FAUST

A Framework for Adaptive Ubiquitous Scalable Tasks

Ole Weidner CCT/LSU

Feb. 3rd. 2009

FAUST

A Framework for Adaptive Ubiquitous Scalable Tasks

APPLICATION LEVEL SCHEDULING

- Why is application level scheduling important?
  - Different applications may have different runtime characteristics - system level scheduling usually does not take this into account
  - New scheduling approaches and algorithms can easily be tested without having them deployed system-wide
  - Multiple distributed resources without a cross-system meta-scheduler can be utilized effectively

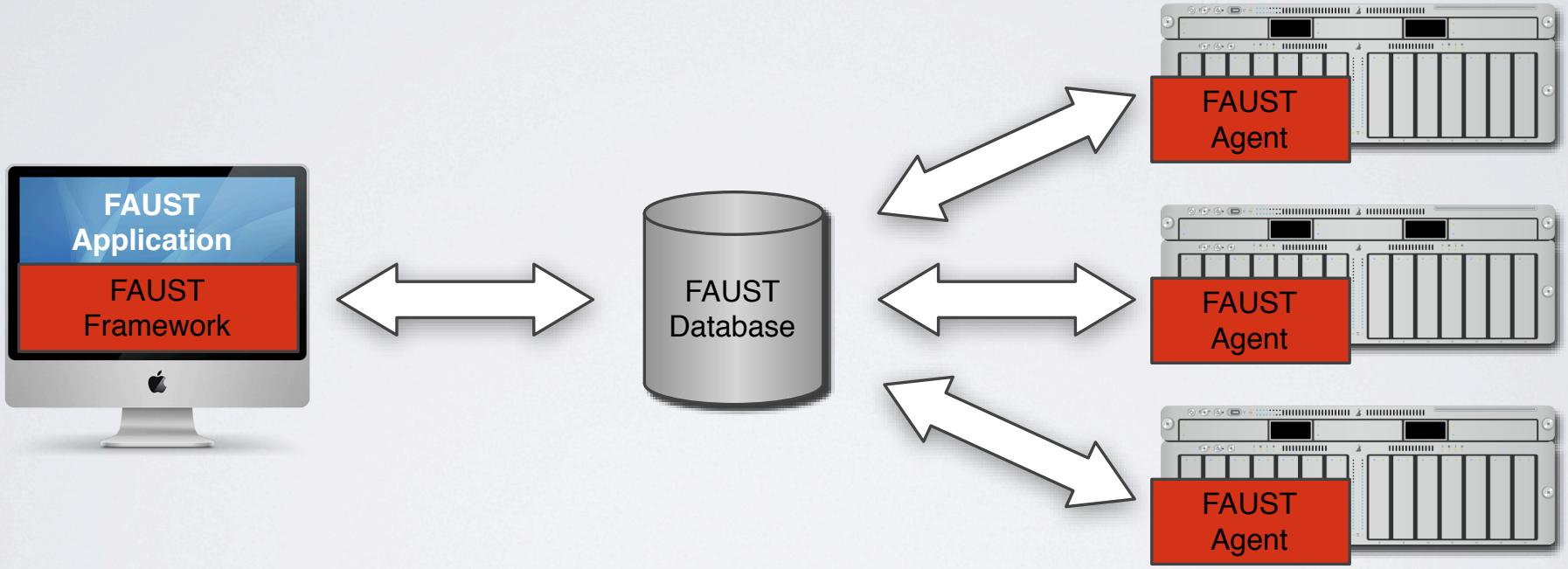
FAUST

A Framework for Adaptive Ubiquitous Scalable Tasks

KEY DESIGN OBJECTIVES

- Create a portable programming *framework* with a SAGA-like interface that can be used to submit, manage and monitor jobs programatically
- Provide a transparent mechanism to collect, interpret and store system performance data
- Expose low-level system and scheduling details *only* if desired / requested by the framework user
- Provide a plug-in mechanism that allows the replacement of scheduling algorithms - no hard-wiring

FRAMEWORK ARCHITECTURE



STATUS QUO

- Framework API is somewhat defined and implemented but probably subject to change
- Agents are work in progress - I'm trying to figure out the perfect set of attributes to describe a system
- Things are progressing - but quite slowly. First usable version of FAUST hopefully available by the end of March

ONLINE RESOURCES

- API documentation: http://macpro01.cct.lsu.edu/~oweidner/FAUST/
- SVN repository: https://svn.cct.lsu.edu/repos/saga-projects/applications/FAUST