MUSIC Beijing Release Planning

Overview

Project Name	MUSIC
Target Release Name	Beijing
Project Lifecycle State	Incubation
Participating Company	AT&T, Intel, Netcracker, Verizon, Windriver (in lexical order)

Scope

What is this release trying to address?

This release of MUSIC provides a service with recipes that individual ONAP components and micro-service can use for state replication, consistency management and state ownership across geo-distributed sites. This is a crucial component enabling ONAP components to achieve geo-redundancy (platform-maturity resiliency level 3).

Use Cases

- Targeted goal for R2: OOF-Homing Optimizer (HAS) uses MUSIC for its state persistence (as a queue) and as a highly available distributed messaging service.
- Stretch goal for R2: ONAP Portal will use MUSIC to store its http session state across sites in a persistent manner.

Minimum Viable Product

MUSIC service that can serve the geo-redundancy needs of ONAP HAS and ONAP Portal while satisfying the platform maturity requirements for the Beijing release. For Beijing this will be run as internal services to both Portal and OOF.

Functionalities

Epics

Key	Summary	T	Created	Updated	Due	Assignee	Reporter	P	Status	Resolution
MUSIC-514	Frankfurt release planning milestone		Oct 04, 2019	Oct 15, 2019	Oct 11, 2019	Bharath Balasubramanian	Jim Baker		OPEN	Unresolved
MUSIC-493	Release Candidate 0 Integration and Test		Sep 12, 2019	Oct 14, 2019	Sep 16, 2019	Brendan Tschaen	Jim Baker		IN PROGRESS	Done
MUSIC-483	Code Freeze		Aug 22, 2019	Sep 24, 2019	Sep 06, 2019	Bharath Balasubramanian	Jim Baker		CLOSED	Done
MUSIC-467	Functionality and API Freeze		Aug 05, 2019	Sep 06, 2019	Aug 16, 2019	Bharath Balasubramanian	Jim Baker		CLOSED	Done
MUSIC-449	El Alto release planning milestone		Jul 18, 2019	Jul 31, 2019	Jul 25, 2019	Bharath Balasubramanian	Jim Baker		CLOSED	Done
MUSIC-199	Make MUSIC a true ORM tool replacing all query-like structures with ORM based constructs like spring-data		Nov 23, 2018	Apr 23, 2019		Sandeep Jha	Bharath Balasubramanian		CLOSED	Won't Do
MUSIC-159	MDBC Implementation		Oct 30, 2018	Apr 23, 2019		Arthur Martella	Garima Mullick		CLOSED	Done
MUSIC-90	Adhere to Cassablanca S3P requirements		Jun 27, 2018	Oct 09, 2018		Thomas Nelson	Bharath Balasubramanian		CLOSED	Done
MUSIC-89	Design document for mdbc integration		Jun 27, 2018	Oct 09, 2018		Bharath Balasubramanian	Bharath Balasubramanian		CLOSED	Done
MUSIC-88	Design to make MUSIC a fully sharded, scale out system by replacing Zookeeper with Cassandra		Jun 27, 2018	Oct 09, 2018		Bharath Balasubramanian	Bharath Balasubramanian		CLOSED	Done
MUSIC-86	Enable automated failure detection and consistent failover across sites for ONAP components using MUSIC through the PROM recipe.		Jun 27, 2018	Oct 09, 2018		Thomas Nelson	Bharath Balasubramanian		CLOSED	Done
MUSIC-83	While MUSIC was consumed internally by components in the Beijing release, in Cassablanca we intend to provide MUSIC as an independent multi-site clustered service		Jun 27, 2018	Jan 29, 2019		Thomas Nelson	Bharath Balasubramanian		CLOSED	Done
MUSIC-47	Address the CLM issues		Mar 07, 2018	Aug 08, 2018		Thomas Nelson	Bharath Balasubramanian		CLOSED	Done
MUSIC-20	Check Code and Move to ONAP.		Jan 25, 2018	Aug 08, 2018		Bharath Balasubramanian	Thomas Nelson		CLOSED	Done
MUSIC-19	Music as a Service		Jan 18, 2018	Jul 04, 2019		Bharath Balasubramanian	Thomas Nelson		CLOSED	Won't Do
MUSIC-8	Jenkins		Jan 18, 2018	Aug 08, 2018		Bharath Balasubramanian	Thomas Nelson		CLOSED	Done
MUSIC-7	Enhance and improve Documentation		Jan 18, 2018	Aug 08, 2018		Bharath Balasubramanian	Thomas Nelson		CLOSED	Done

MUSIC-6	Apply Standards to Logging and API Versioning		Jan 18, 2018	Aug 08, 2018		Bharath Balasubramanian	Thomas Nelson		CLOSED	Done
MUSIC-5	Harden Code		Jan 18, 2018	Aug 08, 2018		Bharath Balasubramanian	Thomas Nelson		CLOSED	Done
MUSIC-4	Ensure SSL communication and SQL Injection.		Jan 18, 2018	Aug 08, 2018		Bharath Balasubramanian	Thomas Nelson		CLOSED	Done

Showing 20 out of 21 issues

Stories

Key	Summary	T	Created	Updated	Due	Assignee	Reporter	P	Status	Resolution
MUSIC-481	Sonar fix: Remove commented out lines of code		Aug 12, 2019	Sep 24, 2019		Mamtha Sabesan	Mamtha Sabesan		DELIVERED	Not Done
MUSIC-448	Support postgres in mdbc		Jul 17, 2019	Aug 27, 2019		Bharath Balasubramanian	Bharath Balasubramanian		CLOSED	Done
MUSIC-447	MUSIC API improvements to allow multiple non-blocking reads to improve performance		Jul 17, 2019	Aug 27, 2019		Bharath Balasubramanian	Bharath Balasubramanian		CLOSED	Done
MUSIC-446	Internal retry mechanisms		Jul 17, 2019	Sep 24, 2019		Bharath Balasubramanian	Bharath Balasubramanian		OPEN	Unresolved
MUSIC-445	Keyspace Based logging		Jul 17, 2019	Sep 24, 2019		Bharath Balasubramanian	Bharath Balasubramanian		OPEN	Unresolved
MUSIC-444	AAF CADI Support		Jul 17, 2019	Sep 24, 2019		Bharath Balasubramanian	Bharath Balasubramanian		OPEN	Unresolved
MUSIC-443	MUSIC Control Panel UI based on ONAP Portal SDK		Jul 17, 2019	Sep 24, 2019		Bharath Balasubramanian	Bharath Balasubramanian		OPEN	Unresolved
MUSIC-442	OOM onboarding for MUSIC as a service utilized by OOF-HAS and F-GPS		Jul 17, 2019	Sep 24, 2019		Bharath Balasubramanian	Bharath Balasubramanian		OPEN	Unresolved
MUSIC-441	Provide mdbc latency comparison with postgres /galera/cockroach		Jul 17, 2019	Aug 05, 2019		Bharath Balasubramanian	Bharath Balasubramanian		CLOSED	Done
MUSIC-440	Provide benchmarks comparing MUSIC with cockroachDB		Jul 17, 2019	Aug 05, 2019		Bharath Balasubramanian	Bharath Balasubramanian		CLOSED	Done
MUSIC-439	Update MUSIC helm charts to onboard OOF and Portal using it as a common service		Jul 17, 2019	Sep 24, 2019		Brendan Tschaen	Bharath Balasubramanian		OPEN	Unresolved
MUSIC-438	Address security vulnerabilities that were brought by Fortify and Blackduck		Jul 17, 2019	Sep 19, 2019		Brendan Tschaen	Bharath Balasubramanian		CLOSED	Done
MUSIC-437	Enhance Music with read /write locks		Jul 17, 2019	Aug 05, 2019		Bharath Balasubramanian	Bharath Balasubramanian		CLOSED	Done

MUSIC-436	Create MUSIC ORM layer		Jul 17, 2019	Aug 05, 2019	Bharath Balasubramanian	Bharath Balasubramanian	↑	CLOSED	Done
MUSIC-276	MDBC Eventual Consistency logic		Jan 19, 2019	Apr 23, 2019	Sunder Tattavarada	Bharath Balasubramanian	↑	CLOSED	Done
MUSIC-171	Unit test cases		Nov 01, 2018	Apr 04, 2019	Brendan Tschaen	Garima Mullick	↑	CLOSED	Done
MUSIC-170	MDBC Architecture		Nov 01, 2018	Apr 04, 2019	Brendan Tschaen	Garima Mullick	↑	CLOSED	Done
MUSIC-169	Table Lock: Read/Write Lock Box		Oct 31, 2018	Apr 24, 2019	Arthur Martella	Garima Mullick	↑	CLOSED	Done
MUSIC-168	Table Lock: Write ownership		Oct 31, 2018	Apr 23, 2019	Brendan Tschaen	Garima Mullick	↑	CLOSED	Duplicate
MUSIC-167	Table Lock: Read ownership		Oct 31, 2018	Apr 23, 2019	Brendan Tschaen	Garima Mullick	↑	CLOSED	Duplicate

