



# Schedconfig and AGIS Evolution

Alden Stradling  
ATLAS Software Week  
1 December 2010

# Schedconfig Headaches

- #1 – ToA
  - Legacy parsing code that I brought over as one block – takes minutes to run
- #2 – BDII
  - Rewritten parsing code, quicker. BDII, however, has inherent lag.
  - Problematic release updating, etc (correlates to the specific gatekeeper, not the site)



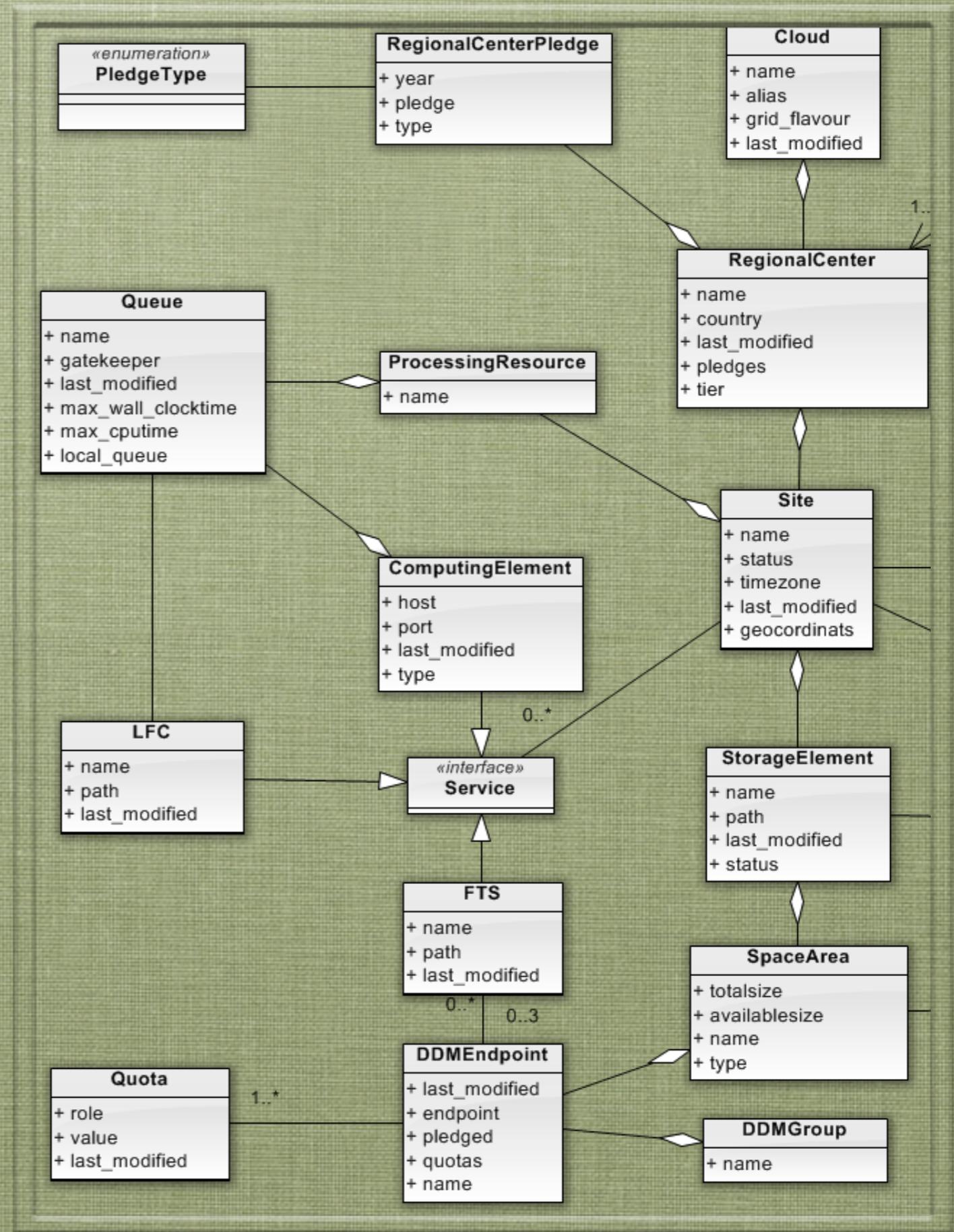
# AGIS to the Rescue

- AGIS proposes to (or already) provides:
  - Tiers of Atlas information in pre-parsed format
  - BDII-type information (gatekeeper, jobmanager, queue, site, region) in a pre-parsed format
  - *Custom methods in the API!* Was quite pleased to hear this in discussion with Roman – rather than writing complex queries against DB, just request the method I need



# AGIS diagram excerpt

The fundamental unit: site (contrast to the basis of queue in schedconfig)



# Concepts to Adapt

- Basic unit of schedconfig is the *queue*.
  - Everything else (site, siteid, cloud) is metadata to the queue.
- Basic unit of AGIS is the site, then the processing resource, then the queue.
  - Not a problem – custom API calls (like methods to get site, DDMendpoint, gatekeeper, quotas, etc. per queue) will bridge the gap neatly



# Advantages

- AGIS + direct release updates to *installedsw* table
  - Allows much more frequent updates, O(5 min)
  - Software status is updated without lag (no more dead jobs)
- Much cleaner code, much simpler debugging
  - Figuring out why something is/is not going into schedconfig via BDII is (at present) a major time sink when things go wrong



# Timeline

- Starting to play with the API right now, and will continue to do so for the next week or so, before making requests to Roman for custom methods
- Put together prototype dev branch code, O(1 week), test
- With methods in hand, finalize design and test further
- Move over to trunk branch with failback to existing codebase. Run for O(weeks) for stability checking
- Disable old codebase
- Total O(1.5 months)



# Caveats

- This is all based on the impression that AGIS is really primetime, and can handle my (low) query rate fine in addition to all other traffic
  - Uptime under load will be something to understand. Is there testing planned/underway?
  - Rate of site info update from BDII inside of AGIS?
  - Python version > 2.5.0 – need to be sure there are no gotchas. 2.4.3 is standard on SLC5.5 machines, and is what my codebase habitually uses. No problem foreseen . . .
- Would be nice to use CERN SVN
  - (I can help with migration)



# Recap

- AGIS integration a real boon
- In principle (especially with API customizations), easy to integrate and test
- Uptime and robustness are still to be understood, but I don't think that it'll be a problem
- Timescale – sometime in the spring, cautious deployment
- Python upgrade for newController. Will do as I do SLC5 testing
- CERN SVN would be a big help

