International Journal of Computer Science and Engineering (IJCSE)
ISSN(P): 2278-9960; ISSN(E): 2278-9979
Vol. 5, Issue 5, Aug-Sep 2016; 25-30

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

EFFICIENT MONITORING AND MANAGEMENT OF MIDDLEWARE PLATFORM

SAVEETHA RUDRAMOORTHY

Senior Architect, Middleware Expert, Amadeus Labs, Bangalore, Karnataka, India

ABSTRACT

Middleware platform comprises of technologies and services which basically interlinks / glues complex integration components, applications, data, and people across enterprise. Middleware platform is very critical in mediating and providing integration services to end customer and backend applications. Hence, it is not acceptable to have any unmanaged component or service, wherein middleware platform should ensure zero downtime with 24/7 support.

This white paper aims in detailing about the efficient ways to monitor and manage middleware platform end-to-end and hence offering complete zero downtime and efficient operational management.

KEYWORDS: Middleware, Horizontal Middleware Management, Log Management Intelligence, Vertical Middleware Management, Infrastructure Management

INTRODUCTION

Following are the key components of typical middleware platform:

Enterprise Service Bus, Service Container, Business Process Management, Business Activity Monitoring, Complex Events Processing, Web Service Gateway and Management and B2B

Customization on these component stack is based on end customer requirements and business values provided. These set of component provides following core capabilities:

- Synchronous, Asynchronous and Batch messaging
- Process Management
- Service orchestration and integration
- Systems management and monitoring
- Access control and throttling

In addition to the core capabilities middleware platform also offers:

- Service Governance
- Security
- Complex Event processing
- Business Intelligence
- Integrated Development, Deployment and Testing Tools

Middleware Reference Architecture

Following diagram depicts typical middleware component reference architecture. Middleware Service provider could decide and orchestrate on one or more components in the below mentioned architecture based on customer needs and business values.

