Earned Value Management

- Earned value management (EVM) is a project performance measurement technique that
- integrates scope, time, and cost data.
- Given a cost performance baseline, project managers and their teams can determine how well the project is meeting scope, time, and cost goals by entering actual information and then comparing it to the baseline.
- A baseline is the original project plan plus approved changes.
- Actual information includes whether or not a WBS item was completed or pproximately how much of the work was completed,when the work actually started and ended, and how much it actually cost to do the completed work.

Earned value management involves calculating three values for each activity or summary activity from a project s WBS.

1. **The planned value (PV),** also called the budget, is that portion of the approved total cost estimate planned to be spent on an activity during a given period. Table 7-3 shows an example of earned value calculations. Suppose a project included a summary activity of purchasing and installing a new Web server. Suppose further that, according to the plan, it would take one week and cost a total of $10,000 for the labor hours, hardware, and software
involved. The planned value (PV) for that activity that week is, therefore, $10,000.

2. **The actual cost (AC)** is the total direct and indirect costs incurred in accomplishing work on an activity during a given period. For example, suppose it actually took two weeks and cost $20,000 to purchase and install the new Web server. Assume that $15,000 of these actual costs were incurred during Week 1 and $5,000

was incurred during Week 2. These amounts are the actual cost (AC) for the activity each week.

3.**The earned value (EV)** is an estimate of the value of the physical work actually Completed.

The **rate of performance (RP)** is the ratio of actual work completed to the percentage of work planned to have been completed at any given time during the life of the project or activity.

TABLE 7-4 Earned value calculations for one activity after Week 1

Activity	Week 1
Earned Value (EV)	5,000
Planned Value (PV)	10,000
Actual cost (AC)	15,000
Cost variance (CV)	10,000
Schedule variance (SV)	5,000
Cost performance index (CPI)	33%
Schedule performance index (SPI)	50%

Earned value formulas

Term	Formula
Earned value (EV)	EV =PV to date RP
Cost variance (CV)	CV = EV AC
Schedule variance (SV)	SV EV PV
Cost performance index (CPI)	CPI EV/AC
Schedule performance index (SPI)	SPI EV/PV
Estimate at completion (EAC)	EAC BAC/CPI

Estimated time to complete Original time estimate/SPI

The earned value calculations in Table 7-4 are carried out as follows:

Cost variance (CV) is the earned value minus the actual cost. If cost variance is a negative number, it means that performing the work cost more than planned. If cost variance is a positive number, it means that performing the work cost less than planned.

Schedule variance (SV) is the earned value minus the planned value. A negative schedule variance means that it took longer than planned to perform the work, and a positive schedule variance means that it took less time than planned to perform the work.

The **cost performance index (CPI)** is the ratio of earned value to actual cost and can be used to estimate the projected cost of completing the project. If the cost performance index is equal to one, or 100 percent, then the planned and actual costs are equal the costs are exactly as budgeted. If the cost performance index is less than one or less than 100 percent, the project is over budget. If the cost performance index is greater than one or more than 100 percent, the project is under budget.

The **schedule performance index (SPI)** is the ratio of earned value to planned value and can be used to estimate the projected time to complete the project.

The cost performance index can be used to calculate the **estimate at completion**

(EAC) an estimate of what it will cost to complete the project based on performance to date. Similarly, the schedule performance index can be used to calculate an estimated time to complete the project.

You can graph earned value information to track project performance.

- Planned value (PV), the cumulative planned amounts for all activities by month. Note that the planned value line extends for the estimated length of the project and ends at the BAC point.
- Actual cost (AC), the cumulative actual amounts for all activities by month.
- Earned value (EV), the cumulative earned value amounts for all activities by month.
- **Budget at completion (BAC)**, the original total budget for the project, or $100,000 in this example. The BAC point is plotted on the chart at the original time estimate of 12 months.
- Estimate at completion (EAC), estimated to be $122,308, in this example. This number is calculated by taking the BAC, or $100,000 in this case, and dividing by the CPI, which, in this example, was 81.761 percent.

2)**Project Portfolio Management**

- Project portfolio management (PPM) is the centralized management of an organization's projects.
- While these projects may or may not be related, they are managed under one umbrella (called a portfolio) to oversee and manage any competing resources.
- Portfolio management in project management also involves the intake process of projects. This includes identifying potential projects, authorizing them, assigning project managers, and including them in the overall portfolio.

Why project portfolio management is important

According to the Project Management Institute, "portfolio management is a way to bridge the gap between strategy and implementation." The portfolio manager's job is to ensure the right projects are being done at the right time to maximize the company's investment.

