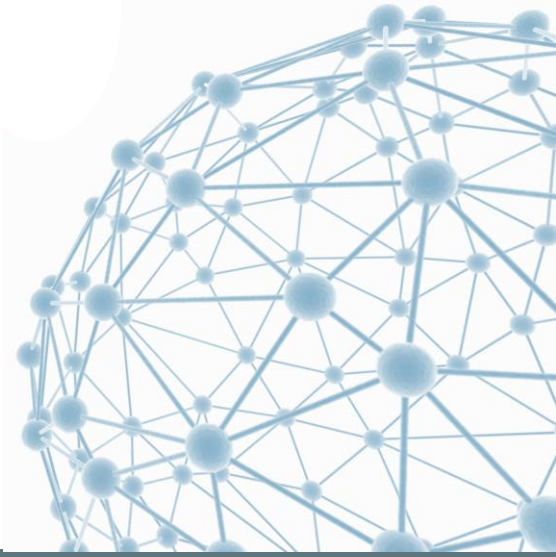



The CMS@home Prototype

Laurence Field
IT/SDC

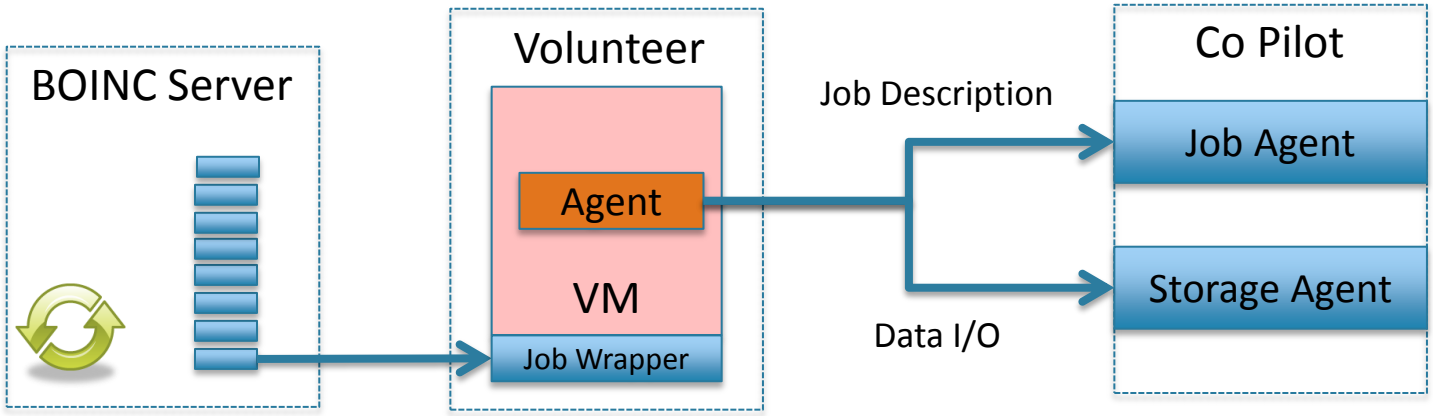
11 November 2014



Motivation

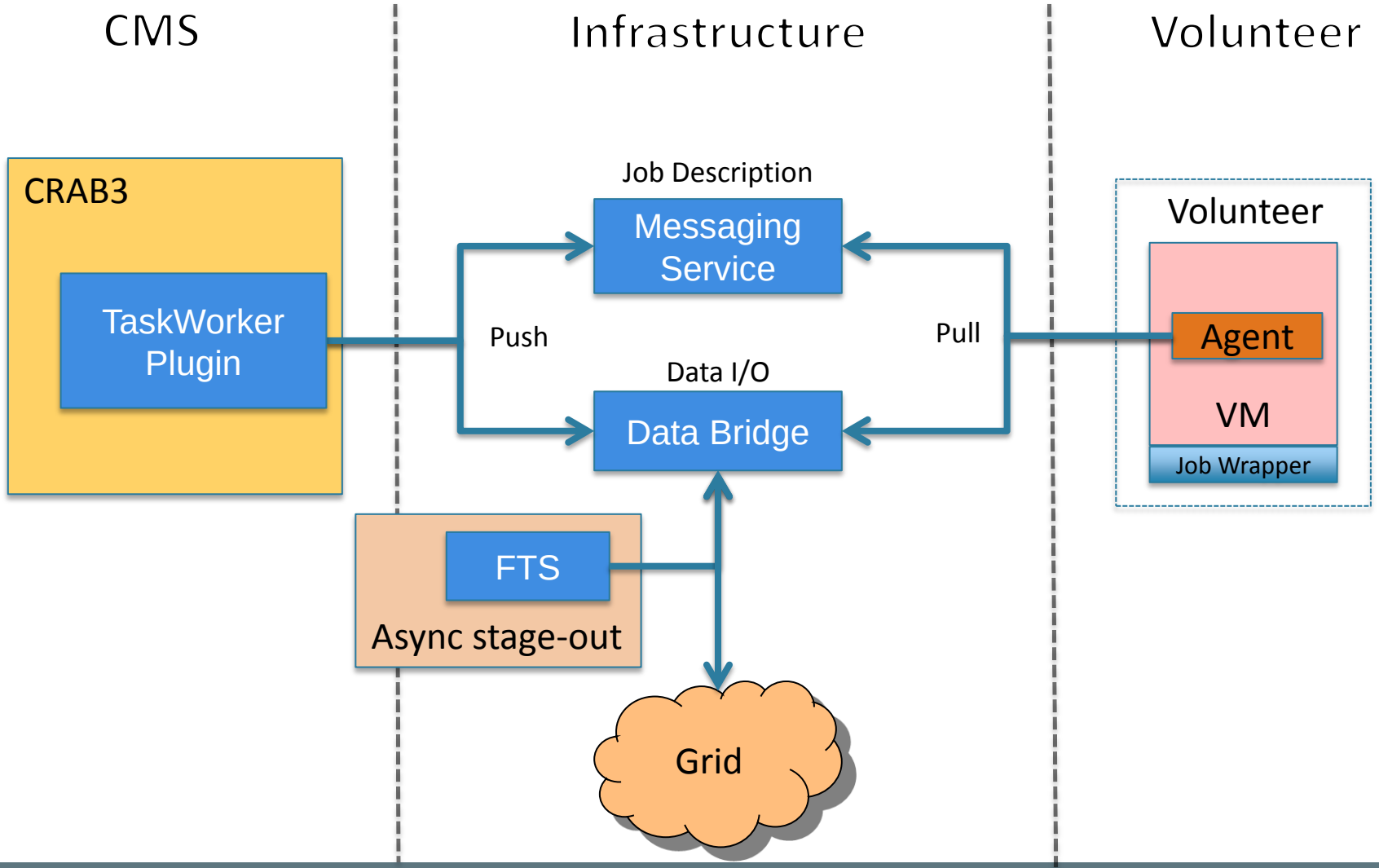
- Summer Student Project
  - Investigate the feasibility
  - Develop a prototype
- Profited from a clustering of expertise in SDC
  - CRAB3
  - Async stage-out
  - Messaging
  - FTS
- And prior experience
  - Test4Theory

Test4Theory Model



- Avoid restarting the VM for every job
  - Reduces CVMFS related network traffic
- Separate VM management and job management
  - Inline with the cloud model
    - Can reuse cloud related tooling
- CoPilot support challenges
  - Aging codebase
    - Dependencies not available in the standard repositories
    - New standardized components available for some functions

