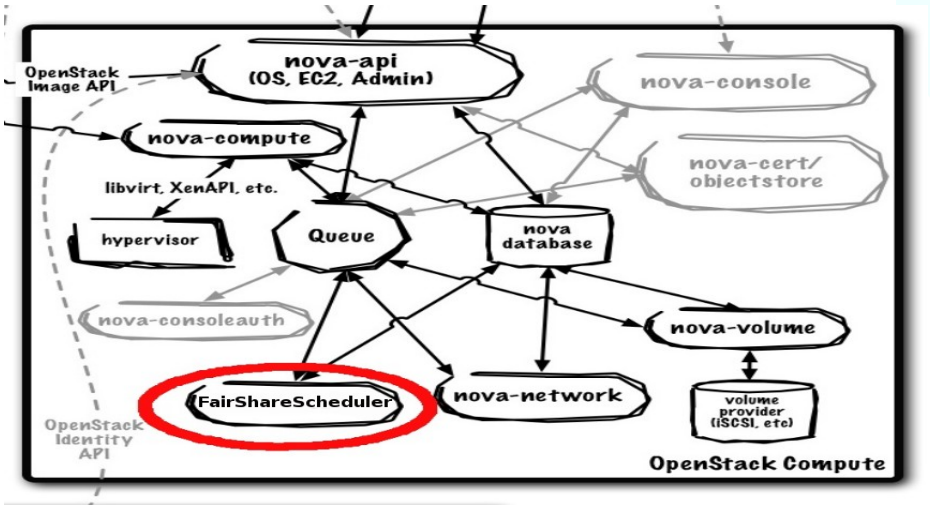
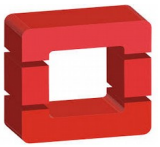
CLOUD AREA PADOVANA



Lisa Zangrando
INFN Padova

FairShare Scheduler update

Partitioning model in OpenStack



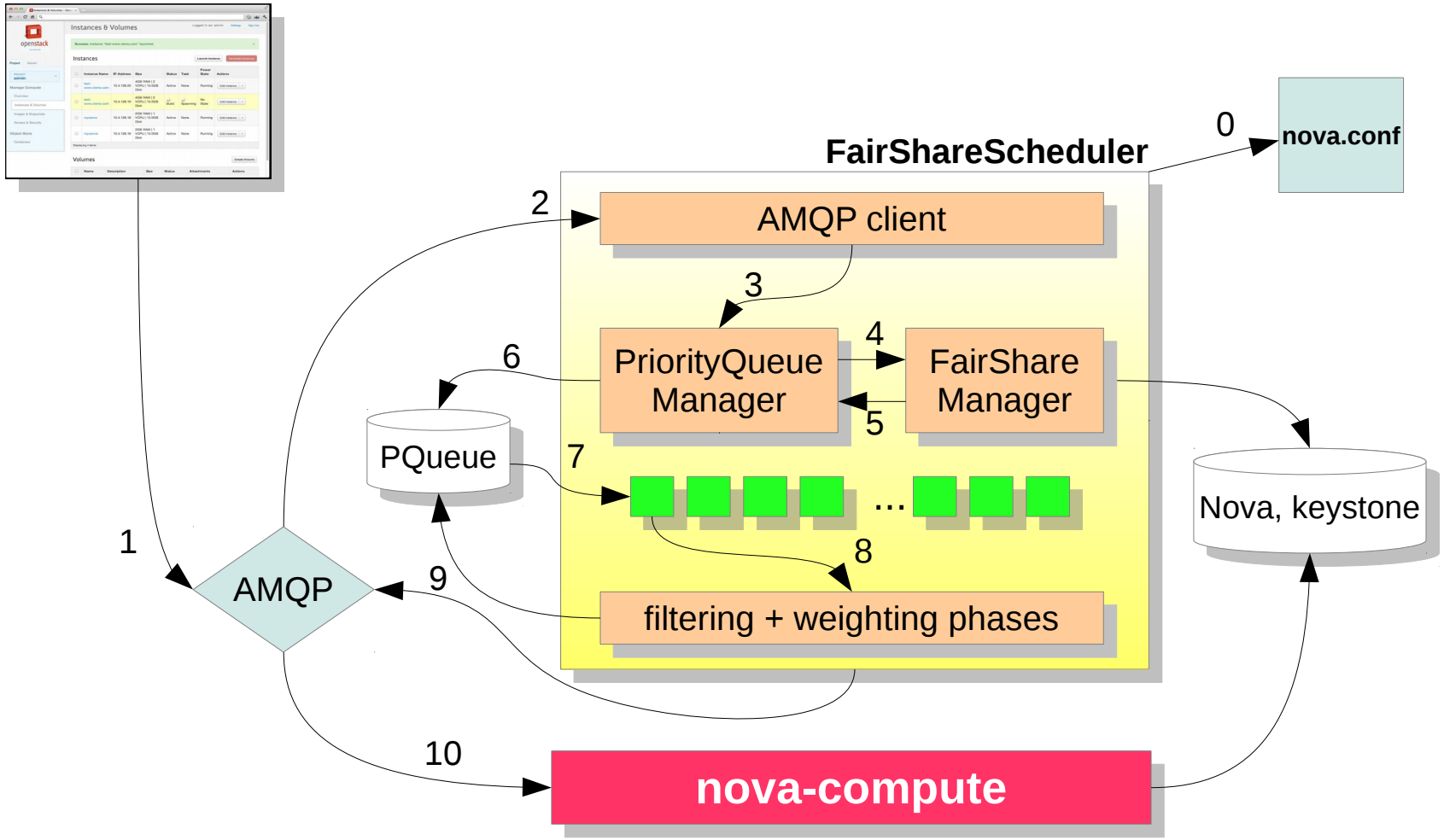
- Currently OpenStack allows just the STATIC partitioning model
  - low global efficiency and increased cost in terms of resource usage
  - it doesn't allow continuous full utilization of all available resources
  - in a scenario of full resource usage for a specific project, new requests are simply rejected

