Continuous Productivity Assessment and Effort Prediction Based on Bayesian Analysis

Seok Jun Yun and Dick B. Simmons

Software Process Improvement Laboratory
Department of Computer Science
Texas A&M University
College Station, TX 77843-3112
Email: {sjy3806, simmons}@cs.tamu.edu

Abstract

Project management is one of the most critical activities in modern software development projects. Without realistic and objective management, the software development process cannot be managed in an effective way. However, difficulty in assessment of project attributes leads a project into failure. Therefore, it is essential to keep providing objective assessment of project attributes as software development evolves. Another important aspect of a software development project is to know how much it will cost. And predicting development effort is central to the project management. However, effort prediction is one of the most difficult tasks in project management. We use Bayesian approach to update productivity and predict effort based on the updated productivity. In this paper, we describe an extended tool that we added to PAMPA 2 (Project Attributes Monitoring and Prediction Associate) to help manage a project.

Keywords - Bayesian Theory, Productivity, Software Engineering, Software Process Improvement, Software Project Management, Knowledge-Based Systems.

1 Introduction

A critical problem facing software development in today's competitive environment is project management. Project management is the primary key to the success of software development. However, project management is not always easy due to difficulty in assessment of project attributes. Currently, most assessment depends on manual procedures. Manual procedures can result in inaccuracies that result from subjective assessment. An incorrect decision based on faulty assessment can result in project failure. Therefore, it is essential to develop a tool to provide objective assessment of project attributes as software development evolves.

Another important aspect of a software development project is to know how much it will cost. And predicting development effort is central to the project management. However, effort prediction is one of the most difficult tasks in project management. Even though many effort estimation models were suggested to date, none of them succeeded to predict development effort accurately. It is recommended to calibrate a model to an organization's own data to increase its accuracy, however, the calibration is difficult to nonexpert.

In this paper, we focus on productivity update instead of calibrating a model. Productivity of a project can be estimated with an effort estimation model, for example, COCOMO II. This productivity is not accurate enough to tell the true effort of the project. Bayesian approach provides a way of updating the productivity with data gathered from the project. And the updated productivity can be used to predict effort of the remaining of the project with better accuracy.

PAMPA 2 was recently developed to describe plans based on an incremental evolutionary project life cycle [12]. Knowledge can be acquired from software development experts and CASE tool database to create a knowledge base. Metrics gathered from CASE tool database can drive a visualization toolkit to assist managers in directing software projects. We developed an expanded tool to use the knowledge base.

The body of this paper is organized as follows. Section 2 introduces productivity. Section 3 explains the features of PAMPA 2. Section 4 describes productivity assessment and section 5 effort prediction. Section 6 illustrates graphical web-based console. Discussion on the tool and conclusion are in section 7 and 8.

2 Productivity

A software project develops a software product (SP). SP status can be observed by tracking SP features, artifacts, known defects, reported problems, testing activity and approved changes. Examples of artifact are user requirements, design documents and source codes. An important artifact is source code used to create the executable file that is delivered to a customer.

Volume attribute is used to describe physical magnitude, extent or bulk of artifacts. Equivalent source lines of code (SLOC), function points, and object points are metrics used to measure volume. Volume can be used to track the progress of development. Effort attribute is the amount of resource expense required to produce an artifact. The effort of personnel is the main cost in a software development project. A widely used effort metric is person-month (PM).

Software productivity is the rate at which software product artifacts are produced in relation to the time, and resource. Software productivity is usually defined as volume divided by effort. Software productivity Pr is expressed as:

$$Pr_i = \frac{V_i}{E_i}, \tag{1}$$

where V_i is the volume of artifact *i*, E_i is the amount of effort expended to produce artifact *i*.

The Internet environment enables development to be distributed across the world. And when the budget for development is limited, employing cheaper labor can decrease the total cost of development. For example, an entry-level programmer's salary ranges from \$167 to \$417 per month in India. That programmer's US counterpart typically commands \$4,167 to \$5,000 per month [5]. Therefore, the salary is an important attribute to account for resource expense in developing an SP in more than two countries. Labor cost L is:

$$L_i = E_i \times S_j, \tag{2}$$

where S_j is salary rate of a person j. After taking labor cost into account, we can change productivity as:

$$Pr_i = \frac{V_i}{L_i}, \tag{3}$$

Productivity in volume per dollar is useful to compare productivity between geographically distributed sites. For example, given their performance are same, programmers in India are 10 times more productive than those in US when you use labor cost instead of effort. Nowadays, many software companies outsource their development work to other countries that have cheaper labor. And labor cost gives a manager a view of controlling resource expense in the multisite development environment.

