International Journal of Management, IT & Engineering
(ISSN: 2249-0558)

# CONTENTS

# Editorial Board

# Editorial Advisory Board

## Title

# KEY PERFORMANCE METRICS FOR IT PROJECTS

## Author(s)

**Dr. S. K. Sudarsanam**

*Professor,*

*VIT Business School, VIT University,*

*Vandalur-kelambakkam Road,*

*Kezhakottaiyur, Chennai – 600 048*

**Abstract:**

Metrics are the numerical data which will help us to measure the effectiveness of Project Execution. In Information Technology industry there are no standard set of metrics for development and sustenance projects which everyone follows. This paper would try and provide a set of key metrics which can be followed by all development and maintenance and support (sustenance) projects. This study would also provide some areas of implementation challenges and how they can be overcome.

**Key words :** BMI (Backlog management index), ETC, FTE, Lines of Code, OTD (on time delivery), Response Time, Resolution Time, Service Level Agreement (SLA), Statement of work (SOW), Throughput, Utilization

## 1. Introduction:

Measurement plays a critical role in effective software development. It provides the scientific basis for software engineering to become a true engineering discipline. As the discipline has been progressing toward maturity, the importance of measurement has been gaining acceptance and recognition. (Lawrence E. Day)

Metrics are derived by comparing two disparate sets of data .Quality Metrics are used at all levels of an organization for meeting various objectives Paul Goodman defines software metrics as "The continuous application of measurement-based techniques to the software development process and its products to supply meaningful and timely management information, together with the use of those techniques to improve that process and its products". (Paul Goodman, 2004)

G. Gordon Schulmeyer defines a metric as "A quantitative measure of the degree to which a system, component or process possesses a given attribute". (Gordon, 1998)

IEEE Standard 1061 [8] lays out a methodology for developing metrics for software quality attributes. The standard defines an attribute as "a measurable physical or abstract property of an entity." A quality factor is a type of attribute, "a management-oriented attribute of

software that contributes to its quality." A metric is a measurement function, and a software quality metric is "a function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality."

Software metrics are numerical data related to software development. Metrics strongly support software project management activities. They relate to the four functions of management as follows:

1. Planning - Metrics serve as a basis of cost estimating, training planning, resource planning, scheduling, and budgeting.

2. Organizing - Size and schedule metrics influence a project's organization.

3. Controlling - Metrics are used to status and track software development activities for compliance to plans.

4. Improving - Metrics are used as a tool for process improvement and to identify where improvement efforts should be concentrated and measure the effects of process improvement efforts.

A metric quantifies a characteristic of a process or product. Metrics can be directly observable quantities or can be derived from one or more directly observable quantities. Examples of raw metrics include the number of source lines of code, number of documentation pages, number of staff-hours, number of tests, number of requirements, etc. Examples of derived metrics include source lines of code per staff-hour, defects per thousand lines of code, or a cost performance index (USC, 2001)

Software developers and testers develop and maintain many critical business applications across many verticals/industries. It is very important that they perform the primary task of meeting the customer requirements and their expectations. It is equally important for them to measure key performance parameters which would help them in knowing how they are performing the tasks and how they can improve and deliver more value to the customer.

We would identify a set of key performance metrics which would help the software developers, leads and managers to measure their productivity, efficiency and effectiveness.

## 2. Definitions:

| No. | Term | Definition |
|---|---|---|
| 1. | Response time | The time delay between the receipt of the ticket and the acknowledgement of the receipt of the ticket |
| 2. | Resolution time | Time delay between the receipt of the ticket and the resolution of the issue/problem |
| 3. | CUT | Coding and Unit Testing |
| 4. | FTE | Full time equivalent (resource used 100% in a project) |
| 5. | OTD | On time delivery |
| 6. | RS | Requirement Specification |
| 7. | LOC | Lines of Code |
| 8. | SOW | Statement of Work |
| 9 | ETC | Effort to complete |

## 3. Metrics Life Cycle:

### 3.1 Metrics Life Cycle:

- Identifying metrics
- Prioritizing metrics
- Classifying metrics
- Identifying data required for the metrics
- Communicating and Training about Metrics
- Capturing and verifying true data
- Analysis and process data as metrics
- Report generation.

