NeuroLOG Achievements and perspectives

All NeuroLOG partners

Paris, September 7, 2009

Financé par ANR

Main project achievements

- **Software integration**
  - Software technologies
  - Deployment and prototype platform
- **Security policies**
  - Distributed access control
  - Data and processing tools resources
- **Distributed data and metadata**
  - Metadata federation
  - Data sets catalog and file streaming
- **Processing tools**
  - Services repository
- **Knowledge management**
  - NeurOntology
- **Applications**

Software integration

Software technologies

- **Software modules**
  - Tomcat server container
  - Java + Mono client
  - WS application code
- **Deployment**
  - Java6 Runtime Environment
    - Hibernate persistence library
  - MySQL (Registry, Site Server)
    - v5.0
  - Certificates authorities
    - Openssl CA
    - Java keytool
  - Mono
  - Data Federator
  - CORESE
  - Web Services (Tomcat + metro)

Packaged with software
External

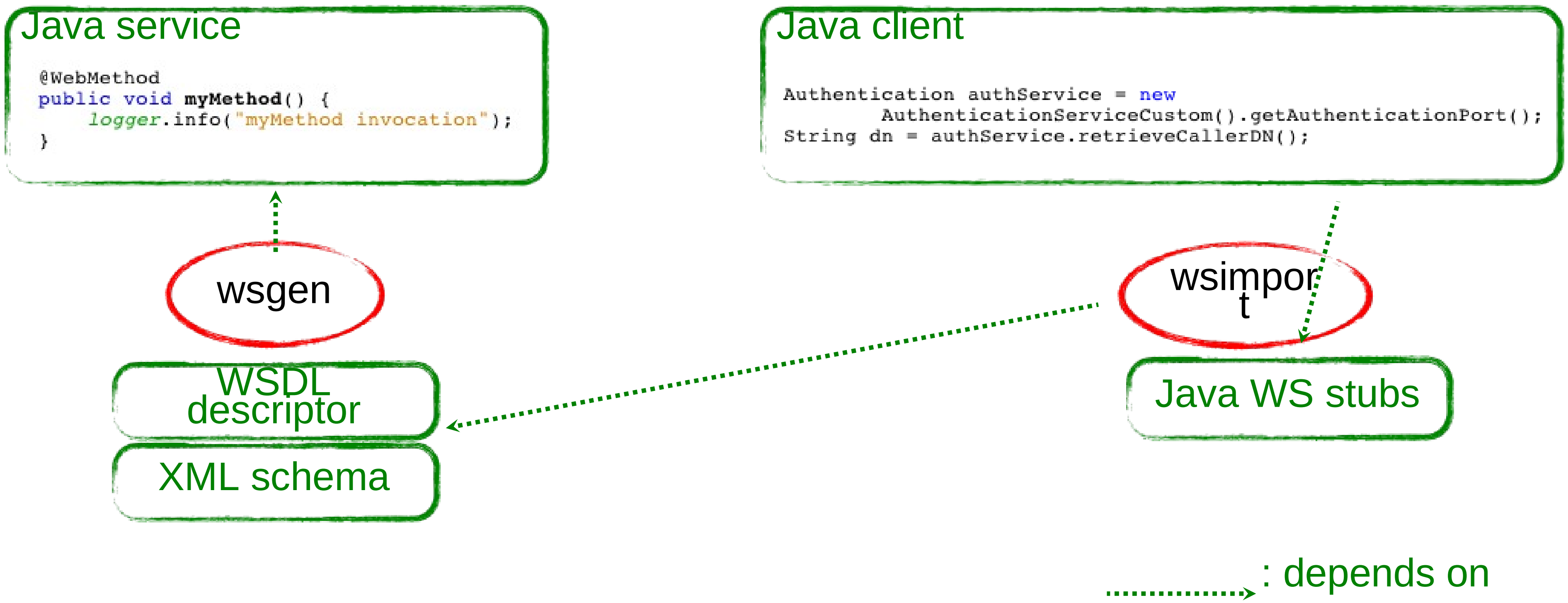
Java RMIs // Web Services 1/2

- **Two framework used in Service Oriented Architecture**
- **Interoperability**
  - RMIs are full «java», it exists some bridges but they have not been designed for interoperability. They exchange java object on a particular port with a particular protocol JRMP.
  - Web Services (WS) exchange standard xml messages and have been designed for interoperability. The protocol used is standard HTTP.
- **Developing with it**
  - RMIs: no need to deal with object serialization
  - WS:
    - take care with translation between objects and xml messages ;
    - automation through code generation steps, but add complexity ;
    - different behaviors between Axis and Sun Metro implementations of WS standard

Java RMIs // Web Services 2/2

- **Deploying services**
  - RMIs are ‘self-contained’, i.e. any java object can act as an RMI server.
  - A third party application container need to be configured to host web services. Huge variability in features provided by containers. Non functional features such as security (authentication, access control) can be provided.

Full JAX-WS development cycle (Metro implementation)



```
@WebMethod
public void myMethod() {
     logger.info("myMethod invocation");
}
```

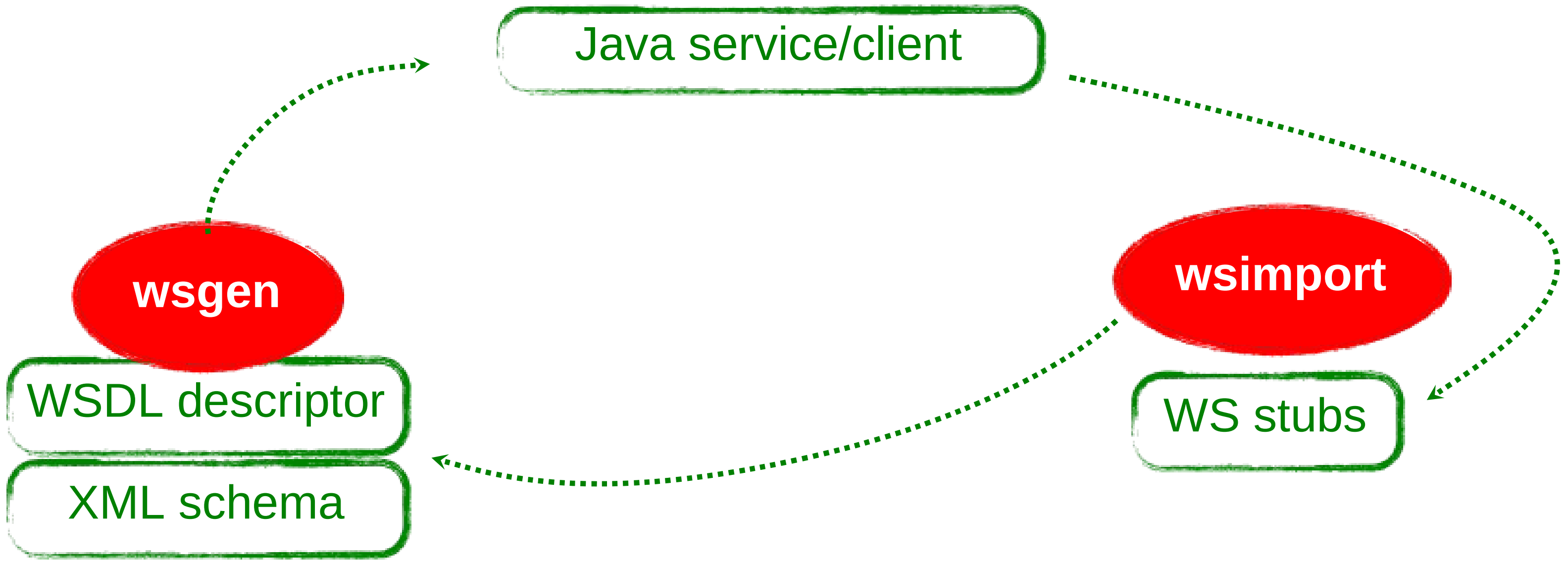
```
Authentication authService = new
        AuthenticationServiceCustom().getAuthenticationPort();
String dn = authService.retrieveCallerDN();
```

- **But what happens if a WS is also client of itself ?**

JAX-WS Metro 2/4

Complex dependencies in a «P2P» design

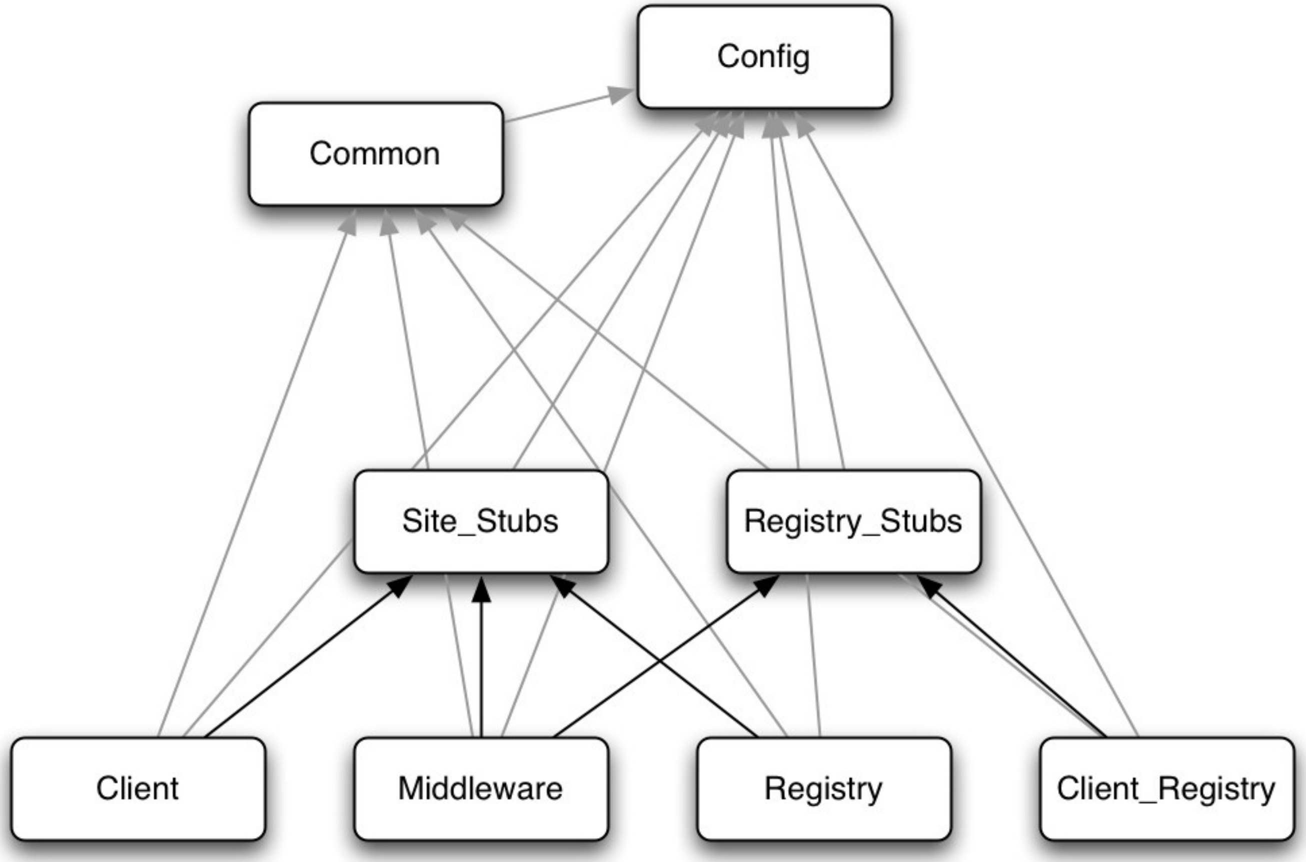
WS build process is not adapted



Final sub-project dependencies

wsgen isolated in the build of «stubs» projects.

wsimport isolated in the build of «clients» projects.

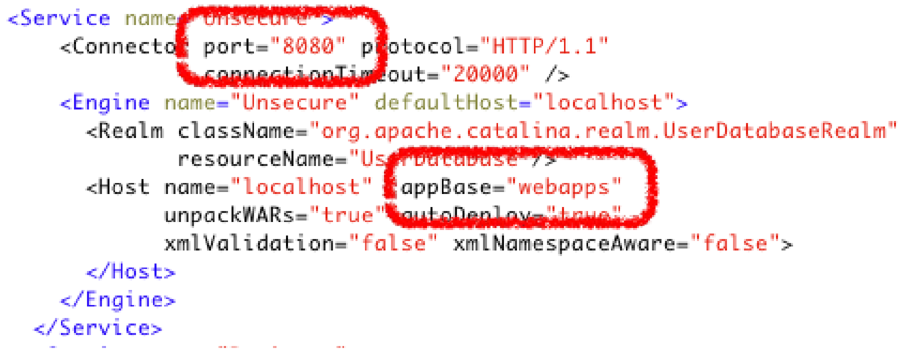


JAX-WS : container and security

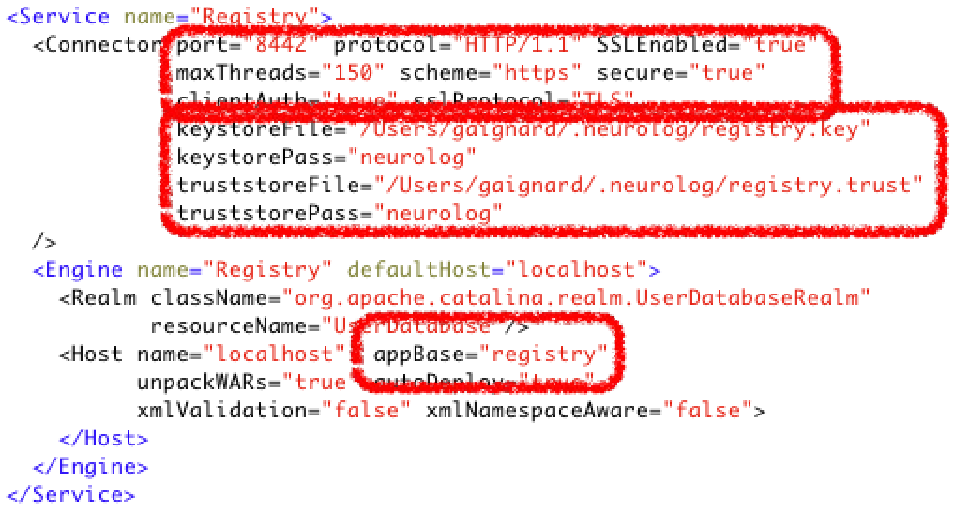
Tomcat choosed for, SSL handling, «lightweight», fame.

But need to be tuned

host at the same time secured and unsecured webservices :



```
<Service name="Unsecure">
    <Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000" />
    <Engine name="Unsecure" defaultHost="localhost">
      <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
             resourceName="UserDatabase" />
      <Host name="localhost" appBase="webapps"
            unpackWARs="true" autoDeploy="true"
            xmlValidation="false" xmlNamespaceAware="false">
      </Host>
    </Engine>
  </Service>
```



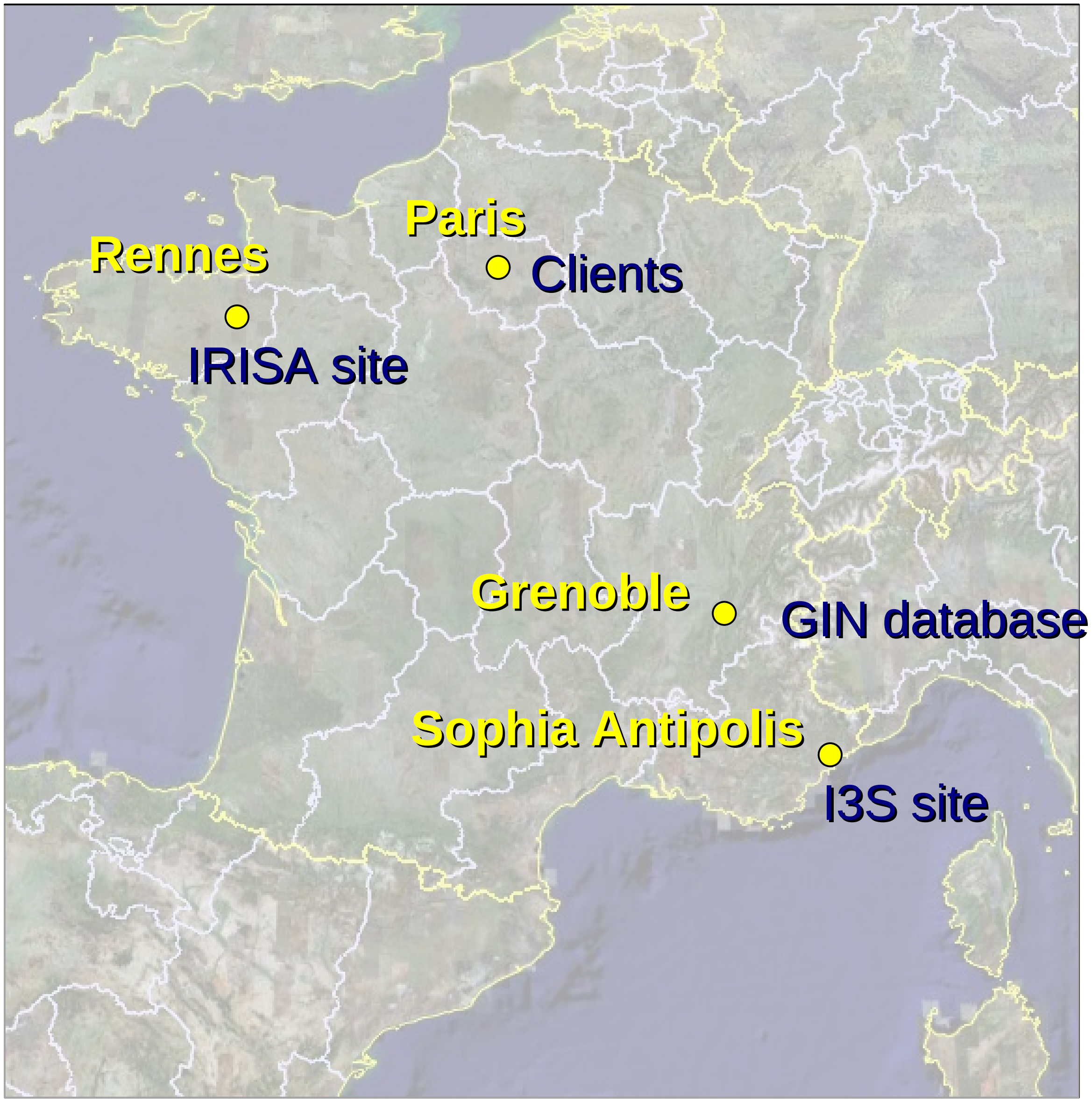
```
<Service name="Registry">
  <Connector port="8442" protocol="HTTP/1.1" SSLEnabled="true"
             maxThreads="150" scheme="https" secure="true"
             clientAuth="true" sslProtocol="TLS"
             keystoreFile="/Users/gaignard/.neurolog/registry.key"
             keystorePass="neurolog"
             truststoreFile="/Users/gaignard/.neurolog/registry.trust"
             truststorePass="neurolog"
  />
  <Engine name="Registry" defaultHost="localhost">
    <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
           resourceName="UserDatabase" />
    <Host name="localhost" appBase="registry"
          unpackWARs="true" autoDeploy="true"
          xmlValidation="false" xmlNamespaceAware="false">
    </Host>
  </Engine>
</Service>
```

A counterpart: clear password *keystorePass* and *truststorePass*

NeuroLOG

Prototype infrastructure

Software technologies for integration of process, data and knowledge in medical imaging

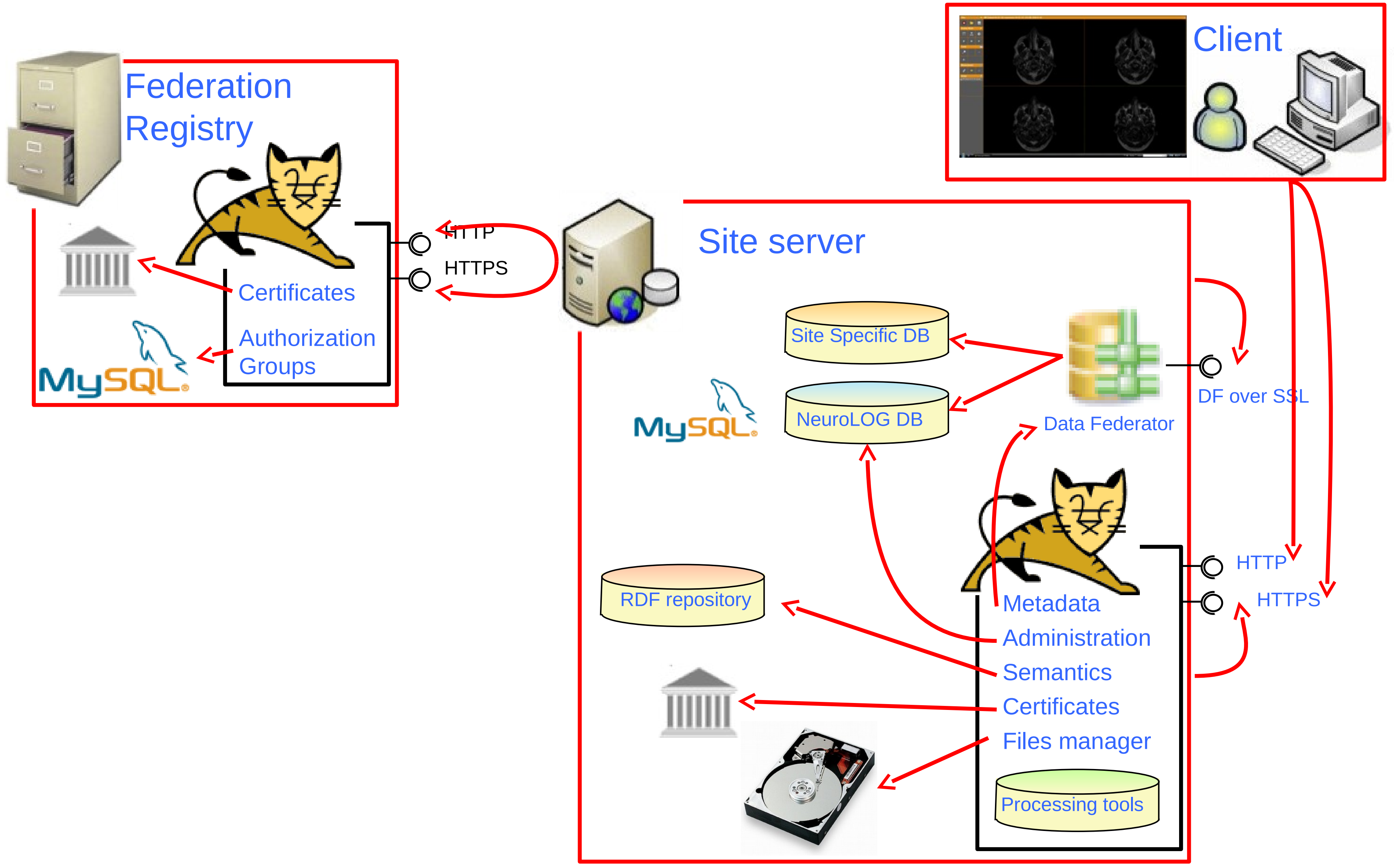


- **3 operational sites**
  - I3S, IRISA, GIN
- **Clients connect from anywhere**

NeuroLOG

Services deployed

Software technologies for integration of process, data and knowledge in medical imaging

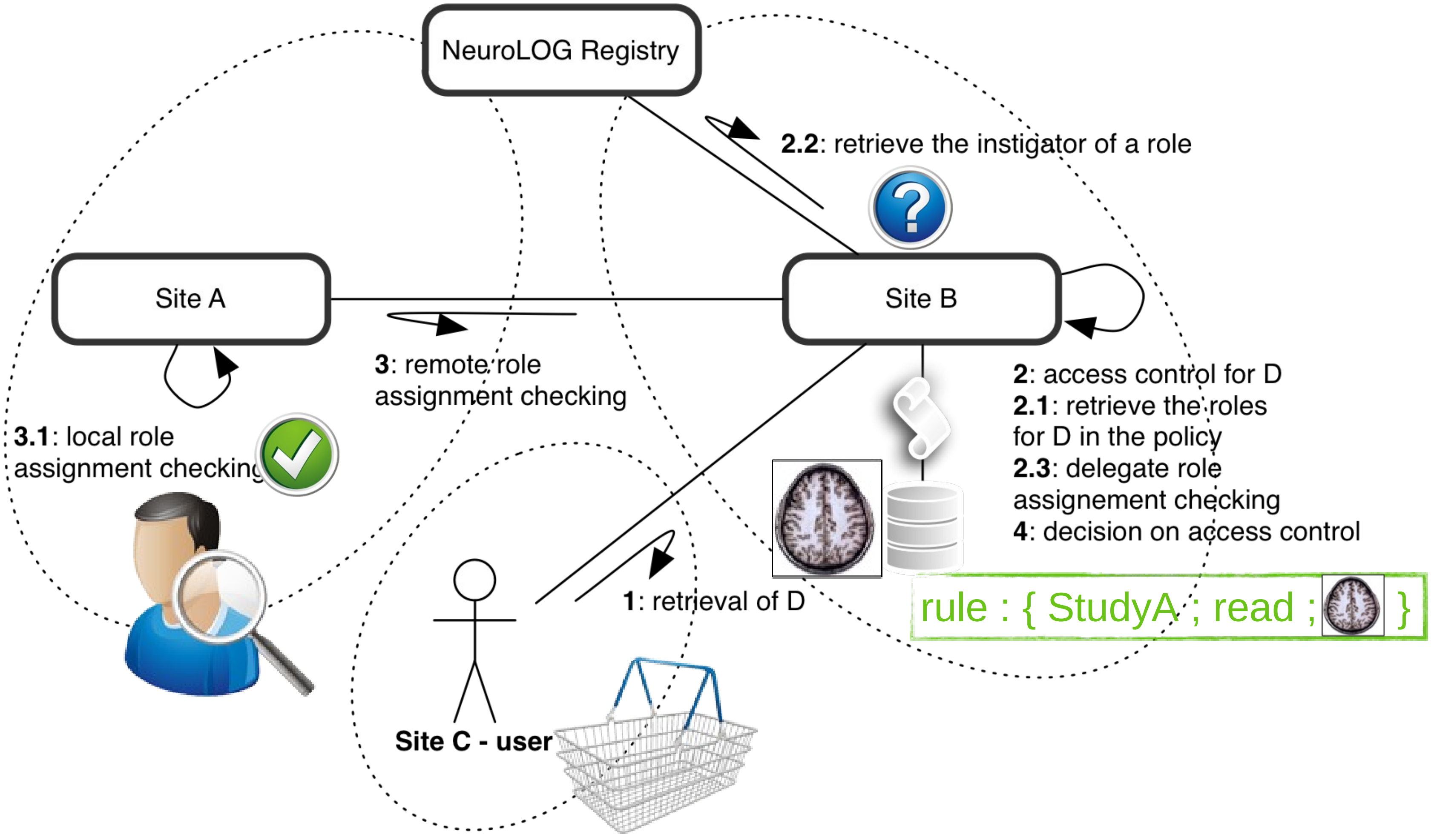


Security policies

NeuroLOG

Distributed access control

Software technologies for integration of process, data and knowledge in medical imaging


NeuroLOG Registry
2.2: retrieve the instigator of a role
Site A
Site B
3: remote role assignment checking
3.1: local role assignment checking
2: access control for D
2.1: retrieve the roles for D in the policy
2.3: delegate role assignement checking
4: decision on access control
rule : { StudyA ; read ; }
1: retrieval of D
Site C - user

Distributed access control

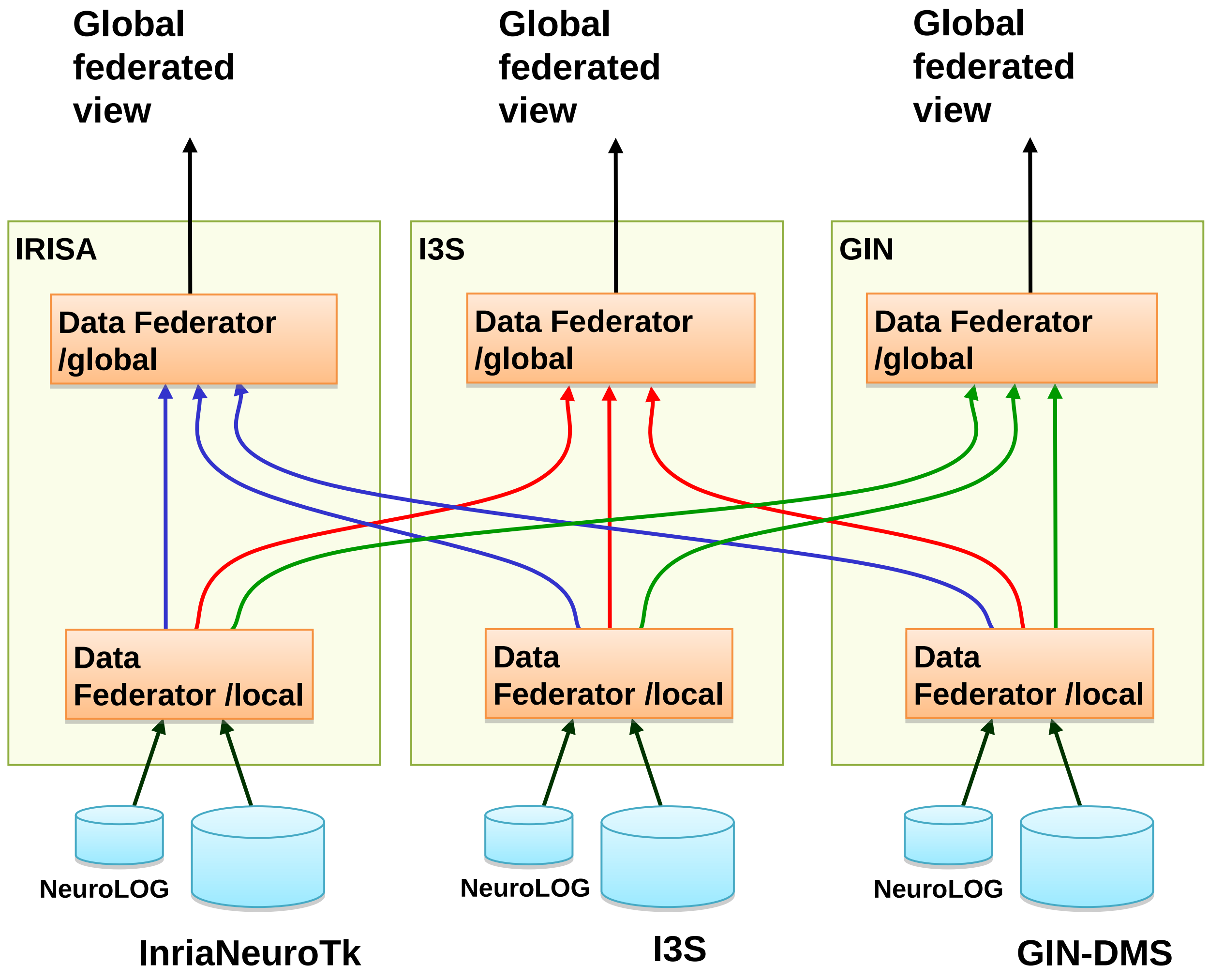
- **Files and processing tools are seen as different kinds of resources**
  - Access control to files is performed by the file manager
  - However, image processing tools are directly exposed through the services container
- **Files access control**
  - The data manager invokes the system resources access control prior to delivering any file
  - Returned files are streamed to the client
- **Processing tools access control**
  - Application code is wrapped with a WS interface (jGASW)
  - The wrapper is instrumented with a code performing a call to the system resources access control prior to each invocation

Security features to implement

- **Activity traces**
  - Non-repudiable traces (files access, etc).
  - Using cryptographically signed traces
  - Instrumenting services and processing tools
- **CPS (professional health card) identification**
  - Card reader and sample ID card (with X509 certificate) available
  - Nation-wide ID control server
  - Need to integrate and use the access control API on client side (attached to card reader)
  - Yet, access control should be performed on server side
- **Local data encryption**
  - Optional

Distributed data and metadata

Data Federator, a technical flavor



Metadata distribution and replication

- **Data Federator mappings**
  - Add prefixes (e.g. site-specific prefix)
  - Merge columns
  - Change column names
  - Change column values
- **Data Federator federation engine**
  - Merge several views from different mapped sources
  - Does not deal with replicated entries on several sources
- **Need for replicated entries**
  - Some entries are local (e.g. *Image*) while others may be found on several sites (e.g. *Patient*)
  - Metadata entries refer to each other through foreign keys
  - Need to achieve compromise between distributed consistency and sites autonomy

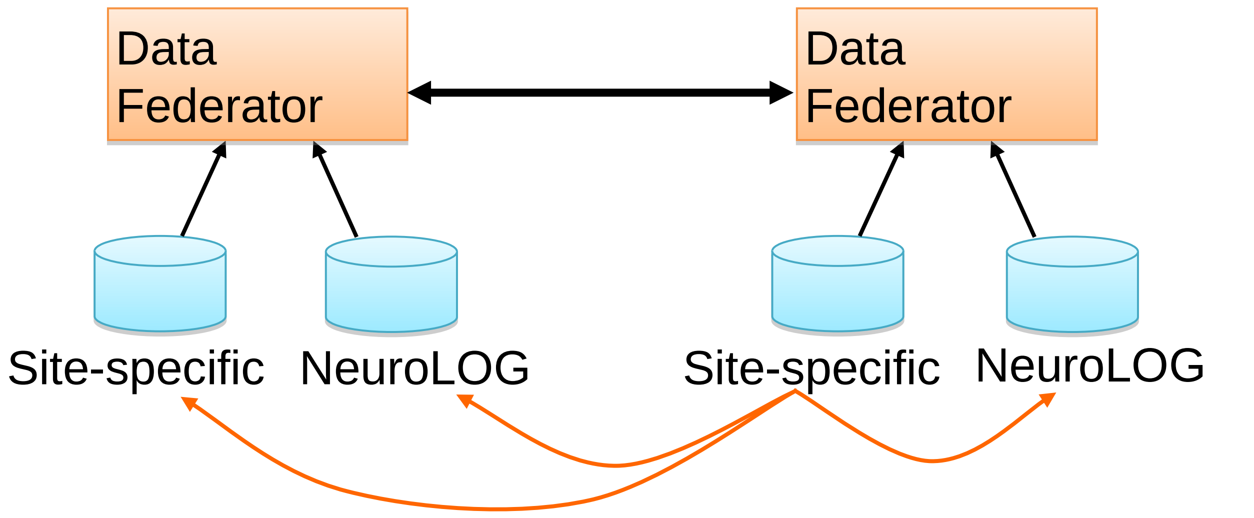
NeuroLOG Metadata Schema Distribution

- **Objectives**
  - to accommodate to site specific databases and data structure;
  - to provide a coherent view of metadata distributed over different sites in potentially heterogeneous databases;
  - to enable the federation of multi-site data sets for clinical studies;
  - to preserve databases coherency; and
  - to ensure sites autonomy through a weak coupling of the databases.
- **Cross-references between entities**
  - Each entity (table line) is indexed by a local identifier (DB key)
  - A global reference is the equivalent of a foreign key for an entity referenced in a different database. Format: <site>:<DB>:<localID>
  - Unlike foreign key, there is no guarantee of coherence between different DBs
  - Some references are critical (e.g. Subject – DataSet) others are not (e.g. data processing provenance information)

Across-site references

Deal with problems:

Manage cross-references



References to remotely defined entities

Consequence

Need for consistency-check and restore procedures

NeuroLOG Metadata Schema Distribution

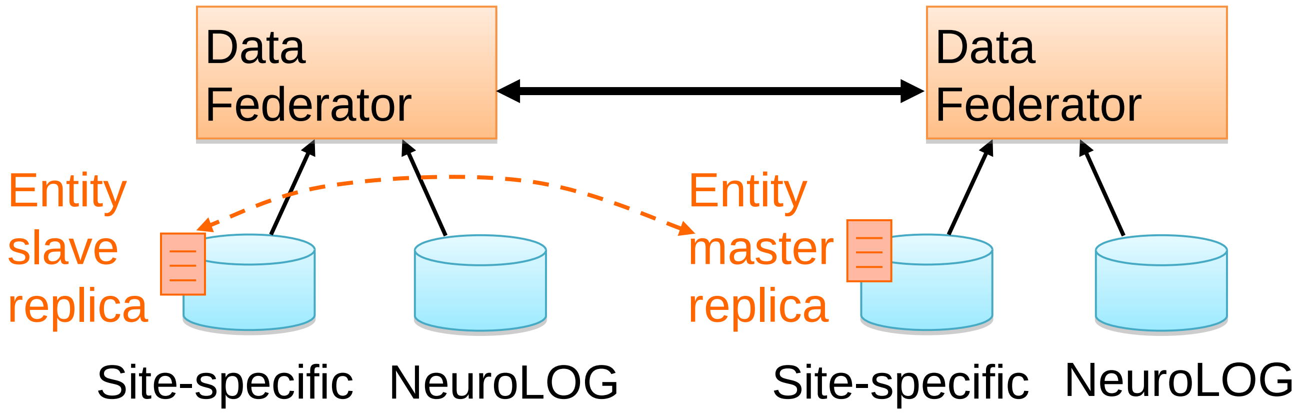
- **Multiple databases coherency may be challenged**
  - Internally to a site: the site-specific DB being managed independently from the middleware.
  - Externally: each site being autonomous.
- **Across-site references**
  - DF enables the mapping of local IDs into global ones
  - Non critical references may use across-site references at the risk of being lost in some cases (a periodic curation process is needed)
  - Critical references have to be controlled: this can only be ensured by locally caching cross-site references
  - Each entity replicated on several sites has a master value (the entity from the master site) and slave replicas (on other sites)
  - Replicated entities will be mapped to the same global ID by DF and only the master entry will appear in the federated tables

