SAS1520-2015

Operations Integration, Audits, and Performance Analysis: Getting the Most Out of SAS® Environment Manager

Bob Bonham and Bryan Ellington, SAS Institute Inc.

ABSTRACT

The SAS Environment Manager Service Architecture expands on the core monitoring capabilities of SAS Environment Manager delivered in SAS® 9.4. Multiple sources of data available in the SAS Environment Manager Data Mart - traditional operational performance metrics, events, and ARM, audit, and access logs - together with built-in and custom reports put powerful capabilities in the hands of IT operations. This paper introduces the concept of service-oriented event identification and discusses how to use the new architecture and tools effectively as well as the wealth of data available in the SAS Environment Manager Data Mart. In addition, extensions for importing new data, writing custom reports, instrumenting batch SAS jobs, and leveraging and extending auditing capabilities will be explored.

INTRODUCTION

Introduced with the SAS 9.4 Intelligence Platform release, SAS Environment Manager provides a framework for SAS administrators and IT operations staff to monitor the performance, health, and operation of their SAS deployment. Designed by SAS for SAS, SAS Environment Manager offers an integrated, operationally focused collection of administration and monitoring tools. As a documented part of the ITIL best practices, the establishment of customer service levels, resource consumption baselines and the centralization and collection of events are critical to maintaining the operational health and stability of an application or solution.

Enterprise class software deployments must be designed to meet an agreed level of service quality, which takes the form of a service level agreement (SLA) that is based on established key performance indicators (KPIs). To avoid SLA violations, it is especially important to include a tightly integrated collection of instruments that are designed to identity faults and service interruptions and to help the IT operations staff identify the root cause of faults quickly, regardless of whether trouble reports are received from customers or from the provider's own service surveillance systems.

SLAs are established to define and clarify the expectation of the service consumer. Key performance indicators are measurable values used to help assess, quantify, and track organizational objectives. Although simple to articulate, SLAs and KPIs can be difficult to instrument in practice, especially with distributed applications and solutions. The following sections outline a service-oriented, event-based architecture that is coupled to application instrumentation, and will aid in the creation of meaningful KPIs. With this objective and understanding, we highlight the broad range of event instruments now available with SAS Environment Manager.

FROM DEVICE TO SERVICE

Maintaining application service levels requires a shift from device-oriented to service-oriented methodologies. Traditional monitoring of system resources (such as CPU, disk, or memory) is generally not sufficient to successfully measure most KPIs. This type of monitoring also does not have the capability to identify and then triage service consumer problems in complex architectures.

Recent implementation strategies have been driven by the popularization of microservices, where complex applications are decomposed into a collection of small, loosely coupled, scalable services that communicate with each other using standard protocols (usually HTTP or HTTPS). There are many benefits to a microservices-based architecture, but new challenges are also introduced. The improved flexibility, capacity, and availability obtained by decoupling, pooling, and scaling small services adds complexity through the use of proxies, service registries, and parallelization of service flows. These complexities require the establishment of a common event correlator mechanism and identification markers within to the inter-service communication flow.

The benefits of KPIs that are implemented with and coupled to service-oriented event architecture are:

- **Reduced resolution time:** The time between the first symptoms of a service problem and the fault repair is reduced, regardless of whether the problem is detected by a user, an alert, or a third-party tool. This benefit is especially important for enterprise class applications that have defined SLAs.
- **Reduced duplication of effort:** If several trouble reports are symptoms of the same fault, fault processing can be performed only once for the root fault, rather than several times for each symptom. If the fault has been repaired, all affected customers can be informed automatically.
- **Impact analysis:** If a fault occurs in a resource, the fault's influence on associated services and affected consumers can be determined. This analysis can be performed for both the short term (during a current a resource failure) and the long term (network optimization).

WHAT IS AN EVENT?

An event is an occurrence or happening that is:

- significant (falls within the scope of the service in question)
- instantaneous (takes place at a specific point in time)
- Boolean (it either has occurred or not)

A simple event can occur during a change of state. The event might be created when a measurement exceeds a predefined threshold or when the measurement is identified as an outlier. Outliers are detected using standard statistical procedures, which are particularly appropriate for data with normal distributions. Examples of simple events include:

- An application availability or health check fails
- A storage device is nearing full capacity
- A user was locked out due to multiple failed login attempts

Composite events, also called complex events, are events that are derived from multiple independent events and can represent more complex situations and activities. Composite events can be based on criteria such as an inference drawn from multiple discrete events or a simple aggregated sum of individual metrics that are accumulated from separate service-related servers.