**Insert Figure 1 here**

Identifying metrics: Each Project team need to identify the set of metrics which they need to capture. Either it would be defined in the Statement of Work or as per agreed SLA with the customer. If not, the project team needs to identify the metrics as per the industry standard

Prioritizing and classifying metrics: The project teams need to prioritize and classify the metrics identified. The metrics can be classified as at project level, Business unit level and organization level and also as weekly or monthly metrics.

Identifying Data required for the metrics: The team need to identify the data required for the metrics

Communicating and training about metrics: The Project Manager need to communicate to the team about the metrics and also need to train the team on data collection and data reporting for metrics.

Capturing and verifying true data: The data need to be captured and need to be verified for correctness. The data need to be the one retrieved from the repository or emails.

Analysis and process data as metrics: The data need to be analyzed and processed for metrics reporting. Also, the calculation of metrics need to be worked in suitable tools (either excel or project management tools)

Report generation: The project management tool or any other tool need to have the facility to generate metrics reporting for the duration required. Normally, teams report the project metrics on a weekly basis.

**3.2 The GQM Approach for implementing Metrics:**

Victor Basili, known as the father of the GQM (Goals, Questions, and Metrics) approach, presents a practical approach to measurement. This is one of the preferred approaches for deriving the appropriate metrics for a given project. The GQM approach facilitates data presentation by defining Goals, Questions and Metrics based on process and product goals. (Basili, 1994)

The different stages of this approach are:

1. Identifying goals, to allow focus on important issues
2. Defining questions, to make goals more specific
3. Derive appropriate metrics that are relevant to the goal
4. Metrics Analysis

- GQM is a systematic approach for integrating goals to the process
- The metrics relevant to process improvement can be effectively identified and tailored to the organization and its goals.
- Measurement provides the most appropriate information to ensure consistency and completeness in the quest for goal attainment.

## 4. Metric Analysis:

A periodic analysis of the processes and productivity enables a project, Business Unit or an organization constantly treads the path of growth. The purpose of metrics analysis is to monitor and enhance performance of the project, Business Unit and organization and the defined process.

Metrics analysis leads to:

- Optimizing the utilization of resources
- Evaluation of performance to customer expectation
- Benchmarking with competition
- Innovation

The benefits of metrics analysis at all levels are:

- Increased customer satisfaction
- Increased productivity
- Repeat and continual business for the organization
- Improvements in competency/skill levels

- Improvement in product performance and process performance
- Improvement in resource utilization
- Better estimates on time, effort, cost, resources, defects, risks, etc

Some important criteria for metrics analysis are:

- **Life-cycle-based analysis**: Projects data is segregated based on life-cycles like Development, Maintenance, Service, Conversion and Testing. Data is analyzed based on this segregation and compared against the organization goals
- **Technology-based analysis**: Project data is classified based on technology areas too. Technology areas such as Web-based technology, Mainframe
- **Language-based analysis**: Projects data is classified based on the language in which the projects are being developed. For example, development languages such as Java, C, ASP, VB

## 5. Advantages of Metrics:

Implementation of software metrics provides lot of advantages in any levels of organization. Software Metrics can aid in improving organizations Processes by

- ➢ Providing insight and early visibility into the "real" status of the efforts
- ➢ Aid in making assessments as to whether progress, productivity and quality goals are being met.

### 5.1 We need Metrics :

To survive a software development organization must make accurate cost estimates and improve productivity, quality and manage critical risks carefully

If you do not know where you are now you certainly won't know where you will be in the future

To achieve accurate measurements of productivity and quality, organizations require metrics collections and analysis

Metrics help to achieve project specific goals, improve existing processes and determine corrective and preventive measures.