Benefits of project portfolio management

By leveraging the benefits of project portfolio management, companies can plan out all the pieces of their project to get the best results. The main advantages of project portfolio management are:

Provides alignment between company objectives and projects

A PMI survey found that a lack of clearly defined goals is the first reason for project failure. Project portfolio management promotes transparent and open discussions amongst the team with a company-first attitude.

Takes the personal bias out of project planning

With PPM, there are no “pet projects.” Subjectivity in project planning is eliminated as PPM focuses on prioritizing projects based on their inherent risk, business goals, resource, and skill availability. Multiple qualitative and quantitative techniques such as ranking models and scoring methods are used to make project decisions.

Makes decision-making easier

Stakeholders may struggle to manage disputes that can arise with different project teams focusing on their priorities and vying for limited resources. By employing a standardized approach to decision-making, PPM subjectively evaluates the demand from competing project teams.

Helps prioritize projects

Good project portfolio managers focus their limited resources on their most valuable projects. When customer requests, regulatory requirements, or strategy demands arise, teams use project portfolio management to work on viable projects that help achieve organizational goals.

Focuses on the big picture

Sometimes project teams concentrate so much on execution that they miss the big picture. While chasing trends, they fail to achieve strategic goals and overwork their teams. When teams adopt project portfolio management, they prioritize and execute only value-delivering projects.

Builds governance and oversight into project management

Project portfolio management builds a natural governance model for all the projects of an organization. While project management looks at a single project, PPM provides a holistic overview of all projects. Project managers can create a contingency plan, employ data-driven techniques, and lead the company on the right path.

Best tools for project portfolio management

While project portfolio management (PPM) may sound simple, putting the PPM life cycle into action can be challenging. Along with a continuous project planning and evaluation process, you will need the right tools to keep all projects moving forward. Here are some widely used tools for project portfolio management:

Decision tree analysis

Decision tree analysis is a great tool for evaluating situations with multiple subjective factors. It illustrates the multiple ways to solve a problem along with their costs, outcomes, and consequences. Use this method to evaluate project outcomes, identify opportunities, manage costs, and solve problems.

Cost-benefit analysis

A cost-benefit analysis is a quantitative method for assessing the risk and reward of a project. Project success occurs when probable benefits are more and the overall cost is lesser.

Objectives matrix

In this method, the overall business strategy is divided into multiple objectives. These smaller objectives are assigned to different projects and scored for evaluation.

Scoring model

Appraising a project using the scoring model is a good way to balance the quantitative and qualitative factors. Some qualitative factors can be market competitiveness, business value, and customer confidence, while qualitative factors include operational cost, revenue value, and return on investment.

Weights and scores are assigned to these factors, and every project gets a total score. It provides a rational way of comparing projects based on their overall scores.

There can be a portfolio for information technology projects, for example, and portfolios for other types of projects. An organization can view project portfolio management as having

five levels, from simplest to most complex, as follows:

1. Put all your projects in one database.
2. Prioritize the projects in your database.
3. Divide your projects into two or three budgets based on type of investment, such as utilities or required systems to keep things running, incremental upgrades, and strategic investments.
4. Automate the repository.
5. Apply modern portfolio theory, including risk-return tools that map project risk on a curve.

COCOMO model

Some parametric models involve very simple heuristics or rules of thumb.

E.g. a large office automation project might use a ballpark figure of $10,000 per workstation based on a history of similar office automation projects developed during the same time period.

More complicated parametric models are usually computerized.

One popular parametric model is the Constructive Cost Model(COCOMO) which is used for estimating software development costs based on parameters such as the source lines of code or function points COCOMO was developed by Barry Boehm, a well-known expert in the field of software development and cost estimating.

Function points-are technology independent assessments of the functions involved in developing a system.

E.g. the number of inputs and outputs, the number of files maintained, the number of updates etc.COCOMO II is a newer, computerized version of Boehm’s model that allows to estimate the cost, effort, and schedule when planning a new software development activity.

Computerized tools, such as spreadsheets and project management software can also be used for cost estimation.

The necessary steps in this model are:

Get an initial estimate of the development effort from evaluation of thousands of delivered lines of source code (KDLOC).

1. Determine a set of 15 multiplying factors from various attributes of the project.
2. Calculate the effort estimate by multiplying the initial estimate with all the multiplying factors i.e., multiply the values in step1 and step2.

The initial estimate (also called nominal estimate) is determined by an equation of the form used in the static single variable models, using KDLOC as the measure of the size. To determine the initial effort Ei in person-months the equation used is of the type is shown below

$Ei=a*(KDLOC)b$

The value of the constant a and b are depends on the project type.

In COCOMO, projects are categorized into three types:

1. Organic
2. Semidetached
3. Embedded

1.Organic: A development project can be treated of the organic type, if the project deals with developing a well-understood application program, the size of the development team is reasonably small, and the team members are experienced in developing similar methods of projects. **Examples of this type of projects are**

simple business systems, simple inventory management systems, and data processing systems.

