## Monitoring Standards for the Producers of Web Services

02-21-2017

Alexander Quang Truong

**Contents**

1. **Summary** ........................................... 2
2. **Metrics** ............................................. 2
3. **Benefits and Explanations of Metrics** .................. 2
4. **Tools for Monitoring** .................................. 3
5. **Suggestions and Guidelines for Producers of New Web Services** .................. 3
   - Key Monitoring Points ........................................ 3
   - SLA ........................................................................ 5

---

Northwestern INFORMATION TECHNOLOGY

---

Monitoring Standards for the Producers of Web Services

02-21-2017

Alexander Quang Truong

**Contents**

1. **Summary** ........................................... 2
2. **Metrics** ............................................. 2
3. **Benefits and Explanations of Metrics** .................. 2
4. **Tools for Monitoring** .................................. 3
5. **Suggestions and Guidelines for Producers of New Web Services** .................. 3
   - Key Monitoring Points ........................................ 3
   - SLA ........................................................................ 5
1. Summary
As a combined effort in Northwestern University Information Technology (NIT), a group of developers across multiple systems and groups which service a multitude of functions has been put together in order to standardize the way web services are monitored through the Enterprise Service Bus (ESB). The workgroup has put together recommendations as to what and how certain characteristics and metrics should be captured to produce useful information for all web services produced by the university either as a stand-alone service or through the ESB as an intermediary. The intent of this document is to provide guidelines for best practices to be adhered to at Northwestern University for monitoring web services and provide our consumers and/or third party vendors with adequate information to consume our services.

2. Metrics
There are many metrics to consider when a web service is exposed. These metrics provide valuable information for all aspects inherent to web services. These metrics include, but are not limited to:

- Performance and Memory Usage
- Access and Usage (Invocations)
- Response Time
- Anticipated Avg. and Peak Loads
- Uptime and Downtime
- Security, Faults, and Violations
- HTTP Status Codes: 403, 404, etc.
- Timeouts/Reconnect attempts

These metrics serve to better understand how robust they are and to what audience is interested in using that web service. From this information, optimizations or enhancements can be made to better and reliably serve our consumers.

3. Benefits and Explanations of Metrics
- Performance and Memory Usage
  - Web service performance and its total memory usage will allow better memory allocation or code refactoring
- Access and Usage (Invocations)
  - Where and how each web service is used should be logged. We can really understand the audience from understanding where most of the access is coming from, at what time these invocations are taking place, what sort of invocations are made, which methods are used more frequently. With this information, we can improve, or rather, pinpoint key and potential optimizations.
• How quickly each service is performing from invocation to response should be logged in a meaningful way.

- Anticipated Avg. and Peak Loads
  - This information should be made available for consumers as to what the web service can handle in terms of load/strain

- Uptime and Downtime
  - How often the service is restarted
  - How often is the service down
  - When was the longest uptime recorded
  - When are most frequent downtimes occurring

- Security, Faults, and Violations
  - Attempts to illegally access the web service (e.g. SQL injection, etc.) should be logged

- 403, 404, etc.
  - Which HTTP Status codes are frequently thrown

- Timeouts
  - Number of allowed attempts when bad arguments, incorrect arguments, or unauthorized access are passed into the service should be explicit to consumers

4. Tools for Monitoring

All touchpoints for web services should be monitored end-to-end. From the time the consumer hits the web service to its delivery to the ESB and then making the roundtrip, everything should be logged that each “checkpoint” was reached with a given status, what occurred, who made the request, and the time it took.

SolarWinds is a good tool to monitor most, if not all, of the aforementioned metrics. An open issue needs to be logged with SolarWinds so that the web service can be registered on its service. SolarWinds also provides another product, Librato, for real-time cloud monitoring.

These tools should offer a way for notifications to be sent out on a case-by-case basis of services going down, memory and/or performance issues, or any other issues wished to be monitored by the producer. If we’re looking at an SLA perspective, these notifications from the monitoring tools should be able to automate tickets to be logged.

Contact Information:

The Northwestern contact for SolarWinds is Martin Milligan. He would be able to coordinate getting the producer service and SolarWinds talking to each other and may have scripts and/or configurations needed to further monitor advanced functionality.

5. Suggestions and Guidelines for Producers of New Web Services

Key Monitoring Points

When creating a new web service, Monitoring is suggested at the following levels:

Web Services Layered Architecture
Web services can be viewed or described in a protocol stack which consists of four main layers. These layers can each be monitored:

- The Service Transport layer is responsible for transmitting data, primarily messages, between application endpoints. This layer may include: HTTP, SMTP, FTP, etc.
- The XML messaging layer is responsible for encoding messages in a common format in which the Service Transport layer will transmit. XML and SOAP are the two most common.
- The Service Description layer is responsible for providing a public interface where it would describe the certain web service. This is most commonly done via a WSDL.
- The Service Discovery layer is responsible for the centralization and publishing of all active web services able to be consumed. This is currently managed by our service registry, but will soon be replaced by a modern management tool, Apigee.

**Recommendation:**

The Provider is recommended to ensure that the WSDL is published to the service registry and it is active in that registry for consumer services to query your published service. The provider should monitor to ensure that all incoming requests are validated with the correct permissions, data parameters, and are processed in a timely manner. These requests and responses at the messaging layer should be monitored with statistics such as: volume, average response time, types of requests, etc. The provider should also be monitoring at the architecture level to ensure that application as well as web servers are up with some sort of ‘ping’ or notification to ensure availability.

All thresholds should be set high for alerting or notification mechanisms to be triggered. This is to allow capturing of critical information about the service rather than daily false negatives being solicited as email or other types of notifications. As aforementioned, the following thresholds should be set high to alert of useful statistics:

- CPU usage %
- IO
- Average throughput
- Response times

This list is not exhaustive and should be considered on case-by-case basis for niche services.

**Contact Information:**

Providers should be explicit in their service as to what consumers should do and how and/or where to best get in contact with the provider of that service. This is to facilitate any troubleshooting where the producer is an expert. Point of contact, email, phone number, etc. should be provided.

**Use of Apigee:**

Apigee will be the primary platform in which producers will be able to market their service to the appropriate consumer audience. Providers are able to setup an alias for the teams to be notified for
certain triggered events regarding their respective services or services in which their data may be connect to. Apigee will provide a small layer of monitoring and as a proxy to for all producing services. There will be initial Apigee config (a config file) which will need to be setup before the producer service goes “live” for public consumption.

**SLA**

An SLA should also be provided with values given in the agreement taken from usage statistics and analysis from monitoring both in Solar Winds and other tools used. The SLA should outline the following, as an example:

- Service Description
- Reliability given as a percent
- Certain thresholds from request and response times given certain payloads
- How to report a problem
- Average turnaround time for resolving defects a different levels of criticality
- The outline of statistics we as providers monitor and which will be provided to consumers

This list is not exhaustive and should be treated as a guide to setup individual SLA’s. These will vary on a service-by-service basis. This SLA should be included in the Apigee platform serving as a documentation repository.