3 PAMPA 2

PAMPA 2 was developed to help a manager view projects by gathering project attributes and presenting project status. As the complexity of software development environment increases, CASE tools such as Rational Rose, RequisitePro, ClearCase, ClearQuest, Test Studio, and MS Project are used to support developers. PAMPA 2 gathers critical project attributes from the CASE tool databases, stores the attributes in the PAMPA 2 KB, and provides status of a project via web-based consoles.

3.1 CASE tools and PAMPA 2

MS Project provides project plan, work breakdown structure (WBS), and resource information such as salary. RequisitePro is a requirements management system. ClearCase is a Configuration Management System (CMS) to help software developers track files and directories used to create an SP. ClearQuest is a change and defect management system. ClearQuest works with ClearCase to track change/defect in an evolving project.

PAMPA 2 gathers plan, process and activity attributes from MS Project, feature attributes from RequisitePro, artifact attributes from ClearCase, and change/defect attributes from ClearQuest, respectively, and stores them in the KB. PAMPA 2 was built on three-tier web based architecture.

3.2 Configuration management system (CMS)

During the development stage, developers create artifacts according to requirements. ClearCase is used for online storage of project artifacts and version control management. We created **Data Transformation** module to access to the storage via COM interface (ClearCase Automation Library).

Today, multiple programming languages are used in developing an SP. And it is needed to compare cost between artifacts created with different programming languages. Capers Jones provided a conversion ratio chart of more than 20 programming languages [6]. The chart shows ratios to convert volume of one programming language to that of another. Therefore, it is possible to equate product volume between programming languages. **Data Transformation** module converts volume metric according to the chart, and stores the equivalent volume into **Volume** of the KB.

ClearCase supports parallel software development and software reuse across geographically distributed project teams. Developers at different locations can use the same Versioned Object Base (VOB). Each site has its own copy, or replica, of that VOB. The set of replicas for a particular VOB is called a VOB family. At any time, a site can prop-

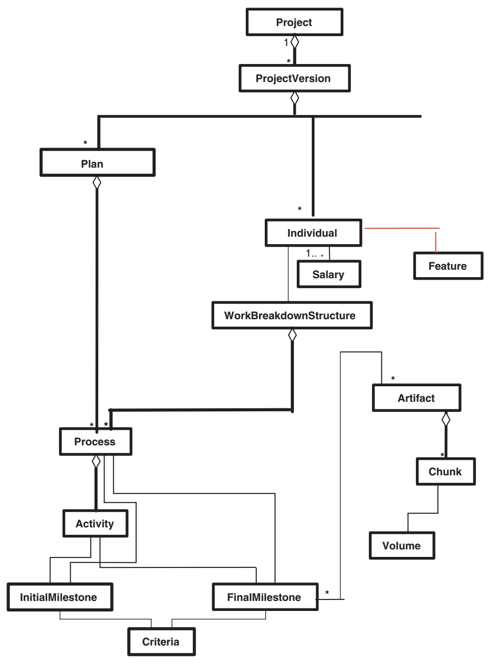

Figure 1. Detailed KB schema on plan

agate changes to other sites, using either an automatic or manual synchronization process.

3.3 Productivity measurement

As shown in Fig. 1, a project plan consists of activities. An activity is the smallest work package that has milestones and assigned individuals (designer, developer, etc). PAMPA 2 stores activity information (id number, activity name, activity type, etc) in Activity, and milestones in **InitialMilestone** and **FinalMilestone**. **Individual** is many-to-many mapping to **Activity**. Each individual has salary rate in **Salary**. The feature attributes for an activity are stored in **Feature**. **Feature** is one-to-one mapping to **Activity**. As a software project evolves, programmers create artifacts according to the requirements. **Artifact** stores the artifact information: artifact type, file name, directory, and programming language to develop the artifact. **Artifact** is one-to-one mapping to **Activity**. An artifact can have several chunks. Volume metric is stored in **Volume**.

The volume and cost gathered in the KB are used to measure productivity. For example, if an activity is completed, we can get effort from the difference of **InitialMilestone** and **FinalMilestone**, salary rate from **Salary**, and volume from **Volume**.