2. Semidetached: A development project can be treated with semidetached type if the development consists of a mixture of experienced and inexperienced staff. Team members may have finite experience in related systems but may be unfamiliar with some aspects of the order being developed. **Example of Semidetached system includes developing a new operating system (OS), a Database Management System (DBMS), and complex inventory management system.**

3. Embedded: A development project is treated to be of an embedded type, if the software being developed is strongly coupled to complex hardware, or if the stringent regulations on the operational method exist. **For Example:** ATM, Air Traffic control.

According to Boehm, software cost estimation should be done through three stages:

1. Basic Model
2. Intermediate Model
3. Detailed Model

1. Basic COCOMO Model: The basic COCOMO model provide an accurate size of the project parameters. The following expressions give the basic COCOMO estimation model:

Effort=a1*(KLOC) a2 PM

Tdev=b1*(efforts)b2 Months

Where

KLOC is the estimated size of the software product indicate in Kilo Lines of Code,

a1,a2,b1,b2 are constants for each group of software products,

Tdev is the estimated time to develop the software, expressed in months,

Effort is the total effort required to develop the software product, expressed in **person months (PMs)**.

Estimation of development effort

For the three classes of software products, the formulas for estimating the effort based on the code size are shown below:

Organic: Effort = 2.4(KLOC) 1.05 PM

Semi-detached: Effort = 3.0(KLOC) 1.12 PM

Embedded: Effort = 3.6(KLOC) 1.20 PM

Estimation of development time

For the three classes of software products, the formulas for estimating the development time based on the effort are given below:

Organic: Tdev = 2.5(Effort) 0.38 Months

Semi-detached: Tdev = 2.5(Effort) 0.35 Months

Embedded: Tdev = 2.5(Effort) 0.32 Months

Example1: Suppose a project was estimated to be 400 KLOC. Calculate the effort and development time for each of the three model i.e., organic, semi-detached & embedded.

Solution: The basic COCOMO equation takes the form:

Effort=a1*(KLOC) a2 PM

Tdev=b1*(efforts)b2 Months

Estimated Size of project= 400 KLOC

(i)Organic Mode

E = 2.4 * (400)1.05 = 1295.31 PM

D = 2.5 * (1295.31)0.38=38.07 PM

(ii)Semidetached Mode

E = 3.0 * (400)1.12=2462.79 PM

D = 2.5 * (2462.79)0.35=38.45 PM

(iii) Embedded Mode

E = 3.6 * (400)1.20 = 4772.81 PM

D = 2.5 * (4772.8)0.32 = 38 PM

2. Intermediate Model: The basic Cocomo model considers that the effort is only a function of the number of lines of code and some constants calculated according to the various software systems. The intermediate COCOMO model recognizes these facts and refines the initial estimates obtained through the basic COCOMO model by using a set of 15 cost drivers based on various attributes of software engineering.

Classification of Cost Drivers and their attributes:

(i) Product attributes -

- Required software reliability extent
- Size of the application database
- The complexity of the product

Hardware attributes -

- Run-time performance constraints
- Memory constraints
- The volatility of the virtual machine environment
- Required turnabout time

Personnel attributes -

- Analyst capability
- Software engineering capability
- Applications experience
- Virtual machine experience
- Programming language experience

Project attributes -

- Use of software tools
- Application of software engineering methods
- Required development schedule

ntermediate COCOMO equation:

E=ai (KLOC) bi*EAF

D=ci (E)di

3. Detailed COCOMO Model: Detailed COCOMO incorporates all qualities of the standard version with an assessment of the cost driver?s effect on each method of the software engineering process. The detailed model uses various effort multipliers for each cost driver property. In detailed cocomo, the whole software is differentiated into multiple modules, and then we apply COCOMO in various modules to estimate effort and then sum the effort.

The Six phases of detailed COCOMO are:

1. Planning and requirements
2. System structure
3. Complete structure
4. Module code and test
5. Integration and test
6. Cost Constructive model

The effort is determined as a function of program estimate, and a set of cost drivers are given according to every phase of the software lifecycle.

Advantages :

- It works on historical data and provides more accurate details.
- Easy to implement with various factors. One can easily understand how it works.

- Easy to estimate the total cost of the project.
- The drivers are very helpful to understand the impact of the different factors that affect project crises.

Disadvantages :

- It ignores the hardware issues as well as the personal turnover level.
- It ignores all the documentation and requirements.
- It mostly depends on time factors.
- It limits the accuracy of software costs.
- It oversimplifies the impact of safety or security aspects.
- It also ignores customer skills, cooperation, and knowledge.