ONAP-OF Project Proposal (6/20/17)

Project Name:

- Proposed name for the project: `ONAP Optimization Framework (ONAP-OF)`
- Proposed name for the repository: `optf`

Subrepositories:

optf/has	-- homing and allocation service
optf/cmso	-- change management scheduling service
optf/osdf	-- optimization service design framework

Project description:

ONAP needs core platform optimization services such as VNF placement and resource allocation (homing), and change management scheduling to function in any multi-site, multi-VIM, and multi-service environment. It could also benefit from a framework which promotes the reuse of software tools and algorithms to allow users to construct new optimization services and to extend/enhance existing platform optimization services.

This project currently provides the following two core platform optimization services, which are built to be service independent, policy driven, and extensible along with an optimization framework to enhance these or creating new services.

a) HAS (Homing and Allocation Service) optimizer: a policy driven service placement and resource allocation service to allow deployment of services and VNFs on a multi-site, multi-VIM infrastructure. This service performs function similar to classic OS schedulers or OpenStack scheduler. The role of the HAS is to select which cloud and which sites the elements of a service should be placed in, while respecting service constraints (latency, availability of specific platform features) as well as platform needs (cost).

b) CMSO (Change management scheduling optimizer): a policy driven workflow schedule optimizer for change management planning. The CMSO helps schedule workflows in time to maximize parallel change management activities, while respecting dependency between the workflows.

This will be delivered as three modules. One for HAS, one for CMSO, and one for the service design framework. The HAS and CMSO can execute both as services on DCAE and independent processes.

The set of platform optimization services will grow over time as the ONAP platform needs arise, and the optimization framework is envisioned to handle this as effectively as possible, with minimal or little new code development for creating new services. The optimization service design framework (OSDF), which can be used to build new optimization applications for users of ONAP, as well as to build new platform optimization services or extend the existing platform services through plugins. To demonstrate its capabilities, OSDF has been used to entirely build the change management scheduling optimizer (CMSO) as well as to build VNF license optimization and connectivity optimization plugins for the homing and allocation service (HAS). The OSDF is intended to allow future applications such as energy optimization in networks, optimal route selection, and radio access network (RAN) runtime performance optimization.

We will describe the current platform optimization services and optimization framework with their architectural fit one by one.

HAS: policy driven service homing and resource allocation on a multi-site, multi-VIM infrastructure. Homing/placement and allocation of resources is one of the fundamental requirements of provisioning a service over the cloud (or even non-cloud) infrastructure. HAS allows designers of services/VNFs to specify their service-specific placement requirements using policy constraints (e.g., geo-redundancy requirements for disaster recovery) and objective functions (e.g., minimize latency) linked to the service model. Then, at service deployment time, HAS collects information from AAI, DCAE, and other sources to determine a placement solution that meets service constraints while considering both the service objective function and the service provider preferences (e.g., cost) and constraints (e.g., available of capacity). Once a placement decision is made, a resource allocation (reservation) decision can be registered in AAI or with the resource manager for the resource if necessary.

HAS can home a request either to a cloud site where new virtual resources are to be created or to an existing service instance. When the services deployed become more complex (e.g., multiple VNFs with different constraints for individual VNFs and the combinations of VNFs) and the cloud infrastructure is large (e.g., dozens or more possible sites), such capability is essential for managing the services and the infrastructure.

HAS will be designed to be used as a building block for both initial deployment, as well as runtime redeployment due to failures or runtime-capacity increase (scale-out). It will be designed to be usable for all platform placement functions, including placements of VMs, containers (e.g., for DCAE micro-services), or VNF specific resources. A plugin model will be provided to allow placements of additional resource types such as licenses, VNF resources. Plugin models will also allow extension by adding new constraint types, optimizer types, and objective functions.

Architecture alignment:

- SO invokes HAS to get a placement and license allocation decision when deploying a new service, or when it is called to redeploy a service upon site failure, or upon increasing the capacity of an already existing service. This is particularly useful in multi-site or multi-VIM environments.

- VF-C/App-C may need to invoke HAS to get a placement decision if an existing VNF must be rebuilt due to failure or increase in capacity.

- OOM may need to invoke HAS to get a placement decision when deploying ONAP components e.g., to get a DCAE micro-service to be placed in proximity to the VNF it is monitoring.

- Policy may need to support HAS by storing placement policies and associating them with service models

- HAS uses information stored in AAI (e.g., available inventory), DCAE (e.g., performance, utilization), SDN-C and other ONAP components to make placement decisions.

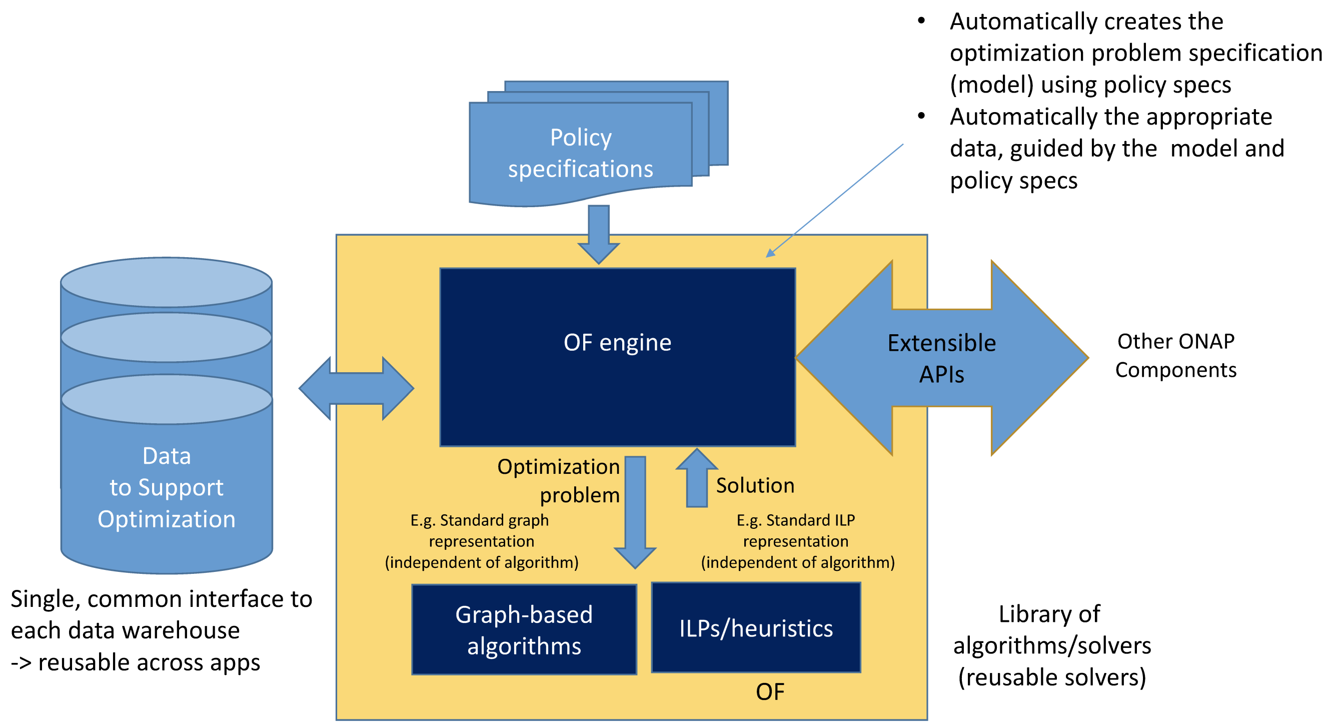
- Multi-VIM: HAS allows placement constraints to be specified that drives workloads to different cloud providers when appropriate (e.g., VNF requires some specific cloud platform) or desired (e.g., VNF requires certain level of reliability or performance that only some cloud providers can meet).

CMSO: VNF change management scheduling optimizer: Change Management (CM) application is responsible for managing and enforcing changes (e. g. device upgrade, configuration change, etc.) in the cloud and network infrastructure. Currently, a major part of CM scheduling is performed manually, which is time consuming, inefficient, and prone to service impacting errors. CMSO provides recommended schedules of changes for upgrading of VNFs under given constraints and current state of schedules and relationships of network elements. The primary challenge is when to schedule changes such that service disruption is minimized. OF offers the CMSO service to the CM application, which can be invoked prior to any change are scheduled. A service designer designs a change request in SDC and configures the schedule requirements through policies. Prior to scheduling changes via Service Orchestrator (SO), the designer makes a call to CMSO from SDC. CMSO collects the existing scheduling information from available ticketing system and vertical dependency information from AAI and calculates a solution to the scheduling application. Finally, the recommended schedule is returned to SDC, which is verified by the designer before committing the schedules to SO.

Architecture alignment:

- AAI (e.g. network topology, cloud sites, service instances, scheduling/ticketing data)
- DCAE (e.g. cloud-level resource utilization)
- SDC (e.g. available VNF license artifacts)
- Policy Service (e.g. rules/constraints)

Optimization Service Design Framework. It a set of design time optimization libraries, tools and microservices (MS) to facilitate and simplify the creation of new specific runtime optimization functionalities. The goal of this framework is to avoid siloed optimization tools and associated duplicated efforts and overheads. Indeed, the current platform services HAS and CMSO use the framework extensively in their own development. Other potential optimization services that can be built using this framework include energy optimization in networks, optimal route selection for various network services, and radio access network (RAN) performance optimization. The figure below illustrates the concept.


• Automatically creates the optimization problem specification (model) using policy specs
• Automatically the appropriate data, guided by the model and policy specs
Policy specifications
OF engine
Extensible APIs
Other ONAP Components
Data to Support Optimization
Optimization problem
E.g. Standard graph representation (independent of algorithm)
Solution
E.g. Standard ILP representation (independent of algorithm)
Graph-based algorithms
ILPs/heuristics
OF
Single, common interface to each data warehouse -> reusable across apps
Library of algorithms/solvers (reusable solvers)

Architecture alignment:

• How does this framework fit into the rest of the ONAP Architecture?

- Offers a set of MS which provide reusable optimization functionality which can optionally be used by other ONAP components if required by a use case
- Provides a framework for building optimization services as a part of the ONAP ecosystem
- Provides adapters to other ONAP systems (e.g. policy, AAI, SDC, etc.) for optimization application developers
- Uses REST and Data Bus interfaces in a service agnostic manner
- Models and artifacts are specified in SDC format, while rules/constraints are specified in Policy Service

• What other ONAP projects does this project depend on?

- AAI (e.g. network topology, cloud sites, service instances, scheduling/ticketing data)
- DCAE (e.g. cloud-level resource utilization)
- SDN-C (e.g. network utilization, available capacity in a VNF instance)
- SDC (e.g. available VNF license artifacts)
- Policy Service (e.g. rules/constraints)

• How does this align with external standards/specifications?

N/A

• Are there dependencies with other open source projects?

- Open source optimization solvers (e.g. GLPK/CBC, OR-Tools, optaplanner – these are pluggable)
- Python eco-system (modules for schema validation, database adapters, etc.)
- While the code is primarily written in Python, we anticipate it to be a mixture of Java and Python

Resources:

- Primary Contact Person
  - Sarat Puthenpura - AT&T
- Names, gerrit IDs, and company affiliations of the committers
  - Sastry Isukapalli - AT&T
  - Sarat Puthenpura - AT&T
  - Shankaranarayanan Puzhavakath Narayanan - AT&T
  - Maopeng Zhang - ZTE
  - Yoram Zini - Cisco

- Names and affiliations of any other contributor
  - Ankit Patel - AT&T
  - Avteet Chayal - AT&T
  - Matti Hiltunen - AT&T
  - Joe D'Andrea - AT&T
  - Rúben Borralho - Celfinet
  - Mark Volovic - Amdocs
  - Manoj K Nair - Netcracker manoj.k.nair@netcracker.com
  - Alexander Vul - Intel
  - Rakesh Sinha - AT&T
  - Max Zhang - AT&T
  - Carlos De Andrade - AT&T
  - Kevin Smokowski - AT&T
  - Ramki Krishnan - VMware
  - Gil Hellmann - Wind River
  - Ikram Ikramullah - AT&T
  - Dileep Ranganathan - Intel
  - Gueyoung Jung - AT&T
- Project Roles (include RACI chart, if applicable)

Other Information:

- link to seed code (if applicable)
- Vendor Neutral
  - if the proposal is coming from an existing proprietary codebase, have you ensured that all proprietary trademarks, logos, product names, etc., have been removed?
- Meets Board policy (including IPR)

Use the above information to create a key project facts section on your project page

Key Project Facts

Project Name:

- JIRA project name: Optimization Framewok
- JIRA project prefix: OPTFRA

Repo name: optf
Lifecycle State:
Primary Contact: Sarat Puthenpura (sarat@research.att.com)
Project Lead: Sarat Puthenpura
mailing list tag [Should match Jira Project Prefix]:

Committers:

Sarat Puthenpura sarat@research.att.com AT&T
Sastry Isukapalli sastry@research.att.com AT&T
Shankaranarayanan Puzhavakath Narayanan - snarayanan@research.att.com AT&T

Maopeng Zhang zhang.maopeng1@zte.com.cn ZTE
Alexander Vul alex.vul@intel.com Intel
Yoram Zini (yzini) yzini@cisco.com Cisco

Vladimir Yanover mailto:vyanover@cisco.com Cisco

*Link to TSC approval:
Link to approval of additional submitters: