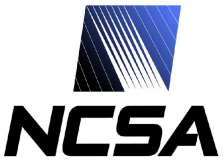
Basic Execution Management input from Globus Alliance

Karl Czajkowski

Univa Corporation



Must Support: Non-trivial Applications

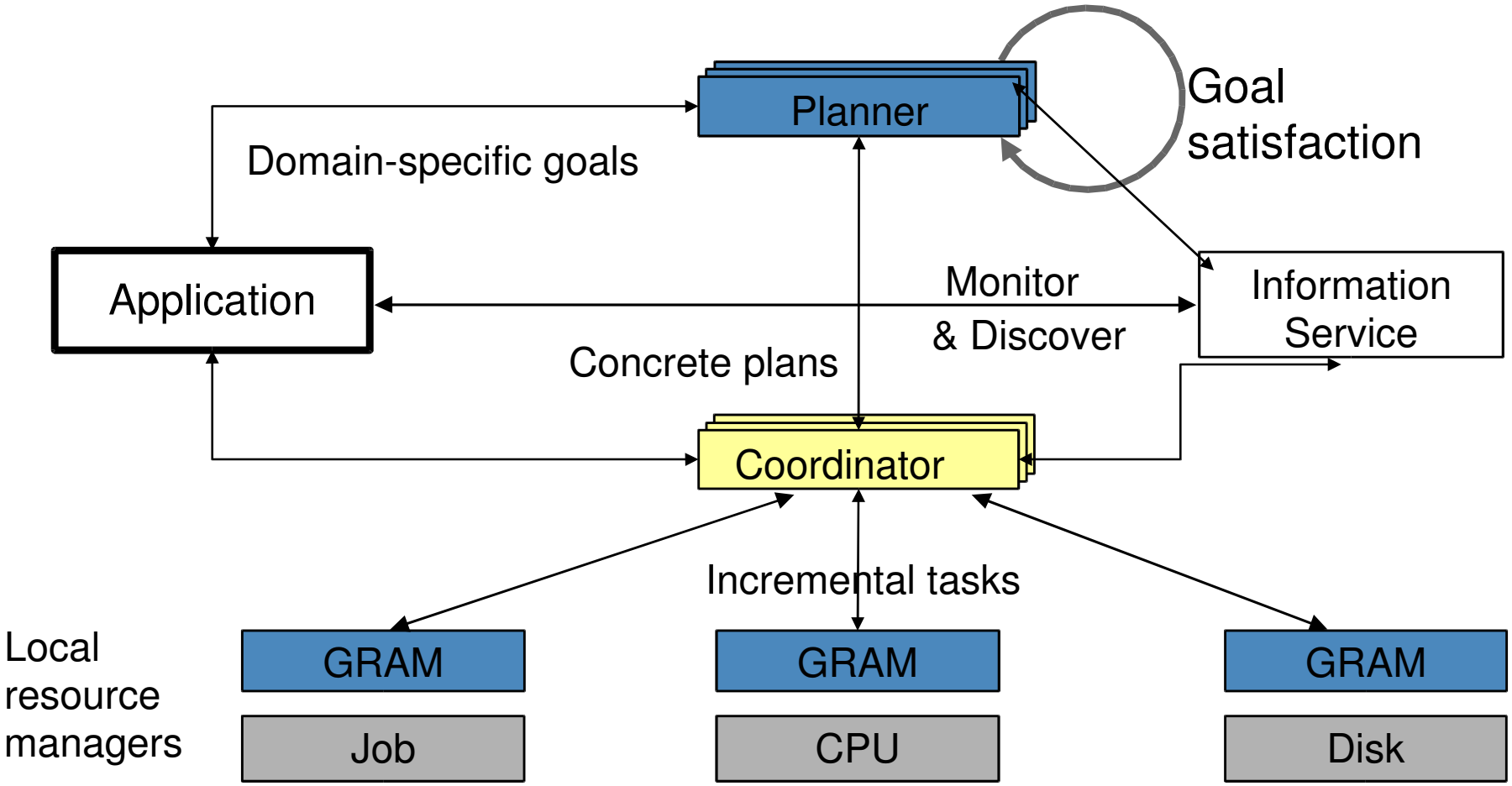
- Real-time or deadline-sensitive jobs
  - Want localized, *very good* resource
- Large jobs
  - Large and/or coupled models
  - Want to *coordinate* a few good resources
- High-throughput job sets
  - Many related jobs from one user/problem
  - Many unrelated jobs from many users
  - Want scalable job control *everywhere*