## 6. Metrics in Projects:

Key Metrics in Sustenance (Maintenance and Support) Projects:

| No | Metrics | |
|---|---|---|
| 1 | Response time - % (for Severity1, 2,3 and 4) | (No. of tickets responded on time/Total no. of tickets responded during the reporting week)*100 |
| 2. | Resolution time - % ( on severity I =1,2,3,4) | (No. of tickets resolved on time/Total no. of tickets resolved during the reporting week)*100 |
| 3. | Application Availability - % | (No. of hours application is available in the week/(7*24))*100 |
| 4. | Backlog Management | (No. of incidents delivered during the week/No. of incidents received during the week)*100 |
| 5. | Service Productivity | No. of incidents delivered with or without code fix/Total effort |
| 6. | Average Resolution time (on Severity) | Total elapsed time of resolved incidents/No. of incidents resolved |
| 7. | Age of open incidents/bugs | Total elapsed time/No. of open bugs |
| 8. | Resolution efficiency | (No. of incidents resolved during the week/(No. of incidents received during the week + No. of open incidents in the beginning of the week) )* 100 |
| 9. | Resolution Rate | Total no. of incidents resolved during the week/Total FTE in the project team (for Sev 1, 2,3 and 4) |

| | | |
|---|---|---|
| 10. | FTR Rate | Tickets resolved first time during the week/Tickets resolved during the week |

Key metrics for Development Projects:

| No. | Metrics | Definition |
|---|---|---|
| 1. | Effort variance | It is calculated as ((Cumulative Actual Effort + Effort to Complete – Cumulative Planned Effort)/Cumulative Planned Effort))*100 |
| 2. | Schedule Variance | (Actual End date – Planned End date)*100/(Planned Duration) |
| 3. | Requirement Volatility | Its calculated as (Requirements added + Modified + Deleted)*100/(No. of Original requirements) |
| 4. | CUT Productivity | Ratio of code Size to CUT Effort |
| 5. | Overall Productivity | Ratio of product size to total effort |
| 6. | Code Review Rate | LOC reviewed / elapsed time |
| 7. | Phase Containment Efficiency | (Errors found in RS Phase/(Errors found in RS phase + Defects found in RS in subsequent phased)*100 |
| 8. | Code Review Rate | Rate of code reviewed per hour (LOC reviewed/Elapsed time) |
| 9. | Milestone Misses | No. of times intermediate milestones were missed |
| 10.. | OTD Misses | No. of times the final delivery is missed (not intermediate milestones) |

| | | |
|---|---|---|
| 11. | FTE Gap | (Total FTE in project – FTE as per SOW) |
| 12. | Defect Density | (Errors + Defects)/Total Size |
| 13 | Post Delivery Defect Rate | Ratio of number of defects reported by the customer during acceptance testing and warranty to actual size of product delivered.<br>(Acceptance + Warranty defects)/ Total Size |

## 7. Sample Metric reports:

Throughput Report: Insert Figure 2 here

Resolution Rate and Trend: Insert Figure 3 here

SLA adherence report and Severity analysis: Insert Figure 4 here

ETC Report: Insert Figure 5 here

OTD (On time Delivery): Insert Figure 6 here

Utilization: Insert Figure 7 and 8 here

## 8. Challenges in Metrics Implementation :

Some of the key challenges in Metrics implementation in IT projects are:

a. Project team Buy in: Need to impress the team that the metrics they capture would help improve their day to day work and improve their productivity and also could serve to show case their efforts and accomplishments in a better manner. Most of the time, the project team members and even some project managers feel that the data collection, analysis and reporting is an overhead for them which they would like to avoid. Also, there is a feeling that this is not a critical task.

