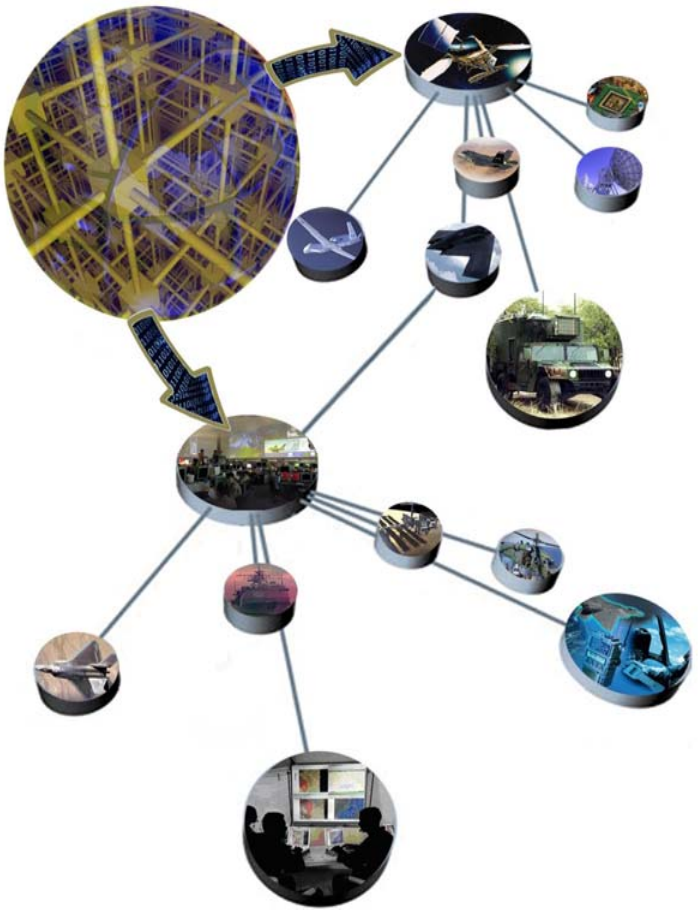
A Grid-of-Grids Service Architecture for Net-Centric Operations: Further Discussion



GSAW Manhattan Beach March 29 2006

Ground System Architectures Workshop

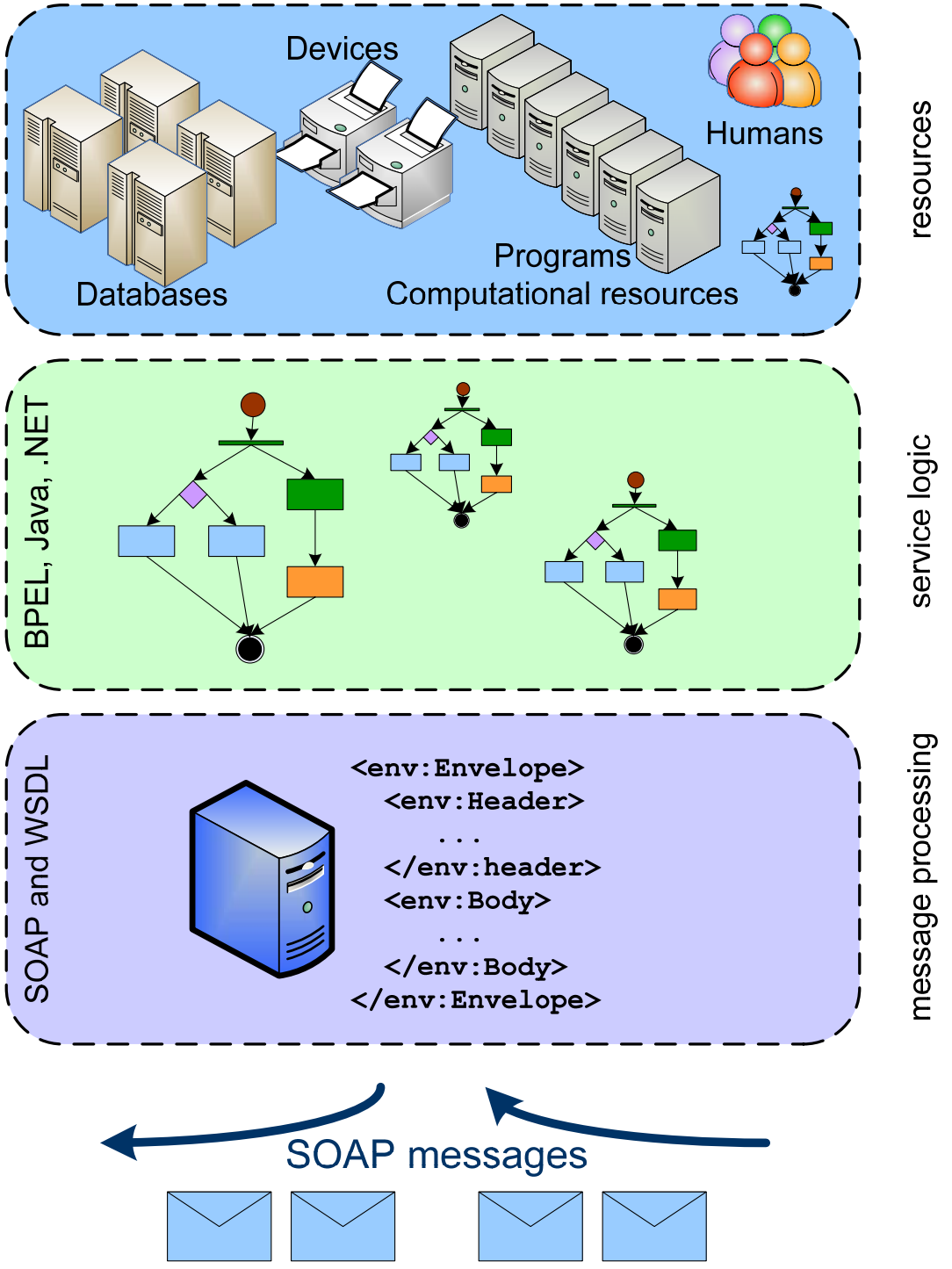
Geoffrey Fox

Anabas Inc. and
Computer Science, Informatics, Physics
Pervasive Technology Laboratories
Indiana University Bloomington IN 47401

gcf@indiana.edu
http://www.infomall.org

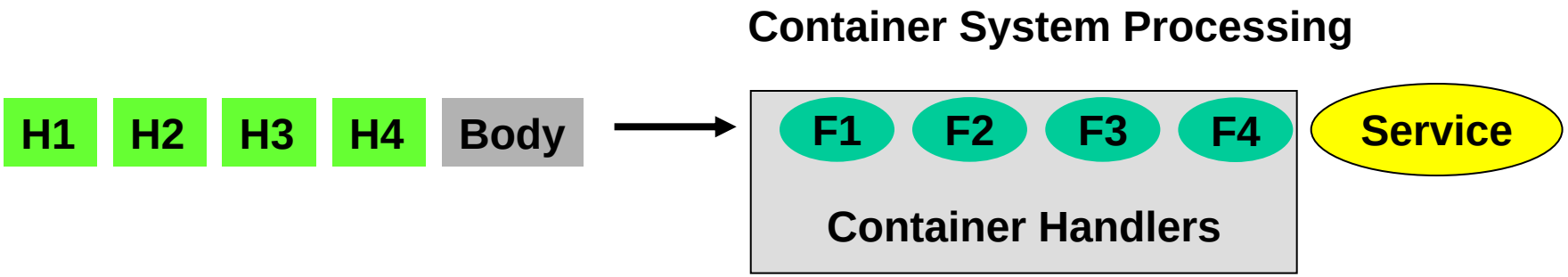
Web services

- **Web Services** build **loosely-coupled, distributed** applications, (wrapping existing codes and databases) based on the **SOA** (service oriented architecture) principles.
- Web Services interact by exchanging messages in **SOAP** format
- The contracts for the message exchanges that implement those interactions are described via **WSDL** interfaces.



What do Web Services Prescribe?

- The specify interfaces for system services (and generally useful services like database)
- They specify an interface language (WSDL) for all services
- They develop containers and frameworks to use to host services
- They specify a message format (SOAP) for ALL messages that defines both application and system actions precisely
- They imply a process be started to define domain specific services
- There are multiple competing activities from Microsoft and IBM to Apache, and IU (for example) developing system and application services
- Unlike for RTI and CORBA, services from different vendors should interoperate



Internet Scale Distributed Services

- Grids use Internet technology and are distinguished by managing or organizing sets of network connected resources
  - Classic Web allows independent one-to-one access to individual resources
  - Grids integrate together and manage multiple Internet-connected resources: People, Sensors, computers, data systems
- Organization can be explicit as in
  - TeraGrid which federates many supercomputers;
  - Information Retrieval Grid which federates multiple data resources;
  - CrisisGrid which federates first responders, commanders, sensors, GIS, (Tsunami) simulations, science/public data
- Organization can be implicit as in Internet resources such as curated databases and simulation resources that "harmonize a community"

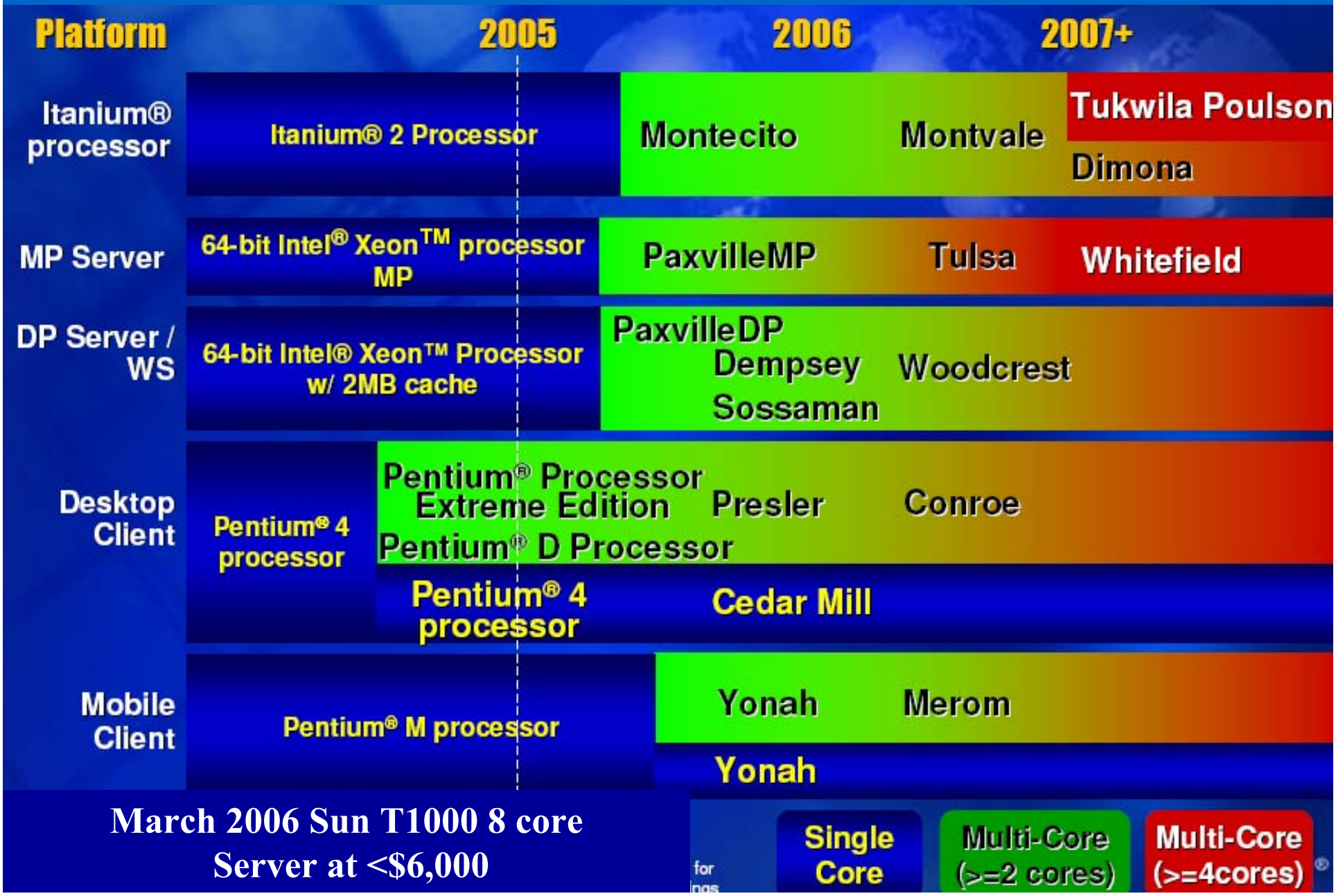
Different Visions of the Grid

- e-Science or Cyberinfrastructure are virtual organization Grids supporting global distributed engineering and science research (note sensors, instruments are people are all distributed)
- Utility Computing or X-on-demand (X=data, computer ..) is a major computer Industry interest in Grids and this is key part of enterprise or campus Grids
- Skype (Kazaa) VOIP system is a Peer-to-peer Grid (and VRVS/GlobalMMCS like Internet A/V conferencing are Collaboration Grids)
- DoD's vision of Network Centric Computing can be considered a Grid (linking sensors, warfighters, commanders, backend resources) and they are building the GIG (Global Information Grid)
- Commercial 3G Cell-phones and DoD ad-hoc network initiative are forming mobile Grids
- Grids support universal Globalization in life, fun, research, business

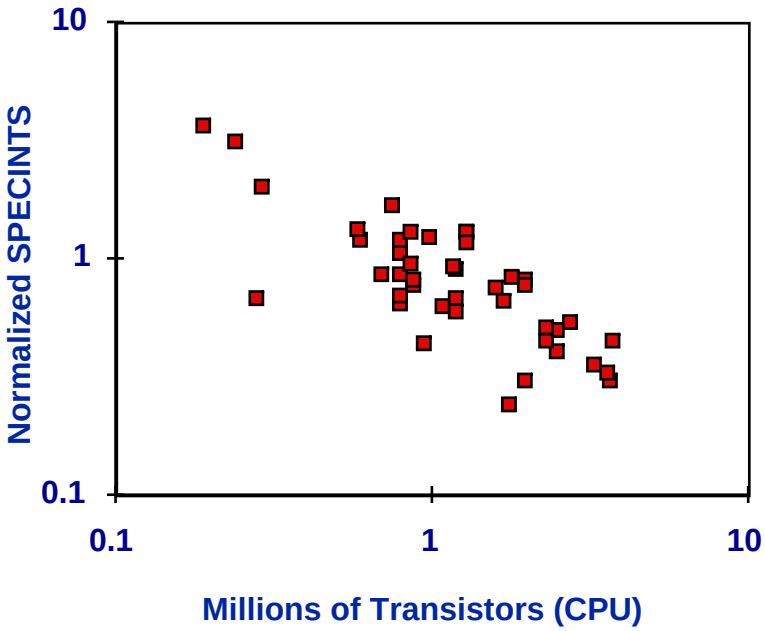
Why use SOA's

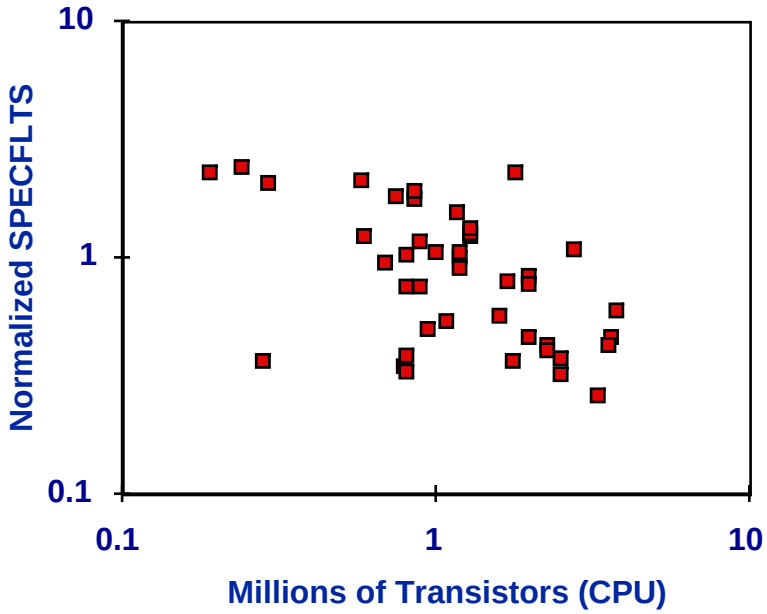
- **Globalization of applications:** Life, Fun, Research, Business, Defense as an International collaborative activity
- **Globalization of Software Production:** Software components including open-source made everywhere
- **Interoperability:** in interfaces and protocol (messages) requires Web Services as only broadly supported SOA
- **Anti-Performance:** if Moore's law gives you a factor X, then use √X for performance, √ X for improved lifecycle (re-use)
- **Software Engineering:** Software paradigms are ways of "packaging" modules/components/objects/methods/subroutines. Services have minimal coupling and best re-use (lowest performance). 1962 Fortran easier re-use than 2006 Java
- **Multicore chips:** requires pervasive concurrency without side effects. Even Microsoft must be able to use 32-128 way parallelism on a chip over next 5 years

Intel Fall 2005 Multicore Roadmap



Performance Per Transistor





- Performance data from uP vendors
- Transistor count excludes on-chip caches
- Performance normalized by clock rate
- Conclusion: Simplest is best! (250K Transistor CPU)

What is Happening?

- Grid ideas are being developed in (at least) four communities
  - Web Service – W3C, OASIS, (DMTF)
  - Global Grid Forum (High Performance Computing, e-Science)
  - Enterprise Grid Alliance (Commercial "Grid Forum" with a near term focus)
- Service Standards are being debated
- Grid Operational Infrastructure is being deployed
- Grid Architecture and core software being developed
  - Apache has several important projects as do academia; large and small companies
- Particular System Services are being developed "centrally" – OGSA framework for this in GGF; WS-* for OASIS/W3C/Microsoft-IBM
- Lots of fields are setting domain specific standards and building domain specific services
- USA started but now Europe is probably in the lead and Asia will soon catch USA if momentum (roughly zero for USA) continues

What do Grids Add?

- Grids use all of the Web Services
- They address management and deployment of large distributed systems of services
  - Internet Scale Distributed Services
  - I will use Grid more simply as a composable coordinated collection of services
- They address security and management issues of virtual organizations crossing multiple administrative domains
- GGF is developing specific services of relevance including job management, many aspects of data and scheduling
  - Not much on sensors, real-time, P2P
- GGF has a good process for developing new higher level specifications

Sources of Grid Technology

- Grids support distributed collaboratories or virtual organizations integrating concepts from
- The Web
- Agents
- Distributed Objects (CORBA Java/Jini COM)
- Globus, Legion, Condor, NetSolve, Ninf and other High Performance Computing activities
- Peer-to-peer Networks
- With perhaps the Web and P2P networks being the most important for "Information Grids" and Globus for "Compute/File Grids"

Philosophy of Web Service Grids

- Much of Distributed Computing was built by natural extensions of computing models developed for sequential machines
- This leads to the distributed object (DO) model represented by Java and CORBA
  - RPC (Remote Procedure Call) or RMI (Remote Method Invocation) for Java
- Key people think this is not a good idea as it scales badly and ties distributed entities together too tightly
  - Distributed Objects Replaced by Services
- Note CORBA was considered too complicated in both organization and proposed infrastructure
  - and Java was considered as "tightly coupled to Sun"
  - So there were other reasons to discard
- Thus replace distributed objects by services connected by "one-way" messages and not by request-response messages

Some ideas to Remember

- Grids are managed Web Services exchanging Messages
- P2P Networks are differently managed and architected services exchanging messages
- Any computer operation involves messages; not all these messages can be isolated
  - With services all messages are explicit and can be examined
- Grid Services extend WS-* Web Service Specifications
- Web Service container replaces computer
- Service replaces process
- A stream is an ordered set of messages
- Service Internet replaces Internet: messages replace packets
- (Sub)Grids replace Libraries

The Grid and Web Service Institutional Hierarchy

4: Application or Community of Interest (CoI) Specific Services such as “Map Services”, “Run BLAST” or “Simulate a Missile”	**XBML XTCE VOTABLE CML CellML**
3: Generally Useful Services and Features (OGSA and other GGF, W3C) Such as “Collaborate”, “Access a Database” or “Submit a Job”	**OGSA GS-* and some WS-* GGF/W3C/….**
2: System Services and Features (WS-* from OASIS/W3C/Industry) Handlers like WS-RM, Security, UDDI Registry	**WS-* from OASIS/W3C/ Industry**
1: Container and Run Time (Hosting) Environment (Apache Axis, .NET etc.)	**Apache Axis .NET etc.**

Must set standards to get interoperability

The Ten areas covered by the 60 core WS-* Specifications

WS-* Specification Area	Examples
1: Core Service Model	XML, WSDL, SOAP
2: Service Internet	WS-Addressing, WS-MessageDelivery; Reliable Messaging WSRM; Efficient Messaging MOTM
3: Notification	WS-Notification, WS-Eventing (Publish-Subscribe)
4: Workflow and Transactions	BPEL, WS-Choreography, WS-Coordination
5: Security	WS-Security, WS-Trust, WS-Federation, SAML, WS-SecureConversation
6: Service Discovery	UDDI, WS-Discovery
7: System Metadata and State	WSRF, WS-MetadataExchange, WS-Context
8: Management	WSDM, WS-Management, WS-Transfer
9: Policy and Agreements	WS-Policy, WS-Agreement
10: Portals and User Interfaces	WSRP (Remote Portlets)

RTI and NCOW needs all of these?

Activities in Global Grid Forum Working Groups

GGF Area	GS-* and OGSA Standards Activities
1: Architecture	High Level Resource/Service Naming (level 2 of slide 6), Integrated Grid Architecture
2: Applications	Software Interfaces to Grid, Grid Remote Procedure Call, Checkpointing and Recovery, Interoperability to Job Submittal services, Information Retrieval,
3: Compute	Job Submission, Basic Execution Services, Service Level Agreements for Resource use and reservation, Distributed Scheduling
4: Data	Database and File Grid access, Grid FTP, Storage Management, Data replication, Binary data specification and interface, High-level publish/subscribe, Transaction management
5: Infrastructure	Network measurements, Role of IPv6 and high performance networking, Data transport
6: Management	Resource/Service configuration, deployment and lifetime, Usage records and access, Grid economy model
7: Security	Authorization, P2P and Firewall Issues, Trusted Computing

The Global Information Grid Core Enterprise Services

Core Enterprise Services	Service Functionality
CES1: Enterprise Services Management (ESM)	including life-cycle management
CES2: Information Assurance (IA)/Security	Supports confidentiality, integrity and availability. Implies reliability and autonomic features
CES3: Messaging	Synchronous or asynchronous cases
CES4: Discovery	Searching data and services
CES5: Mediation	Includes translation, aggregation, integration, correlation, fusion, brokering publication, and other transformations for services and data. Possibly agents
CES6: Collaboration	Provision and control of sharing with emphasis on synchronous real-time services
CES7: User Assistance	Includes automated and manual methods of optimizing the user GiG experience (user agent)
CES8: Storage	Retention, organization and disposition of all forms of data
CES9: Application	Provisioning, operations and maintenance of applications.

The Core Service Areas I

Service or Feature	WS-*	GS-*	NCES (DoD)	Comments
A: Broad Principles				
FS1: Use SOA: Service Oriented Arch.	WS1			Core Service Model, Build Grids on Web Services. Industry best practice
FS2: Grid of Grids				Strategy for legacy subsystems and modular architecture
B: Core Services				
FS3: Service Internet, Messaging	WS2		NCES3	Streams/Sensors
FS4: Notification	WS3		NCES8	JMS, MQSeries
FS5 Workflow	WS4		NCES5	Grid Programming
FS6 : Security	WS5	GS7	NCES2	Grid-Shib, Permis Liberty Alliance ...
FS7: Discovery	WS6		NCES4	
FS8: System Metadata & State	WS7			Globus MDS Semantic Grid
FS9: Management	WS8	GS6	NCES1	CIM
FS10: Policy	WS9		ECS	

The Core Service Areas II

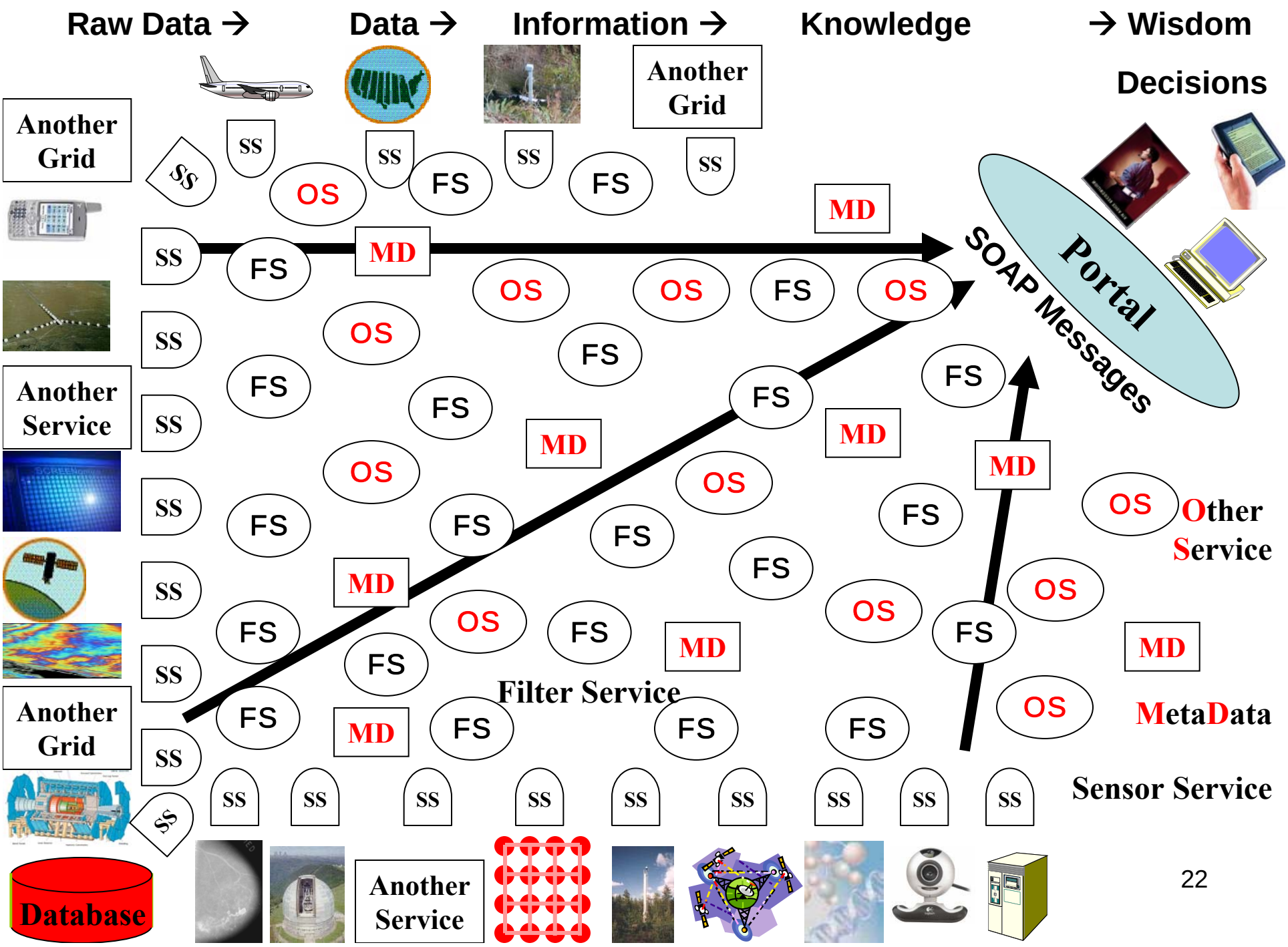
Service or Feature	WS-*	GS-*	NCES	Comments
B: Core Services (Continued)				
FS11: Portals and User assistance	WS10		NCES7	Portlets JSR168, NCES Capability Interfaces
FS12: Computing		GS3		
FS13: Data and Storage		GS4	NCES8	NCOW Data Strategy
FS14: Information		GS4		JBI for DoD, WFS for OGC
FS15: Applications and User Services		GS2	NCES9	Standalone Services Proxies for jobs
FS16: Resources and Infrastructure		GS5		Ad-hoc networks
FS17: Collaboration and Virtual Organizations		GS7	NCES6	XGSP, Shared Web Service ports
FS18: Scheduling and matching of Services and Resources		GS3		

Some Conclusions I

- One can map 7.5 out of 9 NCOW/NCE and GiG core capabilities into Web Service (WS-*) and Grid (GS-*) architecture and core services
  - Analysis of Grids in NCOW/NCE document inaccurate (confuse Grids and Globus and only consider early activities)
- Some "mismatches" on both NCOW and Grid sides
- GS-*/WS-* do not have collaboration and miss some messaging
- NCOW does not have at core level system metadata and resource/service scheduling and matching
- Higher level services of importance include GIS (Geographical Information Systems), Sensors and data-mining

Some Conclusions II

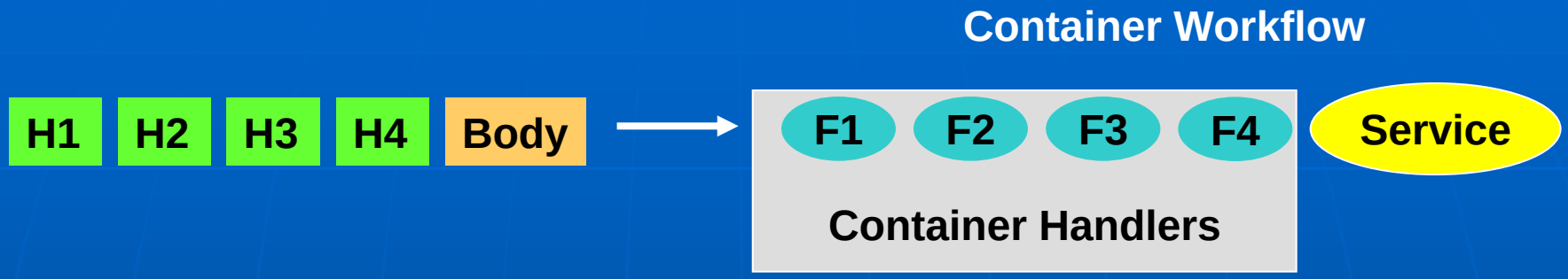
- **Criticisms of Web services in a recent paper by Birman seem to be addressed by Grids or reflect immaturity of initial technology implementations**
- **NCOW/NCE does not seem to have any analysis of how to build their systems on WS-*/GS-* technologies in a layered fashion; they do have a layered service architecture so this can be done**
  - **They agree with service oriented architecture**
  - **They seem to have no process for agreeing to WS-* GS-* or setting other standards for CES**
- **Grid of Grids allows modular architectures and natural treatment of legacy systems**
  - **Note Grids, Services and Handlers are all "just" entities with distributed message-based input and output interfaces**


Raw Data →
Data →
Information →
Knowledge
→ Wisdom
Decisions
Another Grid
Another Grid
Another Service
Another Grid
Another Service
Database
SS
FS
OS
MD
Portal
SOAP Messages
Other Service
MetaData
Sensor Service
Filter Service

Semantic Grid and Services

- Implications of SOA (Service Oriented Architectures) for SG (Semantic Grid)
  - Build services to implement SG
- Implications of SG for SOA
  - Build metadata rich systems of services using SG
- Services receive data in SOAP messages, manipulate it and produce transformed data as further messages
- Meta-data is carried in SOAP messages
- Meta-data controls processing and transport of SOAP Messages
- Knowledge is created from data by services
- The Grid enhances Web services with semantically rich system and application specific management
- One must exploit and work around the different approaches to meta-data and their manipulation in Web Services

Structure of SOAP Messages



- SOAP Messages have System information in the header including WS-Policy based meta-data defining processing options
  - Processed by Handlers
- Application data and meta-data is the body (controversies here!)
  - Processed by the Service itself
- Some meta-data like WS-RF is logically "only in messages"
- Other like that in WS-Context or the SRB are stored in logical equivalent of XML databases
- We only need to preserve semantic structure (XML/SOAP Infoset) so transport in fast XML and store in efficient relational databases

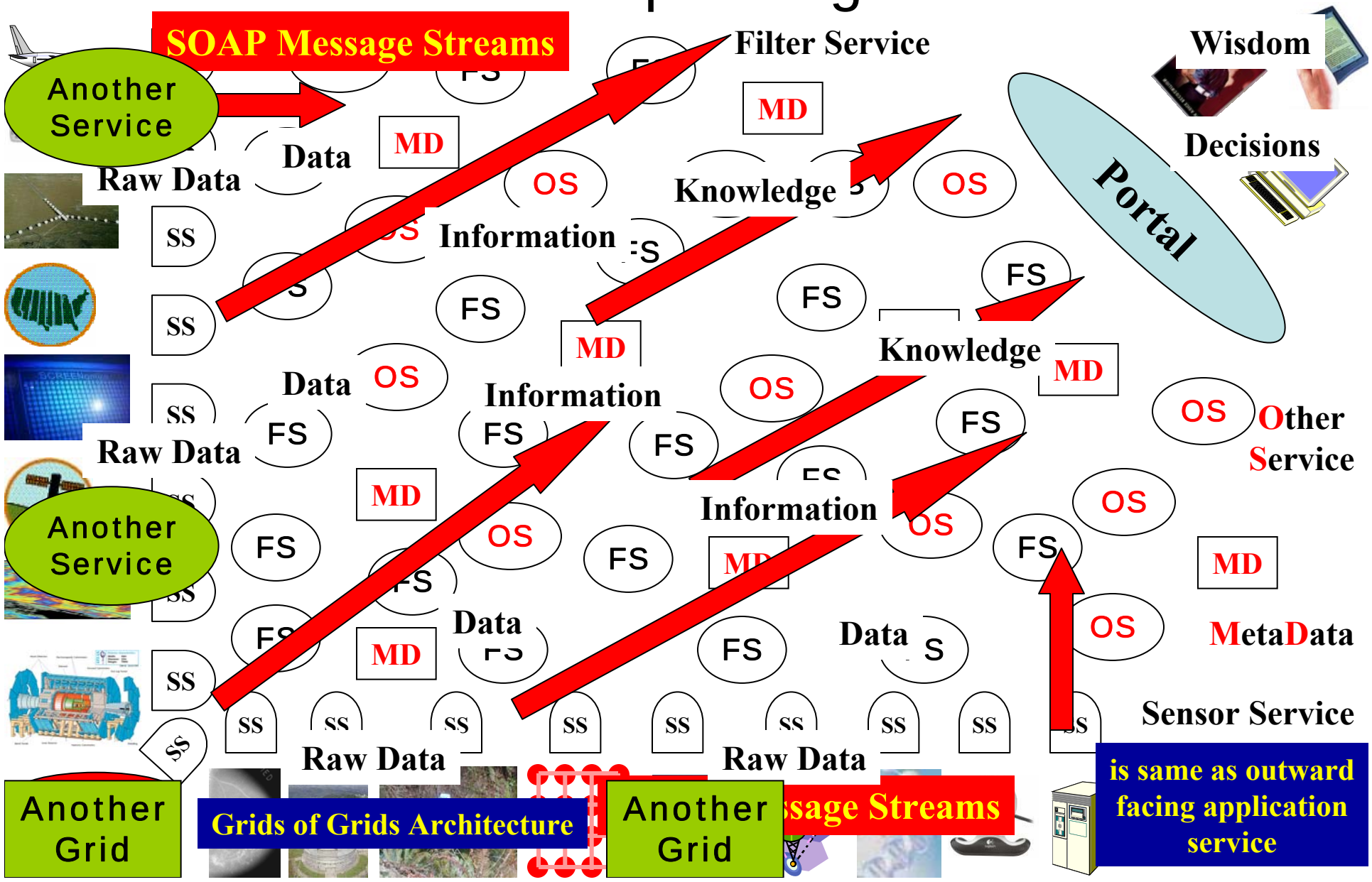
What Type of Services are there?

- There are a horde of support services supplying security, collaboration, database access, user interfaces
- The support services are either associated with system or application
  - We studied the WS-* and GS-* which implicitly or explicitly define many support services
- There are generalized filter services which are applications that accept messages and produce new messages with some data derived from that in input
  - Simulations (including PDE's and reactive systems)
  - Data-mining
  - Transformations
  - Agents
  - Reasoning are all termed filters here
- There are services like "author ontology", "parse RDF" or "attach provenance" that directly support Semantic Grid
- But all services and their interactions are bathed in sea of meta-data and so implicitly need and support the Semantic Grid

It's a Composite Hierarchical World

- Filters can be a workflow which means they are "just collections of other simpler services"
  - One needs meta-data to control the workflow
- Services are programs that accept messages and produce messages
- Grids are a distributed collection of services supporting managed shared resources
  - Management requires meta-data
- Grids are distributed systems that accept distributed messages and produce distributed result messages
  - Can always talk about Grids and view a service or a workflow as a special case of a Grid
- It just requires meta-data to send a message to a Grid and it routed to "correct computer" holding "requested service"
  - Meta-data allows mapping of virtual to real addresses

Semantically Rich Services with a Semantically Rich Distributed Operating Environment

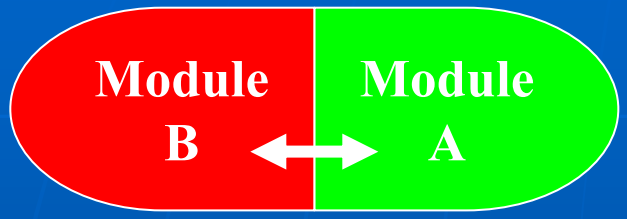


Consequences of Rule of the Millisecond

- Useful to remember critical time scales
  - 1) 0.000001 ms – CPU does a calculation
  - 2a) 0.001 to 0.01 ms – Parallel Computing MPI latency
  - 2b) 0.001 to 0.01 ms – Overhead of a Method Call
  - 3) 1 ms – wake-up a thread or process
  - 4) 10 to 1000 ms – Internet delay
- 2a), 4) implies geographically distributed metacomputing can't in general compete with parallel systems
- 3) << 4) implies a software overlay network is possible without significant overhead
  - We need to explain why it adds value of course!
- 2b) versus 3) and 4) describes regions where method and message based programming paradigms important

Linking Modules

Closely coupled Java/Python …



Method Calls
.001 to 1 millisecond

Coarse Grain Service Model

0.1 to 1000 millisecond latency

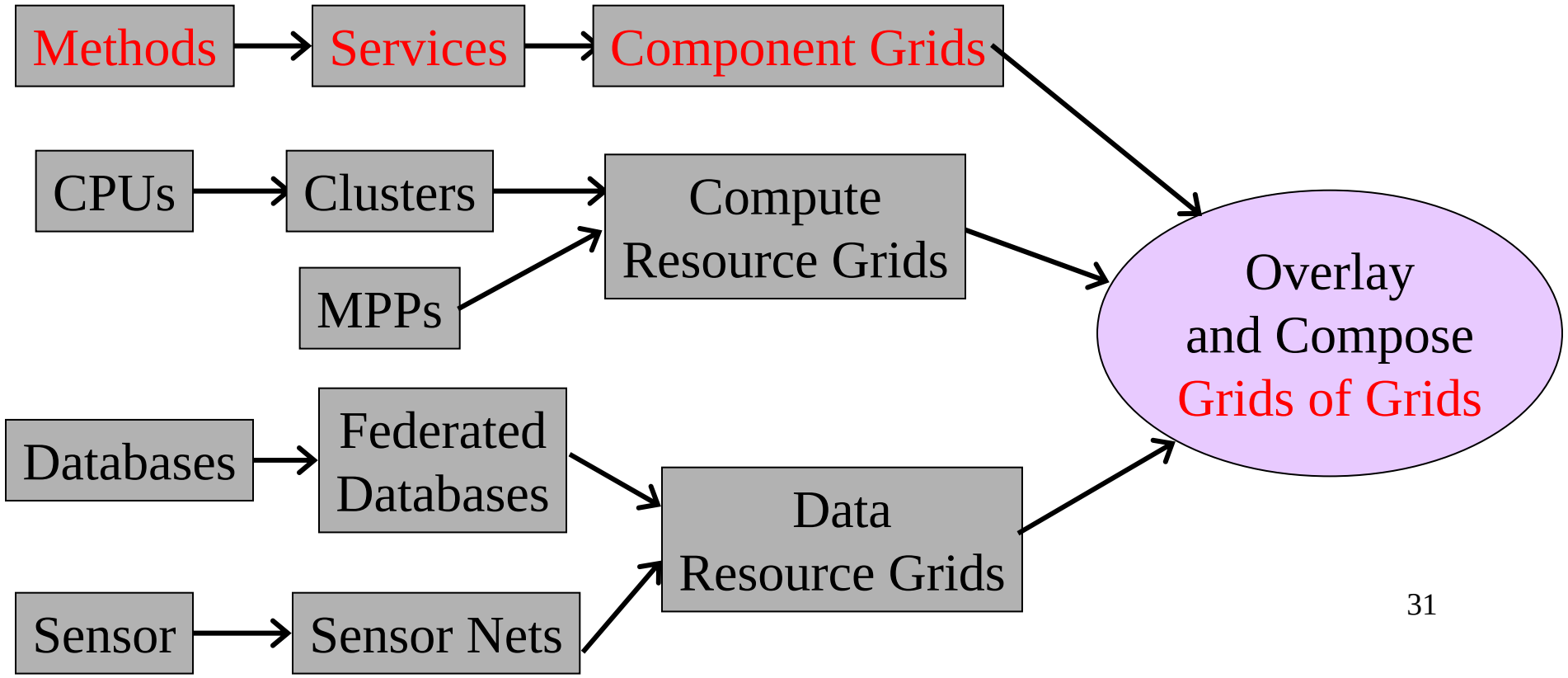
- **From method based to RPC to message based to event-based publish-subscribe Message Oriented Middleware**

What is a Simple Service?

- Take any system – it has multiple functionalities
  - We can implement each functionality as an independent distributed service
  - Or we can bundle multiple functionalities in a single service
- Whether functionality is an independent service or one of many method calls into a "glob of software", we can always make them as Web services by converting interface to WSDL
- Simple services are gotten by taking functionalities and making as small as possible subject to "rule of millisecond"
  - Distributed services incur messaging overhead of one (local) to 100's (far apart) of milliseconds to use message rather than method call
  - Use scripting or compiled integration of functionalities ONLY when require <1 millisecond interaction latency
- Apache web site has many (pre Web Service) projects that are multiple functionalities presented as (Java) globs and NOT (Java) Simple Services
  - Makes it hard to integrate sharing common security, user profile, file access .. services

Grids of Grids of Simple Services

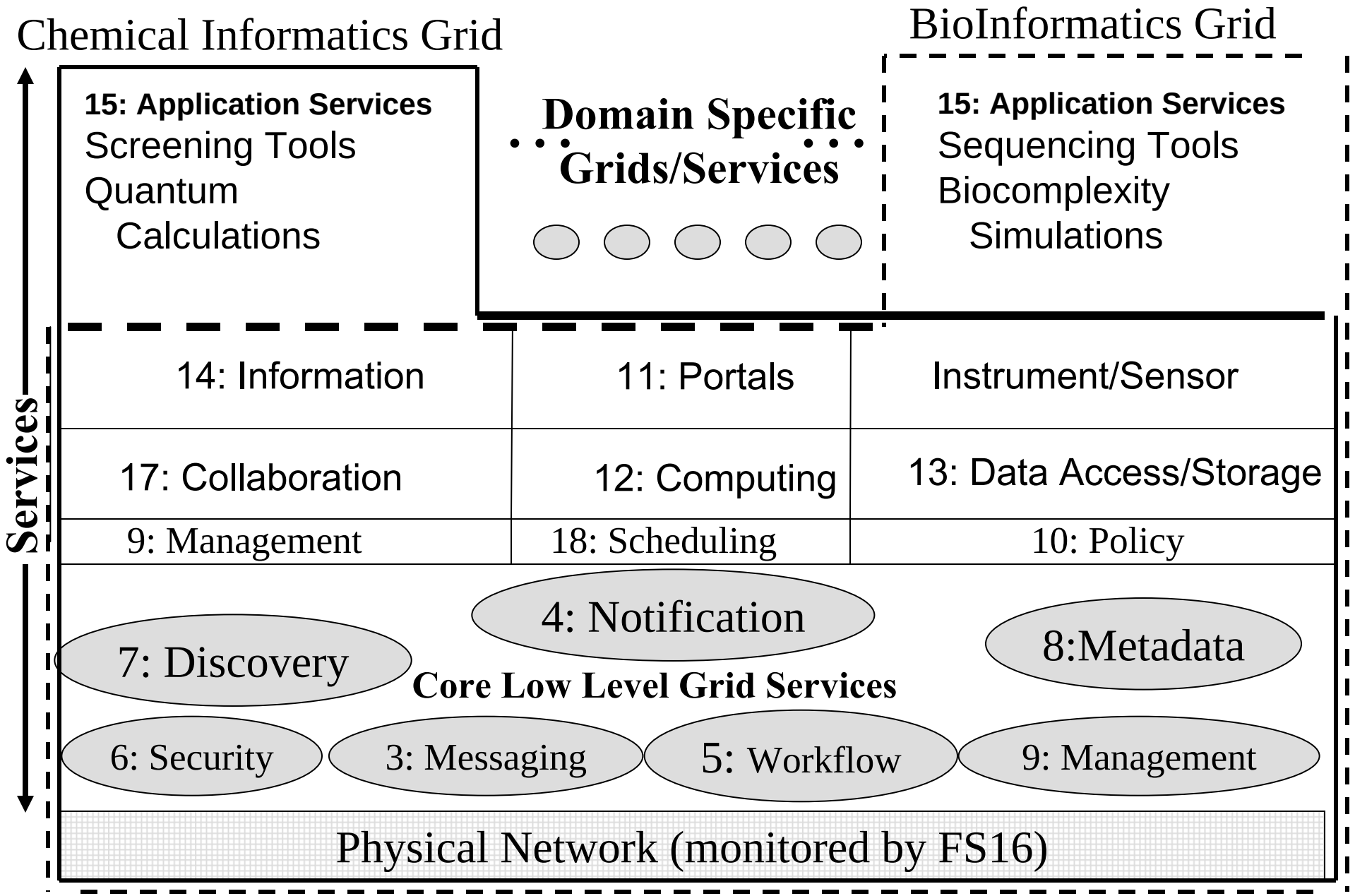
- Link via methods → messages → streams
- Services and Grids are linked by messages
- Internally to service, functionalities are linked by methods
- A simple service is the smallest Grid
- We are familiar with method-linked hierarchy
  Lines of Code → Methods → Objects → Programs → Packages

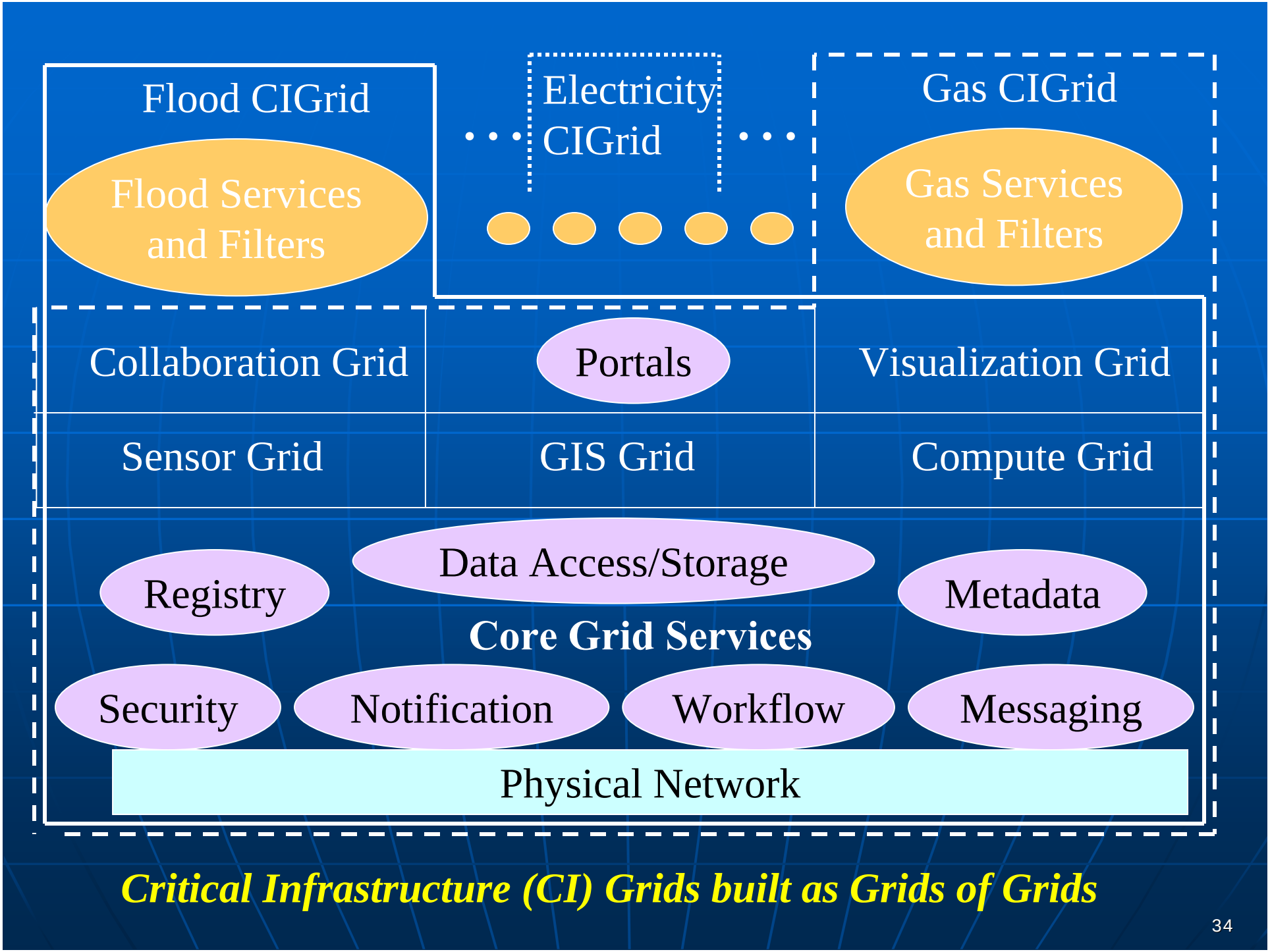


Component Grids?

- So we build collections of Web Services which we package as component Grids
  - Visualization Grid
  - Sensor Grid
  - Utility Computing Grid
  - Collaboration Grid
  - Earthquake Simulation Grid
  - Control Room Grid
  - Crisis Management Grid
  - Drug Discovery Grid
  - Bioinformatics Sequence Analysis Grid
  - Intelligence Data-mining Grid
- We build bigger Grids by composing component Grids using the Service Internet

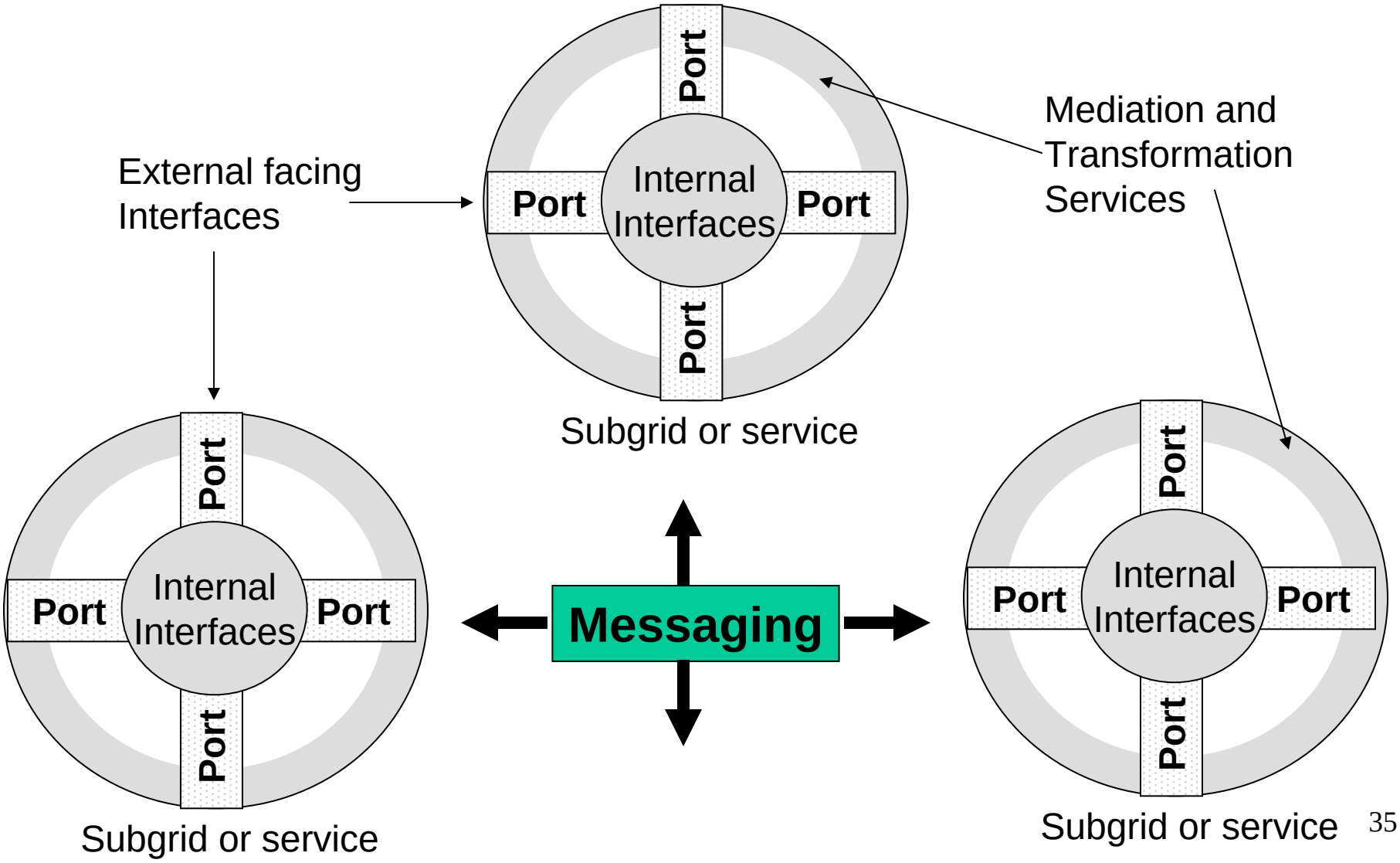
Using the Grid of Grids and Core Services to build multiple application grids re-using common components.



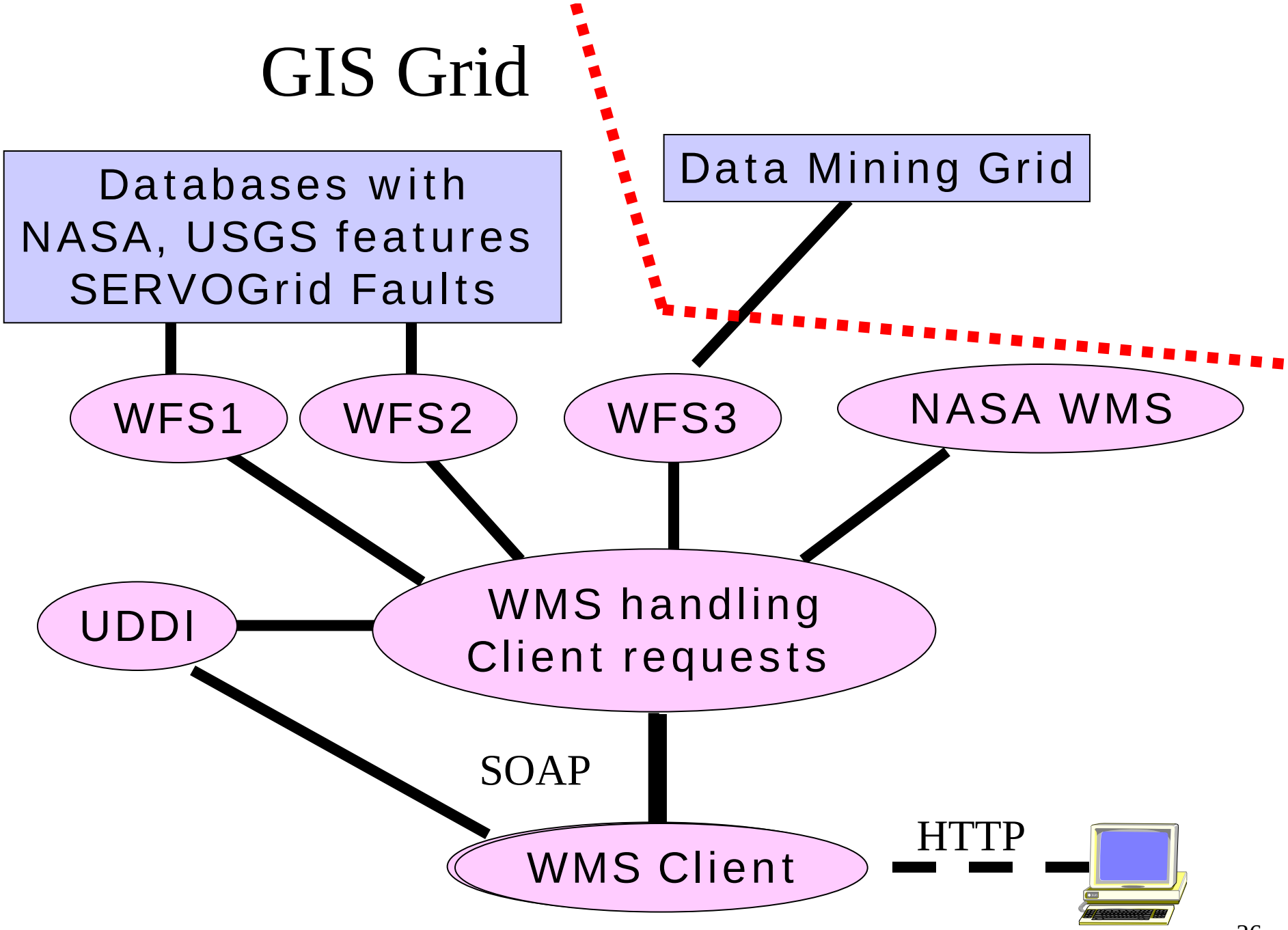


Critical Infrastructure (CI) Grids built as Grids of Grids

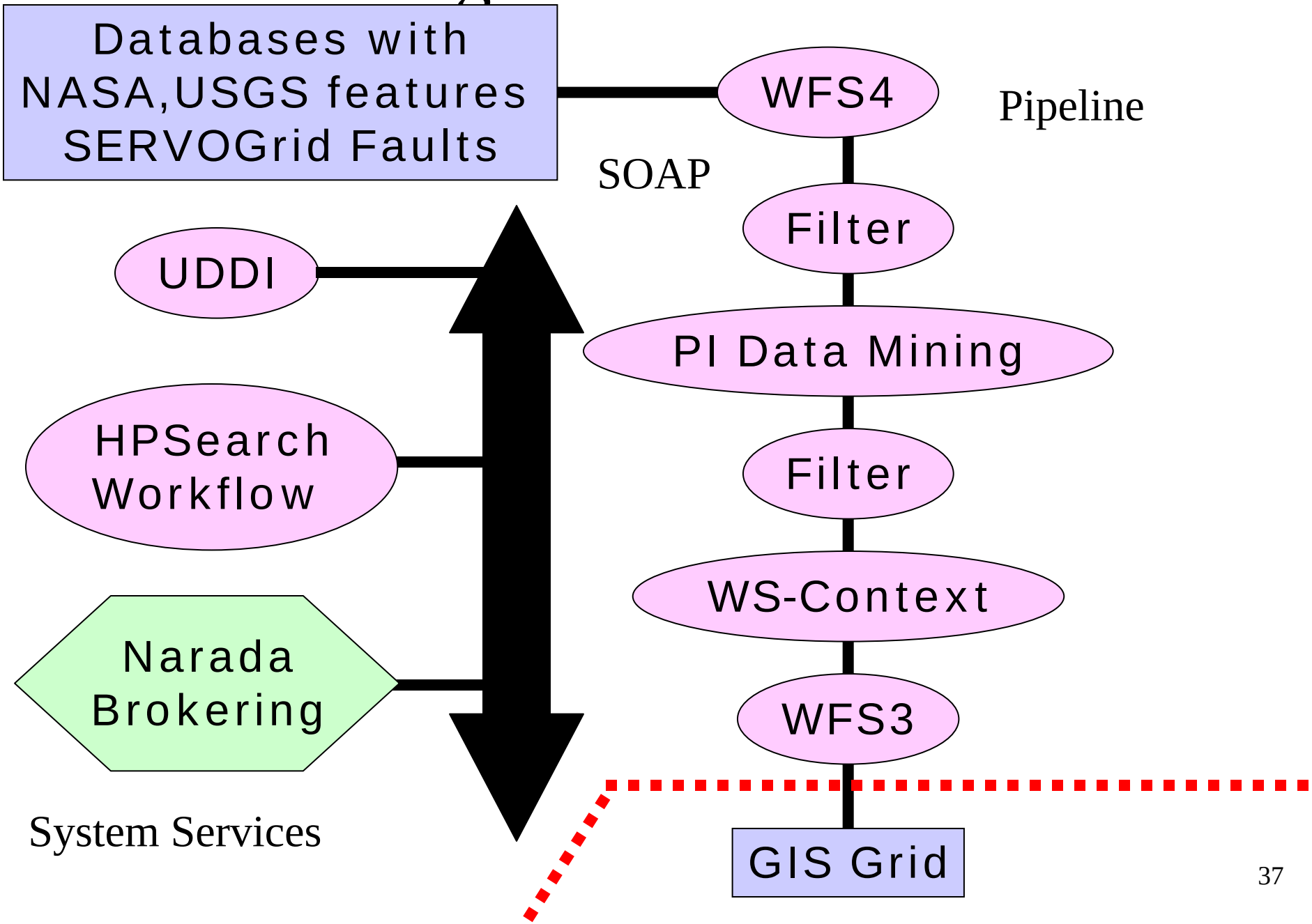
Mediation and Transformation in a Grid of Grids and Simple Services

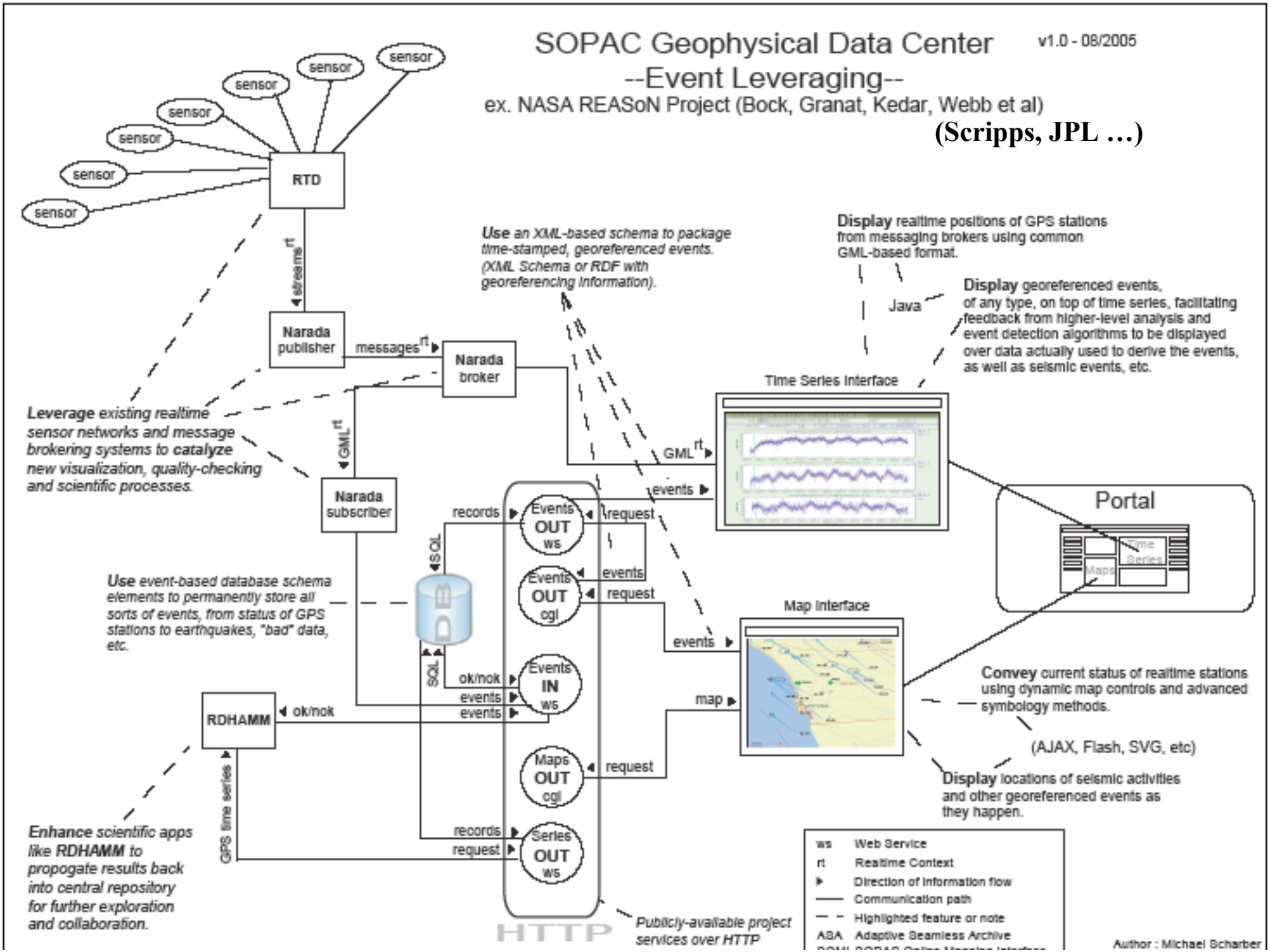


GIS Grid


Databases with
NASA, USGS features
SERVOGrid Faults
Data Mining Grid
WFS1
WFS2
WFS3
NASA WMS
UDDI
WMS handling
Client requests
SOAP
WMS Client
HTTP

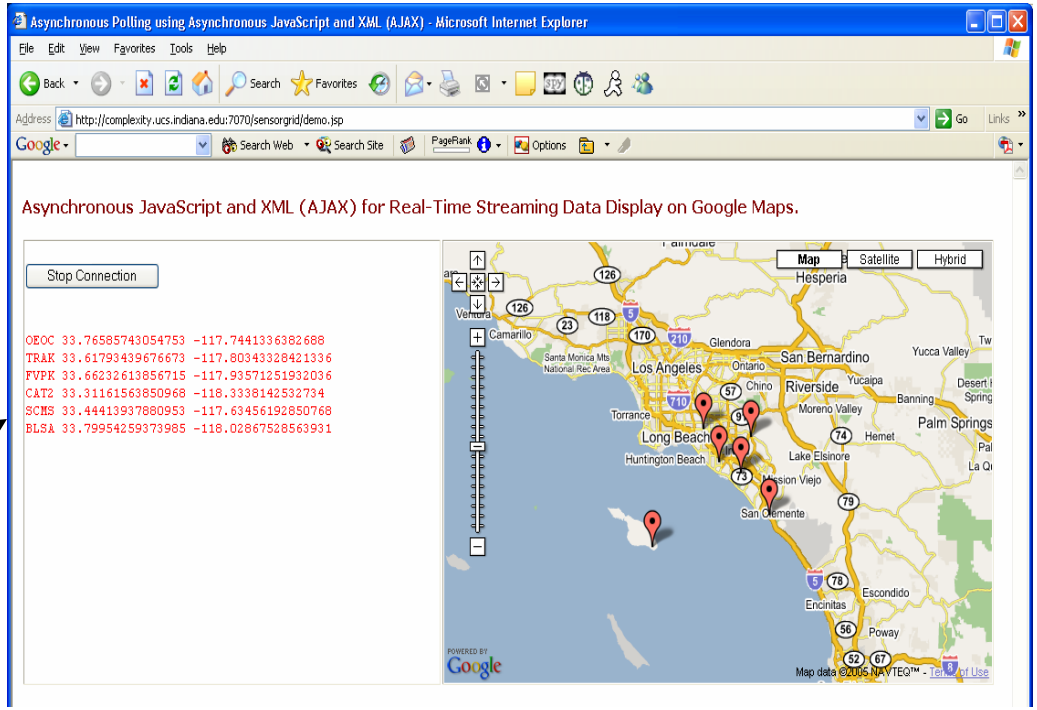
Data Mining Grid in Grid of Grids

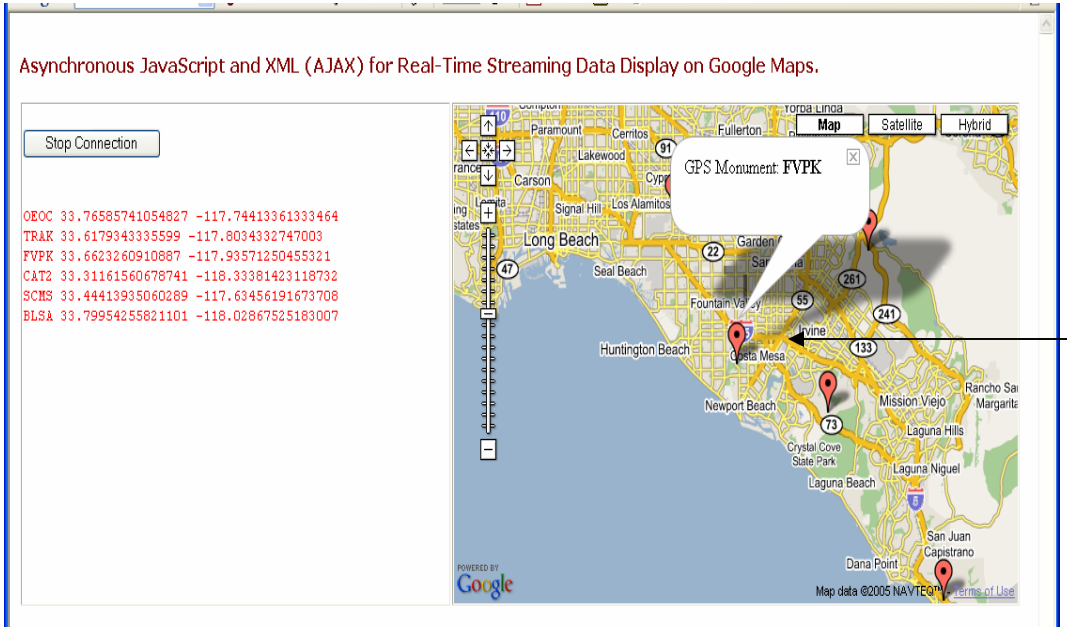



SOPAC Geophysical Data Center
v1.0 - 08/2005
--Event Leveraging--
ex. NASA REASoN Project (Bock, Granat, Kedar, Webb et al)
(Scripps, JPL …)
sensor
RTD
streams rt
Narada publisher
messages rt
Narada broker
GML rt
Narada subscriber
Use an XML-based schema to package time-stamped, georeferenced events. (XML Schema or RDF with georeferencing information).
Display realtime positions of GPS stations from messaging brokers using common GML-based format.
Java
Display georeferenced events, of any type, on top of time series, facilitating feedback from higher-level analysis and event detection algorithms to be displayed over data actually used to derive the events, as well as seismic events, etc.
Time Series Interface
Leverage existing realtime sensor networks and message brokering systems to catalyze new visualization, quality-checking and scientific processes.
GML rt
events
Portal
Time Series
Maps
records
request
Events OUT ws
Events OUT cgi
SQL
DB
Use event-based database schema elements to permanently store all sorts of events, from status of GPS stations to earthquakes, "bad" data, etc.
Map Interface
events
map
Events IN ws
ok/nok
RDHAMM
Convey current status of realtime stations using dynamic map controls and advanced symbology methods.
(AJAX, Flash, SVG, etc)
Maps OUT cgi
Display locations of seismic activities and other georeferenced events as they happen.
GPS time series
Series OUT ws
Enhance scientific apps like RDHAMM to propogate results back into central repository for further exploration and collaboration.
HTTP
Publicly-available project services over HTTP
ws Web Service
rt Realtime Context
Direction of information flow
Communication path
Highlighted feature or note
ASA Adaptive Seamless Archive
Author : Michael Scharber

Real Time GPS and Google Maps

Subscribe to live GPS station. Position data from SOPAC is combined with Google map clients.





Select and zoom to GPS station location, click icons for more information.

Some Grid Performance

- From Anabas Phase I SBIR
- Reduction of message delay jitter to a millisecond.
- Dynamic meta-data access latency reduced from seconds to milliseconds using web service context service.
- The messaging is distributed with each low end Linux node capable of supporting 500 users at a total bandwidth of 140 Mbits/sec with over 20,000 messages per second.
- Systematic use of redundant fault tolerance services supports strict user QoS requirements and fault tolerant Grid enterprise bus supports collaboration and information sharing at a cost that scales logarithmically with number of simultaneous users and resources.
- Supporting N users at the 0.5 Mbits/sec level each would require roughly (N/500)log(N/500) messaging servers to achieve full capability.

Some Next Steps

- **Anabas Phase II SBIR:**
- **Produce a Grid-based implementation for 9 CES for NCOW adding ECS (Environmental Control Services) and Metadata support (UDDI and WS-Context for C2IEDM etc.)**
- **Produce typical Collaboration, Sensor, Datamining and GIS Grids**
- **Produce a Tool to allow composition of services and grids into (larger) Grids (Systems of Systems)**
- **Community Grids Laboratory:**
- **Continue Grids for Earth Science and Sensors with JPL**
- **Build an HLA runtime RTI for distributed event simulation in terms of Grid technology (more extensive than XMSF which links Web services to HLA)**

Location of software for Grid Projects in Community Grids Laboratory

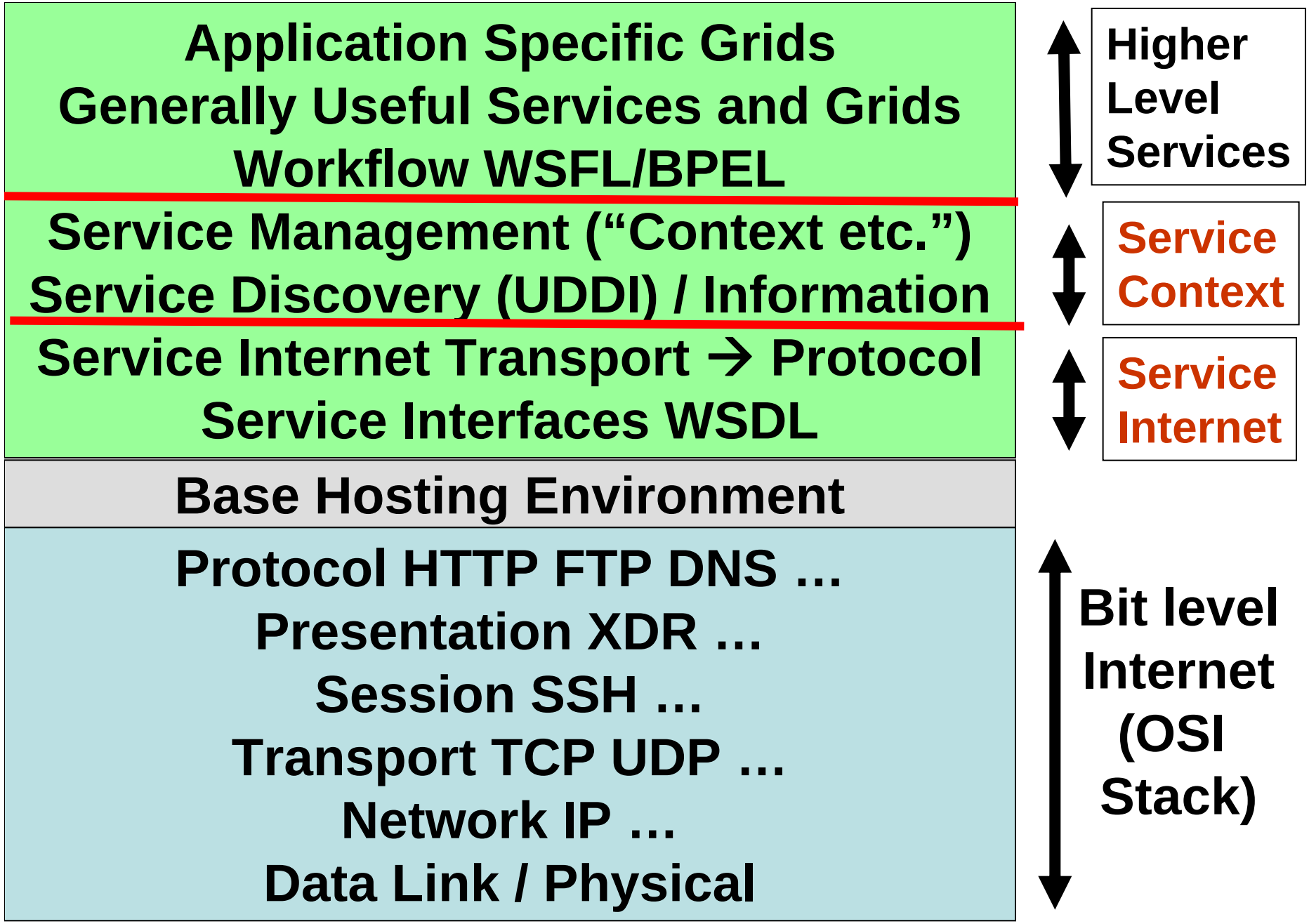
- htpp://www.naradabrokering.org provides Web service (and JMS) compliant distributed publish-subscribe messaging (software overlay network)
- htpp://www.globlmmcs.org is a service oriented (Grid) collaboration environment (audio-video conferencing)
- http://www.crisisgrid.org is an OGC (open geospatial consortium) Geographical Information System (GIS) compliant GIS and Sensor Grid (with POLIS center)
- http://www.opengrids.org has WS-Context, Extended UDDI etc.
- The work is still in progress but NaradaBrokering is quite mature
- All software is open source and freely available

A List of Web Services 1

- **1) Core Service Architecture**
- **XSD** XML Schema (W3C Recommendation) V1.0 February 1998, V1.1 February 2004
- **WSDL 1.1** Web Services Description Language Version 1.1, (W3C note) March 2001
- **WSDL 2.0** Web Services Description Language Version 2.0, (W3C under development) March 2004
- **SOAP 1.1** (W3C Note) V1.1 Note May 2000
- **SOAP 1.2** (W3C Recommendation) June 24 2003

A List of Web Services 2

- **2) Service Internet including messaging**
- **WS-Addressing** Web Services Addressing (BEA, IBM, Microsoft, SAP, Sun) in W3C consideration August 2004
- **WS-MessageDelivery** Web Services Message Delivery (W3C Submission by Oracle, Sun ..) April 2004
- **WS-Reliability** Web Services Reliable Messaging (OASIS Web Services Reliable Messaging TC) March 2004
- **WS-RM** Web Services Reliable Messaging (BEA, IBM, Microsoft, Tibco) v0.992 February 2005 linked to WS-Reliability in OASIS as Web Services Reliable Exchange (**WS-RX**)
- **WS-RM Policy** Web Services Reliable Messaging Policy Assertion (BEA, IBM, Microsoft, Tibco) March 2006
- **WS-RX** Web Services Reliable Exchange (Many members) integrating previous reliability specifications
- **SOAP MOTM** SOAP Message Transmission Optimization Mechanism (W3C) June 2004
- **SOAP-over-UDP** Binding of SOAP to UDP (Microsoft, BEA …) September 2004
- **Many obsolete specifications like WS-Routing and Referral** SOAP Routing Protocol (Microsoft) October 2001



Layered Architecture for Web Services and Grids

WS-* implies the Service Internet

- We have the classic (CISCO, Juniper ….) Internet routing the flood of ordinary packets in OSI stack architecture
- Web Services build the "Service Internet" or IOI (Internet on Internet) with
  - Routing via WS-Addressing not IP header
  - Fault Tolerance (WS-RM not TCP)
  - Security (WS-Security/SecureConversation not IPSec/SSL)
  - Data Transmission by WS-Transfer not HTTP
  - Information Services (UDDI/WS-Context not DNS/Configuration files)
  - At message/web service level and not packet/IP address level
- Software-based Service Internet possible as computers "fast"
- Familiar from Peer-to-peer networks and built as a software overlay network defining Grid (analogy is VPN)
- SOAP Header contains all information needed for the "Service Internet" (Grid Operating System) with SOAP Body containing information for Grid application service

A List of Web Services 3

- **3) Notification and high-level publish/subscribe information dissemination**
- **WS-Eventing** Web Services Eventing (BEA, Microsoft, TIBCO) August 2004
- **WS-EventNotification** (HP, IBM, Intel, Microsoft) March 2006 uses resources to manage subscriptions
- **WS-Notification** Framework for Web Services Notification with **WS-Topics**, **WS-BaseNotification**, and **WS-BrokeredNotification** (OASIS) OASIS Web Services Notification TC Set up March 2004
- **JMS** Java Message Service V1.1 March 2002

- Different from using publish-subscribe to robustly support messaging between Web services
  - Bind SOAP to JMS or MQSeries

A List of Web Services 4

- **4) Coordination and Workflow, Transactions and Contextualization**
- **BPEL** Business Process Execution Language for Web Services (OASIS) V1.1 May 2003 (V1.1) with V2.0 under development
- **WS-CDL** Web Services Choreography Language (W3C) V1.0 Working Draft 17 December 2004
- **WSCI** (W3C) Web Service Choreography Interface V1.0 (W3C Note from BEA, Intalio, SAP, Sun, Yahoo)
- **WSCL** Web Services Conversation Language (W3C Note) HP March 2002
- Workflow is general linkage between services; transactions are a critical special case
- Concept of workflow generalizes traditional workflow processes in business

A List of Web Services 4-Continued

- **4) Transactions, Business Processes and Contextualization**
- **WS-CAF** Web Services Composite Application Framework including **WS-CTX**, **WS-CF** and **WS-TXM** below (OASIS Web Services Composite Application Framework TC)
- **WS-CTX** Web Services Context (OASIS Web Services Composite Application Framework TC) V0.9.2 July 2005
- **WS-CF** Web Services Coordination Framework (OASIS Web Services Composite Application Framework TC) V0.1 April 2005
- **WS-TXM** Web Services Transaction Management (OASIS Web Services Composite Application Framework TC) including **WS-ACID** (V0.1 May 2005), **WS-BP** (Business Process V0.1 May 2005), **WS-LRA** (Long running action V0.1 May 2005)
- **WS-Coordination** Web Services Coordination (BEA, IBM, Microsoft) November 2004
- **WS-AtomicTransaction** Web Services Atomic Transaction (BEA, IBM, Microsoft) November 2004
- **WS-BusinessActivity** Web Services Business Activity Framework (BEA, IBM, Microsoft) November 2004
- **BTP** Business Transaction Protocol (OASIS) May 2002 with V1.1 November 2004
- **ebXML BPSS** Business Process (OASIS) with V2.0.1 pre-Committee Draft review 17 July 2005

A List of Web Services 5

- **5) Security Frameworks and Core Specifications**
- **WS-Security 2004** Web Services Security: SOAP Message Security (OASIS) Standard March 2004.
- **WS-I Basic Security Profile** V1.0 Web Services Interoperability Organization Working Group Draft May 15 2005
- **WS-Security Username Token Profile** Web Services Security Username Token Profile V1.0 OASIS Standard, March 2004
- **WS-Security X.509 Certificate Token Profile** Web Services Security X.509 Certificate Token Profile OASIS Standard, March 2004
- **WS-Security REL Profile** Web Services Security Rights Expression Language (REL) Token Profile OASIS Standard: 19 December 2004
- **WS-I REL Token Profile** V1.0 Web Services Interoperability Organization Working Group Draft 13 May 2005
- **WS-Security Kerberos** Web Services Security Kerberos Binding (Microsoft) December 2003
- **Web-SSO** Web Single Sign-On Metadata Exchange Protocol (Microsoft, Sun) April 2005
- **Web-SSO-Mex** Web Single Sign-On Interoperability Profile (Microsoft, Sun) April 2005
- **WS-SecurityPolicy** Web Services Security Policy Language (IBM, Microsoft, RSA, Verisign) V1.1 July 2005

A List of Web Services 5 - Contd

- **5) Security Capabilities**
- **WS-Trust** Web Services Trust Language (BEA, IBM, Microsoft, RSA, Verisign …) February 2005
- **WS-SecureConversation** Web Services Secure Conversation Language (BEA, IBM, Microsoft, RSA, Verisign …) February 2005
- **WS-Federation** Web Services Federation Language (BEA, IBM, Microsoft, RSA, Verisign) July 2003
- **WS-Federation Active Requestor Profile** Web Services Federation Language Active Requestor Profile V 1.0 (BEA, IBM, Microsoft, RSA, Verisign) July 8, 2003
- **WS-Federation Passive Requestor Profile** Web Services Federation Language Passive Requestor Profile V 1.0 (BEA, IBM, Microsoft, RSA, Verisign) July 8, 2003
- **WS-Authorization** is being developed by IBM and Microsoft and will build on WS-Trust to describe how access to particular web services is specified and managed.
- **WS-Privacy** is being developed by IBM and Microsoft and will build on WS-Policy to describe the binding of privacy policies to Web services and their exchanged data.

A List of Web Services 5 - Contd

- **5) Security Languages**
- **SAML** Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0 OASIS Standard, 15 March 2005
- **WS-Security SAML Token Profile** Web Services Security SAML Token Profile OASIS Standard, 1 December 2004
- **WS-I SAML Token Profile** V1.0 Web Services Interoperability Organization Working Group Draft 13 May 2005
- **XACML** eXtensible Access Control Markup Language (OASIS) V2.0 1 February 2005

A List of Web Services 6

- **6) Service Discovery**
- **UDDI** (Broadly Supported OASIS Standard) V3 August 2003
- **WS-Discovery** Web services Dynamic Discovery (Microsoft, BEA, Intel …) February 2004
- **WS-IL** Web Services Inspection Language, (IBM, Microsoft) November 2001
- Note **WS-Context** as a metadata catalog and **WS-Management Catalog** are examples of related services
- There are many UDDI extensions

A List of Web Services 7

- **7) Metadata and State**
- **RDF** Resource Description Framework (W3C) Set of recommendations expanded from original February 1999 standard
- **DAML+OIL** combining DAML (Darpa Agent Markup Language) and OIL (Ontology Inference Layer) (W3C) Note December 2001
- **OWL** Web Ontology Language (W3C) Recommendation February 2004
- **WS-MetadataExchange** 1.1 Web Services Metadata Exchange (HP, IBM, Intel, Microsoft) March 2006
- **ASAP** Asynchronous Service Access Protocol (OASIS) with V1.0 working draft 2B December 11 2004
- **WS-GAF** Web Service Grid Application Framework (Arjuna, Newcastle University) August 2003
- **WBEM** Web-Based Enterprise Management including CIM (Common Information Model) from DMTF (Distributed Management Task Force) 2004-2005

A List of Web Services 7

- **7) Metadata and State: Resource Framework**
- **WS-RF** Web Services Resource Framework (OASIS) including
- **WS-Resource Framework** Web Services Resource 1.2 (OASIS) Public Review Draft 01, 10 June 2005
- **WS-ResourceProperties** Web Services Resource Properties V1.2 Public Review Draft 01, 10 June 2005
- **WS-ResourceLifetime** Web Services Resource Lifetime V1.2 Public Review Draft 01, 13 June 2005
- **WS-ServiceGroup** Web Services Service Group V1.2 Public Review Draft 01, 10 June 2005
- **WS-BaseFaults** Web Services Base Faults V1.2 Public Review Draft 01, June 13, 2005

Metadata and Service Context

- Consider a collection of services working together
  - Workflow tells you how to specify service interaction but more basically there is shared information or context specifying/controlling collection
- WS-RF and WS-GAF have different approaches to contextualization – supplying a common “context” which at its simplest is a token to represent state
- More generally core shared information includes dynamic service metadata and the equivalent of configuration information.
- One can supports such a common context either as pool of messages or as message-based access to a “database” (Context Service)
- Two services linked by a stream are perhaps simplest example of a collection of services needing context
- Note that there is a tension between storing metadata in messages and services.
  - This is shared versus distributed memory debate in parallel computing

Stateful Interactions

- There are (at least) four approaches to specifying state
  - OGSI use factories to generate separate services for each session in standard distributed object fashion
  - Globus GT-4 and WSRF use metadata of a resource to identify state associated with particular session
  - WS-GAF uses WS-Context to provide abstract context defining state. Has strength and weakness that reveals less about nature of session
  - WS-I+ "Pure Web Service" leaves state specification the application – e.g. put a context in the SOAP body
- I think we should smile and write a great metadata service hiding all these different models for state and metadata

A List of Web Services 8

- **8) Management – original OASIS**
- **WS-DistributedManagement** Web Services Distributed Management Framework with MUWS and MOWS below (OASIS)
- **WSDM-MUWS** Web Services Distributed Management: Management Using Web Services (OASIS) OASIS Standard March 9 2005
- **WSDM-MOWS** Web Services Distributed Management: Management of Web Services (OASIS) OASIS Standard March 9 2005

A List of Web Services 8- Contd

- **8) Management: Microsoft Converged Stack**
- **WS-Management** Web Services for Management (Microsoft, Intel, Sun …) August 2005
- **WS-Management Catalog** The WS-Management Catalog (Microsoft, Intel, Sun …) August 2005
- **WS-ResourceTransfer** Web Service Resource Transfer (HP, IBM, Intel, Microsoft) March 2006
- **WS-Transfer** Web Service Transfer (Microsoft, BEA, Sonic Software etc.) September 2004
- **WS-TransferAddendum** Extensions to Web Service Transfer (HP, IBM, Intel, Microsoft) March 2006
- **WS-Enumeration** Web Service Enumeration (Microsoft, BEA, Sonic Software etc.) September 2004

A List of Web Services 9

- **9) General Service Characteristics**
- **WS-PolicyFramework** Web Services Policy Framework (BEA, IBM, Microsoft, SAP …) September 2004
- **WS-PolicyAttachment** Web Services Policy Attachment (BEA, IBM, Microsoft, SAP …) September 2004
- **WS-PolicyAssertions** Web Services Policy Assertions Language (BEA, IBM, Microsoft, SAP) 18 December 2002 (Superseded by WS-PolicyFramework)
- **WS-Agreement** Web Services Agreement Specification (GGF under development) 9 August 2004

A List of Web Services 10

- **10) User Interfaces**
- **WSRP** Web Services for Remote Portlets (OASIS) OASIS Standard August 2003
- **JSR168:** JSR-000168 Portlet Specification for Java binding (Java Community Process) October 2003
- WSRP specifies the client-service protocol while JSR168 specifies how portlets are implemented for each supported service user-facing Web service ports inside aggregating portalslike JetSpeed, GridSphere or uPortal