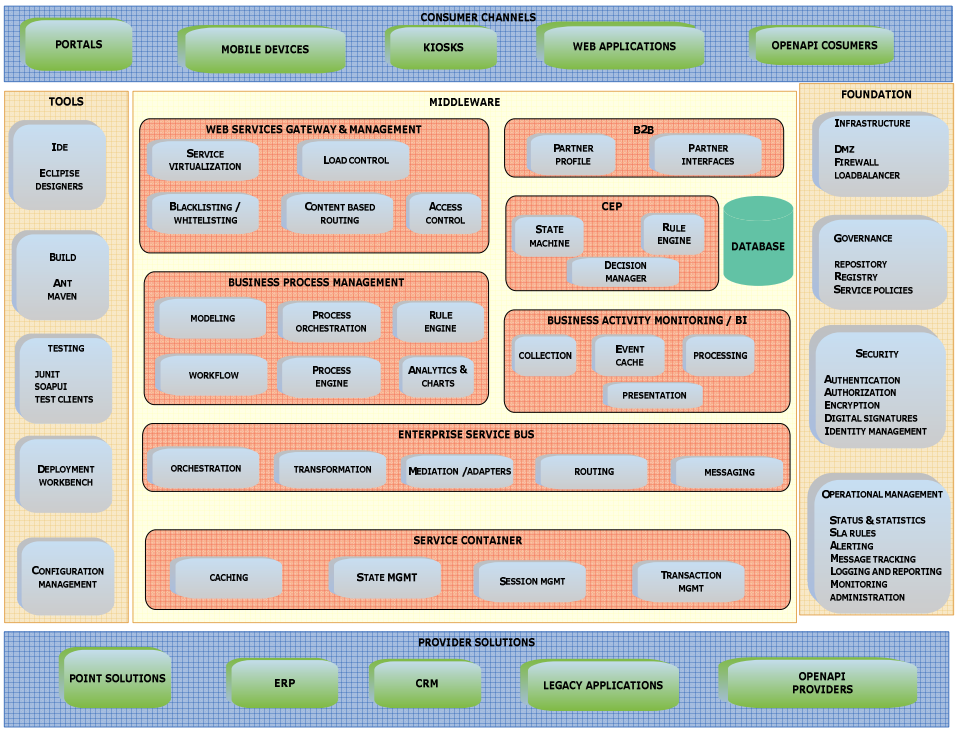

Figure 1: Middleware Reference Architecture

Monitoring Middleware Platform

Monitoring such middleware platform is vital in order to ensure stability of the platform, track end to end business message flow, collect various statistics from audited message to cater customer needs, proactivity identify the issues and automate problem resolution, watch trends, diagnose system bottlenecks etc. Effective middleware management is critical to deliver the expected service to the business.

It is recommended to have 3 layered middleware management architecture:

- Horizontal middleware management
- Vertical middleware management
- Infrastructure management

Horizontal Middleware Management

Horizontal middleware Management provides visibility of message flows across the various components of the middleware platform. By monitoring the messages flowing horizontally, one could find out Total turnaround time of a transaction, success/failure ratio, and platform usage by various clients, SLA variance, and historic metric data and perform trend analysis etc. In addition to that, it is possible to associate functional errors with alerts. We can think of creating high severity/Sev-1 alert if particular business functionality is failing repeatedly in certain duration.

Horizontal management can be accomplished by 3-layered approach:

- The first level is '**Collection'**, wherein message flow is audited by applications hosted on components and audited data is collected.
- The second level is '**Storage'**, where audit data collected are stored on Data repository.
- The third level is the '**Exploitation'**, where stored data is exploited and reported via Reporting solutions.

Following diagram provides an overview how SOA message flows are monitored by Log based intelligence solutions in a typical middleware stack:

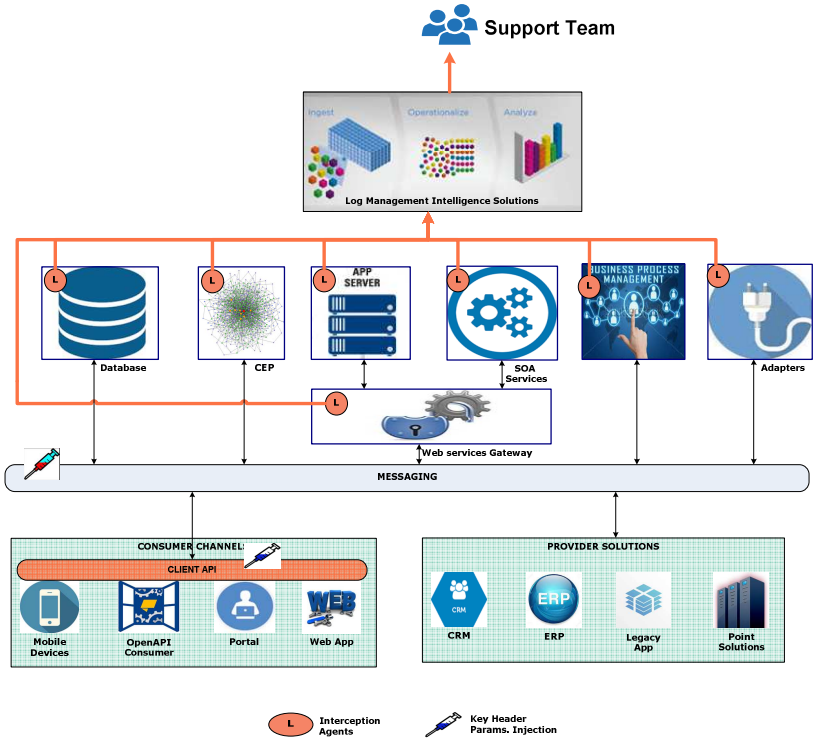

Figure 2: Horizontal Middleware Monitoring and Management

The basic principle of horizontal management is that consumer injecting "unique ID" in the message header in order to trace, inspect and instrument the flow in real time. Advantage of this approach is that if disparate applications are integrated and each application is having unique identification parameter, then messaging infrastructure does not worry about these application specific business ids.