Initial Architecture



TaskWorker Plugin

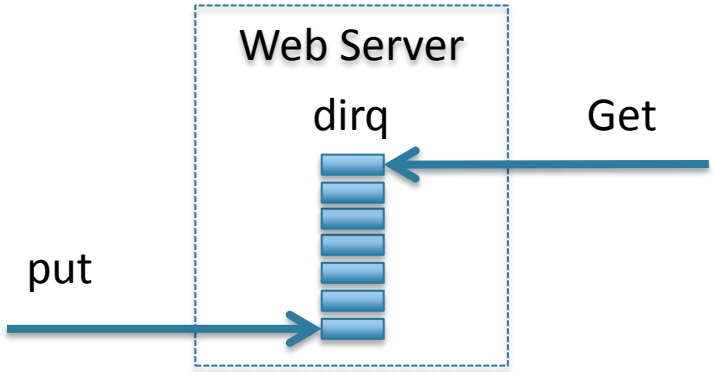
- Task Worker
  - Obtains a task request
    - Fetches work from the job queue
  - Generates the job specifications
  - Submits them to the remote job manager
    - HTCondor
- Handler
  - Responsible for the actual workflow execution
    - New task => handleNewTask handler
      - Implements the operations called Actions
        - e.g. JobSpec creation and submission
      - May be backend specific
- Specific Actions are required to support a new backend
  - The required actions need to be associate to the appropriate handlers
  - Work will be assigned to the right handler which implements the appropriate actions

Apache Plugin

- How to authenticate BOINC users?
  - In the VM, credential provided via /dev/fd0
    - BOINC_USERNAME
    - BOINC_AUTHENTICATOR
- Identity Provider (IDP)
  - BOINC Project DB
    - MySQL
    - User Table
- mod_auth_mysql
  - Maps username/password to DB table
    - AuthMysqlUserTable user
    - AuthMySQLNameField name
    - AuthMySQLPasswordField authenticator
- Enables reuse of apache-based HTTP technology

Message Queue

- Messaging service does not support BOINC authentication
  - Not clear if it is possible or worthwhile to provide functionality
- Standard apache Web server approach
  - mod_auth_mysql to validate BOINC user's credential
  - mod_auth_ssl to validate CRAB3 server's x509 credential
- Two simple cgi scripts
  - put-job.cgi
  - get-job.cgi
- Simple file-based queue
  - python-dirq
- Job descriptions from CRAB3
  - Supports arbitrary file types
    - Garbage in, Garbage out
      - Extensible ☺



