OpenVMS Technical Journal

Adding a Friend to T4 and Friends

Incorporating BEA WebLogic Server 8.1 Performance Data

Pat McConnell

Performance Group, OpenVMS Engineering

Pat.McConnell@hp.com

Overview

This OpenVMS Technical Journal article illustrates an addition to the T4 and Friends family originally introduced in Steve Lieman's "*TimeLine-Driven Collaboration with T4 and Friends: A Timesaving Approach to OpenVMS Performance*".[1] This particular addition to the family is concerned with gathering key BEA WebLogic Server 8.1 performance information and correlating that information with the OpenVMS performance information gathered by T4. The primary purpose of this article is to provide a template for integrating a new monitoring application into the default implementation of T4. The sample BEA WebLogic Server 8.1 monitoring T4 application outlined in this article is available by request from the author.

Introduction

This work was undertaken during the chaos of an internal benchmarking exercise within OpenVMS Engineering, so there were several important influences that should be brought out to frame this sample extension to the default T4 implementation.

First, the internal benchmarking exercise was focused on the effect of a BEA WebLogic Server 8.1 based workload on OpenVMS. Therefore, it had already been decided to utilize T4 to monitor the OpenVMS systems involved in the benchmark, and that obtaining all possible BEA WebLogic Server performance statistics was not a goal of the benchmark. Second, like all efforts of this type, time was at a premium, meaning that the programming involved with this effort had to be expeditious rather than complete and comprehensive. That said, the resulting code and modifications to the DCL procedures comprising T4 could be valuable to anyone interested in correlating OpenVMS and BEA WebLogic Server 8.1 performance metrics.

BEA WebLogic Server 8.1 Performance Metrics

Examination of the BEA WebLogic Server 8.1 documentation indicated that a large, comprehensive set of application performance metrics were available for retrieval via JMX. For this particular set of constraints, it was considered impractical to develop a complete monitoring application that obtained all possible performance metrics. It was important to determine the key WebLogic Server performance metrics that captured the essence of the server's performance. A search of the BEA WebLogic Server 8.1 documentation set revealed the document, *Programming WebLogic Management Services with JMX*, which contained a section entitled, **Best Practices: Commonly Monitored Attributes**. This section detailed a number of BEA WebLogic Server 8.1 attributes that provide a general overview of the performance of a WebLogic 8.1 server. Table 1 briefly names and describes these attributes, but an examination of the aforementioned section within the *Programming WebLogic Management Services with JMX* document is necessary for a thorough understanding of the performance attributes and their access mechanisms.

[1] VTJ Volume Three, January 2004

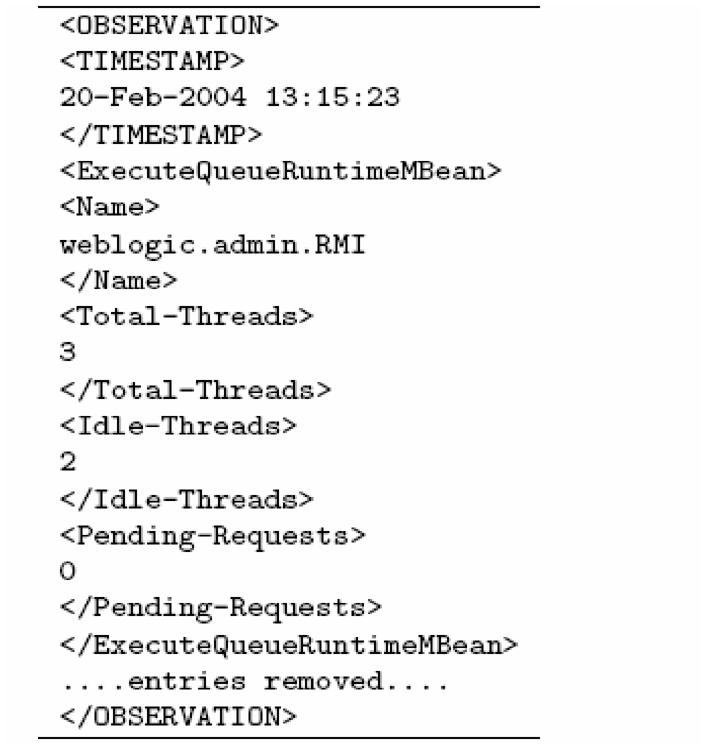
Table 1

Attribute Name	Description
`State`	State of Server
`HeapSizeCurrent`	Server's JVM heap size
`ActiveConnectionsCurrentCount`	Active JDBC connections
`ConnectionsHighCount`	JDBC connection pool high water mark
`LeakedConnectionCount`	Total number of leaked connections
`OpenSocketsCurrentCount`	Current number of open sockets
`AcceptBacklog`	Number of requests that can be backlogged
`ExecuteThreadCurrentIdleCount`	Idle threads in default execute queue
`PendingRequestCurrentCount`	Waiting requests in Server's default execute queue
`ActiveConnectionsCurrentCount`	Active connections to a JDBC connection pool
`ConnectionDelayTime`	Average time to connect to a JDBC connection pool
`FailuresToReconnect`	Connect failures for a JDBC connection pool

Collaborating with T4

The next item of business was to determine how to obtain these metrics within the context of the T4 and Friends framework. The ultimate goal of the T4 and Friends framework is to develop a regularly spaced timeline of all performance metrics. The current implementation of T4 is implemented with the assumption that all metrics of interest are available at each interval of the measurement period. However, in reviewing the key BEA WebLogic Server 8.1 performance attributes mentioned previously, it was clear that there might be a varying number of entries under some of the attributes from one interval to another. Therefore, it was decided that an intermediate XML file would be written by the BEA WebLogic Server 8.1 monitoring program.

The intermediate XML file written by the BEA WebLogic Server 8.1 monitoring program is a minimal one whose main purpose is to capture the run parameters for the monitoring session, and then capture the value of the BEA WebLogic Server 8.1 performance attributes at each measurement interval. A later processing step will convert this XML file to a file format appropriate for T4 analysis. A partial example from a measured system can be found in Figure 1.



```
<OBSERVATION>
<TIMESTAMP>
20-Feb-2004 13:15:23
</TIMESTAMP>
<ExecuteQueueRuntimeMBean>
<Name>
weblogic.admin.RMI
</Name>
<Total-Threads>
3
</Total-Threads>
<Idle-Threads>
2
</Idle-Threads>
<Pending-Requests>
0
</Pending-Requests>
</ExecuteQueueRuntimeMBean>
....entries removed....
</OBSERVATION>
```

Figure 1: Intermediate Partial XML Contents

The BEA WebLogic Server 8.1 monitoring program, `ServerExQInfo`, was designed to accept inputs compatible with those used within the key collection procedure of T4, to regularly sample the BEA WebLogic Server 8.1 performance attributes, and to write to the intermediate XML file. The processing logic of the program is straightforward and consists of the following steps.

- Accept input parameters, validate, instantiate `ServerExQInfo`
- Create monitoring thread, hold execution until start time
- Monitor BEA WebLogic Server 8.1 via JMX MBean accessor functions at each interval
- Write observations to XML output file
- End monitoring and close XML file at the provided end time

Extending T4 Default Implementation

Adding a friend to T4 and Friends often requires modifying the DCL procedures that implement T4, so a quick review of T4 processing logic follows.

- `T4$CONFIG` is run to gather parameters
- `T4$COLLECT_V33` is submitted to a batch queue
- `T4$COLLECT_V33` executes on a batch queue to spawn collectors
- `T4$COLLECT_V33` then does post-processing at the end of the monitoring period
- A composite T4 CSV file is then available for analysis

