework for Risk Management. Wharton School Publishing, USA.

Fleming, Q.W. and J.M. Koppelman, 2005. Earned Value Project Management. 3rd Edn., PMI, Karachi.

Jenkins, A.M., J.D. Naumann and J.C. Wetherbe, 1984. Empirical investigation of systems development practices and results. Inform. Manage., 7: 73-82.

Jorion, P., 2001. Value at Risk: The New Benchmark for Managing Financial Risk. McGraw Hill, New York.

Jorgensen, M. and D.I.K. Sjoberg, 2003. An effort prediction interval approach based on the empirical distribution of previous estimation accuracy. Inform. Software Technol., 45: 123-126.

Jorgensen, M., 2004. Top-down and bottom-up expert estimation of software effort. Inform. Software Technol., 46: 3-16.

Jorgensen, M., 2005a. Evidence-based guidelines for assessment of software development cost uncertainty. IEEE Trans. Software Eng., 31: 942-954.

Jorgensen, M., 2005b. Practical guidelines for expert-judgment-based software effort estimation. IEEE Software, 22: 57-63.

Jorgensen, M. and T.M. Gruschke, 2009. The impact of lessons-learned sessions on effort estimation and uncertainty assessments. IEEE Trans. Software Eng., 35: 368-383.

Känsälä, K., 1997. Integrating risk assessment with cost estimation. IEEE Software, 14: 61-67.

Li, Y.F., M. Xie and T.N. Goh, 2008. A study of analogy based sampling for interval based cost estimation for software project management. Proceedings of 4th IEEE International Conference on Management of Innovation and Technology, Sept. 21-24, National University of Singapore, pp: 281-286.

Little, T., 2006. Schedule estimation and uncertainty sur-rounding the cone of uncertainty. IEEE Software, 23: 48-54.

Madachy, R., 1994. Knowledge-based risk assessment and cost estimation. Proceedings of 9th Knowledge-Based Software Engineering Conference, Sept. 20-23, California, USA., pp: 172-178.

McConnell, S., 1996. Rapid Development: Taming Wild Software Schedules. Microsoft Press, USA.

Mendes, E., S. di Martino, F. Ferrucci and C. Gravino, 2008. Cross-company vs. single-company web effort models using the Tukutuku database: An extended study. J. Syst. Software, 81: 673-690.

Molokken, K. and M. Jorgensen, 2003. A review of surveys on software effort estimation. Proceedings of the 2003 International Symposium on Empirical Software Engineering, Sept. 30-Oct. 01, IEEE Computer Society, Washington DC, USA., pp: 223-223.

Molokken, K., M. Jorgensen and S.S. Tanilkan, 2004. A survey on software estimation in the norwegian industry. Proceedings of 10th International Symposium on Software Metrics, Sept. 11-17, IEEE Computer Society, Washington DC, USA., pp: 208-219.

Parkinson, N., 1955. Parkinson's law, Economist. http://www.economist.com/business-finance/management/displaystory.cfm?story_id=14116121.

Silva, D.G. and B.T. Abreu, 2009. A simple approach for estimation of execution effort of functional test cases. Proceedings of the 2009 International Conference on Software Testing Verification and Validation, April 1-4, IEEE Computer Society, Washington DC, USA., pp: 289-298.

Standish, G., 1994. The standish group report. http://www.projectsmart.co.uk/docs/chaos-report.pdf.

Suri, P.K. and R. Soni, 2007. Simulating the potential effect of risk management on project scheduling. Inform. Technol. J., 6: 8-13.

Yang, B., H. Hu and L. Jia, 2008. A study of uncertainty in software cost and its impact on optimal software release time. IEEE Trans. Software Eng., 34: 813-825.