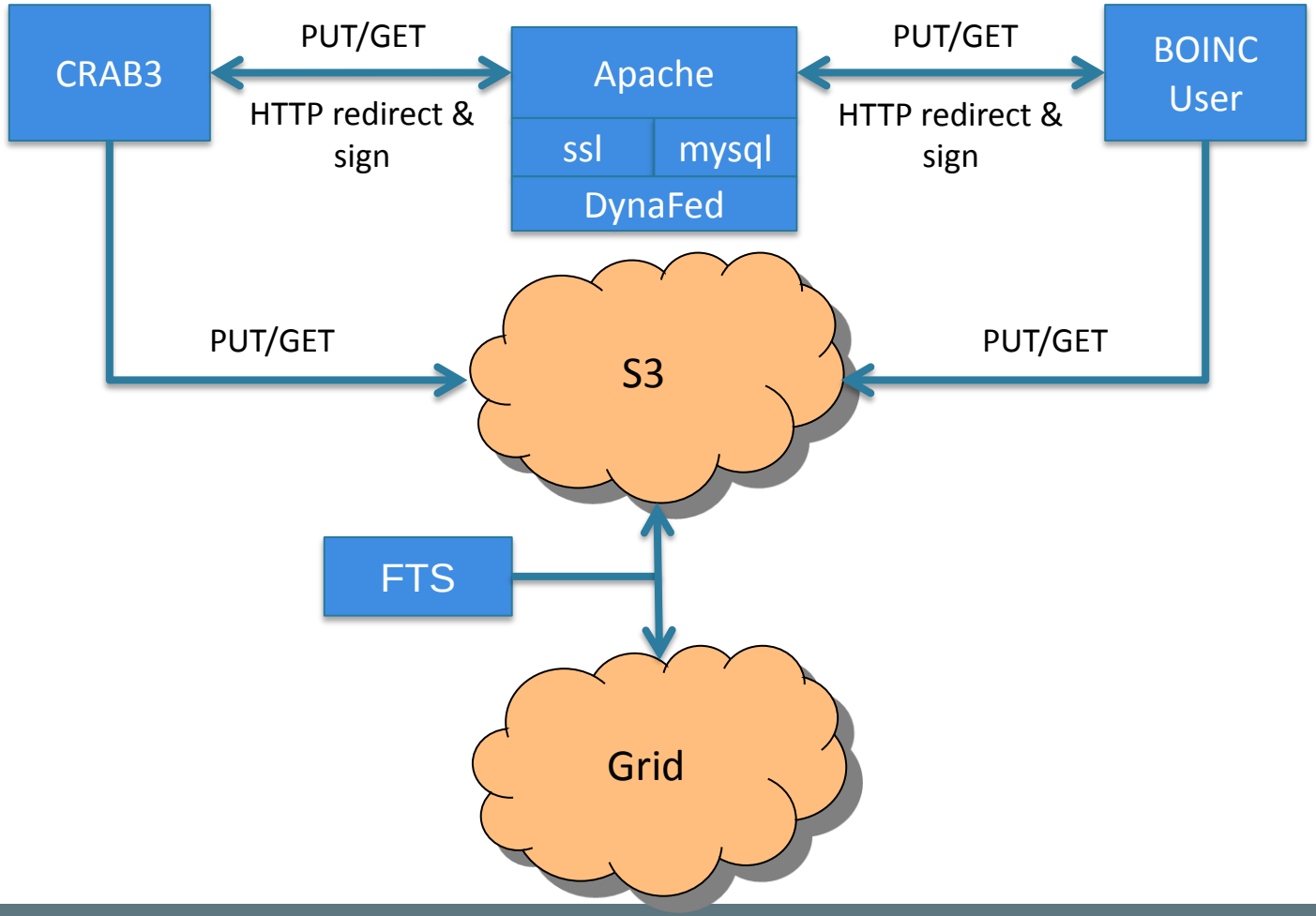
The Data Bridge

- Spans authentication domains
  - BOINC user's credential
  - Grid x509 credentials
- Scalable data I/O
  - With sandboxing capabilities
    - Data Isolation
- Simple apache-based prototype
  - Supports HTTP PUT/GET
    - mod_auth_mysql to validate BOINC user's credential
    - mod_auth_ssl to validate CRAB3 server's x509 credential
- HTTP Federation
  - Possibility to reuse standard DM tools

Dynamic HTTP Federations

- The *Dynafed* system implements federated storage over HTTP
  - In testing in LHCb and Canada (Atlas)
  - Federates WebDAV or S3 enabled storage systems
  - Apache front end
- Can be used as a data bridge
  - S3 storage backend(s)
  - Acts as a security gateway, authenticating clients either via X509 or *BOINC auth*
    - Clients then redirected directly to the storage
  - Great scalability potential
    - Global system, smart replica selection (availability, proximity)
- http://svnweb.cern.ch/trac/lcgdm/wiki/Dynafeds