- We (INFN-PD) started to address the problem by developing a pluggable scheduler, named FairShareScheduler, as extension of the current OpenStack scheduler (i.e FilterScheduler)

- FairShareScheduler provides:
  - queuing mechanism for handling the user requests
  - fair-share algorithm based on the SLURM Priority MultiFactor strategy

Lisa Zangrando
<lisa.zangrando@pd.infn.it>

FairShareScheduler: the high level architecture



Lisa Zangrando
<lisa.zangrando@pd.infn.it>

Integration in OpenStack

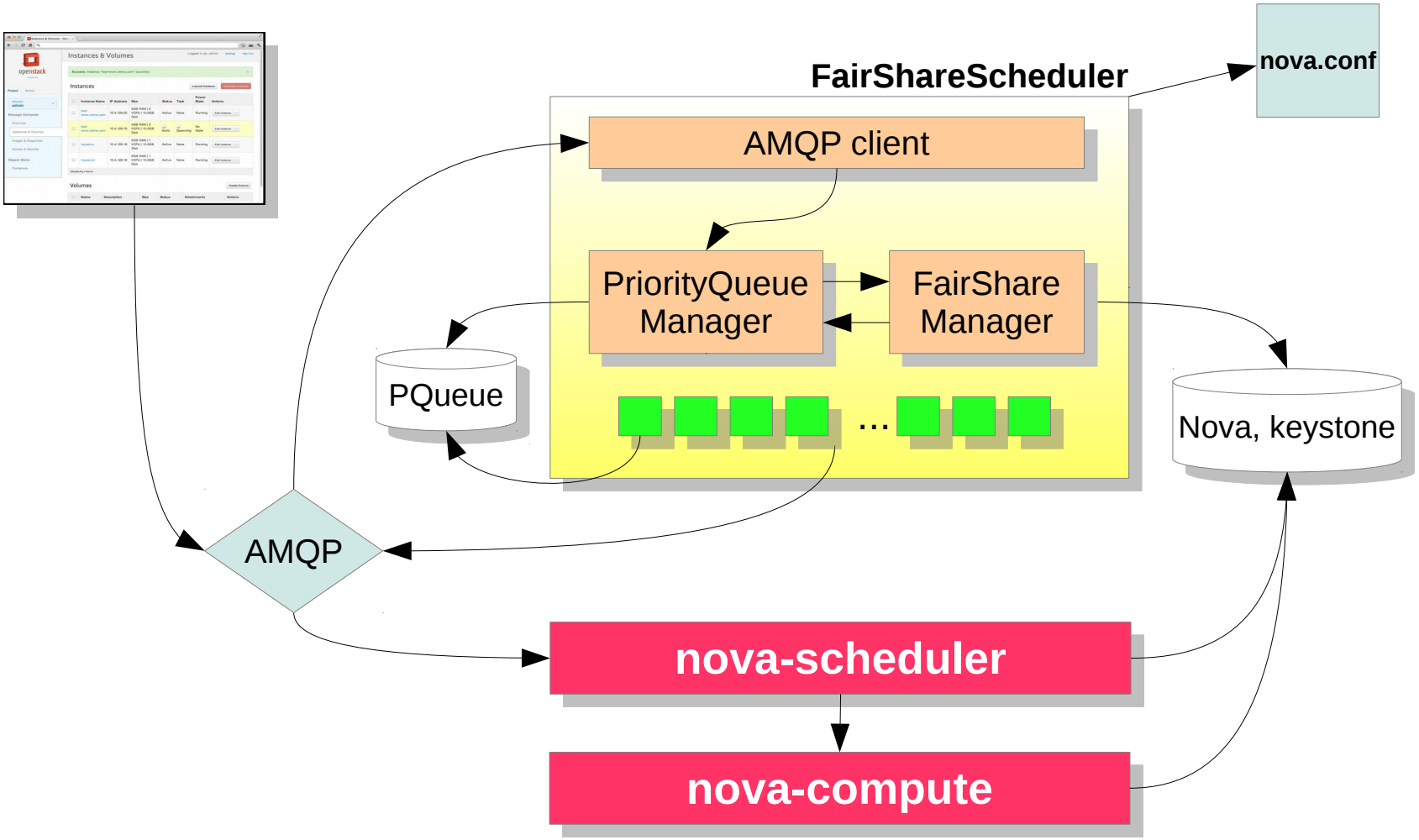
- Adapting the FairShareScheduler every time a new OpenStack version is released is not feasible
  - the goal is to integrate our work in the official distribution