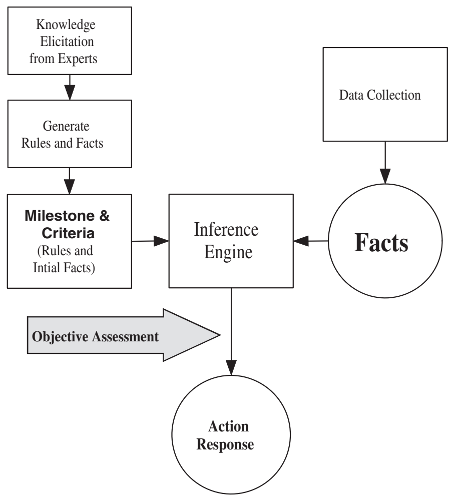

Figure 2. Expert system

4 Productivity assessment

4.1 Expert system

An expert system crystallizes and codifies the knowledge and skills of experts into a tool that can be used by non-specialists [7]. An expert system consists of a knowledge base and an inference engine. The knowledge base contains the domain-specific knowledge of a problem. The inference engine consists of procedures for processing the encoded knowledge of the knowledge base together with any further specific information at hand.

Knowledge in the form of rules and facts is acquired from experts. PAMPA 2 stores the knowledge in the KB. Inference engine analyzes project data (**Facts**) with the knowledge (**Rules and Initial Facts**), and reports assessment as shown in Fig. 2. Therefore, when a software project encounters problems such as progress delay and resource deficit, expert system assists a manager in making appropriate decisions.

4.2 Rules and initial facts

Many factors affect a project. Factors exist as rules and facts which can be acquired from experts' knowledge. In this paper, we will discuss factors primarily related to productivity as shown in Table 1. A software development expert defines the factors, and sets the values of the factors as initial facts for each activity. Estimated Effort is the amount of effort allocated to an activity in plan, which has LowerLimit, Expected and UpperLimit value. The Expected value means that an activity would be best to finish in the time. And the LowerLimit and UpperLimit values

Table 1. Factors

Factors	Unit	Values
Est. Effort	PM	LowerLimit, Expected, UpperLimit
Est. Productivity	Volume/PM	LowerLimit, Expected, UpperLimit
Est. Cost	Dollar	LowerLimit, Expected, UpperLimit
Est. Artifact Volume	Volume	Expected

give an interval within which the completion of an activity is expected to fall with a marginal effect on plan. If an activity takes longer than UpperLimit or finishes earlier than LowerLimit, then plan for the activity and other following activities should be adjusted accordingly. The values of Estimated Productivity, Estimated Cost, and Estimated Artifact Volume can be set as well.

Once defined and set, **Rules and Initial Facts** are stored in the **Criteria** of the KB. Expert system uses these **Rules and Initial Facts** and tests them on **Facts** (Fig. 2). **Facts** are project attributes gathered from CASE tool databases. Expert system monitors new **Facts** and tests the **Rules and Initial Facts** on them without intervention of a manager.

5 Effort prediction

5.1 Productivity update

Over the past two decades there has been considerable activity to provide effort estimation models. Most of the models have been drawn from completed software projects. Unfortunately, there is anecdotal evidence that effort estimation models have high error rates [9] [10] [11]. As a result, it is recommended to calibrate the model to an organization's own actual data to increase the model's accuracy [2]. Bayesian approach was used to calibrate the cost drivers of the COCOMO II model. Bayesian approach permits an investigator to use sample data to update prior (expert-judgment) information in a logically consistent manner. However, a lot of expertise was needed to determine the prior information of the 17 cost drivers.

In this paper, we use Bayesian approach on productivity update. To make a plan, we estimate productivity of a project using one of the effort estimation models and

Table 2. Estimated productivity

Man.	Pr	Man.	Pr	Man.	Pr
A	296	D	242	G	414
B	435	E	446	H	340
C	309	F	268	I	414

store the value in Estimated Productivity. As the project evolves, we measure productivity from artifact volume and effort as shown in Fig. 1. We have prior information about productivity (Estimated Productivity), and sample data (measured productivity). Thus we can use Bayesian approach to integrate the information about productivity. And the updated productivity can be used to predict effort for the remaining of the project.

5.2 Bayesian approach

Bayesian approach provides a mechanism for updating initial probability statements about parameters with the sample data observed [4].

$$p(\theta|y) \propto p(y|\theta)p(\theta), \tag{4}$$

where $p(\theta)$ is the prior distribution, and $p(y|\theta)$ is the sampling distribution. The posterior distribution $p(\theta|y)$ is proportional to the product of prior and sampling distribution. Bayesian inference about a parameter θ is, therefore, conditional on observed sample data.

5.3 Prior distribution

To determine the prior distribution, we use the COCOMO II cost estimation model. The COCOMO II is an algorithmic model to estimate effort [1]. It requires volume of artifacts and cost drivers to estimate effort in PM. A cost driver is a model factor that affects the effort to complete a project. There are 17 cost drivers in COCOMO II. Effort multiplier (EM) is a value of rating level of a cost driver. And effort is estimated with the model:

$$E = 2.94 \times V \times \prod_{i=1}^{n} EM_i, \tag{5}$$

where $V = Size^E$. By the way, productivity is estimated from the model directly. This model provides productivity estimate for a project:

$$Pr = \frac{1}{2.94 \times \prod_{i=1}^{n} EM_i}. \tag{6}$$

Table 3. Posterior productivity

Sample Data	Mean	Confidence Interval
10	320	292, 351
20	312	295, 330
30	309	296, 322
40	308	301, 314

The estimation of the COCOMO II depends on human judgment on the EM of the cost drivers. Therefore, the judgment can be different between humans. We surveyed 9 project managers to estimate productivity of a web-based software development project. The results are shown in Table 2. The 9 estimates can be used to create a prior distribution. The mean value of productivity is 351 SLOC per PM. This is the value stored in Estimated Productivity.

The distribution of productivity is well known of its positive skewness [1] [8]. To approximate the normal distribution, natural log transformation is applied to productivity. We created a prior distribution with mean μ, 5.84, and variance σ^2, 0.051 in log form.

5.4 Posterior distribution

In this paper, we use a two-parameter univariate normal sampling model to make inferences about mean and variance of productivity. And we assume that the mean and variance are interdependent. After observing n sample data, the marginal posterior distribution of μ is:

$$p(\mu|y) \sim N(k, \sigma^2/(\mu_0 + \mu_n)), \tag{7}$$

where $k = (n_0\mu_0 + n\bar{y})/(n_0 + n)$ is the precision weighted average of the prior and sample data mean, σ^2 is the variance, μ_0 is the prior sample size, μ_n is the sample size, and n_0 is the prior sample size [3].

The variance is the inverse of the precision. The precision is an important parameter, because the higher the precision, the more highly concentrated are observations expected to be around the mean. The precision has a Gamma distribution. The marginal posterior distribution of τ is:

$$\tau = \sigma^{-2} \sim G(v/2, v\sigma_n^2/2), \tag{8}$$

where v is the posterior degrees of freedom, and σ_n^2 is the sample variance [3].

In this paper, we collected 40 sample data from the project. Every 10 sample data, we obtained the posterior distribution, and transformed it to calculate posterior productivity. The results are shown in Table 3. The table shows productivity mean and its 95% confidence interval. Posterior productivity replaces the value in Estimated Productivity whenever a new value is obtained. We observed that productivity was stabilized after 20 sample data. After completing the project, we concluded that the true productivity of the project is 308 SLOC per PM.

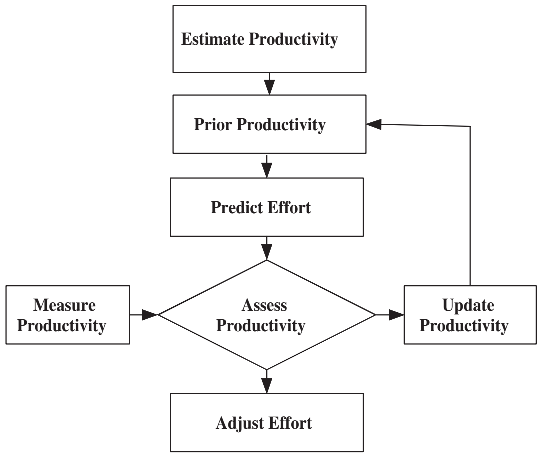

Figure 3. Prediction and adjustment

5.5 Prediction and adjustment

After getting the posterior productivity, we can predict effort of the remaining of the project, and store new value in Estimated Effort. By the way, we use the measured productivity to adjust the effort of the finished activity. With the effort prediction and adjustment, we can calculate the total effort of the project: actual effort of finished activities plus estimated effort of unfinished activities. And the posterior distribution can act as the prior distribution for the next observed sample data. Therefore, this procedure is continuously performed till the project ends.

6 Graphical web-based console

Graphical charts are also provided to help project management. Gantt chart is one of the visual aids to monitor development progress. Gantt chart shows activity names, milestones and schedules. Each activity has two bars: one shows planned duration and the other one progress. In addition to the Gantt chart, we created Activity Network chart. Activity Network chart shows the relationships between activities as well as activity information: activity name, milestones and duration.