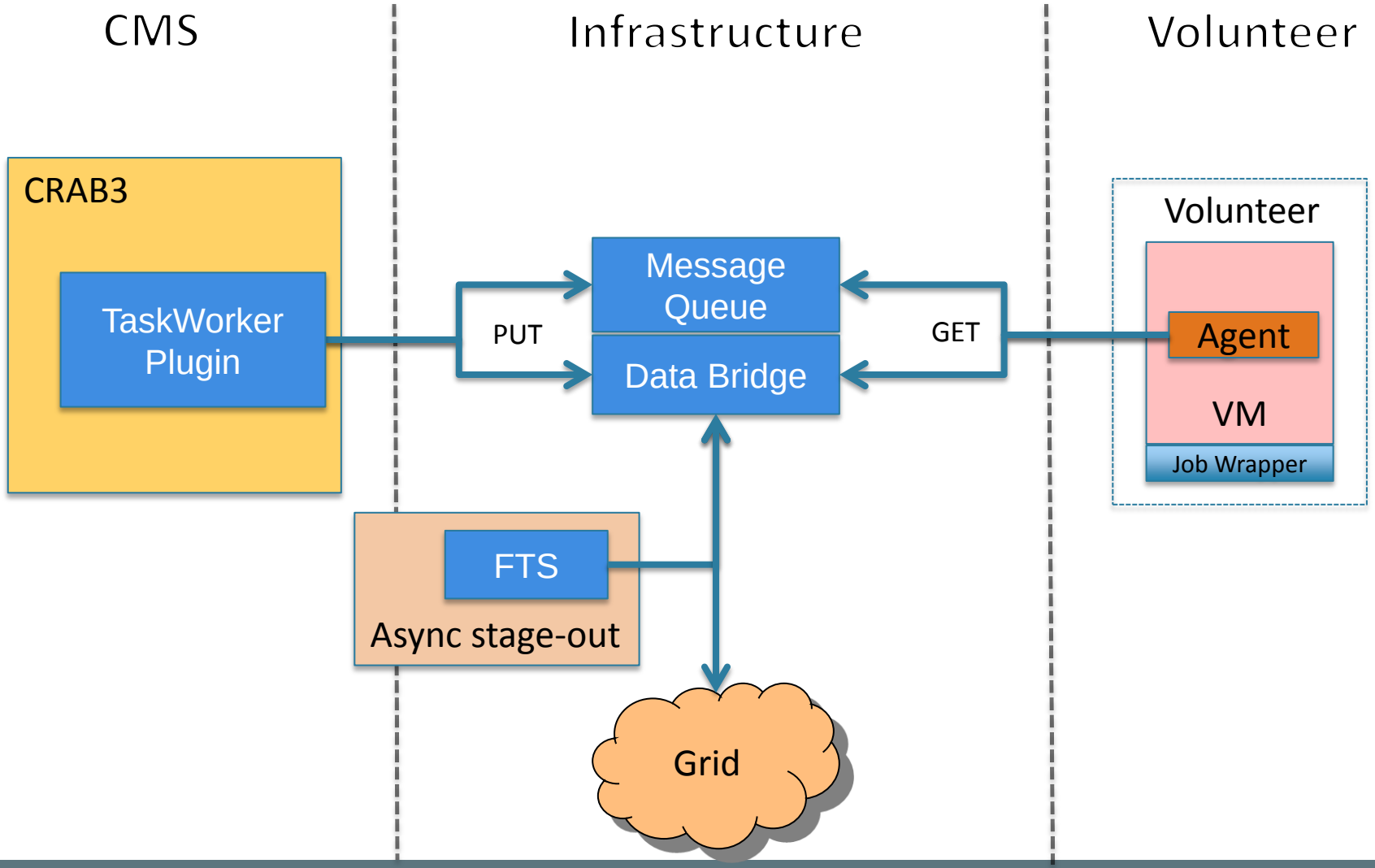
The Data Bridge



VM and Job Agent

- CernVM3
  - Contextualized using CernVM Online
- CVMFS configuration
  - Mount cms and grid
- Add BOINC user
  - Credentials read from /dev/fd0
    - name, authenticator
- CMSJobAgent.py workflow
  - Gets new job description
  - Downloads required input files
  - Runs job
  - Uploads output files
- CMSJobAgent.sh
  - Checks if CMSJobAgent.py is running
    - If not starts it
- cms-agent cron
  - Runs CMSJobAgent.sh every minute

Current Architecture



Extending the platform

- The Data Bridge as a common component for BOINC projects
  - For VM-based approaches requiring external job injection
- Data movement external to BOINC
  - Can support high data I/O requirements
- BOINC & DataBridge Recipe
  - PUTjob description
  - PUT data
  - Create Job Agent
    - GET job description
    - GET input data
      - Provided by the job description
    - Run Job
    - PUT output
  - Read data from output *bucket*
    - Similar to HLT
      - Fitter bad data etc.
- Could be used in the storage-less IaaS providers

Summary

- Advanced prototype for CMS@home
  - Following the proven Test4Theory model
- Developed the concept of the Data Bridge
  - Reused HTTP federation component for S3
    - Added BOINC authentication
- Added a simple message delivery function
  - For the job description
- Provide an image along with a job agent
- Extended CRAB3 to support his approach
  - Using a Task Work plugin
- Towards a platform for volunteer computing

Acknowledgements

- Hendrik Borras
- Daniele Spiga
- Hassen Riahi
- Adrien Devres
- Fabrizio Furano
- Oliver Keeble