Monitoring and Alerting Backlog

as of 4/30/2013 delivery to AIR

The following captures the known outstanding functionality left for Monitoring and Alerting as well as any known remaining technical tasks. The schedule for completion of the backlog is not determined. External dependencies have been noted where given as reason for being in the backlog. Please note that the component cannot be considered finished until this remaining backlog is completed. Scheduling for completion of these tasks will need to be coordinated by all involved stakeholders.

Authentication/Authorization

This task remains as an open issue until such time the SSO/Authorization mechanism for Smarter Balanced RFP#11 (SB11) is decided upon and delivered. Based on the following assumptions, the level of effort remaining will be a matter of applying the authentication pattern to the web application that hosts the web services, as well as annotating the web service method with:

```
@Secured(Role="roleGoesHere") org.springframework.security.access.annotation.Secured
```

Assumptions:

- Authentication will be handled by a separate component.
- The authentication method will be supported by Spring Security
- Authentication will be on a component client basis. Example: integrating components will be authenticated and granted authorization (by the security component) in the form of a statically defined role.
- The web service endpoints will be secured (via the @Secured annotation) and configured to be protected by a statically defined role, and improper access will result in a HTTP 401 (unauthenticated) or a HTTP 403 (unauthorized).
- Additional business (non-role) based security rules have not been specified, and will require additional effort if required.

Configuration Component/Service Locator

This is a Smarter Balanced RFP #11 level issue. At the time of authoring the Monitoring and Alerting component, the direction was unclear as to the location and implementation of this service. It was the intent of the development team to design a functional solution for now, that can be refactored in the future. There are several where a configuration component will be needed.

1. **Component lookup**: Currently a property file is used to configure the Monitoring and Alerting webapp with the location of the Monitoring and Alerting rest services it requires. Eventually, there will be a need to discover the component's URL via a service locator component.
2. **Custom configuration**: given the nature of Monitoring and Alerting there are many small

options that must be configured on a component (and even server deployment) level. Please see installation instructions for more details on the property files and individual properties which are required.

Dependency Configuration

Monitoring and Alerting has an external dependencies configurations (MongoDB) that allow for three flexible configuration options. Please see installation instructions for more details on the configurations needed.

1. **Embedded Property File:** Include the configurations on the class path (as resources) in the deployed application. This is least secure, but has the least moving parts. Currently the files are only included as test resources for executing junit tests. The file name is mongo.properties.
2. **Environment Variable Key/Values:** Set the required keys (those that would be contained in the property files) as environment variables. A little more secure, but still not ideal.
3. **External Property file:** Create a secured property file on the server's file system. Pass the location of this file to the application as environment variables. This is most secure, allowing for restricted access to only the process ID the application is running under. It does require a little configuration/coordination at deploy time, but avoids exposing credentials as part of the artifact/runtime environment configuration.

XML vs JSON format

At this time the services were developed to accept and return JSON as the primary data interchange format. Most endpoints also serve XML, but JSON should be used when binding to the services. The format is able to be modified when feedback is received from integrations with other components.

UI--Monitoring and Alerting Webapp

Standard Look and Feel: Given emerging standards regarding look and feel, the Monitoring and Alerting webapp will need to be reskinned. Currently, the webapp utilizes Bootstrap for responsive design (layout) as well as themes (color, button look, etc.).

IE8: There is an outstanding question regarding the requirement for IE8 for internal (non-student facing) user interfaces. Given that Monitoring and Alerting is a non-student facing application, any IE8 related layout issues were deferred pending the decision of whether IE8 will need to be supported.

Deployment

Fault Tolerance

Monitoring and Alerting is required to be a high availability component in the SB11 Architectural Report. The ability to make a component highly available will be extremely dependent upon the deployment environment. The deployment model for the SB11 has not been finalized, and as

such, plans for clustering/and failover can not be finalized for Monitoring and Alerting. Special concerns apply to the Hyperic environment given the HQ server, and the utilization of JMX beans for metric reporting. Depending upon the clustering method chosen, caching for notification processing (against alerts and logs) will have to be implemented.

Hyperic Datastore

The hyperic server utilizes Postgres for it's internal datastore. The default installation of Hyperic includes an embedded version of Postgres. They Hyperic installation instructions specify an external installation of Postgres to support clustering of the Hyperic HQ server. This will need to be addressed when Monitoring and Alerting becomes a high availability component.

Technical Tasks

There are various technical tasks, and investigation projects that were not considered critical path to required functionality. These tasks have been noted in internal DRC task tracking. If the schedule allows and priorities dictate, these items can be scheduled to be worked on.