Control center was created to help project management. Control center provides four dial charts for managers to quickly discern the true status of a project. It consists of **Schedule**, **Progress**, **Productivity**, and **Resource** meter. **Schedule** meter shows time spent in a schedule,

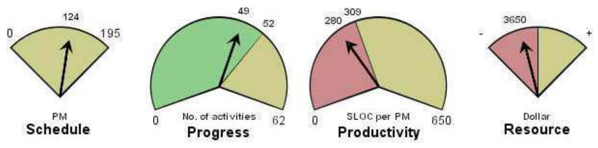

Figure 4. Control center

Progress meter progress of a project, **Productivity** meter productivity, and **Resource** meter resource balance.

Schedule meter has an arrow which points to the expended time in plan. In Fig. 4, we can tell the project spent 124 out of 195 PM. **Progress** meter has a line and an arrow. The line indicates the number of activities scheduled to finish to date. And the arrow points to the number of finished activities. Currently, 49 activities were completed out of 52 activities scheduled to finish to date. There are total 62 activities in the project. **Productivity** meter has a line and an arrow. The line indicates the estimated productivity. And the arrow points to measured productivity. Current productivity is 280 whereas estimated productivity is 309. **Resource** meter has two areas. The left area shows that resource is deficient, and the other one shows otherwise. The arrow points to the center line when resource is consumed as planned. We tell the project is suffering from budget deficits of 3,650 dollars.

Control center gives a manager a quick view of project status. Gantt and activity network charts show detailed views of each activity. Therefore, the three charts will be used in combination to monitor project management.

7 Discussion on the tool

We applied the tool in several projects. From the study we can state that:

- Attributes gathering: The automatic attributes gathering feature of PAMPA 2 increases effectiveness of project attributes assessment and prediction.
- Adopting labor cost: It gives a manager a view of controlling resource expense in multisite development environment.
- Productivity assessment: Objective assessment of project attributes helps a manager easily find out problems and take corrective actions.
- Effort prediction: Bayesian approach provides a convenient way of updating productivity. And the accuracy of effort prediction increases as more data gathers.
- Web-based console: It helps a manager monitor project status via Internet.

We have discussed the benefits of the tool in project management. Nonetheless, its use is limited to those projects that use CASE tools. However, CASE tools are widely used in industry, and PAMPA 2 can be calibrated easily with any CASE tools.

8 Conclusion

This paper describes a tool of productivity assessment and effort prediction based on Bayesian approach that we added to PAMPA 2. An expert system and graphical web-based console were created to assess measured productivity compliance to estimated productivity, predict effort, and then make effort adjustment based on measured productivity.

Project management is the primary key to the success of software development. With the aid of the productivity assessment and effort prediction tool, successful project management is possible because of continuous objective assessment and prediction which help a manager make timely adjustments to control a project.

References

[1] B. Boehm, C. Abts, A. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece. *Software Cost Estimation with COCOMO II*. Prentice-Hall, Upper Saddle River, NJ, 2000.

[2] S. Chulani, B. Boehm, and B. Steece. Bayesian Analysis of Empirical Software Engineering Cost Models. *IEEE Trans. Software Eng.*, 1999.

[3] P. Congdon. *Bayesian Statistical Modeling*. John Willey & Sons, Upper Saddle River, NJ, 2001.

[4] A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis*. Chapman & Hall, Boca Raton, FL, 1995.

[5] N. Goth. Bottlenecked in Bangalore. *Red Herring Communications*, 1997.

[6] C. Jones. *Applied Software Measurement: Assuring Productivity and Quality*. McGraw-Hill, NY, 1996.

[7] G. Joseph and R. Gary. *Expert Systems: Principles and Programming*. PWS Publishing Company, Boston, 1998.

[8] M. Katrina. *Applied Statistics for Software Managers*. Prentice-Hall, Upper Saddle River, NJ, 2002.

[9] C. Kemerer. An Empirical Validation of Software Cost Estimation Models. *Comm. ACM*, pages 416–429, 1987.

[10] B. Kitchenham and N. Taylor. Software Cost Models. *ICL Technology J.*, pages 73–102, 1984.

[11] Y. Miyazaki and K. Mori. COCOMO Evaluation and Tailoring. *Proc. Eighth Int'l Conf. Software Eng.*, 1985.

[12] D. Simmons and C. Wu. Plan Tracking Knowledge Base. *Proc. of the Twenty-Forth Annual Int'l Computer Software and Applications Conf.*, pages 299–304, 2000.