Changes to T4$COLLECT.COM

The first modification made to the T4 DCL procedures to integrate the BEA WebLogic Server 8.1 performance monitoring functionality was made to the T4$COLLECT_V33 procedure. Within this procedure, the first change was to determine whether to initiate BEA WebLogic Server 8.1 monitoring. Figure 2 shows where in the procedure the change was made, as well as the code that determines whether monitoring is desired. Figure 3 illustrates the contents of the file that controls which BEA WebLogic Server 8.1 instances are monitored.

```
$! NOTE THAT MANY LINES HAVE BEEN DELETED TO THIS POINT
$! TO ILLUSTRATE WHERE TO PLACE WebLogic Server HOOK
$! INTO T4$COLLECT_V33.COM
$!
$! First of all, spawn the MONITOR data collection ...
$!
$ Mem_Size = "("+F$String(f$GetSyi("MemSize")/128)+"Mb"
$ Avail_Cpus = " with ''F$GetSyi("AvailCpu_Cnt")' cpu(s))"
$ Spawn/NoSymbols/NoLogicals/NoWait/Process="''This_Pid'_MON" -
 Monitor/Record=T4_'This_Node'_'Today'_'St_Et'_Mon.Dat -
 /Interval='P6 -
 /Flush_Interval='P6 -
 /Begin="''Start_Time'" -
 /End="''End_Time'" -
 /Comment="''F$GetSyi("Hw_Name")' ''Mem_Size'''Avail_Cpus'" -
 /NoDisplay           All_Classes
$
$
$!  WebLogic Server hook - if present, pass exactly the same driving parameters
$!  to WebLogic Server code.
$!
$ if F$Search("t4$sys:wlstltTab.wls") .Nes. "" Then -
    @t4$sys:wlstlt.Com "''P3'" "''Start_Time'"     "''End_Time'" -
                    "''P6'" "''Today'_''St_Et'" "''P7'" "''This_Pid'"
$!
$!
$! MANY LINES DELETED FROM THIS POINT IN T4$COLLECT_V33.COM
$! TO ILLUSTRATE WHERE TO PLACE WebLogic Server HOOK
$!
$ ENDSUBROUTINE
```

Figure 2: Changes to T4$COLLECT Procedure

```
$ WLStlt.wls - file that controls T4 initiated monitoring of
$ WebLogic Server instances.
$ WLSName,WLSURL,WLSPort,WLSAdmin,WLSAdminPassword,Monitor This Instance?,BEAHome
mydomain,t3://monitor.this.system,7001,weblogic,weblogic,Y,dkd400:[bea]
```

Figure 3: Control of Monitoring File Contents

WLSTLT.COM

This modification to `T4$COLLECT_V33` invokes the DCL procedure, `WLSTLT.COM`, which creates a DCL procedure for each BEA WebLogic Server 8.1 instance that the control file indicates should be monitored. Each of these created DCL procedures will be submitted to the batch queue specified in `T4$CONFIG.COM`. These created procedures are the workhorses of the BEA WebLogic Server 8.1 instance monitoring, using `ServerExQInfo` to access the BEA WebLogic Server 8.1 performance metrics, and recording the observations in an XML file.

Creating a Composite T4 CSV File

At the end of the monitoring session, the T4 procedure `T4$COLLECT_V33` creates a T4 CSV file that aggregates all of the observations made by the various T4 monitors. In this sample T4 and Friends application, the XML file output is not available to T4 to do automatic post processing, so a different approach is required. At the end of each `WLSTLT.COM` execution, the XML file output is converted to a T4 CSV style file, and then each of these files are appended to the T4 CSV file produced via `T4$COLLECT_V33` with the T4 utility, `T4$APRC`. Once this work is done, the resulting composite T4 CSV file contains not only OpenVMS system performance metrics in a timeline format, but the corresponding BEA WebLogic Server 8.1 performance metrics as well. Figure 4 illustrates this situation by showing the output of a sample visualization program where both an OpenVMS and BEA WebLogic Server 8.1 metric are plotted.

