# Smart Sceduler (Solver Scheduler): Complex constraint based resource placement

Yathiraj Udupi (yudupi@cisco.com), Debo Dutta(dedutta@cisco.com)

This blueprint proposes a new Nova Scheduler Driver that provides an extensible mechanism for making smarter resource placement decisions (i.e., scheduling decisions) by modeling the placement request as a constraint optimization problem.

The existing Nova FilterScheduler schedules instances to hosts based on certain filter criteria such as available disk space, ram, cpu cores, etc. The filters are implemented in python and are run sequentially to filter out the hosts, and then eventually sorted based on the supported weight functions. The filter and weight based solutions currently supported by the FilterScheduler are capable of handling simple constraints. However for complex constraints, building a Filter or a Weight function might be very complex.

Tenants can have complex business rules and policies that govern the data center resources, and the resource placement decisions should satisfy the complex constraints. For example, tenants may expect all the storage to reside locally where the compute is, or may expect to minimize the network bandwidth usage. Tenant policies may also request to minimize the distance between VMs, VMs and storage, etc that also rely on the network topology for making resource placement decisions. These complex constraints can cover metrics not just limited to compute (from Nova), but can also include storage (Cinder), and network metrics.

Complex constraint solvers and frameworks such as PULP, CVXOPT, COIN-OR are available in open sources and have fast implementations in C, C++. Hence a natural solution to the above problem is to design a pluggable scheduler that leverages existing solvers. We believe that this will open new avenues for complex constraint (and objectives) based resource placement in large OpenStack deployments.

Arriving at smart, global, and optimal resource placement decisions in Openstack covering multiple Openstack services is a big effort and a single blueprint is not sufficient to get there. This blueprint is an initial effort in this direction and provides a new way of scheduling compute resources in Nova.

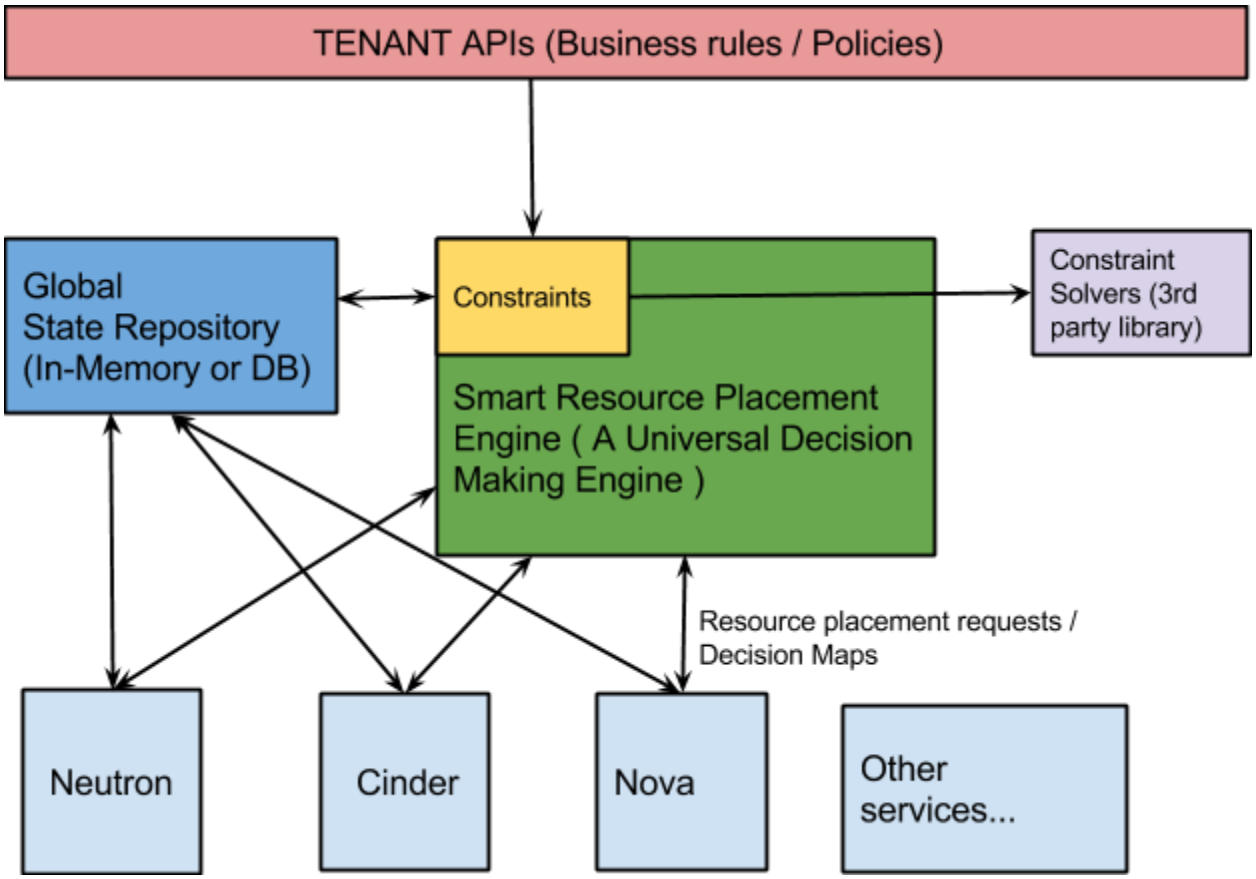
For the big picture, we feel the following solutions are needed:

- A smart resource placement decision making engine that is universally applicable for

all kinds of resource placement decisions, and can communicate with all the openstack services.