Showing 20 out of 62 issues

Longer term roadmap

In the long term we hope that MUSIC will be common, shared state-management system for all ONAP components and micro-services to manage geo-redundancy. For example, we envisage the use of MUSIC for multi-site state management in SO (to store Camunda state across sites), <SDN-C, AppC> (to store ODL related state across sites) , A&AI (to store its graph data) and most other ONAP components that need to manage state across sites. Further, we envision that these services will use the MUSIC recipes (mdbc, prom, musicCAS, musicQ) to achieve the goal of a multi-site active-active federated ONAP solution.

Release Deliverables

Indicate the outcome (Executable, Source Code, Library, API description, Tool, Documentation, Release Note...) of this release.

Deliverable Name	Deliverable Description
Source code and REST API	The entire source code for the MUSIC service and the corresponding REST API to access it.
Compilation scripts	Script to generate the MUSIC war file that can be deployed in the Apache Tomcat webserver
Installation guide	Document that describes how MUSIC can be installed in containers
Tool description	Complete description of the inner workings of MUSIC and how it performs state management
Basic Benchmarks	Basic performance benchmarks for the MUSIC operations
API documentation	REST API documentation in Swagger
Test Cases	Junit test cases covering sufficient parts of MUSIC code

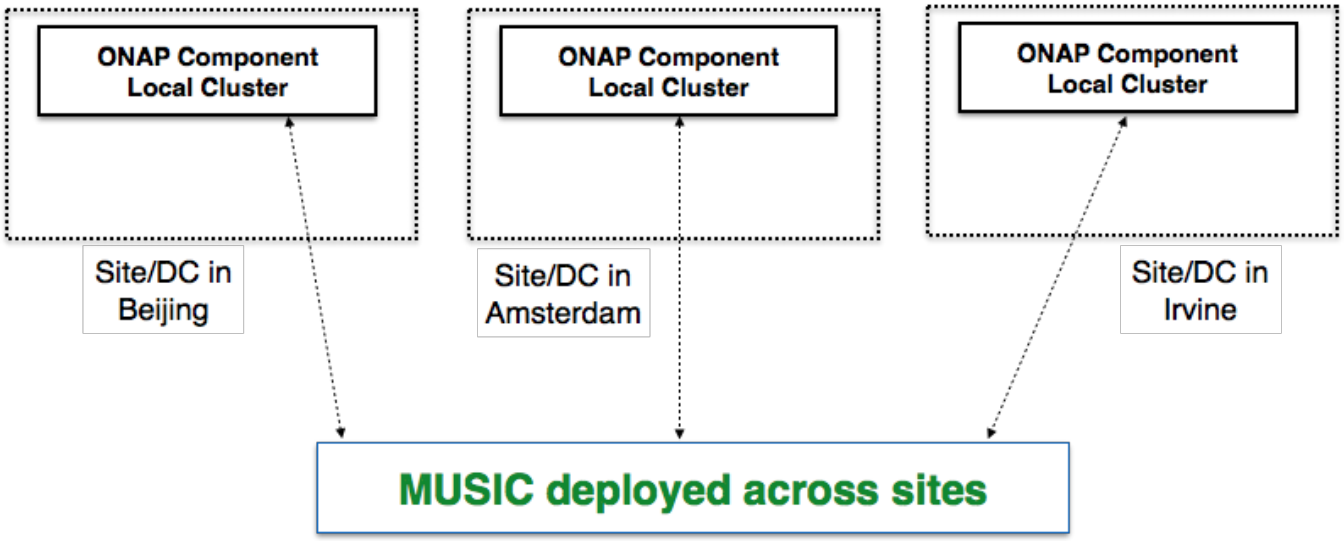
Sub-Components

NA.

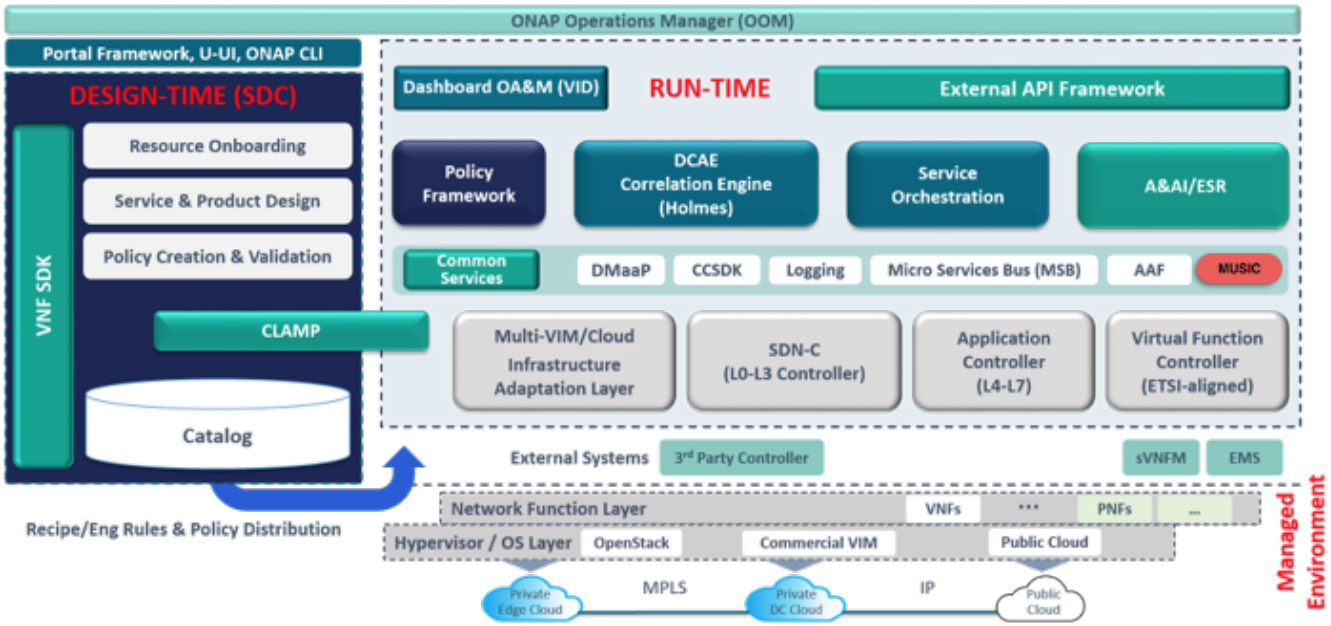
Architecture

High level architecture diagram

In this figure the ONAP components targeted for R2 are ONAP HAS and Portal (stretch goal).



MUSIC will be available as a common service like DMaap or AAF as shown in the red, oblong box below:



Platform Maturity

Refering to CII Badging Security Program and Platform Maturity Requirements, fill out the table below by indicating the actual level , the targeted level for the current release and the evidences on how you plan to achieve the targeted level.

Area	Actual Level	Targeted Level for current Release	How, Evidences	Comments
Performance	1	1	This file shows basic performance benchmarks performed for MUSIC on a 10 node cluster.	• 0 -- none • 1 – baseline performance criteria identified and measured • 2 & 3 – performance improvement plans created & implemented
Stability	1	1	As shown in this file, our experimental runs were all over 1 hour.	• 0 – none • 1 – 72 hours component level soak w/random transactions • 2 – 72 hours platform level soak w/random transactions • 3 – 6 months track record of reduced defect rate
Resiliency	2	2	Within each container we have scripts that will detect failure of MUSIC and restart it. However, if the entire container fails, we will need OOM to bring it up.	• 0 – none • 1 – manual failure and recovery (< 30 minutes) • 2 – automated detection and recovery (single site) • 3 – automated detection and recovery (geo redundancy)
Security	2	2	▪ SSL Communication's between Cassandra Cluster Nodes. ▪ REST over HTTPS with AAF for Authentication.	• 0 – none • 1 – CII Passing badge + 50% Test Coverage • 2 – CII Silver badge; internal communication encrypted; role-based access control and authorization for all calls • 3 – CII Gold
Scalability	1	1	Among the MUSIC components [tomcat, zookeeper, cassandra], new MUSIC nodes with the tomcat and cassandra can be added seamlessly to scale the cluster (MUSIC itself is state-less). Zookeeper nodes ideally should not be scaled since there are major performance implications. However, this can be done with reconfiguration.	• 0 – no ability to scale • 1 – single site horizontal scaling • 2 – geographic scaling • 3 – scaling across multiple ONAP instances
Manageability	1	1	Using EELF with logback as the logging provider.	• 1 – single logging system across components; instantiation in < 1 hour • 2 – ability to upgrade a single component; tracing across components; externalized configuration management
Usability	1	1	Use SWAGGER for the REST API and Installation Docs. Will need to enhance and update the documentation.	• 1 – user guide; deployment documentation; API documentation • 2 – UI consistency; usability testing; tutorial documentation

- API Incoming Dependencies

NA.

- API Outgoing Dependencies

API this project is delivering to other projects.

API Name	API Description	API Definition Date	API Delivery date	API Definition link (i.e. swagger)
MUSIC API	The REST API used to store state and manage access to it through a locking service.	TBD.	TBD.	Waiting for project approval.

- Third Party Products Dependencies

Third Party Products mean products that are mandatory to provide services for your components. Development of new functionality in third party product may or not be expected.
List the Third Party Products (OpenStack, ODL, RabbitMQ, ElasticSearch,Crystal Reports, ...).

Name	Description	Version
Cassandra	Higly-available key-value store that will maintain state. http://cassandra.apache.org/	3.2
Zookeeper	Distributed coordination service used to provide the locking service for MUSIC. https://zookeeper.apache.org/	3.4.6
Tomcat	Web-server that will host the MUSIC code and support the REST API. http://tomcat.apache.org/	9

Testing and Integration Plans

The following type of tests will be ensured for MUSIC in this release:

- Unit tests: Junit test cases will be added incrementally as part of code delivery ensuring static code analysis as recommended by common SW delivery practices. MUSIC is written in Java, so coverage target percentage will be evaluated along project. Unit Tests will be automated (as part of the build), so that execution will happen before every MUSIC delivery. This includes writing test cases for the new components and those for the existing ones not yet covered under unit testing. A python based Unit Testing framework will be integrated into the project, binding all developers to start with Unit Tests when developing each module.
- Functional Test cases: MUSIC will be tested according to the functionality committed to the Beijing release with a specific focus on the functionality utilised by ONAP HAS and ONAP Portal for MUSIC-based state management.
- End to End Test cases: MUSIC will be integrated within the whole ONAP components architecture and dedicated set of E2E test cases will be setup. Purpose, scope and details on E2E testing will be planned/defined subsequently by working with the Integration team, and for the test cases in the scope of this release.

Gaps

None identified so far.

Known Defects and Issues

None identified so far.

Risks

None identified so far.

Resources

Udated the Resources Committed to the Release centralized page.

- ## Release Milestone

The milestones are defined at the Release Level and all the supporting project agreed to comply with these dates.

- ## Team Internal Milestone

This section is optional and may be used to document internal milestones within a project team or multiple project teams. For instance, in the case the team has made agreement with other team to deliver some artifacts on a certain date that are not in the release milestone, it is erecommended to provide these agreements and dates in this section.

It is not expected to have a detailed project plan.

Date	Project	Deliverable
To fill out	To fill out	To fill out

- ## Documentation, Training

- Highlight the team contributions to the specific document related to he project (Config guide, installation guide...).
- Highlight the team contributions to the overall Release Documentation and training asset
- High level list of documentation, training and tutorials necessary to understand the release capabilities, configuration and operation.
- Documentation includes items such as:
    - Installation instructions
    - Configuration instructions
    - Developer guide
    - End User guide
    - Admin guide
    - ...

Note

The Documentation project will provide the Documentation Tool Chain to edit, configure, store and publish all Documentation asset.

Other Information

- ## Vendor Neutral

If this project is coming from an existing proprietary codebase, ensure that all proprietary trademarks, logos, product names, etc. have been removed. All ONAP deliverables must comply with this rule and be agnostic of any proprietary symbols.

- ## Free and Open Source Software

FOSS activities are critical to the delivery of the whole ONAP initiative. The information may not be fully available at Release Planning, however to avoid late refactoring, it is critical to accomplish this task as early as possible.
List all third party Free and Open Source Software used within the release and provide License type (BSD, MIT, Apache, GNU GPL,...).
In the case non Apache License are found inform immediately the TSC and the Release Manager and document your reasoning on why you believe we can use a non Apache version 2 license.

Each project must edit its project table available at Project FOSS.

Charter Compliance

The project team comply with the ONAP Charter.