Scope: BES is Step 3 Only

1. Discovery
   - "What is out there? (of relevance) (to me)…"
   - Finds BES providers
2. Inspection
   - "How do relevant providers compare?"
   - Compare policies, status, etc.
3. Agreement
   - "Will/did I get what I need executed?"
   - The core Execution Management problem

…Process can iterate due to adaptation

(GRAM/BES are Bottom Layer)



WS-Agreement

- Ongoing standardization effort
  - In GGF's GRAAP-WG
  - Several issues raised in public comment period
    - Need 3-6 months to address and reenter public comment?
- Generalizes GRAM ideas
  - Service-oriented architecture
  - Container/host becomes *Service Provider*
  - Tasks/jobs become *Negotiated Services*
  - Mgmt state presented as *Agreement* services
- Supports composition w/ domain terms

JSDL

- Job Submission Description Language
  - Provides an ontology for “job to be run”
- Requires external spec to put it in context
  - WS-Agreement can use JSDL as “service terms”
- A mix of abstract and concrete job bits
  - Executable+args+environment
    - Assume application already provisioned at host
  - Abstract “portable” job profiles
    - Try to cope w/ localized differences

Need to Clarify Timeline

- Theoretically: WS-Agreement + JSDL + epsilon
  - Delivery could be delayed by dependencies
    - WS-Agreement needs another revision
    - JSDL might as well, to address concrete job issues?
  - Scoping concerns
    - Is current JSDL 1.0 target complete enough? Too big?
    - No practical experience w/ WS-Agreement yet
- Conservative/fast: base on existing system(s)
  - We would help slice/dice WS-GRAM WSDL
  - Other vendors offering to help?

