Measuring the Effectiveness of Service Assurance Solutions

Different Service Assurance solutions cater to different operational processes in a service provider’s environment. Most service providers measure the effectiveness of their operations in terms of the availability of their OSS or Service Assurance solutions, where availability is equated to operational effectiveness. However, availability can only indicate the presence of an entity for a defined purpose, it cannot measure the effectiveness of the entity in fulfilling the defined purpose.

Hence, there is a need for a standard mechanism to monitor and report the effectiveness of Service Assurance solutions against their defined purpose.

This paper intends to introduce the concept of measuring and managing the effectiveness of Service Assurance solutions. The objectives of this paper are:

1. To define standard metrics to measure the capabilities of a Service Assurance product
2. To define standard performance (or) effectiveness metrics/ reports for the Operational Support Systems (OSS) Service Assurance solutions
About the Author

S Jayavardhan

Jayavardhan has been working with TCS since 2002. Jay holds a Bachelors degree in computer science and M.Tech in computer science from Central University, Hyderabad. Jay has several years of experience in the telecom domain involving in development and support for TCS in-house asset in Network Management System and also associated with leading network operators and equipment vendors for service assurance. Jay is currently associated with TCS Service Assurance COE and working on areas like Hosted OSS/BSS, Device Assurance and Service Assurance for IPTV. He is a certified IBM Netcool Deployment Professional, ITIL foundation certified and a certified Java professional. His other interests include IPTV, LTE and Mobile applications.
# Table of Contents

1. Introduction ........................................ 4  
2. Proposed Approach ............................... 5  
3. Conclusion ......................................... 9  
4. Appendix A –  
   Sample metrics proposed for Fault Management .... 10  
5. Appendix B –  
   Javadoc instance for defined interfaces .......... 10  
6. References ........................................ 10
Introduction

Just as any organized body requires a Manager to monitor and manage the underlying organizational entities, the Telecom ecosystem requires a Manager (or) manager of managers to monitor and control the underlying not-service-aware network elements. With the advances in the Telecom and Information technologies, the word abstraction has become the mainstay. Network abstraction is the latest mantra, for all service providers. The same underlying network is used by the service provider to provide multiple services. For example, Triple Play and Quad Play (Video, Voice, Data and mobility) services are provisioned to customers on the same underlying network. Due to this abstraction at the service level, traditional Service Assurance solutions no longer cater to the complex service monitoring needs of service providers. The process of Telecom Assurance Management has shifted its focus from Resource Assurance to Service Assurance. This shift in focus has forced many traditional vendors and new product vendors to come up with service aware assurance solutions to manage services, in addition to managing the resources.

Standard metrics for each functional area under Service Assurance can be identified and the values for these can be published by the product vendors. This will provide transparency regarding the product capabilities, and will help the service providers make an informed decision while selecting a Service Assurance solution. The standard metrics will also enable measuring a product’s implementation and solution effectiveness by providing overall effectiveness values, instead of quarterly status reports with metrics like uptime and availability which are captured manually.

The following functional areas under Service Assurance have been considered to arrive at the definition of metrics and interfaces:

- Fault management
- Performance management
- Service quality management
- Trouble ticketing
- SLA management
**Proposed Approach**

The approach comprises three phases, as depicted in the figure below (Figure 1):

The three phases are described below:

- **Identification of metrics in each functional area**
- **Refining the interface specification & metrics data model**
- **Definition of java methods & interfaces**

![Figure 1 Service Assurance solution measurement approach](image)

**Identification and Definition of Metrics**

The metrics to be identified should be SMART:

- **Specific** to the domain and category
- **Measurable** to enable objective extracting of value
- **Achievable** to derive values for the defined metric from the solution
- **Relevant** to measuring the effectiveness of solution in a particular functional area and category
- **Time based** to provide values for configuration of a solution, and on continuous basis, to capture the performance and health of the solution

Metrics are classified into three categories as shown in Figure 2 below. These metrics help obtain an understanding of the effectiveness of the Service Assurance solution.

1. Solution configuration metrics
2. Solution health metrics
3. Solution performance metrics
Solution Configuration Metrics: These metrics are the baseline configuration parameters. These parameters provide the capabilities or features of the Service Assurance solution. These values are the characteristics of a solution (a specific solution and specific version) and will not change during the life time of this solution except by changes to the version, or by upgrades. These are usually captured once for every version of the solution. With every new release or patch updates, the values can be captured again to identify the changes in the solution configuration. Sample configuration metrics for a sample Service Assurance solution (Fault Management Solution) include Technologies Supported, Maximum Number of Network Elements Supported, and Platforms Supported.

Solution Health Metrics: The metrics identified under this category represent the status of the components of the solution and thereby the status of the solution. The values for these metrics are the most objective among the three categories of metrics and determine the health of the solution. Sample metrics for a sample Service Assurance solution (Fault Management Solution) include Uptime, Availability, and Component Status.

Solution performance metrics: These metrics represent the dynamic parameters of the solution components. The values for the identified metrics determine the performance of the solution. The values for these metrics change with time and are used to measure the effectiveness of the solution continuously. Sample metrics for a sample Service Assurance solution (Fault Management Solution) include Total number of alarms and Number of dropped notifications.

Appendix A contains a sample set of performance metrics for Fault Management solution and the excel sheet in Appendix C contains the list of metrics identified for each functional area. Each of these functional areas is implemented as a solution in the Service Assurance domain. The values for the defined metrics will give the effectiveness of the solution.
The overall effectiveness of the solution can be measured based on the values for the above defined categories of metrics, as depicted in Figure 3.

The value for configuration metric in the graph will be a weighted average of the values for different metrics under Service Assurance solution configuration category. Similarly, cumulative values can be obtained for performance and health/fault metrics and overall Service Assurance solution effectiveness can be measured as depicted in the graph. In the graph below the Service Assurance solution is assumed to be functionally complete with respect to its capabilities, but is average with respect to performance and health.

![Figure 3: Solution effectiveness index (Configuration=5, Performance=4, Fault=2)](image)

**Definition of java methods & interfaces**

As depicted in Figure 1, the next step, after identification of metrics in each functional area, is defining the java interface specifications to capture values for these metrics. Along the lines of OSS/J (Reference 1) an interface specification can be defined in java programming language. The defined specification contains methods which can be used to capture values for the metrics which are identified and defined in the previous chapter. Since the metrics were identified in the functional areas of the FM, PM, SQM, SLAM and TT this chapter also deals with definition of interfaces for these functional areas.

- Metrics for Fault Management interface specification
- Metrics for Performance Management interface specification
- Metrics for Trouble Ticketing interface specification
- Metrics for Service Quality Management interface specification
- Metrics for Service Level Agreement Management interface specification
The framework is packaged hierarchically as shown below:

![Diagram of the package hierarchy](image)

**Figure 4 Solution effectiveness self management java interface hierarchy**

As noted initially, service providers are required to implement these specifications. The implementation can obtain values for each metric from these solutions in a specific way depending on the platform, application or database of the product. These interfaces are organized under the package `com.tcs.selfmgmt.telcoassurance` as shown in Figure 4. Each functional area under Service Assurance is defined as an interface and the metrics to be collected are defined as methods in the interfaces. Refer to the example below for the package name, sample interface and a sample method in the interface specification.

Example:

```java
Package com.tcs.selfmgmt.telcoassurance.faultmgmt;

Interface FaultMgmtSelfMgmtKPIInterface {
    Int GetNumberOfAlarms();
}
```

Appendix B contains the same java interfaces specification and the zip file in Appendix C contains the complete set of java interfaces and javadoc for the defined interfaces. The interfaces contain methods which can be used for collection of metrics defined in section on “Identification & definition of metrics”. Note: Extract the contents of the zip file to view the interfaces as well as javadoc for the interfaces.
Implementation of defined interfaces to measure Service Assurance solution effectiveness

The Java interfaces are required to be implemented by the product vendors and/or service providers. As depicted in Figure 5, they are required to implement the defined interfaces and code the logic to extract the values for the defined metrics. These values will enable the Service Assurance solution to self-monitor, thereby helping to measure its effectiveness.

The self monitoring of the Service Assurance solution can be continuous or intermittent. The monitoring is carried by self management probes which are Java classes implementing the interfaces defined in the earlier section Definition of Java methods & interfaces. There are a few methods in the interfaces which return static values for metrics. For example: the configuration metrics. These metrics can be queried in real time from the Service Assurance solution. However, to measure the effectiveness of the Service Assurance solution, values are to be extracted continuously for performance metrics and health metrics. The values for these metrics provide information about the effectiveness of a Service Assurance solution through parameters like availability, uptime and Number Of Dropped Notifications.

**Conclusion**

The emergence of trends like service/business management and Service Provider IT (SPIT) is resulting in more choices with respect to managers, which further reinforces the need for a solution effectiveness framework. These individual solution effectiveness indicators with respect to each solution area can lead to overall operational effectiveness of the service provider. These interfaces will only assist the service provider in measuring his operational effectiveness and not to set any benchmarks in the industry. The concept has the potential of application in other domains as well, with the definition of new interfaces.
## Appendix A – Sample metrics proposed for Fault Management

<table>
<thead>
<tr>
<th>S.No</th>
<th>Operational Metric</th>
<th>Automated/ Manual</th>
<th>Effectiveness Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Total number of alarms</td>
<td>Automated</td>
<td>% Reduction in alarms</td>
</tr>
<tr>
<td>2</td>
<td>Number of Network Elements monitored by Fault Management</td>
<td>Automated</td>
<td>% improvement in coverage of network</td>
</tr>
<tr>
<td>3</td>
<td>% of faults covered by Fault Management</td>
<td>Manual</td>
<td>% improvement in alarm and actual fault coverage</td>
</tr>
<tr>
<td>4</td>
<td>Number of dropped notifications</td>
<td>Manual</td>
<td>% Faults Processed</td>
</tr>
<tr>
<td>5</td>
<td>Number of invalid alarms</td>
<td>Manual</td>
<td>% improvement in accurate notifications</td>
</tr>
<tr>
<td>6</td>
<td>Number of alarms cleared per day</td>
<td>Automated</td>
<td>% improvement in number of cleared alarms</td>
</tr>
<tr>
<td>7</td>
<td>Mean time to clear alarm</td>
<td>Manual</td>
<td>% reduction in mean time to clear alarm</td>
</tr>
<tr>
<td>8</td>
<td>Number of alarms acknowledged per day</td>
<td>Automated</td>
<td>% improvement in number of acknowledged alarms</td>
</tr>
<tr>
<td>9</td>
<td>Mean time to acknowledge alarm</td>
<td>Manual</td>
<td>% reduction in mean time to acknowledge alarm</td>
</tr>
</tbody>
</table>

## Appendix B – Javadoc instance for defined interfaces

```java
public interface FaultMgmtKPIInterface {
    public int getNumber OFAlarmClearPerDay();
    public int getNumber OFInvalidAlarms();
    public int getNumber OFDroppedNotifications();
    public int getNumber OFFaultsProcessed();
}
```

This is the Fault Management package. It is the sub package of telcoassurance in service_assurance project. This management looks after and monitors the faults in the system. It has an interface which contains various methods for fault management.

**Author:**

TCS ASSURANCE TEAM

**Method Summary**

- `getNumber OFAlarmClearPerDay()`: This method gives the number of alarms cleared per day.
- `getNumber OFInvalidAlarms()`: This method gives the number of invalid alarms.
- `getNumber OFDroppedNotifications()`: This method gives the number of dropped notifications.
- `getNumber OFFaultsProcessed()`: This method gives the number of faults processed.

### References

About TCS Telecom Industry Solution Unit

TCS’ Telecom Business Unit is the second largest vertical contributing higher percentage to the overall TCS revenues. With a dedicated pool of professionals and an accumulated experience and ongoing associations with world-class Telecom service providers and equipment manufacturers, TCS has acquired unparalleled understanding of the Telecom domain.

TCS helps wireline, wireless, broadband, and cable service providers redefine their markets with innovative solutions that help them become more agile, reduce fixed operations costs, and introduce next generation services. TCS sets customers apart from their competitors with instant access to industry solutions, best-in-breed technology, assets, and frameworks.

Contact
For more information about TCS’ Telecom Solutions, contact global.telecom@tcs.com

Subscribe to TCS White Papers
Feedburner: http://feeds2.feedburner.com/tcswhitepapers

About Tata Consultancy Services (TCS)
Tata Consultancy Services is an IT services, consulting and business solutions organization that delivers real results to global business, ensuring a level of certainty no other firm can match. TCS offers a consulting-led, integrated portfolio of IT and IT-enabled infrastructure, engineering and assurance services. This is delivered through its unique Global Network Delivery Model™, recognized as the benchmark of excellence in software development. A part of the Tata Group, India’s largest industrial conglomerate, TCS has a global footprint and is listed on the National Stock Exchange and Bombay Stock Exchange in India.

For more information, visit us at www.tcs.com