- Interacted with OpenStack developers (NOVA, GANTT and BLAZAR teams)
  - this was not easy, e.g. because they have problems understanding our use case
  - moreover the future of some of these projects is not very clear

- At the OpenStack Summit 2014 (Paris) the Nova-Scheduler leader suggested us to consider the FairShareScheduler as an external manager which interacts with Nova-Scheduler
  - this approach should simplify the integration process
  - architecture redesign and redevelopment work required
  - new stackforge project creation

Lisa Zangrando
<lisa.zangrando@pd.infn.it>

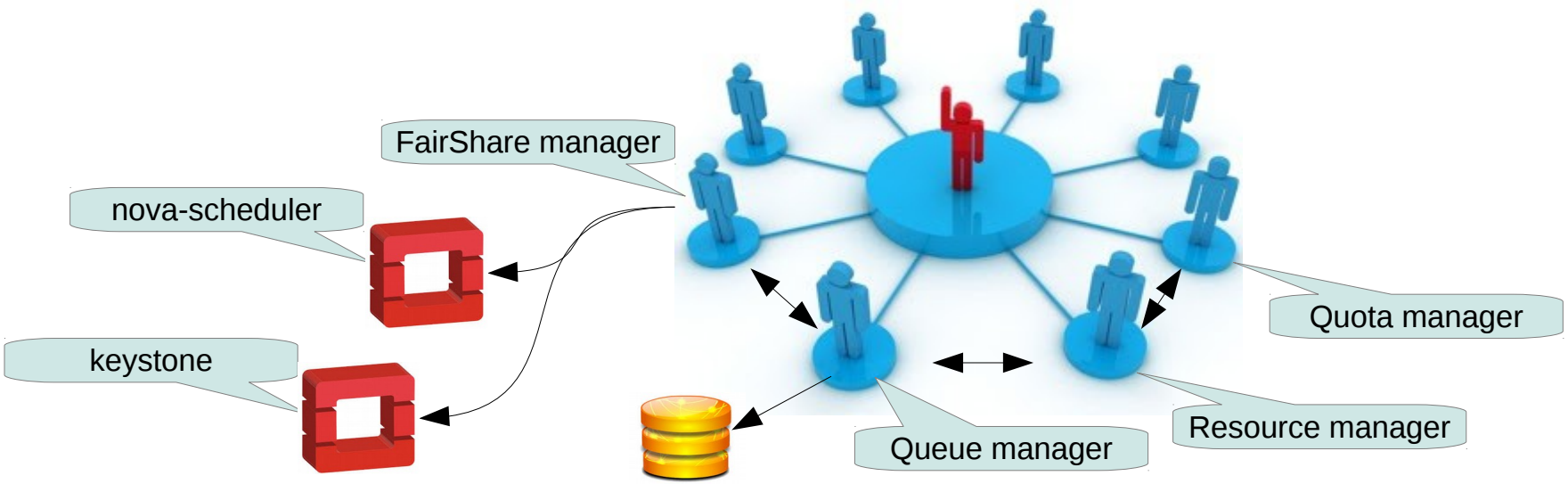
FairShareScheduler: architecture redesign



Lisa Zangrando
<lisa.zangrando@pd.infn.it>

The evolution

- The possibility of designing the FairShareScheduler as indipendent project allows us to evolve it a little bit more as a new OpenStack service which handles pluggable managers
  - APIs for interacting with the service and the managers
  - a manager is a specific and independent task executed periodically or interactively
    - you can implement your manager by using the provided API
    - one manager will provide the same capabilities of the current FairShareScheduler



Lisa Zangrando
<lisa.zangrando@pd.infn.it>

The manager interface

```
class Manager(Thread):
    def getName(self): # return the manager name
    def getStatus(self): # return the manager status
    def isAutoStart(self): # is AutoStart enabled or disabled?
    def setup(self): # allows custom initialization
    def destroy(self): # invoked before to destroy the manager
    def execute(self, cmd): # executes user command synchronously
    def task(self): # executed periodically at fixed rate
```


Lisa Zangrando
<lisa.zangrando@pd.infn.it>

The evolution: status

- Development started (IceHouse, Juno)
  - foreseen two months of work for the first prototype
- New project at StackForge as soon the prototype is ready
  - interaction with other OpenStack projects not required
- StackForge provides a place for OpenStack contributors to create and maintain unofficial projects using the same tools and procedures as the official ones
  - it includes: Gerrit code review, Jenkins CI, GitHub repository, IRC...
  - it is a good first step for exposing new projects but doesn't guarantee eventual OpenStack incubation and integration
  - the new project must be self sufficient
- Incubation and integration phases need the approval of the OpenStack reviewers
- University of Victoria available to help with testing

Lisa Zangrando
<lisa.zangrando@pd.infn.it>

Questions?

Lisa Zangrando
<lisa.zangrando@pd.infn.it>