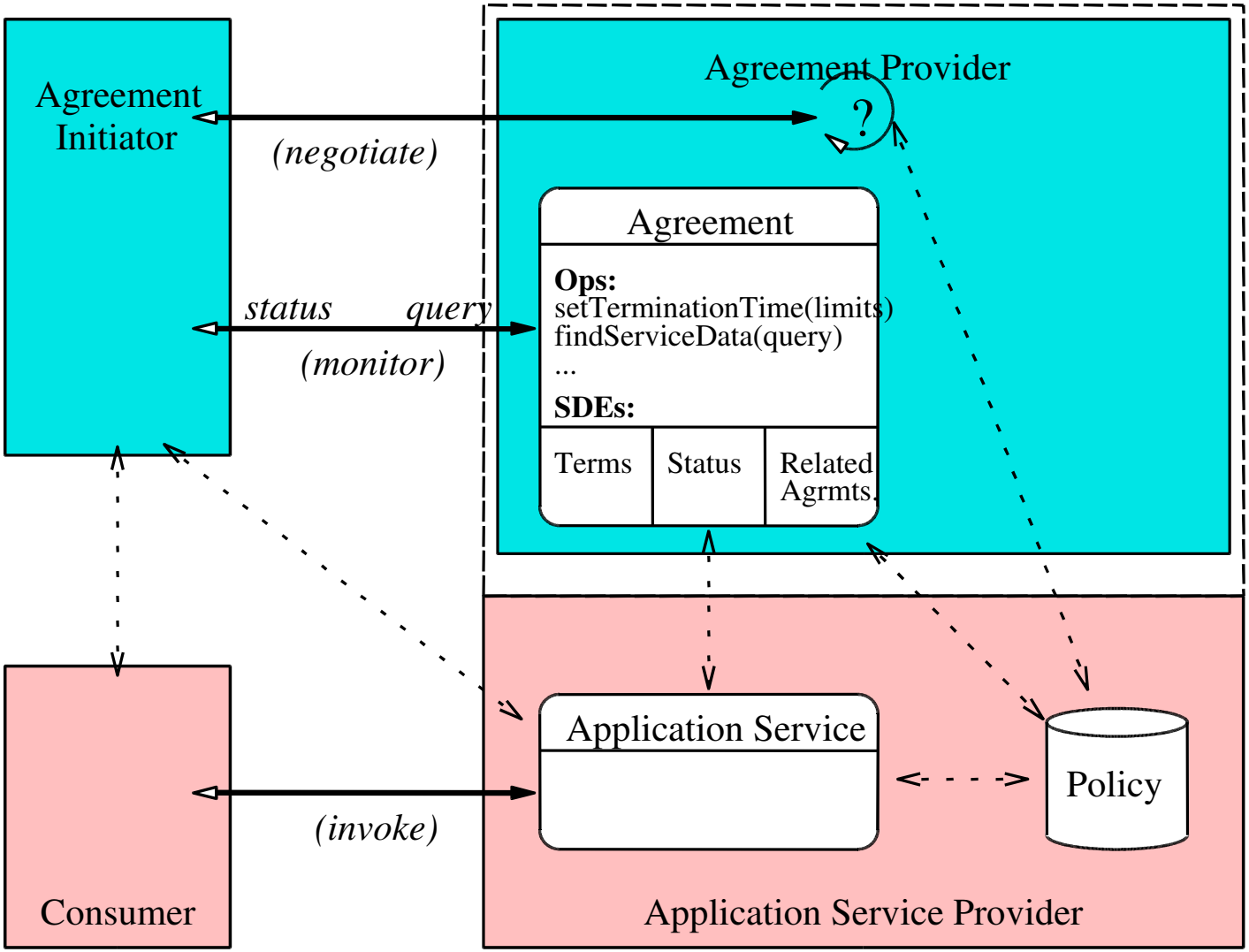
State of the Art

- Discovery is very hard and immature
  - Some viable information gathering systems
  - But information models have gaps
    - Lots of low-level “buttons and knobs” stuff, e.g. CIM
    - Some overly abstract stuff, e.g. GLUE, GRAM today
    - Complexity already a barrier to entry
  - RM policy: personalized scope/relevance
- BES needs some basic metadata/properties
  - Introspect for “dialect” support
- Assume more will be developed over time

Caution!

- Must choose scope wisely and stick to it
  - Many groups flounder on recurring debates
    - Old issues revisited for new participants
    - Old issues tripped over from new angle (and not noticed)
  - *The real work is in rendering abstract archictectural consensus into concrete syntax w/o fatal warts and flaws*
- Compositionality a mixed blessing
  - Defer decisions on difficult problems
    - Can BES address file staging? Credential mgmt?
  - But, more opportunities for non-interop
    - “Power set” of different specs/extensions...

WS-Agreement Entities



Simple Negotiation

- AgreementFactory::createAgreement()
  - Coarse-grained
  - Conventional fault/response model
  - Batch negotiation of complex terms
  - Idiom: enables one-shot job submission
- Agreements can be chained
  - Establish stateful context of Agreements
  - New Agreement depends on or claims context
- Need companion specs for advanced scenarios

WS-Agreement is a Protocol

- WS-Agreement is a message model…
  - Not a software component
- …applicable to previous examples
  - Interface standard between components
  - Improve interoperability of other systems
  - To enable composition/federation

(Possible WS-Agreement conversion examples:

  - GRAM, Condor
  - Workflow, economic scheduling
  - PBS, LSF, CSF)

Specifying Terms: Who and What?