Convenience Client API layer can be provided to consumers to perform this Unique id injection and to abstract other messaging complexities. Unique id could be 1 key or set of keys. If the platform caters to multiple customers and each customer having multiple applications, then unique id could be combination of

"CustomerID" + "ApplicationID" + "CallID"

Wherein sample key for a telecom domain could be,

CustomerID = Airtel or BSNL

ApplicationID = Customer MgmtApp or BillingApp

CallID = Random Unique identifier

Again, for example, if billing solution is serviced by multiple applications and application uses multiple unique devices, then this unique id could be combination of

“CustomerID” + “Domain” + “ApplicationID” + “DeviceID” + “CallID”

These elements can be set as individual tag elements for SOAP request or Query parameter if ReST request is made by client. ‘Envelope Wrapper’ pattern is followed here as these elements are purely used for middleware management and wrapped under SOAP header, wherein payload of the message can be part of SOAP body.

One sample SOAP header message could be:

```
<soapenv:Envelope xmlns:soapenv=”http://schemas.xmlsoap.org/soap/envelope/”>
<soapenv:Header xmlns:ca="http://example.com ">
<ca:CustomerID>BSNL</ca:CustomerID>
<ca:Domain>BILLING</ca:Domain>
<ca:ApplicationID>BILLING_APP_1</ca:ApplicationID>
<ca:DeviceID>DEVICE1</ca:DeviceID>
<ca:CallID>874555885</ca:CallID>
</soapenv:Header>
</soapenv:Body>
</soapenv:Envelope>
```

Still, if the client is unable to perform unique id injection, then middleware platform entry component can simulate this unique id. Once request is sent to service provider, it is expected that service provider retain these unique id header parameters in the reply as well. Thus a transaction can be tracked end to end with same id.

Thus, these messages with unique id flowing through middleware components can be audited into various log sources like text file, database, EMS queues, documents etc. This covers the ‘Collection’ part.

‘I*nterception agents’* installed on middleware components can intercept the log file, EMS queue or database and push the information into centralized log management data repository, covering the ‘Storage’ part.

Reporting tools built on top of log repository will provide reports on message flow, SLA variance, Latency etc, covering ‘Exploitation’ part.

Log Management Intelligence Solutions

There are various commercial and open source vendor offerings which will operate in above principal of Collection, Storage and Exploitation. Tibco Loglogic suite, Splunk, Elasticsearch Logstash Kibana (ELK), LogRhythm etc are few solution offerings from various vendors catering log based business intelligence.

Vertical Middleware Management

Vertical SOA Management provides the set of tools to administer individual middleware components centrally, and to operate the platform in a highly automated manner. For load and scalability reasons, production class middleware platform contains more than one instance of middleware components. Vertical monitoring and management talks about monitoring each of this instance and manage centrally.

Following diagram provides an overview how vertical management can be performed on middleware

components:

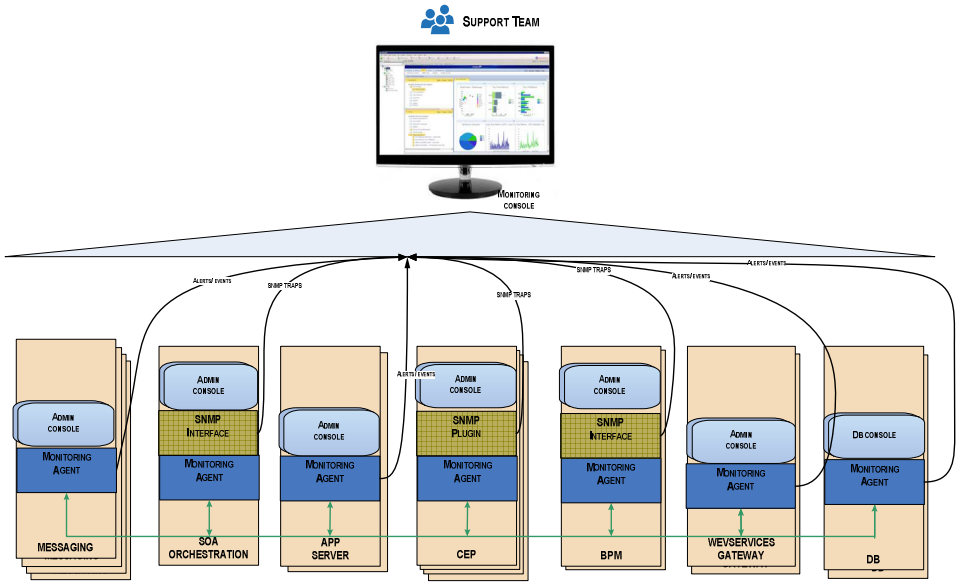

Figure 3: Vertical Middleware Monitoring and Management

VERTICAL MANAGEMENT TOOLSET

Individual Components' Administration Tools

Most middleware components come with sophisticated inherent administration tools to manage the breadth and depth of the components features. Custom administration tools can be built to overcome a shortcomings with few of the components. These component admin tools assists one to perform configurations, deploy applications, monitor the status of applications etc

Vertical Monitoring Software / Scripts

There are few middleware components having inherent capability to perform vertical monitoring and reporting. Additional software monitoring agents or customized scripts can be installed on servers where middleware components are installed. There may be few components capable to produce SNMP traps. In these cases, these monitoring agents captures SNMP traps and report.

Typical support one could expect out of these agents are:

- Monitoring and reporting of resource usage like CPU, N/W availability, Virtual memory, Disk utilisation, Table space utilisation etc
- Monitoring the component process state (Active or Inactive) and restart the process if it is failing. Additionally, alert the operational management personalif continuous failures are happening.
- Monitor the applications 'Health' status hosted on middleware by sending 'Echo' messages
- Monitor and report if certain log statements appear repeatedly on middleware component event logs
- Associate the reporting with incident management, wherein automatic 'Incident Requests' can be raised based on severity.

Few vendors who provide such vertical monitoring software are Tibco, Nagios, Zabbix etc

Centralized Management Console

It is ideal to have centralized management console on top of all middleware admin consoles. This central console will give complete snapshot of all the middleware component statuses and one could drill down into specific component status if required.

Infrastructure Management

Infrastructure management covers data central automation in terms of monitoring of server hardware, operating systems, storage and network components (including HW load balancers), as well as database administration.

Infrastructure monitoring agents like 'Hyperic' can be installed on all the VMs and Servers where middleware components are installed. These Hyperic Agents do more or less the same as the Sightline Agents, reporting Server availability / performance information / disk utilization / memory utilization etc. If there is an error, e.g. file system filled 85%, then the Hyperic server can raise automatic incidents. For storage management, Netapp On Command Manager, IBM XiV can be used.

CONCLUSIONS

With the 3 layered middleware monitoring and management approach discussed above, one could manage the platform efficiently and cost effectively.

It is easier to extend this 3 layered approach to any new middleware component. Also, with aid of central management console, operations team gets the view of complete server, software and application statistics, end to end message flow, Low/Medium and High priority alerts etc.

REFERENCES

1. http://www.slideshare.net/Zubin67/open-source-middleware-reference-architecture
2. http://www.ibm.com/developerworks/library/ar-archtemp/
3. https://blog.profitbricks.com/top-47-log-management-tools/
4. http://mashable.com/2015/11/17/network-server-tools/#uPJ.bpnNIgqZ

AUTHOR DETAILS

Saveetha is a Senior Architect, Middleware Expert with Amadeus Labs based out of Bangalore, Karnataka. She is Tibco and webMethods certified with over 16 years of experience in enterprise integration and middle ware across domains like Telecom and Airline.

You can find Saveetha on LinkedIn: https://www.linkedin.com/in/saveetha-rudramoorthy-06aa296