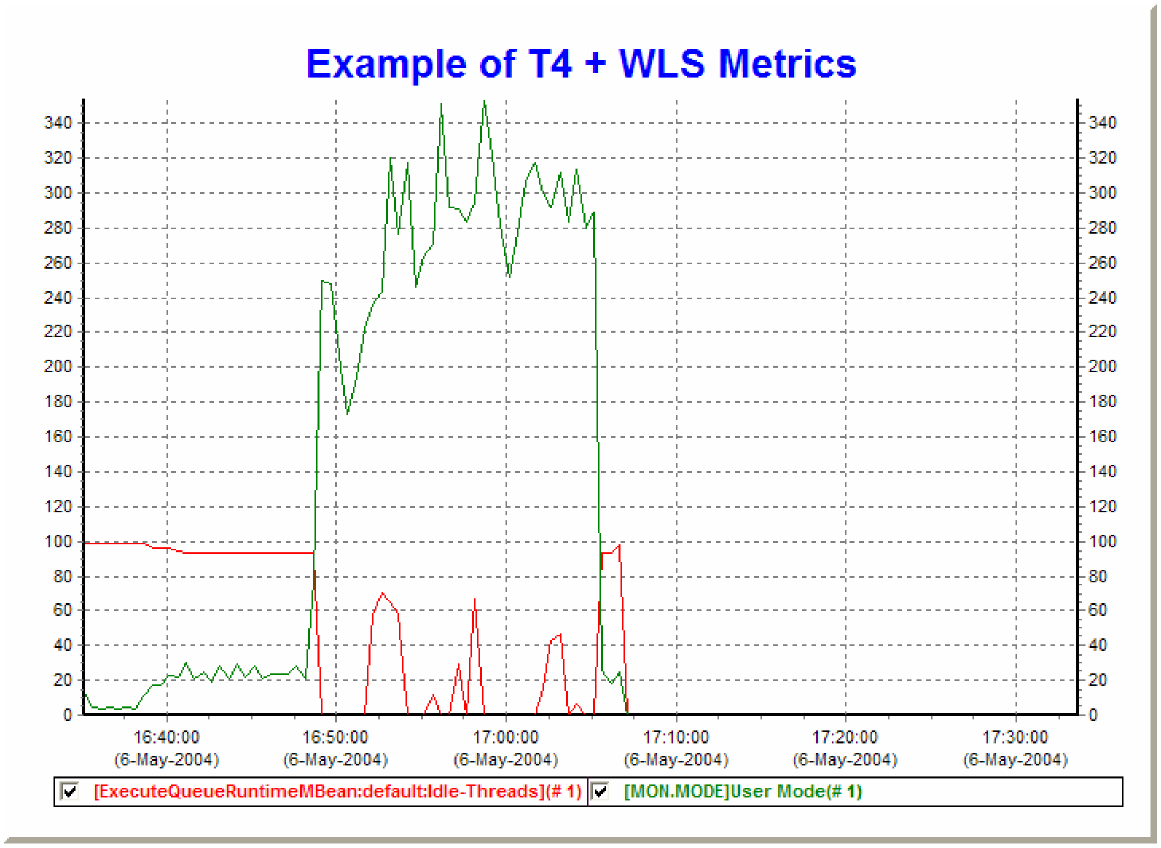


Figure 4 - Example of Merged T4 and WLS Statistics

Summary

This article outlined the steps necessary to extend the default T4 implementation to include the ability to obtain key BEA WebLogic Server 8.1 performance metrics, and then to integrate the BEA WebLogic Server 8.1 information into the composite T4 CSV for later processing by members of the T4 and Friends family. The steps that are outlined within this article can be abstracted to develop other T4 extensions for whatever an OpenVMS performance analyst requires.

For more information

"*TimeLine-Driven Collaboration with T4 and Friends: A Timesaving Approach to OpenVMS Performance,*" VMS Technical Journal, Volume Three - February 2004

This article is the standard reference for understanding the Timeline Collaboration concept which has driven the development of T4 and Friends. A reading of this article will give the necessary background for the development of extensions to the default T4 implementation.

BEA WebLogic Server Programming WebLogic Management Services with JMX

Release 8.1

Revised: July 18, 2003

This manual presents the use of JMX by BEA WebLogic Server 8.1 to provide access to its performance metrics. It is a clearly written reference that contains numerous Java examples.

SYS$ETC

As mentioned in the body of the article, a default T4 implementation is included in this library from OpenVMS v7.3-2 onward.