In a service provisioning **domain**

(e.g. “computational jobs”)

- A **standard** specifies **domain** terms/concepts
- A **provider** specifies its support for
  - *some or all* of the domain standard terms
  - a given term, *specifically*
    - Within *behavioral* constraints
    - Within *negotiability* constraints
    - With extra fields/sub-terms
    - Arbitrary term *properties*: e.g. *optional* or *required*
- *A **client** discovers compatible providers*

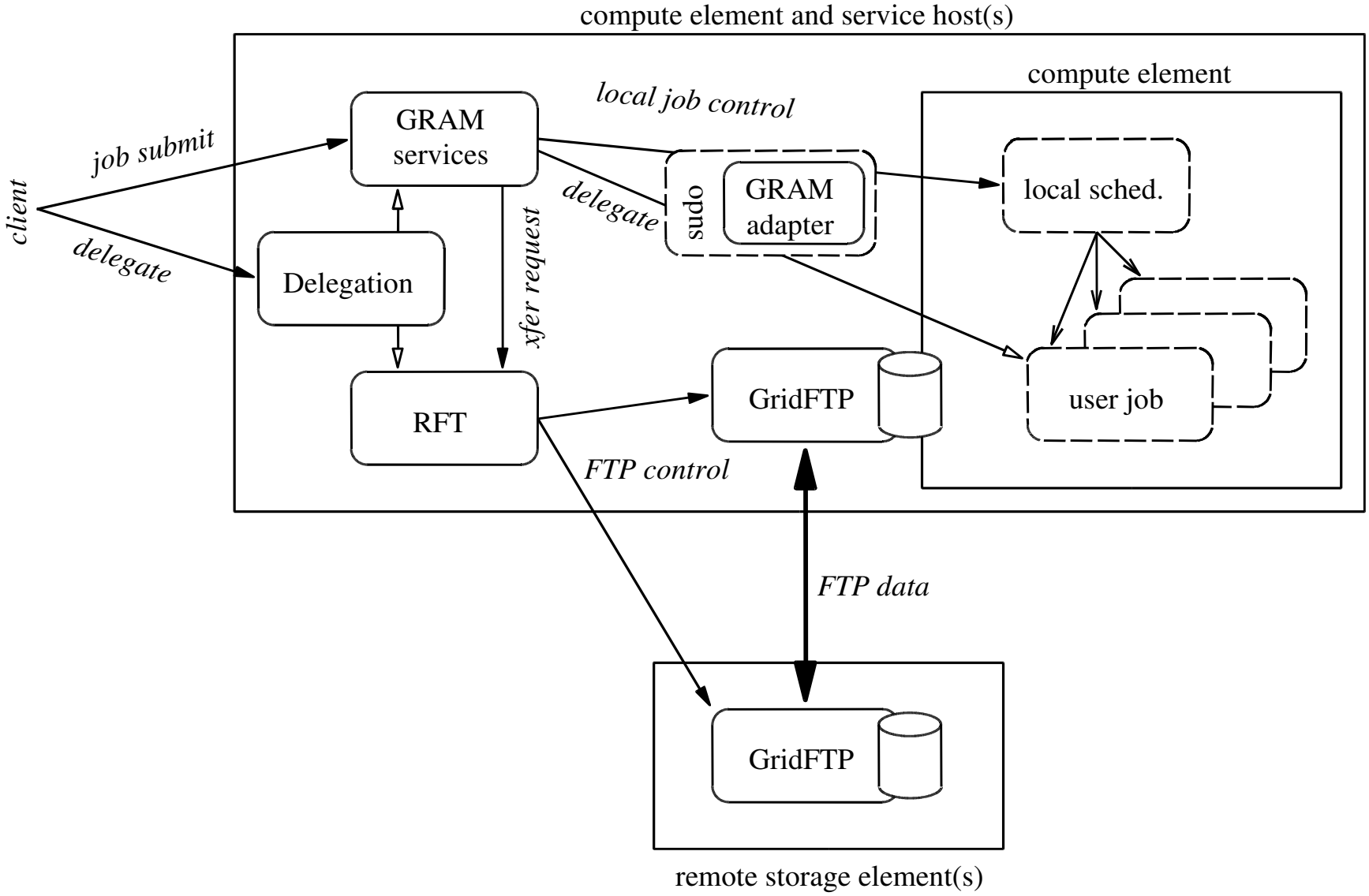
Agreement-based Jobs

- Agreement represents “queue entry”
  - Commitment with job parameters etc.
  - This is the management “proxy” for job
- Agreement Provider
  - i.e. Job scheduler/Queuing system
  - Management interface to service provider
- Service Provider
  - i.e. scheduled resource (compute nodes)
- Provided Service is the Job computation

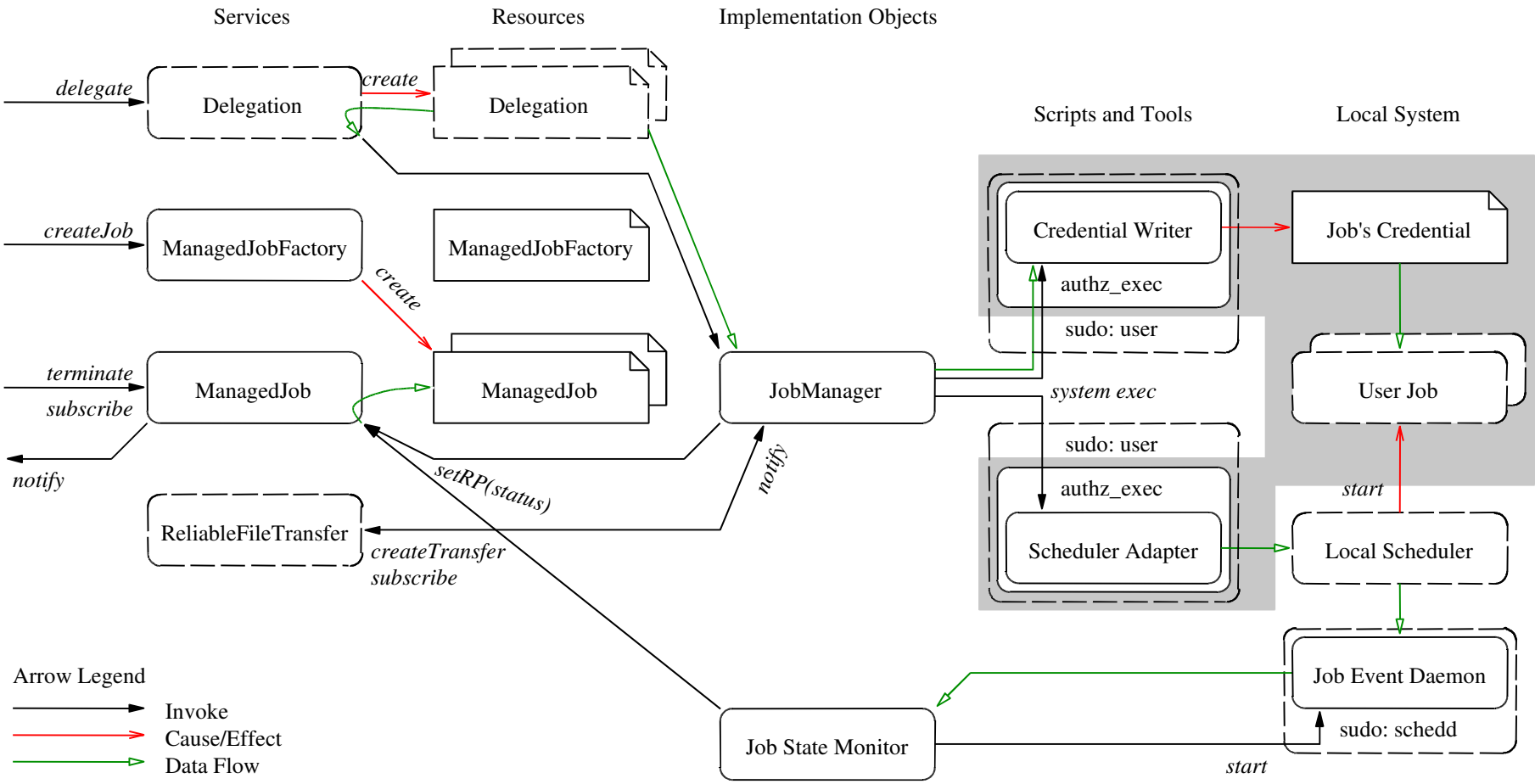
BES as WS-Agreement

- Write a BES spec
  - A profile combining WS-Agrmt + JSDL
  - Influence JSDL or spec. extensions?
  - Be conservative about generating new dialects
- Get some implementation experience
  - We know how to adapt WS-GRAM
  - Tie into existing ManagedJob resource impl.
- Put BES in comment period
  - Determine if experiences warrant revisions

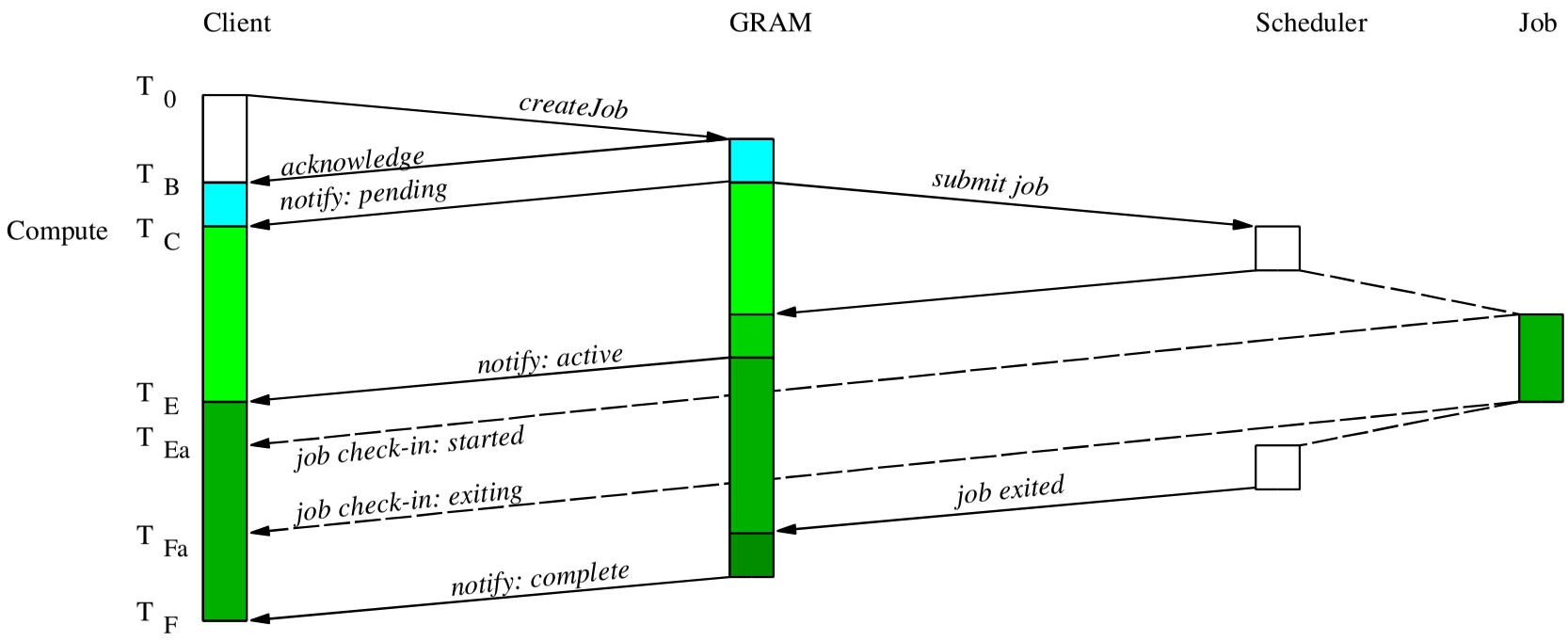
WS-GRAM Approach



WS-GRAM Software Map



WS-GRAM Base Protocol



WS-GRAM Full Protocol