

Open Standards and Architectures

Richard Sinnott

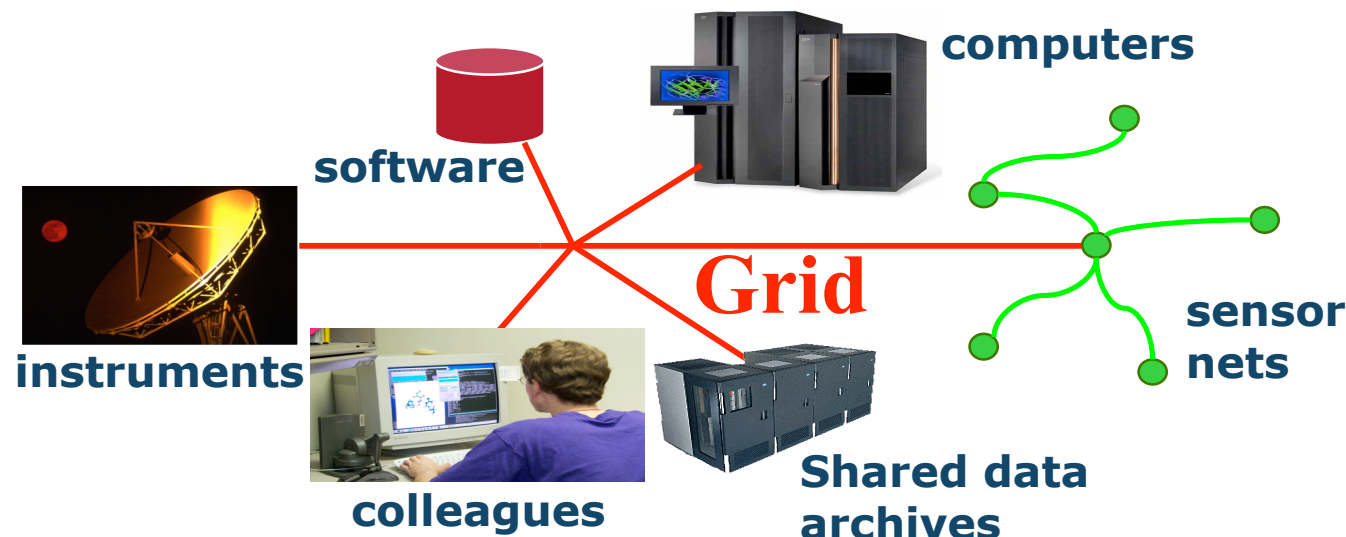
<http://csperkins.org/teaching/2004-2005/gc5/>

Overview

- Grid driving force for open standards and architectures
 - Standards making bodies and the process itself
 - ongoing standards development
 - groups at GGF, OASIS, IETF, ...
- Background to Open Grid Service Architecture (OGSA)
- OGSA Guiding Philosophy and Associated Constraints
- OGSA framework itself
 - Including Open Grid Service Infrastructure (OGSI)
 - and coming soon ... Web Service Resource Framework (WSRF)

Grid Standards Needs

- Grid systems and applications aim to integrate, virtualise, and manage resources and services within distributed, heterogeneous, dynamic “virtual organizations”
- Requires disintegration of barriers that separate different computing systems within/across organizations, so that computers, applications, data, and other resources can be accessed as and when required, regardless of physical location



Grid Standards Needs ...ctd

- Key to the realization of this Grid vision is standardization
 - Standards needed so that the diverse components, services, data can be
 - discovered,
 - accessed,
 - allocated,
 - monitored,
 - accounted for,
 - billed for,
 - ... and in general managed as a single virtual system, even when provided by different vendors used by different organizations
- Standardization is critical to create:
 - interoperable, portable, and reusable components and systems
 - also help to establish “good practice”
- Open Grid Service Architecture (OGSA) represents Grid community standardisation efforts/vision
 - <http://www.globus.org/ogsa>

Prelude to OGSA Framework

- OGSA on-going effort
 - Various standards bodies, groups involved in putting “meat on the bones” of OGSA framework
 - Previously driven mostly by Global Grid Forum (GGF)
 - Meet 3 times per year to discuss/agree/explore/develop/understand requirements for needed standards
 - Recent developments are moving standards making process towards Organization for the Advancement of Structured Information Standards (OASIS)
 - Also aspects of involvement of
 - Internet Engineering Task Force (IETF)
 - World Wide Web consortium (W3C)

Prelude to OGSA Framework ...ctd

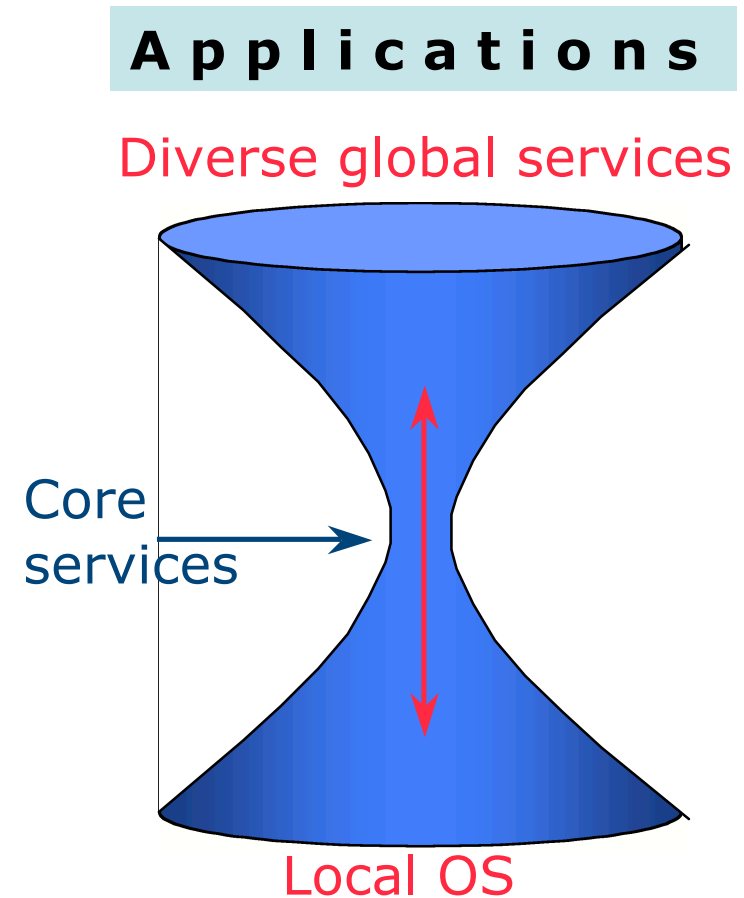
- Implementations of (parts of) OGSA under development
 - We will be exploring some of these in course
 - Globus Toolkit (GT3.3), PERMIS, ...
 - **A word of warning....**
 - OGSA is not yet fully defined and there are no real OGSA Grids out there right now
 - Two projects in UK looking at OGSA Grids
 - But really these are just trialing some relevant software
 - In short OGSA does not yet formally exist but is a vision for how a Grid framework might look where some components services have been implemented
 - » **...don't be surprised if software systems not much more than beta versions**

General Philosophy behind OGSA

- Purpose of OGSA not to design system satisfying every user's needs
 - OGSA “*being designed*” to provide users and implementers greatest flexibility in supporting their applications
 - OGSA avoiding providing “the solution” to a wide range of system functions.
 - Idea is to allow users to select the *kind* and the *level* of functionality for their application, and let them make their own trade-offs
 - e.g. some users concerned with privacy, others concerned with integrity of data, others with ...
 - Some users are content with password authentication,
 - ...others might need stronger user identification such as signature analysis, fingerprint verification
 - » Both of these are examples of differences in the *kind* of security functionality.
 - Size of cryptographic key is issue of *level* of security
 - » Without changing nature of the security provided, users can get a greater degree of security by paying higher cost of using a longer key or a stronger algorithm

History of OGSA Framework

- Previous architectures for Grid systems, e.g. Globus toolkit version 2 and earlier - given as “bag of services”
 - Not vertical solution
 - designed for components with APIs in C
 - Largely Unix/Linux based
- Idea was hourglass model of services
 - Focus on architecture issues
 - Propose set of core services as basic infrastructure
 - Use to construct high-level, domain-specific solutions
 - Design principles
 - Keep participation cost low
 - Enable local control
 - Support for adaptation
 - “IP hourglass” model



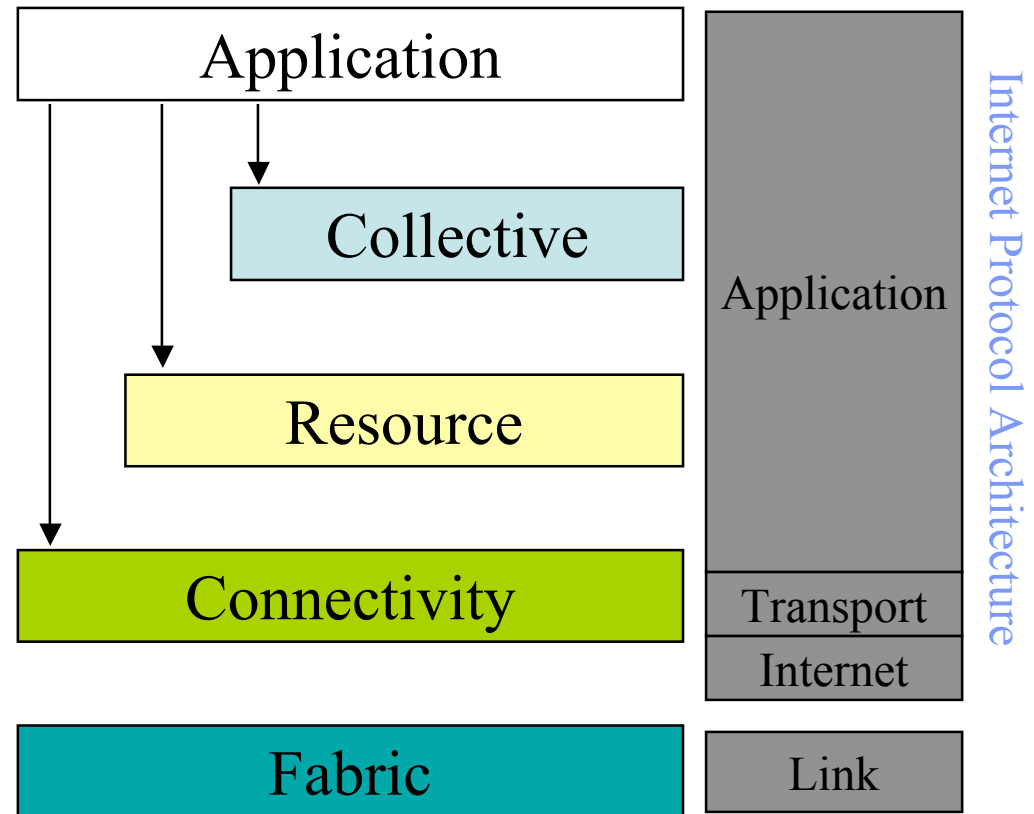
History of OGSA Framework ..ctd

“Coordinating multiple resources”:
ubiquitous infrastructure services, app-
specific distributed services

“Sharing single resources”: negotiating
access, controlling use

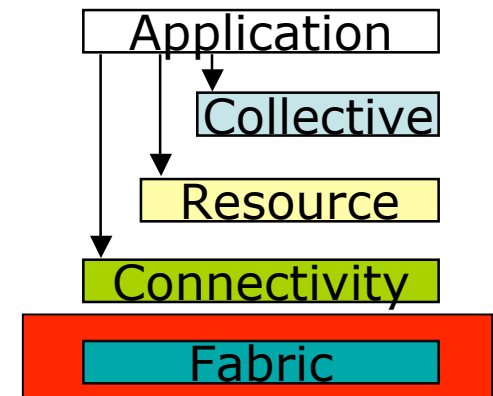
“Talking to things”: communication
(Internet protocols) & security

“Controlling things locally”: Access to,
and control of, resources



Fabric Layer Protocols & Services

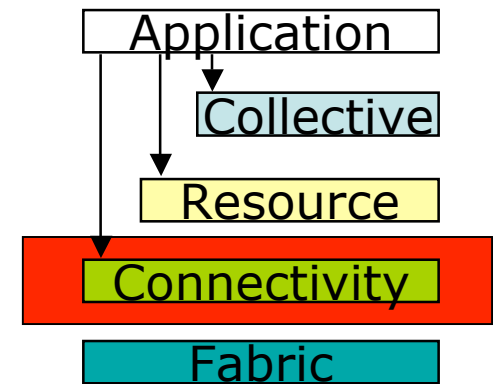
- Diverse mix of resources that may be shared
 - Individual computers,
 - job pools,
 - file systems,
 - archives,
 - metadata catalogs,
 - networks,
 - sensors,
 - ...etc



- Few constraints on low-level technology
- Defined by interfaces not physical characteristics
 - More in lecture 15

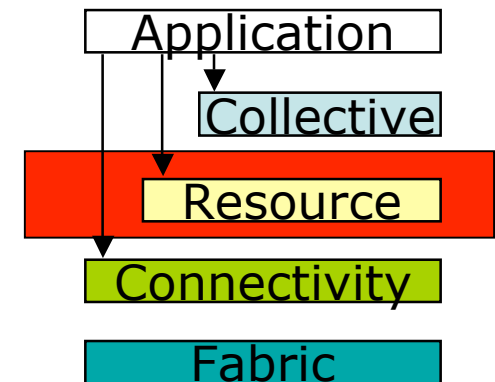
Connectivity Layer Protocols & Services

- Communication
 - Primary focus on internet protocols (IP, DNS, ...)
- Security: Grid Security Infrastructure (GSI)
 - Uniform authentication, authorization,
 - and message protection mechanisms
 - in multi-institutional setting
 - Single sign-on, identity mapping
 - Public key technology, SSL, X.509
 - Supporting infrastructure:
 - Certificate Authorities, certificate & key management, ...
 - More in later lectures 7-9
 - (note did not really support authorisation!)



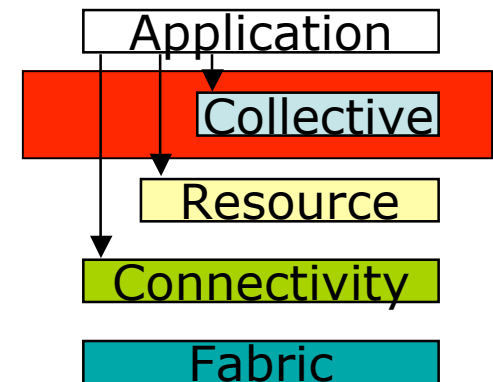
Resource Layer Protocols & Services

- Grid Resource Allocation Management (GRAM)
 - Remote allocation,
 - reservation,
 - monitoring,
 - control of compute resources
- GridFTP protocol (FTP extensions)
 - High-performance data access & transport
- Grid Resource Information Service (GRIS)
 - Access to structure & state information
- Grid Information Index Services (GIIS)
 - Access to collections of GRIS information sets
- Others
 - Catalog access,
 - code repository access,
 - accounting, etc.
- All built on connectivity layer
 - More in later lectures 4-6



Collective Layer Protocols & Services

- Meta-directory services
 - Custom views on dynamic resource collections assembled by a community
- Resource brokers
 - Resource discovery and allocation
- Co-reservation and co-allocation services
- Workflow management services
- ...
 - More in later lectures 10-12



Problem Solved?

- Lets just adopt the hourglass “bag of services” and let applications writers plug into them as they like
 - Not quite...
 - Significant missing functionality, e.g.
 - Databases, sensors, instruments, workflow, ...
 - Virtualization of end systems (hosting environments)
 - Little work on total system properties,
 - e.g. model did not easily support
 - » Dependability, end-to-end QoS, ...
 - » Reasoning about system properties
 - Main technology providers (Globus) = academic oriented
 - Becoming more and more difficult to manage
 - » (Argonne Team ~15 people)
 - Focus on Unix/Linux
 - » Commodity grids (CoG) for port of subset of GT, e.g. in Java
 - Business going other way
 - Web services, XML, B2B, ...

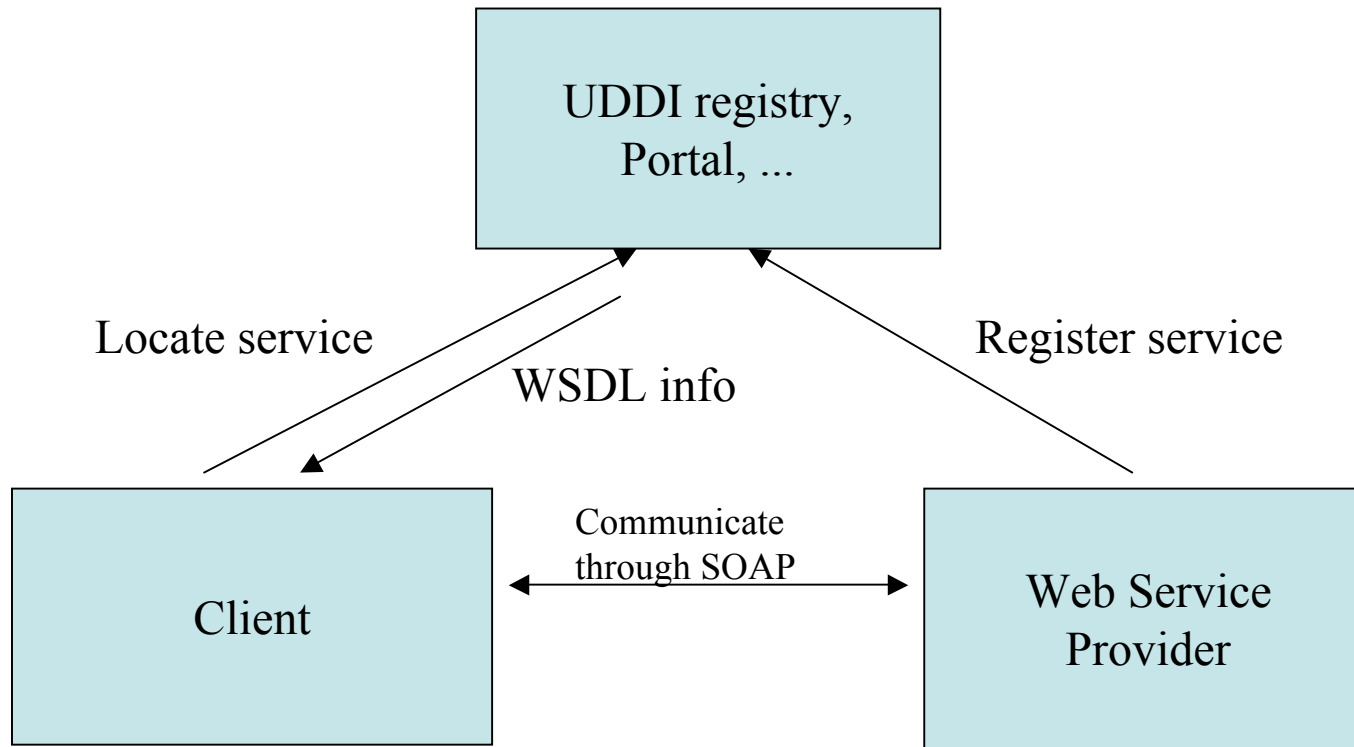
OGSA Framework Considerations

- Limitations of “bag of services” based on need for open standards driven architecture
 - “bag of services” driven predominantly by technology solutions
 - Wider community required more open framework to compose functionality, develop new systems without being tied to specific code base
- Open Grid Services Architecture (OGSA) proposed as solution to this
 - architecture/framework of Grid/web services
 - Grid service is a web service with some specific additional functionality
 - planned that core set of persistent services available
 - specific Grids create new instances of “transient” Grid services on demand

Web Services

- (Much more details in next lecture)
 - Not a new idea
 - Publish, find, bind
 - (ring a bell with other distributed systems approaches???)
 - Core languages and technologies involved
 - XML
 - Simple Object Access Protocol (SOAP)
 - Simple API for XML (SAX),
 - Document Object Model (DOM)
 - Web Services Description Language (WSDL)
 - Universal Description, Discovery and Integration of Web Services (UDDI)
 - ...
- ... plus many APIs, language, meta-language flavours being defined on an almost daily basis
- We will meet some of them later in the course...

Web Services



WSDL info = how to interact, what operations, what XML grammar flavour, ...

Service Oriented Architecture is common approach

Will look more about these technologies in later lectures

OGSA Framework

- OGSA represents framework describing core set of infrastructure components described by Grid (web) services
 - *Pre-March 2004* basic idea was that all Grid services would share some common functionality
 - This commonality known as *Open Grid Service Infrastructure (OGSI)*
 - Technologies implemented which supported this (GT3)
 - Arguments against OGSI though in March 2004 and now suggestion to move to purer web service approach (Web Service Resource Framework)
 - » arguments based on re-use of web service tools, use of XML schema...
 - » ... big business involvement
 - Need to know OGSI as this is what tools support right now

OGSI Outline

- OGSI defined minimal, integrated set of extensions and interfaces needed for OGSA
 - included
 - how Grid service instances are named and referenced
 - base, common interfaces (and associated behaviors) that all Grid services implement
 - additional (optional) interfaces and behaviors associated with factories and service groups

OGSI Outline

- OGSI proposed component model extending WSDL and XML Schema definition to support
 - stateful Web services
 - Web services do not generally remember state related to previous invocations
 - persistent and transient services
 - Services can be created on demand
 - extension of Web services interfaces
 - inheritance of portTypes
 - asynchronous notification of state change
 - Clients can subscribe to services and be informed of state changes
 - references to instances of services
 - Grid Service Handles and Grid Service References
 - collections of service instances
 - Service groups (basic service indexing)
 - service data
 - can be used for accessing service state information or meta-data for service
- Will be explored in more detail in the next lecture

OGSA Framework

- Precisely what OGSA will finally look like not yet clear
 - Various requirements shaping OGSA though
 - Must make provision for:
 - site autonomy
 - extensibility
 - scalability
 - usability
 - meta data management and discovery (reflection)
 - security for both users and resource providers
 - resource management and exploitation of resource heterogeneity
 - multi-language support
 - notions of identity
 - fault tolerance
 - ...

OGSA::Autonomy

- *Site autonomy*
 - OGSA not for single monolithic system
 - Composed of resources owned and controlled by an array of organizations
 - no “big brother” centralized mgt
 - Organizations expect control over their own resources
 - » Orgs. specify how much resource can be used, when it can be used, and who can and cannot use the resource, under what circumstances ...etc
 - Autonomy of implementation
 - Sites must be able to choose which implementations of OGSA components to use (*note likely to be several!!!!*)
 - e.g. because they “trust” one implementation over another,
 - for performance reasons
 - ...

OGSA::Extensibility

- *Extensible core*
 - cannot know all current/future needs of users, distributed system developments
 - OGSA needs to be extensible with mechanisms/policies to replace, update components
 - will permit OGSA evolution
 - allow users to construct their own mechanisms and policies to meet specific needs
 - OGSA components themselves need to be extensible and replaceable
 - allowing different implementations to be developed and used
 - scalability
 - because OGSA systems may consist of millions of hosts, it must have a scalable software architecture
 - cannot assume centralized structures/servers

OGSA::Usability

- *Usability of computational environment*
 - OGSA must mask the complexity of the hardware environment and of communication/synchronization of parallel processing
 - machine boundaries should be invisible to users
 - if OGSA is not transparent and easy to use then it will provide little benefit over existing mechanisms of distributed computing
 - » could be argued that a Grid with no transparency is not a Grid!!!
 - Note however that some applications will require the capability to make low-level decisions and to interface with low-level system mechanisms
 - OGSA must accommodate ***both*** classes of end user/application
 - those just want to get their work done and not worry about details
 - other users needing to tune their applications

OGSA::Security

- *Security for users and resource owners*
 - Security must be built firmly into OGSA from the outset
 - There is no one security policy perfect for all users
 - Security must fit in to existing security solutions
 - since cannot replace existing host OS, cannot significantly strengthen existing operating system protection and security mechanisms
 - But OGSA must ensure existing mechanisms are not weakened
 - OGSA must provide mechanism for users to select policies that fit their needs
 - OGSA should not define the security policy or require a “trusted” OGSA
 - » Users define the policy and OGSA provides suitable “hooks”

OGSA::Management

- *Management and exploitation of resource heterogeneity*
 - OGSA must support interoperability between heterogeneous hardware and software components
 - Applications written in variety of languages on a variety of HW/OS
 - OGSA must transparently overcome issues of heterogeneity
 - In addition, some architectures are better than others at executing particular applications
 - e.g. batch based farms exploiting application parallelism vs SMP based apps
 - OGSA must ensure that these affinities, and the costs of exploiting them, can be factored into scheduling decisions and policies

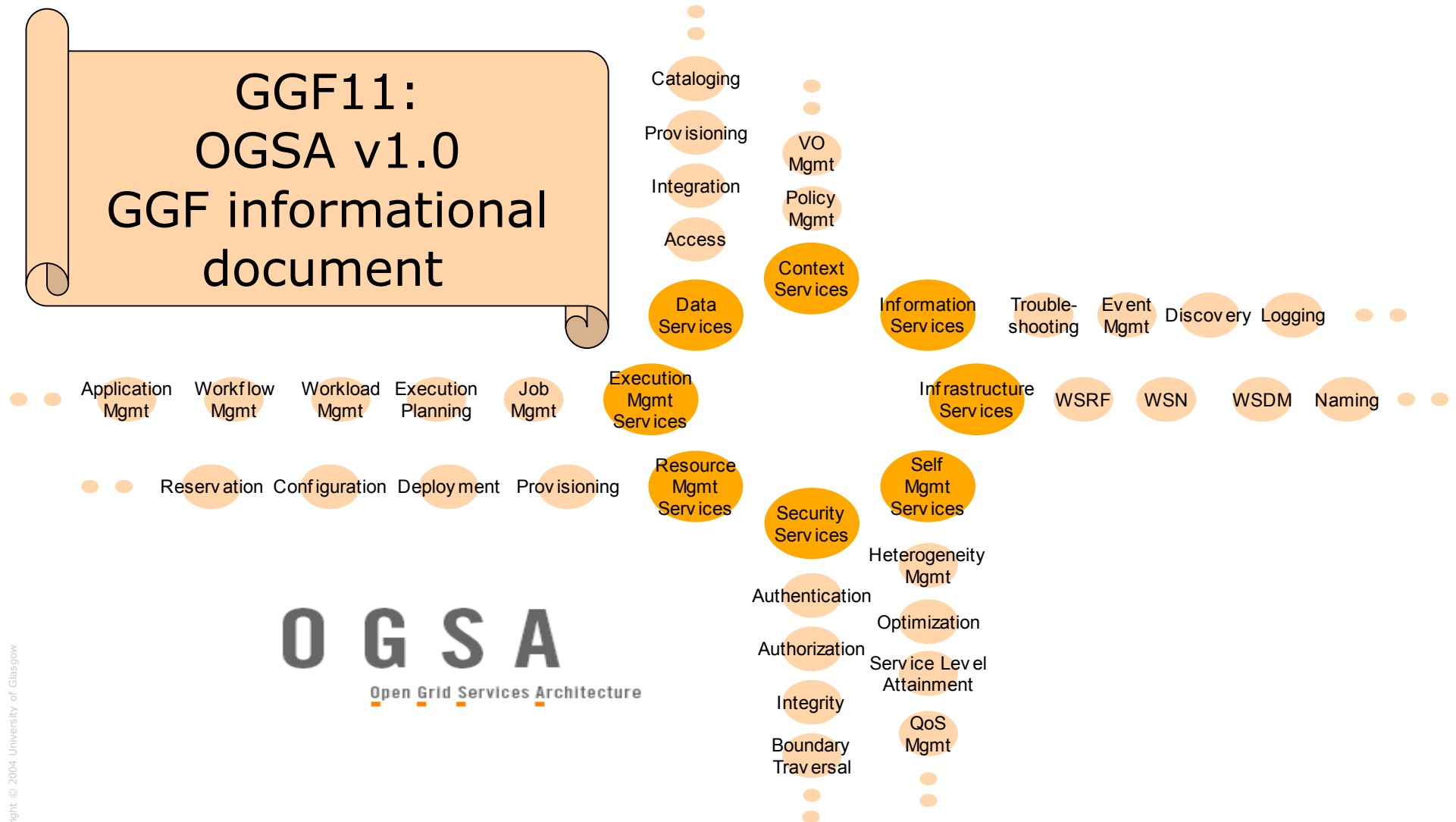
OGSA::Fault Tolerance

- *Fault-tolerance*
 - OGSA should support systems where at any given instant
 - hosts
 - communication links
 - disks
 - ...can fail
 - OGSA should deal with failure and dynamic re-configuration
 - for OGSA components themselves,
 - ... and the applications making use of OGSA

OGSA Use Case Scenarios

- Currently GGF group defining OGSA looking at specific scenarios that must be supported
 - How do I establish identity and negotiate authentication?
 - How is policy expressed and negotiated?
 - How do I discover services?
 - How do I negotiate and monitor service level agreements?
 - How do I manage membership of, and communication within, virtual organizations?
 - How do I organize service collections hierarchically to deliver reliable and scalable service semantics?
 - How do I co-ordinate usage of data and compute resources to achieve optimized usage for my application?
 - How do I monitor and manage collections of services?
 - ...

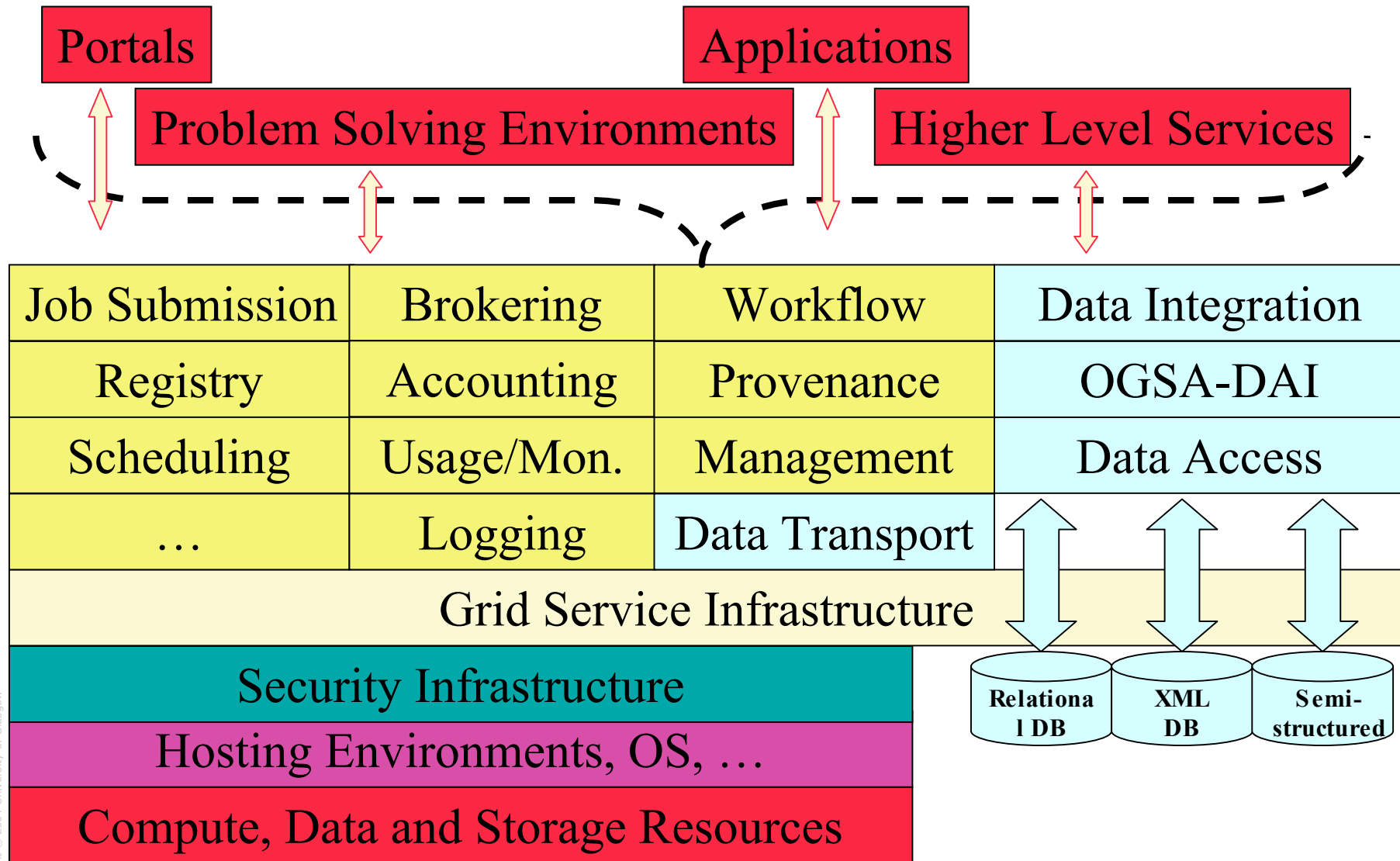
OGSA Framework



OGSA Issues

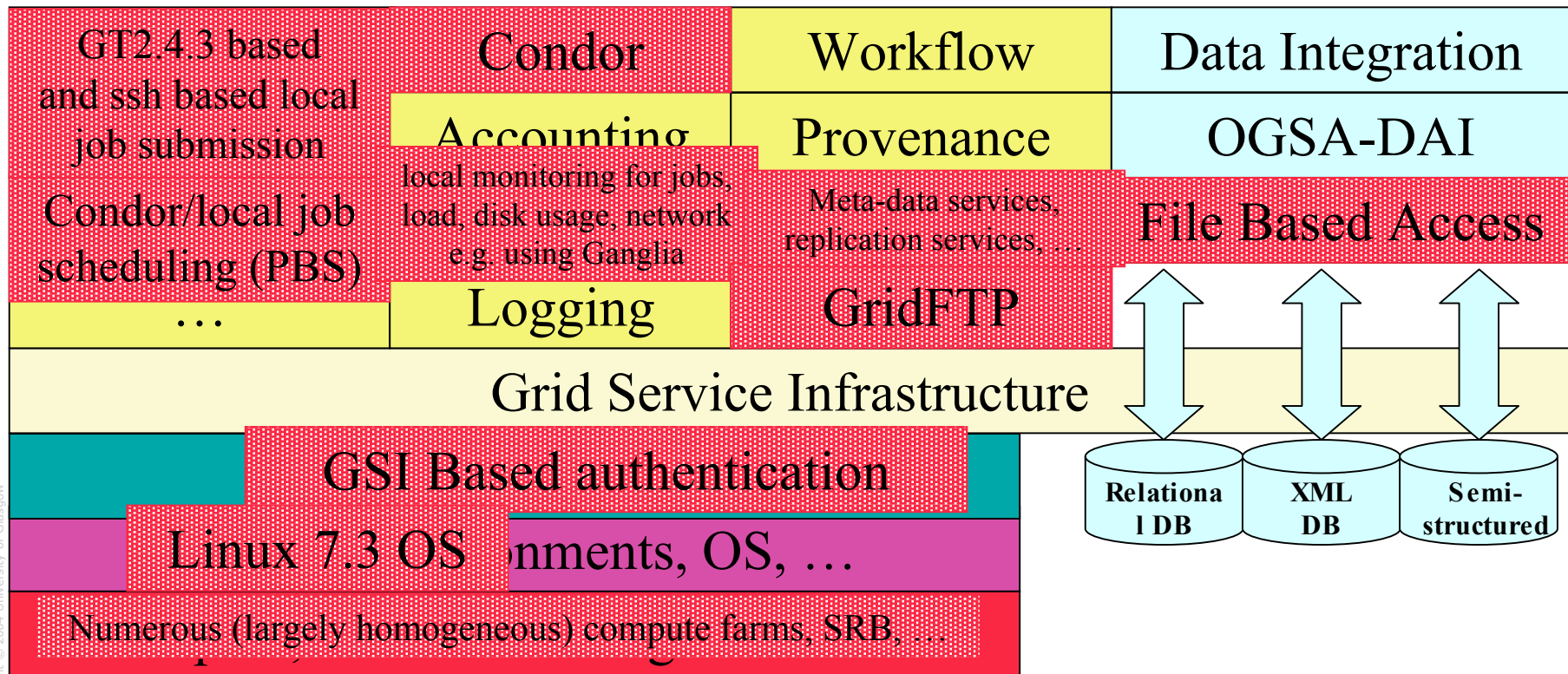
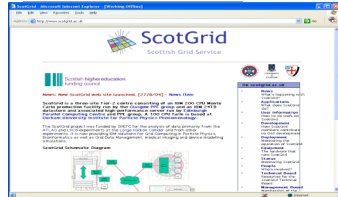
- Why is development of OGSA problematic?
 - Numerous groups working concurrently doing things that have never been done before
- Consider defining “policy service”
 - What resources for what people in what context
 - Could be done via scheduling services?
 - Could also be done via resource monitoring and allocation services?
 - Could be done through security services?
 - Could engineer applications to achieve this directly?
 - In short many ways to address the problem – there is no “correct solution”
 - OGSA trying to know to come up with a framework that allows choices to be made

Conceptualisation of OGSA Framework



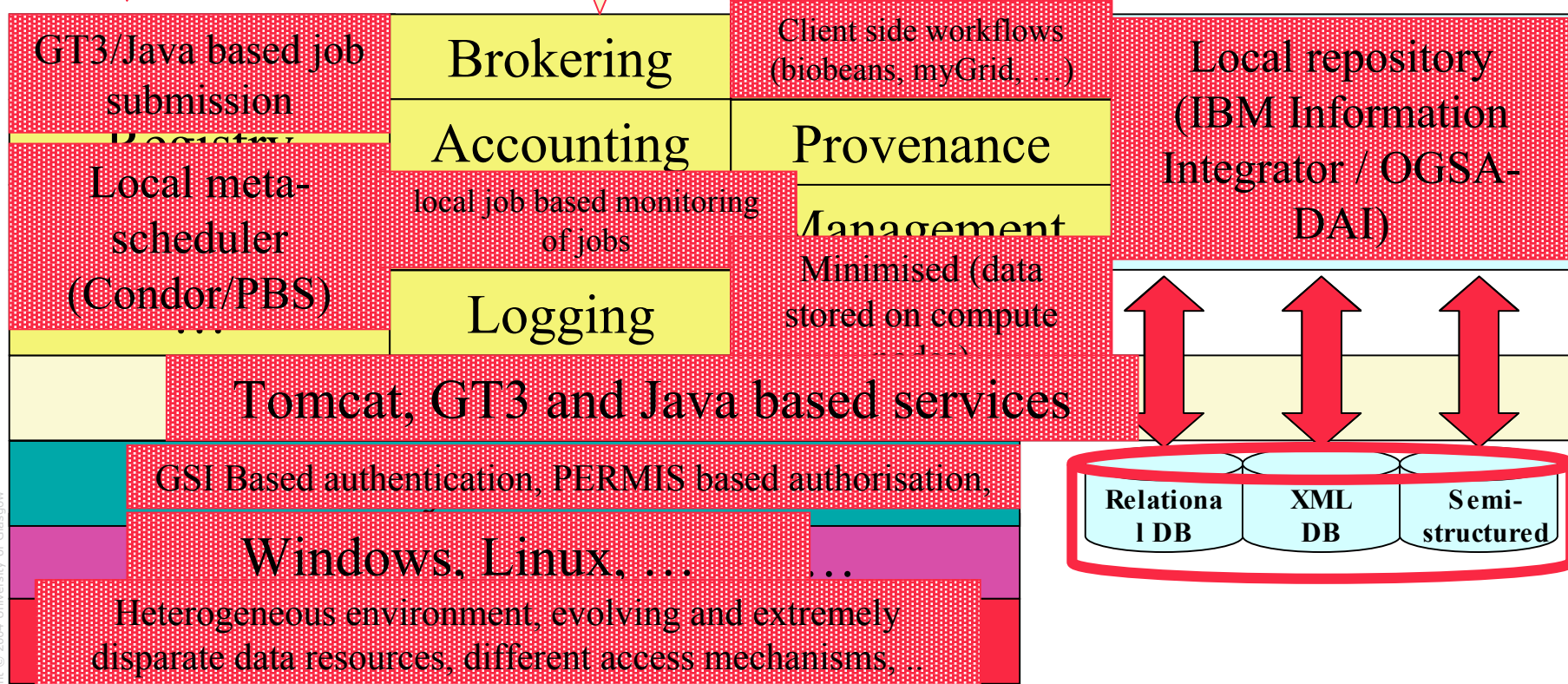
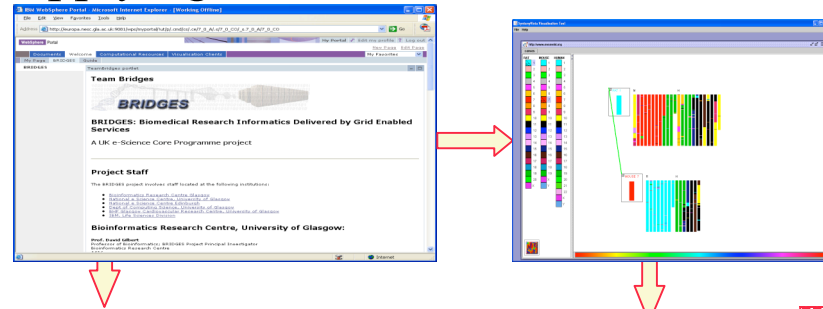
Examples of OGSA

Applying OGSA for High Energy Physics Grids



Examples of OGSA

Applying OGSA for Bioinformatics Life Sciences



OGSA Conclusions

- Very much a work in progress
 - Challenges in developing standards in complex area
 - Challenges in software engineering, paradigm shifts
 - Challenges in understanding scientific domain to help shape software/standards
- Not really a standard, more a set of guidelines
 - Issues of conformance, compliance, consistency to OGSA architecture?
 - Only possible once detailed APIs established (WSDL)
 - Then “might” be possible to think about conformance test suites etc