- A support for handling business rules and policies of the tenants that eventually translate to the complex constraints and feed into the decision making engine.
- All services need to be able to communicate with the universal decision making engine to request for decisions and get decision results.
- A resource state repository for near-real-time updated states of all resources.

A high-level architecture diagram is presented below that addresses this idea of smart resource placement and applicable not just to the Nova service, but also other Openstack services such as Cinder and Neutron.




## Solver Scheduler: A Smart Resource Placement Engine

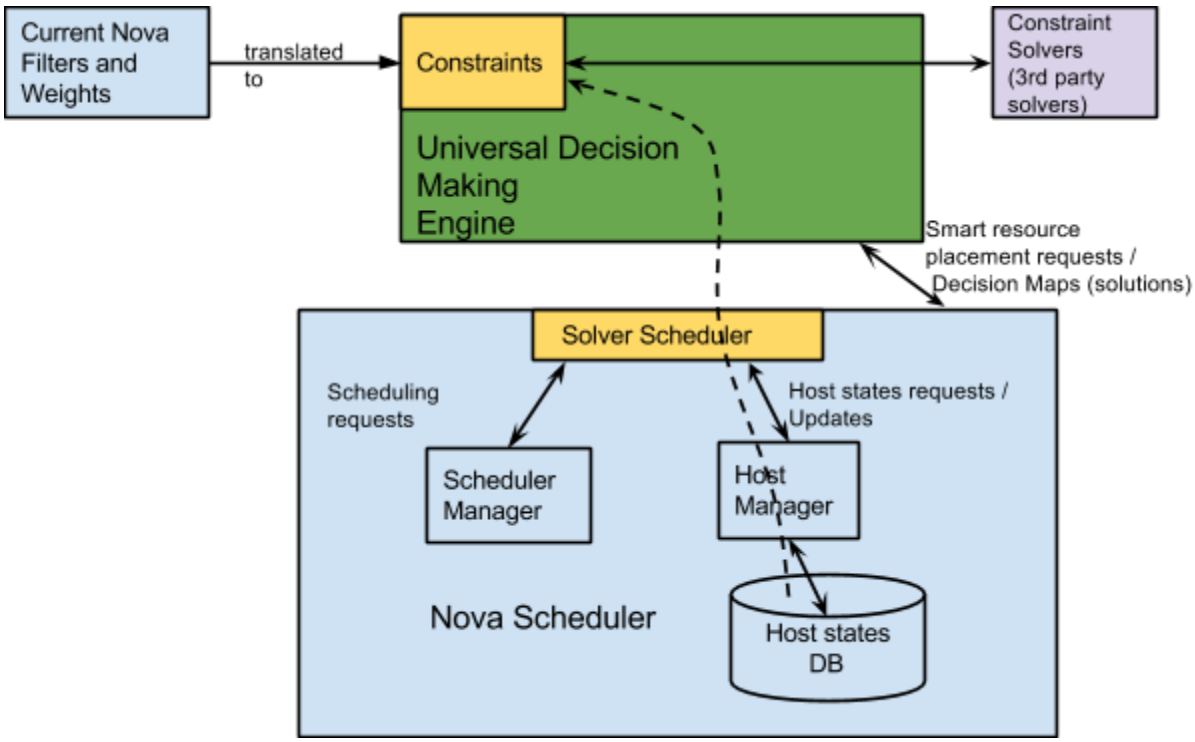
Our focus is on this module and the key purpose of this component is to solve for constraints and make resource placement decisions. The existing openstack services will plugin to this module for getting placement decisions while scheduling resources. We are first addressing this within Nova, and presenting Solver Scheduler as a pluggable complex constraint based

placement engine for scheduling compute resources.

Solver Scheduler provides a way to solve for resource placement by modeling the request as a constraint optimization problem. In this approach, we are working with a set of constraints that have to be satisfied, and if there are multiple options that are feasible, our goal is to find the best matching solution that optimizes a cost metric (could be a minimization metric or a maximization metric).

**Key Design Points:**

- ***Not disruptive, and works with the current Nova architecture:*** The key aspect of this solution is that it is not disruptive of the existing Nova scheduler. This constraint based Solver scheduler fits into the existing Nova scheduling process utilizing the host states from DB. The requests to provision VMs can be made to be routed (by changing the scheduler driver configuration) via the Solver scheduler, as opposed to the Filter scheduler. The existing Filters can be implemented as constraints and fed to the solving engine, and the weights can be listed as a minimization or maximization cost metric, and the solver will compute the whole problem providing a decision map with instance, host tuples. This will be used by the Nova Scheduler component to complete the VM provisioning.



- ***Using pluggable Solvers:*** As part of the blueprint, we have provided a LP(Linear programming)-based solver implementation for the placement engine. However our architecture will support pluggable solvers, and hence allowing for other implementations.

- ***Using pluggable constraints and costs:*** Our architecture allows for plugging in various constraints and costs, and the main solver will combine all of the constraints and costs defined while making scheduling decisions. This allows for translating high level requirements or policies into constraints and costs to be used by the constraint solver.

- ***Can consume resources from across services:*** Individual constraints or costs defined can consume metrics from other Openstack services. This can be achieved by using the respective clients provided for other services such as cinderclient, ceilometer client etc.

References:

1. Solver Scheduler: Complex constraint based Resource Placement - Blueprint
https://blueprints.launchpad.net/nova/+spec/solver-scheduler
2. Unified Constraints-based Smart resource placement