The senior management and quality managers need to work closely with project team and impress upon them the importance of metrics collection, analysis and reporting for the projects, business unit and organization as a whole.

b. Data collection: Many of the project teams do not have problem management tool or any of the service management repositories. The data collection need to be done manually using excel sheets and would require lot of manual editing and efforts. The customers can be impressed upon the need to have such repository which would make the data collection and metrics reporting a pleasant job.

c. The stop clock facility in problem management tools will help report the correct response and resolution times in cases where issues were on hold awaiting feedback or answers from the customer. Otherwise, the resolution time needs to be calculated manually which would take significant amount of time of the project manager.

d. SLA awareness and interpretation: The project teams need to be aware of the SLAs and proper training need to be provided to teams to ensure that they know the SLAs which they need to comply with and also the interpretation of metrics/SLAs. Many a times the way the project teams interpret an SLA would be different from the way it is originally defined. So, there would be errors in the SLA reports due significant changes in the SLA metric interpretation.

e. Utilization metric reporting: The project teams tend to include idle time as well in the utilization metric which will not give the right picture on utilization. If utilization is low in one area, the team could be used in other area or can work on some more applications to ensure 100% utilization. There is a feeling in project managers that this will affect their project costs. The value provided to customer in terms of improved productivity will be the key and the team can take up additional work from the customer instead of billing the idle hours to the customer.

f. Threshold values for metric not available (in SLA) .In such cases the project manager need to work with the senior management in the organization or with customer managers to define the threshold for all the key metrics reported in the project. This would help measure the metric performance

## 9. Conclusion:

Metrics plays vital role in maintaining quality of product and process in any organization. The challenge is in identifying the right metrics, analyzing and implementing them. Once its implemented, the managers can monitor and track the key performance parameters of projects and take proactive steps to address the non-compliance areas and improve the customer service levels.

## 10. References:


- **Paul Goodman,** SOFTWARE METRICS: Best Practices for Successful IT Management, 2004
- **USC, (2001).** Software Metrics Guide. University of Southern California. http://sunset.usc.edu/classes/cs577b_2001/metricsguide/ metrics.html
- **Gordon Schulmeyer**. Handbook of Software Quality Assurance, Prentice Hall 3rd Edition, 1998
- **Basili, Victor R., Gianluigi Caldiera, and H. Dieter Rombach**, "The Goal Question Metric Approach", in Encyclopedia of Software Engineering, Wiley, 1994.

Figure 1:

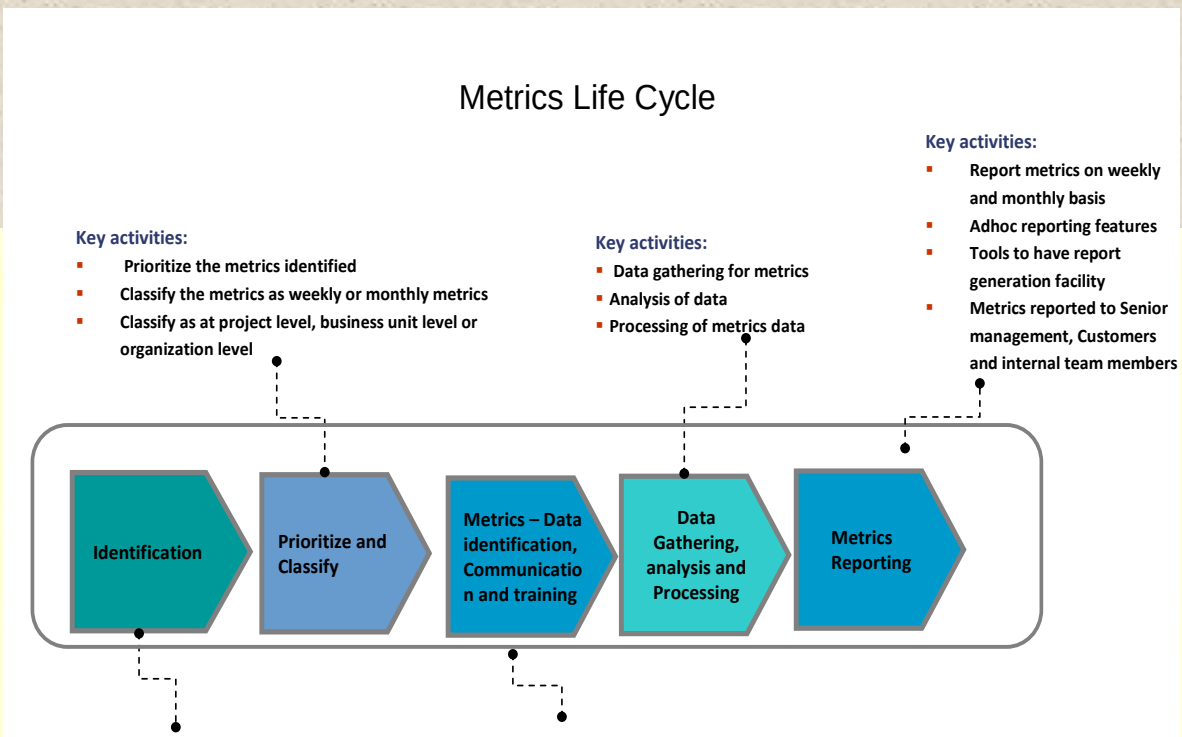
Metrics Life Cycle
Key activities:
Prioritize the metrics identified
Classify the metrics as weekly or monthly metrics
Classify as at project level, business unit level or organization level
Key activities:
Data gathering for metrics
Analysis of data
Processing of metrics data
Key activities:
Report metrics on weekly and monthly basis
Adhoc reporting features
Tools to have report generation facility
Metrics reported to Senior management, Customers and internal team members
Identification
Prioritize and Classify
Metrics – Data identification, Communicatio n and training
Data Gathering, analysis and Processing
Metrics Reporting
Key activities:
Identify the set of metrics to be captured
As per Project Statement of work or agreed service levels or as per industry standards
Key activities:
Identify data for the metrics
Communicate the proposed metrics to team
Train the team on data identification, analysis and reporting

Figure 2:

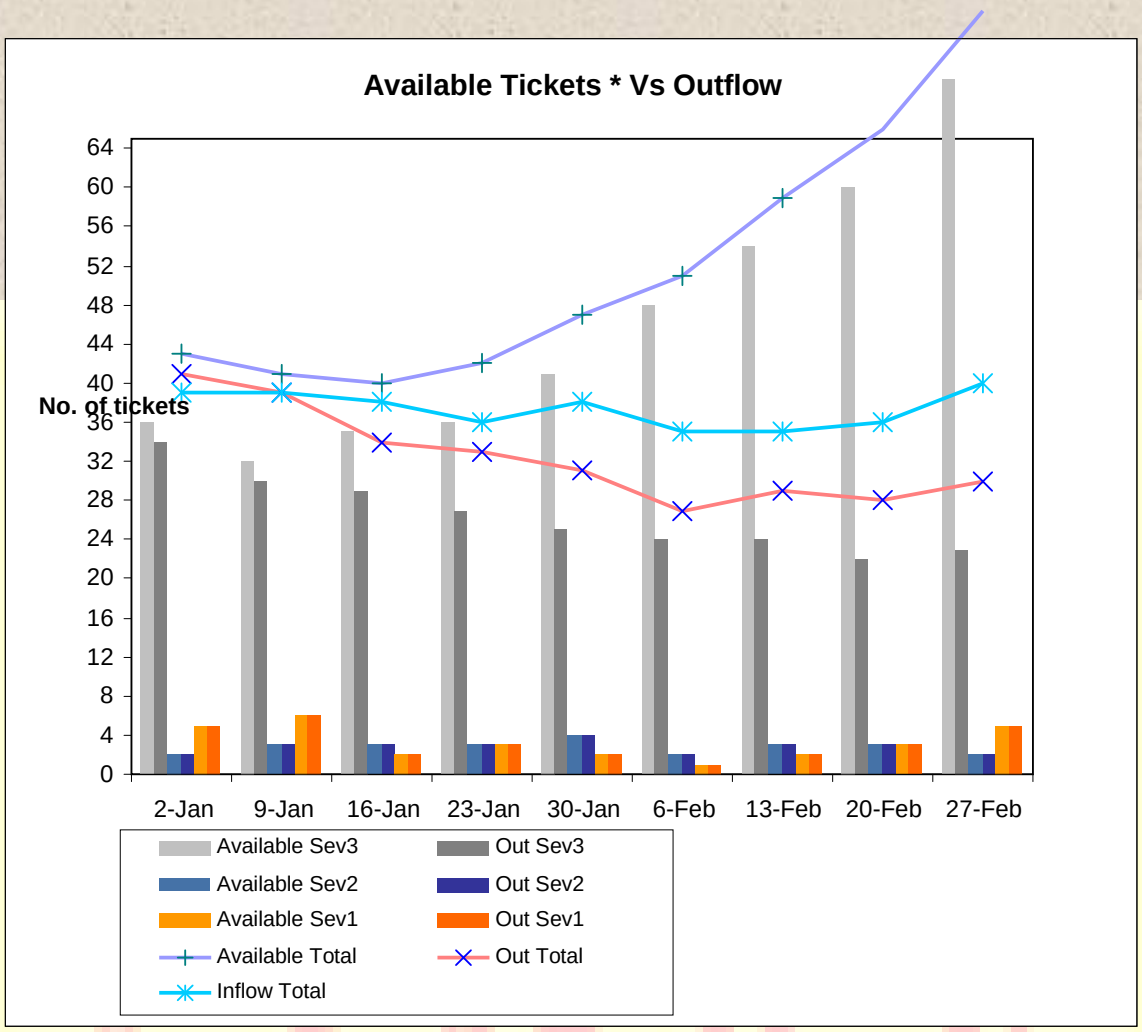
Available Tickets * Vs Outflow
No. of tickets
64
60
56
52
48
44
40
36
32
28
24
20
16
12
8
4
0
2-Jan
9-Jan
16-Jan
23-Jan
30-Jan
6-Feb
13-Feb
20-Feb
27-Feb
Available Sev3
Out Sev3
Available Sev2
Out Sev2
Available Sev1
Out Sev1
Available Total
Out Total
Inflow Total

Figure 3:

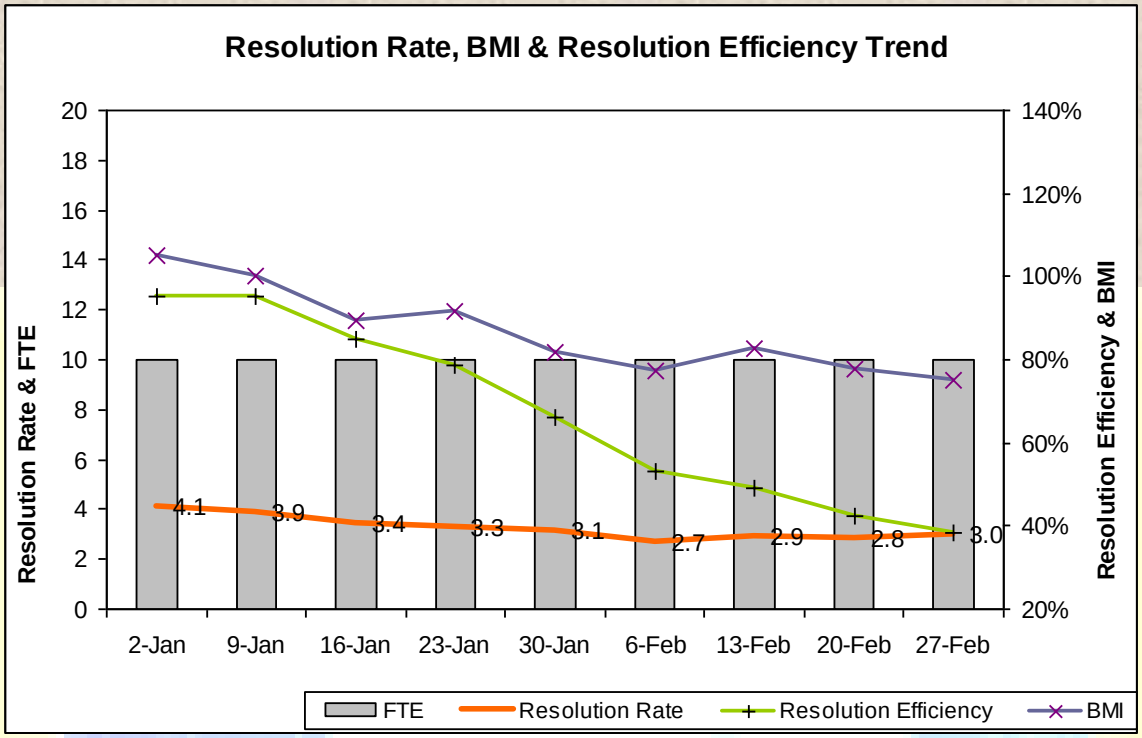


Figure 4:

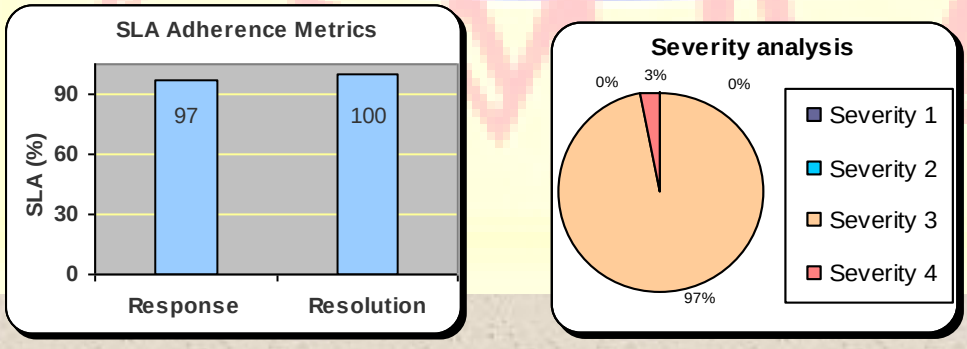

Figure 5: ETC report:

| SOW No. | Project Name | GOC | Key Competency | Estimated Effort (Hrs) (A) | Effort till date (Hrs) (B) | Effort spent last week (Hrs) | Effort to complete (Hrs) (C) | Effort variance % (B+C-A)/A |
|---|---|---|---|---|---|---|---|---|
| | Project X | | Base Technology | 3200 | 1920 | 200 | 1320 | 1% |
| | Project Y | | Base Technology | 19200 | 9280 | 800 | 9920 | 0% |
| | | | | | | | | |

Figure 6: On Time Delivery:

| Wk Ending | Vendor | GOC | SOW No. | Project Name | Key Competency | Project milestones | | | % OTD till date (B/A) | Final Milestone Date | | | Reason for OTD miss and other Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Total milestones | Planned till date (A) | On time (B) | | Contract | Actual | Final OTD | |
| 16-Mar | Vendor 1 | | | Project X | | 9 | 3 | 3 | 100.00% | 20-Apr-09 | | | |
| 16-Mar | Vendor 1 | | | Project Y | | 6 | 4 | 1 | 25.00% | 20-May-09 | | | |
| 16-Mar | Vendor 1 | | | Project Z | | 5 | 5 | 5 | 100.00% | 13-Mar-09 | 13-Mar-09 | Yes | |
| 16-Mar | Vendor 1 | | | Project M | | 5 | 5 | 4 | 80.00% | 13-Mar-09 | 15-Mar-09 | No | |

Figure 7: Utilization

| Vendor | SOW No. | Engagement Type | Project/Service Name | GOC | Key Competency | FTE | Hours Available during last week (A) | Hours Worked during last week (B) | Idle Time during last week (Hrs) | Utilization % (B/A) | Cumulative Utilization % | Utilization Forecast % Week 1 | Utilization Forecast % Week 2 | Utilization Forecast % Week 3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vendor 1 | | Project | Project X | | | 17 | 2640 | 2480 | 20 | 94% | 92% | 95% | 96% | 94% |
| Vendor 1 | | Service | Service Y | | | 20.5 | 3280 | 3200 | 16 | 98% | 96% | 98% | 97% | 98% |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |

Figure 8: