Introduction to Grid Computing

Anca Jurcut
E-mail: anca.jurcut@ucd.ie
UCD School of Computer Science and Informatics
University College Dublin, Ireland

What is the "Grid"?

- Middleware designed to enable the sharing of resources on a very large scale: files, computers, software, data and sensors.
- Typically shared by groups of users in different organisations who are collaborating to solve a problem:
  - via sharing of data
  - via sharing of computing power
- Distributed system that is loosely coupled, heterogeneous, and geographically dispersed.

What is the “Grid”?

- Based in the idea of computing resources being consumed like electricity is from the power grid.
- Resources supported by heterogeneous computer hardware, operating systems, programming languages and applications.
- Management required to co-ordinate the use of resources.
  - Are clients getting what they need?
  - Can services afford to allow access?
- Security also an important consideration.

Example: World Wide Telescope

- The World Wide Telescope is a project that demonstrates the type of data-intensive application that the Grid is most useful for.
- Project owned by Microsoft Research to deploy data resources that are shared by the astronomy community.
  - Archives of observations covering a particular time period, part of the electromagnetic spectrum and part of the sky.
  - Made by different instruments distributed worldwide.

Example: World Wide Telescope

- Each team's data is gathered locally (order of terabytes and growing exponentially).
- As data is gathered, it is analysed and stored as derived data for use by others: common data labelling/format required.
  - Metadata to describe when it was collected, where it was collected and the instrument used.
  - Derived data specifies parameters used when filtering from the raw data.

Example: World Wide Telescope

- Calculation of derived data requires heavy computational support.
- Frequently has to be recalculated as techniques improve.
- Considerable expense to the data owner.
- Aim of the project is to unify the world's astronomy archives into a giant database containing astronomy literature, images, raw data, derived datasets and simulation data.

Requirements of a Grid

1) **Remote Access to Resources** (information required from the archives).
2) **Processing where it is stored and managed**: Typical query involves a small quantity of data from one or more massive archives.
3) **Dynamic Service Creation:** Resource manager of an archive should create service instances dynamically to deal with particular sections of data that are required.

Requirements of a Grid

4) **Metadata**: to describe

- characteristics of the data in the archive (e.g. area of the sky, date, time, instrument used).
- characteristics of the service (e.g. cost to use, location, publisher, load, space available).

5) **Directory Services**: to find services based on the above metadata.

6) **Management Software**: to manage queries, data transfers, reservation of resources. Should take into account that resources may need to be rationed.

Requirements of a Grid

- Web Services can deal with the first two requirements (remote access/local processing).
- Requires that each application provides a service description that includes a set of methods for accessing the data.
- Grid middleware deals with the rest of the requirements (including security, if relevant).
- **Note:** World Wide Telescope is *data-intensive*. Grids also used for *computationally-intensive* tasks also (e.g. image analysis)

Open Grid Services Architecture

- OGSA is a standard for grid-based applications.
- Framework within which the requirements above may be met.
- Based primarily on *web services.*
- Resources managed by *application-specific grid services*.

Interlude: Web Services

- Web services standardise the messages that entities in a distributed system must exchange in order to perform various operations.

- At the lowest level, this standardization concerns the protocol used to transport messages (typically HTTP), message encoding (Simple Object Access Protocol: SOAP), and interface description (Web Service Definition Language: WSDL).
  - A client interacts with a Web service by sending it a SOAP message; the client may subsequently receive response message(s) in reply…

- At higher levels, other specifications define conventions for:
  - securing message exchanges, management, discovery and choreography.

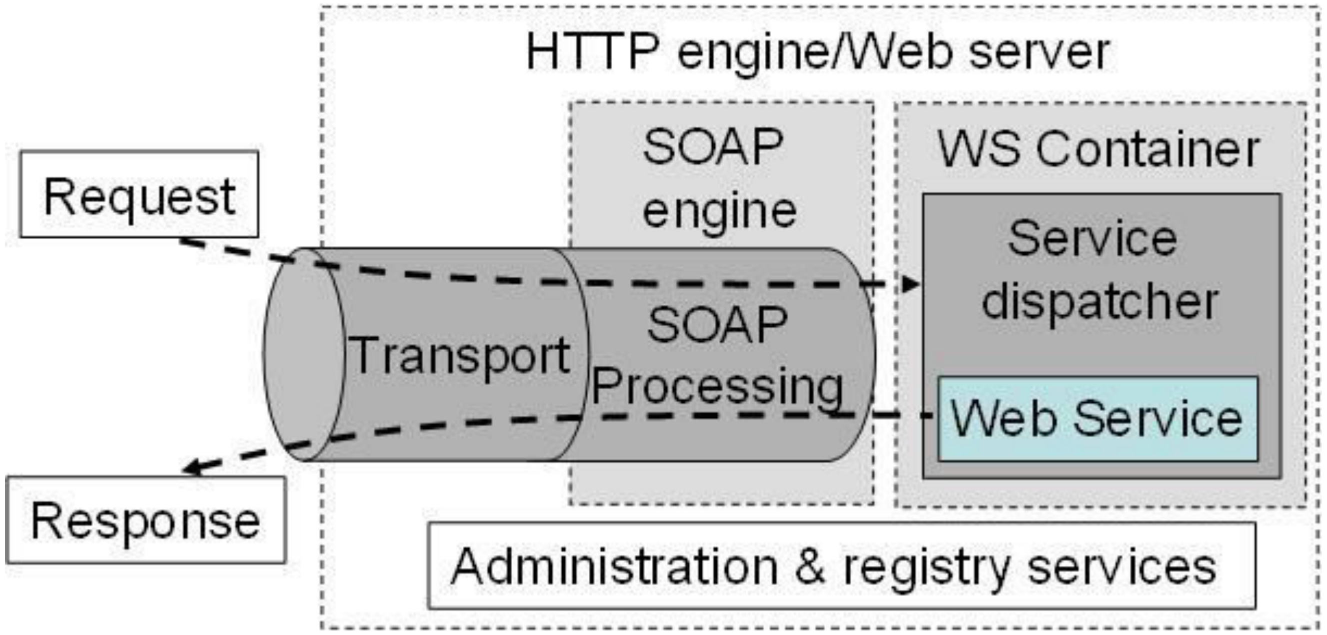
Implementing Web Services

- From the client perspective, a Web service is simply a network-accessible entity that processes SOAP messages.
  - However things are somewhat more complex under the covers.
- To simplify service implementation, it is common for a Web services implementation to distinguish between:
- The Web Services framework:
  - the hosting environment (or container)
  - the (domain-independent) logic used to receive a SOAP message and identify and invoke the appropriate code to handle the message, and potentially also to provide related administration functions.
- The Web Service Implementation:
  - the (domain-specific) code that handles the message.

Implementing Web Services

- This separation of concerns means that the developer need only provide the domain-specific message handling code to implement a new service.

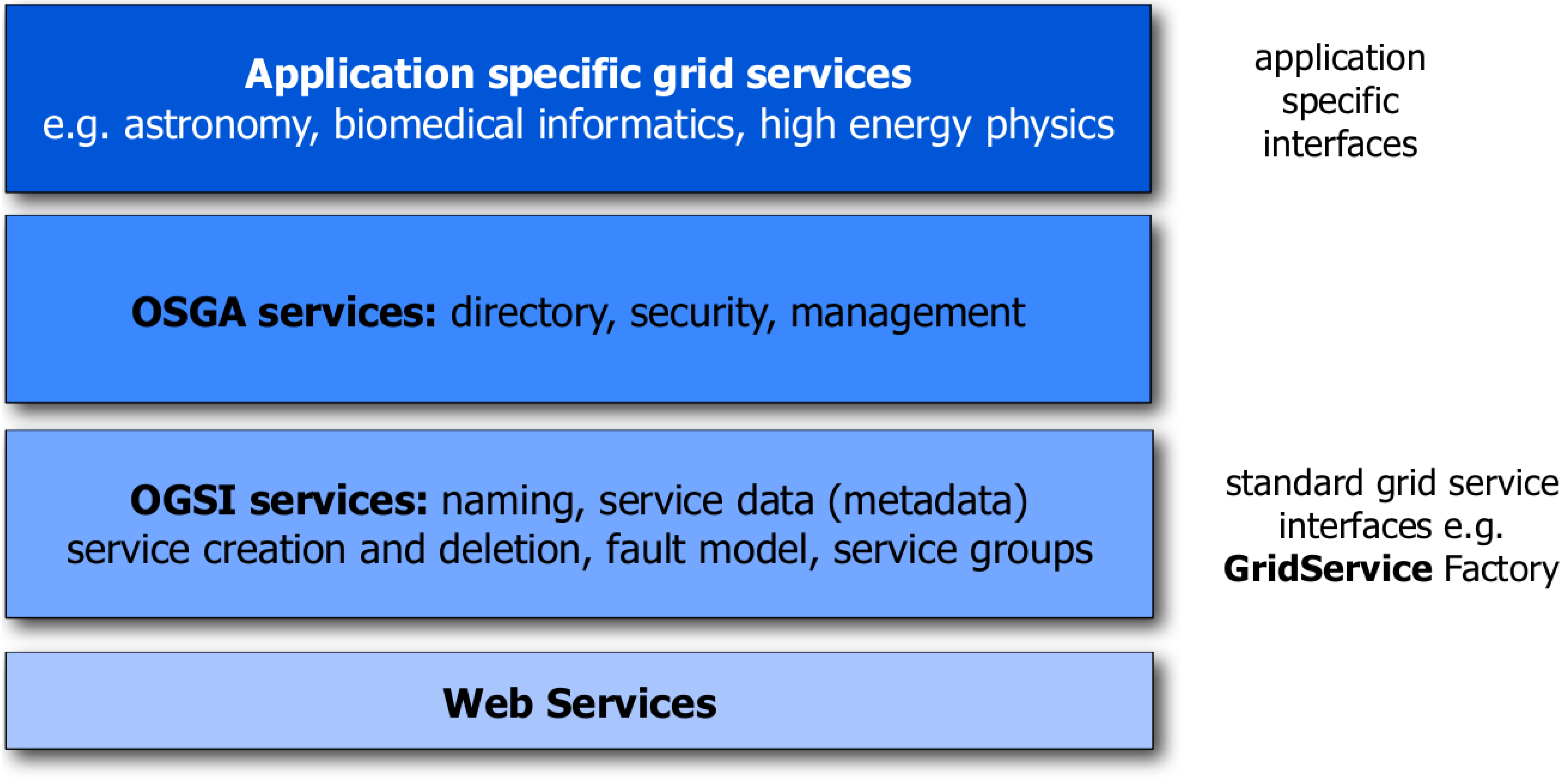
- It is also common to further partition the hosting environment logic into:
  - that concerned with transporting the SOAP message, and
  - that concerned with processing SOAP messages.



WS Specifications: Architecture

- XML is used extensively within Web services as a standard, flexible, and extensible data format.
- SOAP 1.2 provides a standard, extensible, composable framework for packaging and exchanging XML messages between a service provider and a service requester.
  - SOAP is independent of the underlying transport protocol, but is most commonly carried on HTTP
- WSDL 1.1 is an XML document for describing Web services.
  - Standardized binding conventions define how to use WSDL in conjunction with SOAP and other messaging protocols.
  - WSDL interfaces can be compiled to generate proxy code that constructs messages and manages communications on behalf of the client application.
  - The proxy automatically maps the XML message structures into native language objects that can be directly manipulated by the application.
  - The proxy frees the developer from having to understand and manipulate XML.

Open Grid Services Architecture


Application specific grid services
e.g. astronomy, biomedical informatics, high energy physics
OSGA services: directory, security, management
OGSI services: naming, service data (metadata)
service creation and deletion, fault model, service groups
Web Services
application specific interfaces
standard grid service interfaces e.g. **GridService** Factory

Application level grid services

1. Web services that implement standard grid-service interfaces in addition to their application-specific interfaces.
   a) interface to a set of data (*service data*) that contains metadata about the service (e.g. recent results, average values, characteristics of the data or other service characteristics).
   b) the context in which a service runs must provide a *factory* with the ability to create, start and stop service instances. Relies on the naming facilities of the OGSI layer

Application-level grid services

2. Make use of standard grid services: provided as two layers (OGSI and OGSA).

   a) OGSA Layer includes:

      - Directory service: allows clients to select suitable services based on metadata collected from services instances that have registered with the directory service.
      - Management service: monitors services, deal with failures and control service instance lifetimes.
      - Security services: provide single logins, delegation as well as authentication and encryption.

Application-level grid services

b) Open Grid Services Infrastructure (OGSI) layer includes:

- Implementation of naming scheme for service instances.
- Definition of standard *service data* elements that must be implemented by every application-level grid service instance, along with operations to get and set their values.
- Definitions of the interface to the factory to create new instances.
- Fault model for use by all grid services
- Notification services: enable services to publish information about service data and others to subscribe
- Service groups so services can cooperate

OGSI Layer

- Service instances are created dynamically.
- Two-level naming scheme:
  - Long-lived globally unique identifier Grid Service Handle (GSH). Starting a new instance in the same state later can re-use the GSH.
  - Grid Service Reference (GSR): refers to the location of the service (how and where it can be contacted). Often a WSDL document.

OGSI Layer

- Service Data
  - Clients can request a service instance to return information about its current state (e.g. capacity, free space, errors, recent results, load)
  - Requires that each service instance must store this data and offer operations to access it.
  - Standard set of operations provided in the *GridService* interface: includes operations to access the names of other interfaces supported, names of service data elements, identity of the factory that created it, GSH/GSR, termination time.
  - Every grid service must implement *GridService*

OGSI Layer

- Service Creation and Deletion:
  - Standard *Factory* interface specifies operations for creating transient service instances on demand
  - Application requests a new instance: specifies the description of the service required and the end of its lifetime (can be extended on request).
  - Factory creates new instance, registers it with a *handle resolution service* (resolves GSH to GSR)
  - Instance ends when its task is finished, or on request of its creator.
  - Service instances can be shared by clients.

OGSI Layer

- Fault Model
  - Common approach for reporting of faults.
  - Used by all OGSI-level services.
  - Recommended for application-level grid services.
  - XML scheme that requires at least a timestamp and the originating service to be specified.
  - Optional extras: fault description, fault, error code.

OGSI Layer

- WSDL interface extensions
  - Standard WSDL does not include mechanisms to define service data.
  - OGSI extends it to define the names of types of service data elements.

OGSA Services

- Higher level services built on OGSI services.
- Variety of OGSA services will meet specific requirements of application-level services.
- Some are so widely-applicable that they are relevant to any system.
    - e.g. directory services, management, monitoring, security.
    - Common part of grid toolkits

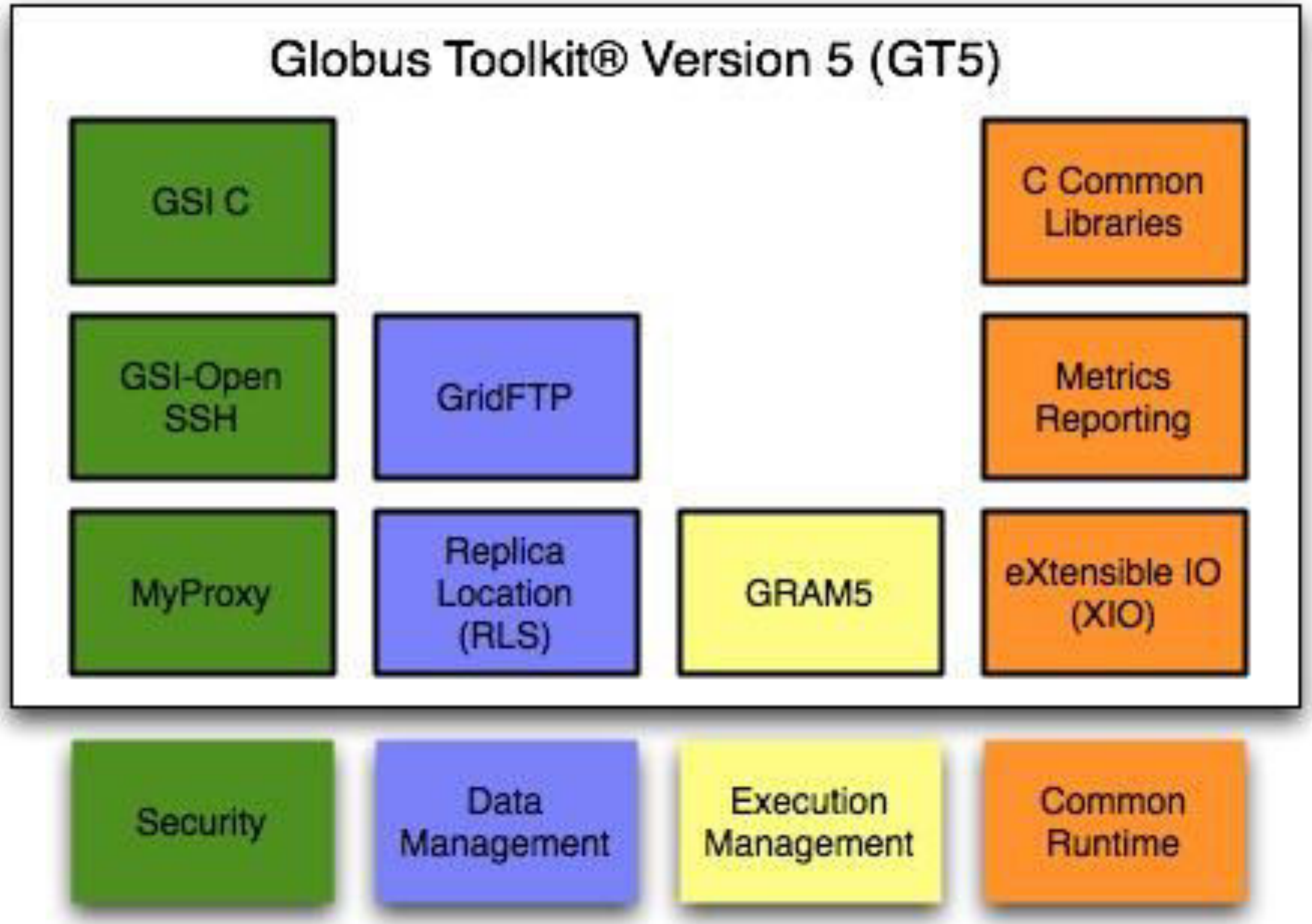
OGSA Services

- Management Services
  - Management of any resource that can be shared or exploited.
  - Arranging for services to be used and monitoring their state.
  - Issues include task submission, Quality of Service agreement, advance reservation.
    - May involve *Service Level Agreements* (SLAs), which make guarantees to clients as to the type of service being provided and allow the resource owner to maintain control over how it's used and how much information is exposed to the client.

Globus Toolkit

- Globus Project began in 1994 to provide software that integrates and standardises the functions required by a family of scientific applications (including directory services, security and resource management).
- First version of the toolkit appeared in 1997.
- OGSA came from second version.
- Major version is version 5 (2010)
- Latest release October 2014 (version 6.0)
- Maintained by Globus Alliance as an open source project (www.globus.org)

Globus Toolkit v5 (GT5)



GT5: Security

- Concerned with authentication, message protection, authorisation and secure logs for policy enforcement.
- Authentication and authorisation capabilities both with and without web services.
- Both built on standard X.509 certificates, used to identify entities like users and servers and support delegation of privileges to other entities on a temporary basis.
- Good discussion of DS security: http://www.globus.org/toolkit/docs/5.0/5.0.0/security/key

GT5: Execution Management

- Concerned with the initiation, monitoring, management, scheduling and/or coordination of remote computations.
- **Grid Resource Allocation and Management (GRAM):** used to locate, submit, monitor and cancel jobs (computing tasks).
  - Not a job scheduler, but rather a set of services and clients for communicating with a large range of different job schedulers using a common protocol.
  - Reliable operation, stateful monitoring, credential management and file staging are important.

GT5: Data Management

- Concerned with location, transfer and management of distributed data.
- **GridFTP:** High-performance, secure, reliable data transfer protocol designed for high-bandwidth wide-area networks.
- **Replica Location Service (RLS):** provides for the registration and lookup of replica information.
  - **Catalog service:** records mappings from logical names to target names, which may be physical locations or another level of logical naming.
  - **Index service:** index of logical names indicating which catalogs have mappings for them.

GT5: Common Runtime

- Base of the system, to provide a standard method of communication.
- **XIO:** Extended I/O library providing a single API (open/close/read/write) supoprting TCP, UCP, file, HTTP, GSI, GSSAPI_FTP, TELNET and queuing. New drivers can be added into the framework by developers.
- **C Common Libraries:** Abstraction layer for data types, libc system calls and data structures used throughout the toolkit that may be useful for application developers.

Globus Toolkit v6.0 (GT6)

- This version retains binary and protocol compatibility with the previous major versions (GT 5.0 and GT 5.2) of the Globus Toolkit.
- The main thrust of development for GT6 was to reduce the complexity of building, testing, and distributing the Toolkit and to merge the various operating system ports into one source distribution.
- This release contains native RPM and Deb packages for Linux and a Mac OS X Installer.app package, as well as binary tarball/zipfile distributions for Linux, Mac OS X, and Windows.

Globus Toolkit v6.0 (GT6)

- The main highlights of this release are:
  - Retain source (API), binary (ABI), and protocol compatibility with GT 5.0 and GT 5.2
  - Migrate from CVS to Git for version control https://github.com/globus/globus-toolkit
  - Simplified build by eliminating GPT and library flavors
  - Integrate testing into the native package build process
  - Add support for el7 (CentOS 7, RHEL 7)
  - Add binary distribution for windows -- mingw (client only) and cygwin
  - Add binary package and tarball for Mac OS X
  - Make binaries and scripts more easily relocatable (don't need to set GLOBUS_LOCATION for most cases)
  - Globus Connect Server compatibility with SUSE Linux Enterprise Server 11 GridFTP
  - Wider support for UDT MyProxy - Updated to MyProxy v6.0rc3. GSI-Enabled OpenSSH
  - Updates for building on windows Supported with native RPM or Debian Packages: CentOS 5, 6, 7; Fedora 19; 20 - Red Hat Enterprise Linux 5, 6, 7; Scientific Linux 5, 6; Debian 6, 7 - Ubuntu 10.04, 12.04, 14.04 - SuSE Linux Enterprise Server 11

Summary

- The Globus Toolkit is the defacto standard for grid computing.
- It is founded on the Web Services Core.
- Four key services are built on top of this:
  - Execution Management,
  - Data Management
  - Security.
  - Common Runtime.
- Most current work on grid computing assumes the use of the Globus Toolkit.