Examples of composite events include:

- The number of invalid login attempt events that is higher than a normal threshold
- Slow application response time

- Slow microservice transaction
- High TCP/IP error counts or dropped connections

APPLICATION RESPONSE MEASUREMENT (ARM)

ARM is an Open Group standard for monitoring complex transaction flows across loosely coupled service-oriented architectures that are common in enterprise software. The ARM standard is vendor-neutral, has multiple API bindings, and is targeted toward managing the performance of distributed applications.

Applications that are ARM-instrumented produce correlator records at code boundaries that application developer considered to be interesting. An ARM record identifies the application, the transaction, and the user, and it provides the status of each transaction when it completes. Performing sequence performance analysis of these transactions helps to construct KPIs from the user's perspective.

Although they have traditionally been called ARM records or ARM transactions, we can consider each correlator record as a primitive ARM event. Composite ARM events can be constructed from the collective set of ARM transactions associated with a job or other unit of work.

BRING IT TOGETHER

Coupling an information store or data mart (composed of an inventory of resources, primitive events, and application response measure events) with the capability to construct composite events provides the means to perform these tasks:

- Recognize resource constraints, bottlenecks, and hotspots
- Develop meaningful KPIs in terms of the consumer or business
- Understand capacity constraints
- Triage multi-node, parallelized microservices

These, in turn, give the analyst or systems administrator the instruments needed to answer key questions about the heath and operational integrity of an application. Some of these questions are:

What is the application response time?

The only performance criteria more important than satisfactory response times is determining whether a transaction is hung or is failing. In fact, from the user's perspective, the only two things that really matter are whether an application is working and how responsive it is. Response times are important elements in service level agreements and are the building blocks for KPIs.

Who uses the application, and how many of which transactions are used?

What events are generated, which service are involved?

How can I tune the application?

Application-oriented workload instrumentation provides an accurate understanding of the "who", "when", and "what" of the workload and is an essential tool in effective resource management and capacity planning.

Where are the bottlenecks?

If response times are unsatisfactory, where is the time being spent?

Would a faster network have an impact?

Measuring application transactions, event requests, and the main transactions visible to the user enables you to quantify the application run time and gain insight to the root causes.

The application or environment can be tuned to perform better by understanding which child transactions and event requests contribute the most to the total response time – from there, administrators can take steps to minimize the execution paths that are taking the longest. A faster communications link might alleviate a communication bottleneck and be warranted if there is a strong correlation to the overall application response time.

The following sections examine the capabilities of SAS Environment Manager as a necessary first step in the development of a service-oriented event identification and notification system.

SAS ENVIRONMENT MANAGER DATA MART

In computing, extract, transform, and load (ETL) refers to a process of taking data stored in one location, transforming it for quality or normalization purposes and loading it into a data store. Although simple in concept, effective use of the data directly correlates both to the variety of information sources and to the quantity of the transforms and is the principal reason for the development of the SAS Environment Manager Data Mart.

The SAS Environment Manager Data Mart is an information store, using data collected and correlated from three distinct sources:

- **Agent collected metrics (ACM):** Metric data that is continuously collected from the hosts and software components of a deployment
- **Events and alerts:** Data from events (which can occur at any time and can originate from many sources) and alerts (which are generally more urgent and are triggered by pre-defined conditions, such as metric thresholds, resource availability, or event messages)
- **Log records:** Data generated from nightly batch processing of log files (including application, audit, ARM, and system log files)

AGENT COLLECTED METRICS (ACM)

The SAS Environment Manager agent runs on every host within a SAS deployment. Its function is to detect and monitor the host, processes, logs, and system resources and transmit that information to the SAS Environment Manager Server. The metrics that are collected vary depending on the resources that the agent has discovered or has determined are running in the environment. Monitored resources range from high-level items such as the operating environment to low level and very specific items such as the garbage collection associated with an instance of the SAS Web Application Server. SAS Environment Manager includes plugins that allow the agent to monitor SAS resources such as the SAS Metadata Server and the SAS Stored Process Server. Other plugins allow SAS Environment Manager to monitor components that can be found in a SAS deployments such as a Hadoop deployment, web servers, or Postgres databases.

EVENTS AND ALERTS

SAS Environment Manager plugins define the types of events that should be recorded for monitored resources. Events are surfaced through the SAS Environment Manager Event Center web interface. There are five primary sources of events:

- Events created through the event importer
- Messages written to a log file associated with a monitored resource
- Changes made to monitored configuration files or directories
- Control actions initiated by administrators through SAS Environment Manager
- Alerts that have been triggered based on customizable conditions

The information contained in an event might vary by source and context. For example, when a disk space usage alert is triggered, the important information might include the actual disk space being used. On the other hand, if an event has been recorded because an error message was written to a monitored log file, the name of the log file and the message text are obviously relevant. To make reporting easier, the SAS Environment Manager Data Mart uses a standard set of fields for every event and alert, regardless of type. This set includes fields to identify the resource involved, the affected component (such as the name of the log file or the name of the alert triggered), the user involved, the time when the event occurred, and any additional information or messages that are generated.

LOG RECORDS

Log identification, collection, and parsing provide a third information store. Periodically, the log collector, which is a SAS Environment Manager component that is deployed alongside the SAS Environment Manager agent, identifies new log files (or new messages in an existing log file) and transfers those new logs to a central repository. Batch jobs execute each night to pull the new log information to the central repository for further processing. The log files are parsed, the important information is extracted, and the information is written to tables within the SAS Environment Manager Data Mart. Some of the technology behind these operations is derived from an earlier component, the SAS® Audit, Performance and Measurement Package. This package has been available as an experimental download from the SAS support website for several years. Like the SAS Audit, Performance and Measurement Package, the technology in SAS Environment Manager extracts two main types of information:

- Metadata audit information
- Performance measurements that capture resource consumption and information about how various SAS resources(such as content and servers) are being used

Like the SAS Environment Manager plugins, the SAS Environment Manager log parsing architecture is designed to be extensible and modular. Rather than one massive program that needs to understand every possible source and format of log file, individual parsers are designed to handle specific types of logs. For example, one parser might be responsible for parsing the log files generated by the SAS Stored Process Server while another handles web server logs, and another processes log files associated with a specific SAS solution. This modular approach also allows the work to be broken up into separate job flows that can be executed in parallel. Regardless of which parser processed a given log file, the resulting information is normalized and added to the SAS Environment Manager Data Mart.

DATA MART CONSTRUCTION

SAS Environment Manager Service Architecture constructs a data mart that consists of three SAS libraries and the six data structures (or patterns) described below.

The tables within the data mart can be categorized based on the type of data they contain and how they are structured. These categories can help clarify the types of reporting and analysis that might be most appropriate for the data.

THE RESOURCE DATA PATTERN

This is the pattern represented by the reporting data sets. Each record describes measurements taken at a point in time for a single resource. The simplest record structure would be: a date/time field, a resource identifier field and a resource measure.

THE JOB DATA PATTERN

This is the pattern recognized and classified by the log processing ETL. Each record describes a job or job step and provides summary level information for the resources consumed during the execution of that job or job step. A simplified record structure would be: job name, job start time, job end time, job duration, and total amount of resource consumed.

THE EVENT DATA PATTERN

Primitive events and alerts are stored in the SAS Environment Manager Data Mart in a normalized structure. Each record describes a specific event for a specific resource. A simplified record structure would be: resource, date/time, event type, who, what and message text.

THE INVENTORY DATA PATTERN

The inventory data pattern contains no measurements. It includes metadata about only the resources and measurements for which data is available from the SAS Environment Manager Data Mart. Each record describes a resource or measurement any additional descriptive attributes.

RESOUCE AVAILABILITY DATA PATTERN

Typically generated by some form of service “ping”, the availability record data pattern maintains one active record per resource. The record contains the resource ID, begin timestamp, end timestamp, and status. If there is a status change of state, the end timestamp is updated and a new record is constructed reflecting the current status state.

ALERT DATA PATTERN

Similar to an event, alerts are triggered and created based on event, metric, or control actions. Each record describes a specific alert and its triggering criteria. A simplified record structure contains: alert name, alert type, trigger criteria, date/time, who, what, and message text.

COMBINING PERFORMANCE AND EVENT DATA

For many types of events, criteria such as composite event duration, time-stamping, and identification of simple event recurrence are significant. For example, a given event might occur multiple times. Simply applying event operators on each event to produce a composite event,

instead of recognizing the multiple occurrence of a single event might produce an ambiguous result. Understanding and handling more complex elements of event collection and processing are outside of the scope of this paper.

You can also use the event sources to create composite alerts that are based on one or more alert creation criteria. For example, alert condition criteria can include greater than, less than, equals, or not equals, with each of these criteria being compared to either an absolute value or the percentage of a derived baseline.

EVENTS IN PRACTICE

Now that events, event criteria, and data patterns have been defined, this section discusses how to use SAS Environment Manager Data Mart, the Report Center, and the analytic capabilities of SAS, to better quantify an event in context with the collected samples and the distribution histories of the measurements.

SAS Environment Manager can create events based on inputs or triggers from these five information sources:

- event importer (including outlier identification)
- active log pattern matching
- control actions
- configuration changes or file modifications
- alerts

The following table, “Weekly Event Totals by Type and Source”, displays a summary chart of events, organized by event type and by source. Note that the table has three event types: alert, control, and log file message trigger.

Weekly Event Totals by Type and Source

		Weekly Count		
		08MAR15	15MAR15	22MAR15
eventType	Source			
ALERT	APM ETL Processing	0	2	0
	Object Spawner - ptnode22	0	5	0
	SASMeta - Metadata Server	4	6	0
	ptnode22	2028	2029	71
	ptnode23	3	1	0
	tc Runtime SASServer1_1	0	0	2
	All	2035	2043	73
CONTROL ACTION	Source			
	APM ETL Processing	1	0	0
	Log Centralization	1	0	0
	SAS Deployment Agent 1.0	6	13	0
	All	8	13	0
LOG FILE MSG	Source			
	EMI Framework Event Importer	54	131	1
	Object Spawner - ptnode22	0	6	0
	SAS Deployment Agent 1.0	0	192	1
	SASApp - OLAP Server	0	2	0
	SASMeta - Metadata Server	406	7	0
	All	460	338	2

The chart shows that some alerts are generated from event importing (for example, APM ETL Processing). Other triggered alerts also occur when a resource metric meets specific criteria, such as crossing a given threshold value. Each event, its occurrence time, and its description are recorded in the SAS Environment Manage Data Mart events table.

In addition, derived alerts and composite events can be triggered by post analytics, scoring, and outlier assessment routines. The composite event can then be written to the event importer file and loaded by using the event importer service.

Example alert sources can include:

- file system and file mount monitoring
- resource threshold triggers
- application audit log forensics
- access connections report
- application response report

EVENT IMPORTER

SAS Environment Manager Service Architecture includes the capability to import events using a simple text format. Although SAS Environment Manager provides a rich set of capabilities, production deployments frequently include custom or third-party tools for monitoring. The event importer provides a simple integration mechanism that allows external tools to easily create events in SAS Environment Manager by appending information to a text file.

The text file used by the event importer has the following characteristics:

- One line per event
- | (vertical bar) is the separator character between fields
- There are four fields, all required:
    - Timestamp (using the format yyyy-MM-dd'T'HH:mm:ssZ)
      for example, `#2015-03-10T09:45:24-05:00`
    - Level (DEBUG, INFO, WARN, or ERROR)
    - Source (100 characters)
    - Message (4,000 characters)

A default event importer is created as part of the SAS Environment Manager Service Architecture initialization in this location:

```
[LevRoot]/Web/SASEnvironmentManager/emi-framework/Events/sasev.events
```

Other instances of the event importer can be created manually. See *SAS Environment Manager User's Guide* for details.

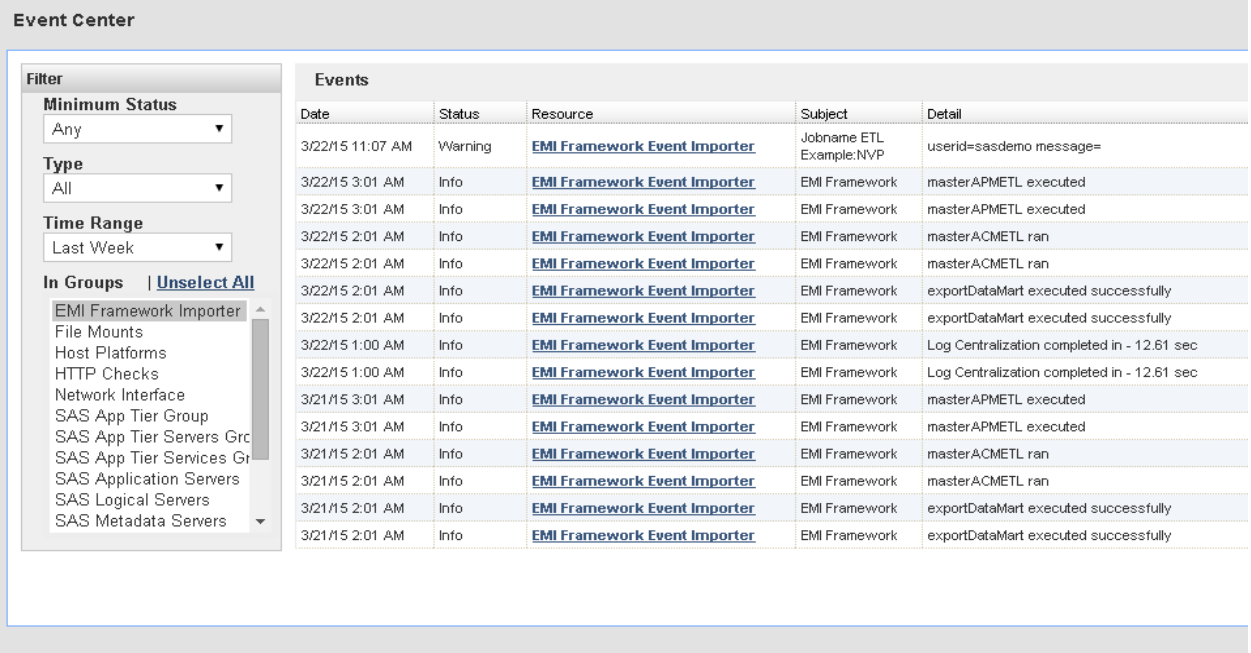
Updates to the text file occur in "near real time." It might take up to a couple of minutes for events to appear in SAS Environment Manager Event Center.

Here are some example lines in the event importer file:

```
"2015-02-07T20:32:35-05:00|INFO|MyApp|Startup completed in 13 seconds"
"2015-02-14T13:17:52-05:00|WARN|Auth|User [demo1] not authorized to [delete] [file1.txt]"
"2015-03-04T08:43:10-05:00|ERROR|MyApp|Unable to connect to database [CONFIGDATA]"
```

You can use the SAS macro "%evevent" to generate an event using the event importer. The syntax is described below:

```
evevent macro

  Usage:

      msglevel - Supports DEBUG, INFO, WARN, and ERROR. TRACE mapped to DEBUG

      msgtext  - Message String

      src      - Source message

      floc     - File path override to write event message.

%evevent(src=ETL,msglevel=INFO,msgtext=Master ETL completed);
%evevent(src=ETL,msglevel=WARN,msgtext=Second Step of TESTJOB Started);
```



ACTIVE SAS APPLICATION LOG MONITORING

The SAS Environment Manager agent and custom SAS plugins actively watch log files for significant messages. By default, log monitoring is supported and enabled for the following SAS application servers:

- Metadata Server
- Object Spawner
- Connect Spawner
- SAS Workspace Server
- Stored Process Server
- Pooled Workspace Server
- OLAP Server

All error messages and select warning messages (such as authentication failures or account lockouts) that appear in the log will create corresponding events in SAS Environment Manager. You can extend or customize the default set of patterns that are used to identify significant messages to include messages that are not tracked by default.

For example, suppose you need to create an event when the following message appears in the SAS Metadata Server log:

```
2015-03-04T04:55:14,900 WARN  [00000008] :cfgsas1 - The server is being switched to
an administrator mode because of the errors and warnings noted above.
```

To create the event, edit **<LevRoot>/SASMeta/MetadataServer/sev_logtracker_plugin.**properties to add the information in bold:

```
# User lockout warnings
level.warn.1=.*Access to this account.*is locked out.*
```

```
# User Login Failed
level.warn.2=.*New client connection.*rejected from server port.*
# Metadata Admin Mode – Just increment number and add the pattern
level.warn.3=.*server is being switched to an administrator mode.*
```

The changes will be automatically picked up within a few minutes, without the need to restart the SAS Metadata Server or the SAS Environment Manager agent. If this message appears in the log, an event will be created that can be seen in the SAS Environment Manager Event Center and, like all events, in the SAS Environment Manager Data Mart.

Further details about customizing event creation from watching log files can be found in *SAS Environment Manager User's Guide*.

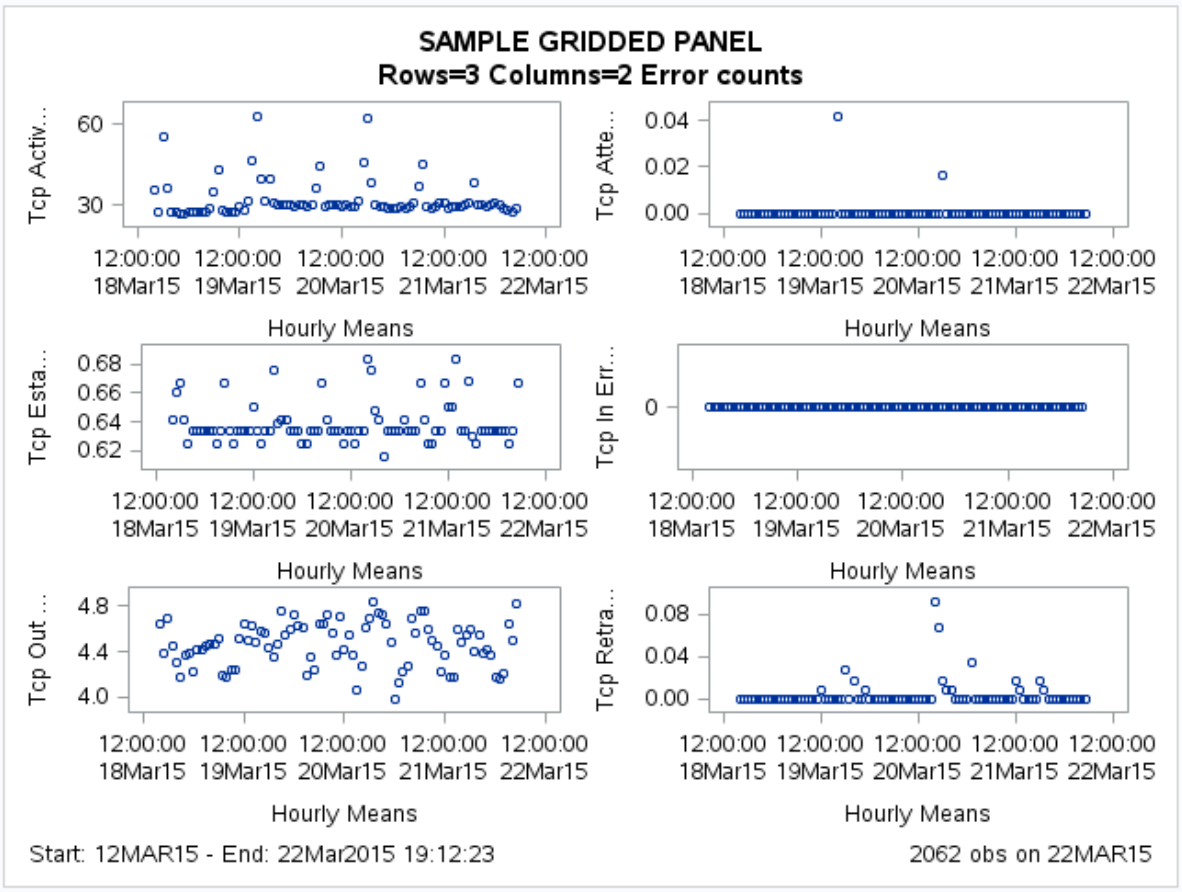
FILE SYSTEM AND FILE MOUNT MONITORING

SAS Environment Manager provides the tools needed to monitor both local and remote file systems. When you use the SAS Environment Manager Service Architecture to enable enhanced monitoring, a set of alert conditions is established for a variety of key resources that monitor the temp directory, the default SASWORK directory, SASHome, and the SAS deployment root directory (level root). These resources are automatically discovered and monitored by the SAS Environment Manager plugins, but the alert definitions are part of the enhanced monitoring in SAS Environment Manager Service Architecture.

Administrators can also create custom resources (which are called platform Services in SAS Environment Manager) to monitor other file systems or even mounted directories (such as NFS, Samba, and SMB). After you create the resources, you can set alert policies for them, which will cause the inventory, metrics, and alerts for these resources to be included in the SAS Environment Manager Data Mart. This data can be extremely valuable for both active monitoring in SAS Environment Manager as well as for forecasting and capacity planning using the SAS Environment Manager Data Mart.

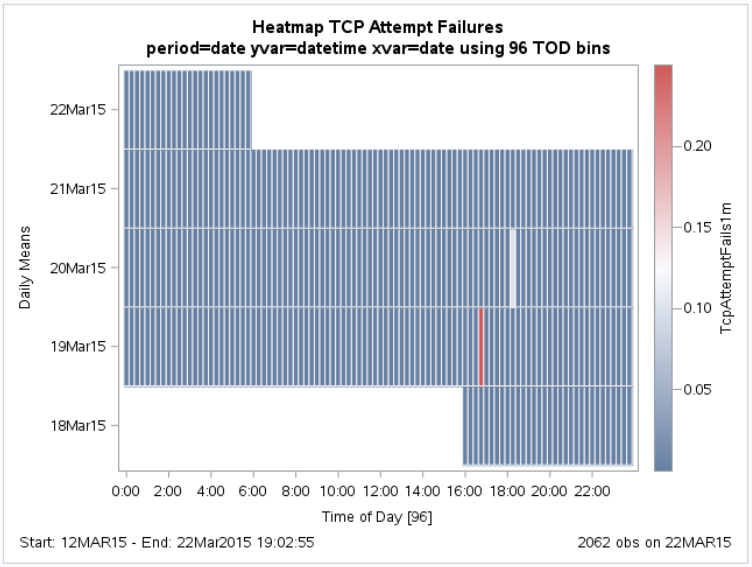
RESOURCE THRESHOLD ALERTS

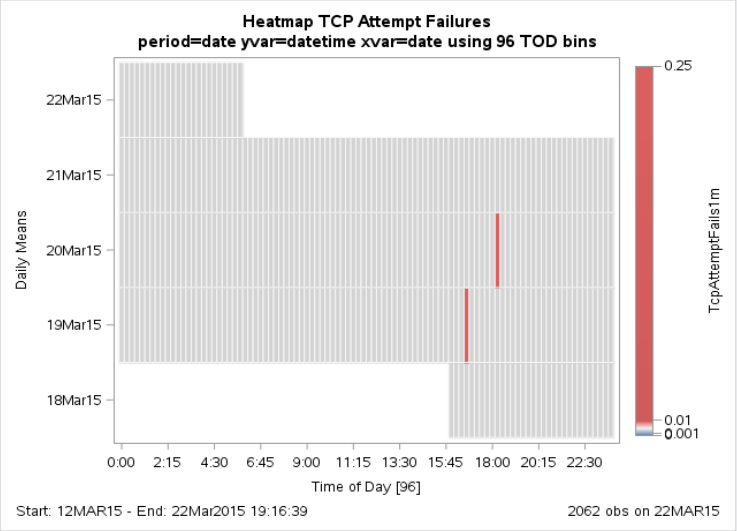
An example of a collection of platform resource metrics is presented below in an ODS PROC Template, 2 x 3 GRIDDED plot. This type of plot allows each plot in the grid to have a unique vertical axis for each plot. In this example, the plots display variable TCP error measurements, including TCP In Errors, TCP Retransmits, and TCP Attempts. Plot 2, which is in the upper right, shows two TCP Attempt failure outliers, one occurring around 2 PM on March 19 and the other occurring around 5 PM March 20. However, the date and time of the actual event is difficult to precisely determine.



SAS Environment Manager Service Architecture includes numerous time series analytics and reports that take advantage of the capabilities of SAS analytics. For example, we can use a heat map to display more precise information about the outliers in Plot 2 of the previous example.

The following two charts are heat maps of the TCP Attempt failure outliers in the previous example. The first heat map was produced with an automatic population scaling. The second was produced by providing a single threshold of 0.01 attempts per minute. Both maps clearly present the precise time and date of the two outliers.



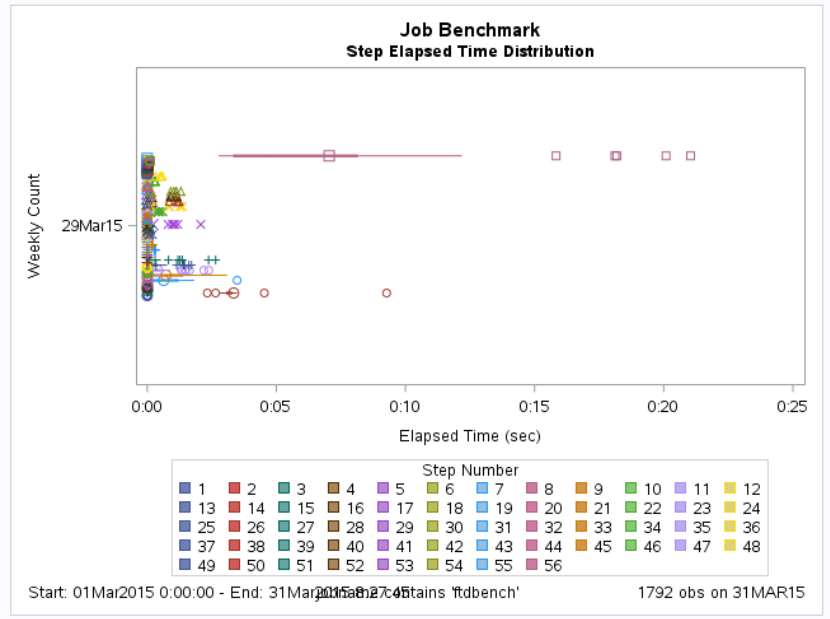


SAS JOB PERFORMANCE, PUTTING IT ALL TOGETHER!

The final example leverages many capabilities of SAS Environment Manager, including:

- Resource monitoring
- Processing of SAS job log records
- SAS DATA step code
- %evevents macro and the event importer.

The following HBOX distribution graph presents a collection of job steps (PROC steps), showing variations in the run time for each step. Most steps run very quickly, and their run times are consistent and repeatable. However, step 8, which is represented by purple squares, shows a much broader variance in the step's elapsed run time.



The question we need to answer is this ; does this variance matter, or is it a sign of a resource constraint?

The SAS jobs log parser processes and collects records from both stimer and fullstimer, including CPU, I/O, context switches, and memory usage. In addition, some procedures write additional tuning information in the log, which suggests a potential need for performance tuning.

For example, the fullstimer report contains the following information.

```
NOTE: Processing on disk occurred during summarization. Peak disk usage wa
      REALMEMSIZE may improve performance.
NOTE: There were 1000000 observations read from the data set FTD9.FTD.
NOTE: The data set WORK.HSIS has 999995 observations and 109 variables.
NOTE: Compressing data set WORK.HSIS decreased size by 34.56 percent.
      Compressed is 8499 pages; un-compressed would require 12988 pages.
NOTE: PROCEDURE SUMMARY used (Total process time):
      real time           2:16.27
      user cpu time       29.42 seconds
      system cpu time     15.10 seconds
      memory              727529.60k
      OS Memory           736628.00k
      Timestamp           03/31/2015 12:03:41 PM
      Step Count                        9  Switch Count  0
      Page Faults                       0
      Page Reclaims                     126471
      Page Swaps                        0
      Voluntary Context Switches        89672
      Involuntary Context Switches      140
      Block Input Operations            0
      Block Output Operations           12905432
```

PROC SUMMARY writes this note to the log, suggesting possible performance tuning:

```
NOTE: Processing on disk occurred during summarization. Peak disk usage was
approximately 3206 Mbytes. Adjusting MEMSIZE and/or REALMEMSIZE may improve
performance.
```

The SAS jobs parser flags the note from PROC SUMMARY and sets the tuneopp flag to 1 for the specific SAS job running the specific PROC SUMMARY job step.

You can then combine the SAS jobs parser, the value for tuneopp, and the SASJob_INFO data set in a %evevent macro. The macro creates a notification of a potential performance tuning issue through triggering and importing the event notification.

For example, this %evevent macro statement flags areas where the log has identified that tuning might be needed:

```
%macro chkjobslog(ds=);
   %local dsid rc;
   %let dsid = %sysfunc(open(&ds));
   %do %while (%sysfunc(fetch(&dsid)) = 0);
%let jobname=%sysfunc(getvarc(&dsid,%sysfunc(varnum(&dsid,jobname))));
%evevent(src=SASJOBS_INFO, msglevel=WARN, msgtext=Job [&jobname]
Review Tuning);
   %end; /* end of task loop */
   %let rc = %sysfunc(close(&dsid));
   %if &rc ne 0 %then %put %sysfunc(sysmsg());
%mend chkjobslog;
```

Running the %chkjobslog macro against the KITS.SASJOBS_INFO data set creates an event record for each SASJOBS_INFO record that has been flagged with the possible tuning condition.

```
%chkjobslog(ds=KITS.SASJOBS_INFO);
```

The chkjobslog macro then records an event in the sasev.events file, such as the following:

```
20150331T211840+0000|WARN|userid=sasdemo  message=Job [bench] Review Tuning
```

CONCLUSION

SAS Environment Manager Service Architecture provides IT operations with a variety of powerful tools for creating, importing, storing, and analyzing event data. Event data flows from multiple sources, including logs, SAS jobs, and alerts, into the SAS Environment Manager Data Mart. These new tools enable this rich set of event data to be combined with the wealth of operational metric data that is collected by SAS Environment Manager. The SAS Environment Manager Data Mart empowers IT operations staff to be more knowledgeable, proactive, and responsive in order to meet established service level agreements. Moving forward, SAS Environment Manager Service Architecture will continue to evolve to support even more sources and more composite events, and to operate in near real time.

REFERENCES

Bonham, Robert. and Gregory Smith. 2014. "Log entries, Events, Performance Measures, and SLAs: Understanding and Managing your SAS Deployment by Leveraging the SAS Environment Manager Data Mart." *Proceedings of the SAS Global Forum 2014 Conference.* Cary, NC: SAS Institute Inc. Available

http://support.sas.com/resources/papers/proceedings14/SAS064-2014.pdf.

Buchmann, Alejandro and Annika Hinze. 2010. *Principles and Applications of Distributed Event-Based Systems.* Hershey, PA: Information Science Reference.

Johnson, M. W. 2000. "ARM 3.0 - Enabling Wider Use of ARM in the Enterprise". *Computer Measurement Group's 1999 International Conference (CMG99).* Research Triangle Park, NC: Tivoli Systems.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Bob Bonham
100 SAS Campus Drive
Cary NC 27513
SAS Institute, Inc.
Bob.Bonham@sas.com

Bryan Ellington
100 SAS Campus Drive
Cary, NC 27513
SAS Institute, Inc.
Bryan.Ellington@sas.com