Master / slaves replicated entities

Deal with problems:

Several sites may contain the same entity

Need to identify replicas of a single entity



Impact on mappings:

- Filter out slave entities
- Replace slave entity identifier by master entity identifier (primary key and foreign keys)

NeuroLOG

NeuroLOG Metadata Schema Distribution

- **Current status**
  - DF mapping with replicated entities tested and validated
- **To be implemented**
  - Data curation procedure
  - Data migration procedure (to preserve coherency in case of scheduled deletions)

Distributed data and metadata

- **Done**
  - Metadata federation
    - DF configuration processes automation
    - WD development for metadata and data search
  - Site specific mappings for IRISA and GIN
  - Data sets catalog and file streaming
- **Todo**
  - IFR49 mapping under construction, to be finalized
  - IFR49 DF and site servers to be deployed
  - Integration with grid for file exchanges
  - Migration to NeuroLOG ontology v2 and corresponding new relational schema
  - Dynamic scheme / extensibility
  - Fault tolerance

Knowledge management

Work completed

Ontologies

- Specification of the ontology design methodology – L2.1 and of the architecture of the semantic query component – L2.2, Jul. 07
- OntoNeuroLOG v.1.0 (Federated relational schema + OntoSpec + OWL-Lite manifestations) – May 08
  - The ontologies are available both at the MIS and the NeuroLOG websites
- OntoNeuroLOG v.2.0 (Federated relational schema + OntoSpec) – Aug. 09

Semantic processing software

- First release of semantic query component – Aug. 09
  - Evaluation of SPARQL queries (based on CORESE)
  - Based on OntoNeuroLOG Version 1.0
  - integrated in the NeuroLOG client

Future work (end of 2009)

Ontologies

- OntoNeuroLOG v.2.0 (OWL-Lite manifestations) – Sept. 09
- OntoNeuroLOG v.2.1 (Federated relational schema + OntoSpec + OWL-Lite manifestations) – Dec. 09
    - MR protocols
    - Processing tools (input, output, pre- and post-conditions)
    - Essential clinical data
    - Brain functions and pathology (basic)

Semantic processing software

- Second release of the semantic query component
    - Based on OntoNeuroLOG Version 2.0
    - Deployed in a client-server architecture
- First release of metadata manager

Future work (2010)

Ontologies

- OntoNeuroLOG v.3.0 (OntoSpec manifestation + partial implementation) – Feb. 2010
  - Mathematical functions (associated to Datasets)
  - ROI and ROI annotations
  - Pipelines (composition of services)
  - Anatomy

Semantic processing software

- Second release of metadata manager – mid 2010
- Service composition manager – mid 2010

Publications

G. Kassel. Vers une ontologie formelle des artefacts. In Actes des 20èmes Journées Francophones d'Ingénierie des Connaissances (IC'2009), Hammamet (Tunisie), 25-29 mai 2009.

P. Lando, A. Lapujade, G. Kassel, F. Fürst. An Ontological Investigation in the Field of Computer Programs. In Communications in Computer and Information Science, Springer Berlin Heidelberg, vol. 22, 2008, p. 371-383.

L. Temal, M. Dojat, G. Kassel and B. Gibaud. Towards an Ontology for Sharing Medical Images and Regions of Interest in Neuroimaging, Journal of Biomedical Informatics, 41(5),766-778, 2008.

B. Gibaud B., G. Kassel. Intégration de ressources en neuroimagerie (images et outils de traitement) : le projet NeuroLOG. Journée STIC Santé Inserm « Ontologies médicales : utilité et place des ontologies fondationnelles ? », Paris, 18 Novembre 2008.

G. Kassel, P. Lando, A. Lapujade, F. Fürst. Des Artefacts aux Programmes. Proceedings of the *1ères Journées Francophones sur les Ontologies* (JFO 2007), 18-20 October 2007, Sousse (Tunisia), p. 281-300.

P. Lando, F. Fürst, G. Kassel, A. Lapujade. "Premiers pas vers une ontologie des programmes informatiques. Proceedings of the 18th *Journées Francophones d'Ingénierie des Connaissances* (IC 2007), Grenoble, France, july 2007.

L. Temal, P. Lando, M. Dojat, F. Fürst, B. Gibaud, G. Kassel, A. Lapujade. "OntoNeuroLOG : une ontologie modulaire et multi-niveaux pour gérer l'hétérogénéité sémantique des métadonnées. Actes de la journée thématique "*Ontologies et Gestion de l'Hétérogénéité Sémantique*" du GDR I3, Grenoble, 3 juillet.

P. Lando, A. Lapujade, G. Kassel, F. Fürst. Towards a general ontology of computer programs". Proceedings of the 2nd *International Conference on Software and Data Technologies* (ICSOFT 2007), area: Knowledge Engineering, 25-27 July 2007, Barcelona (Spain).

Processing tools

Done

The java General Application Service Wrapper (jGASW) features includes:

- Rich description of CLI applications and execution procedure using a personalized WSDL for each tool
- Tool's dependencies encapsulation (libraries and extra files needed to execute the application)
- Instrumentation of service execution according to the interface (local, grid) enabling mixed compositions in pipelines
- Strong mapping between types of services and arguments of applications
- GUI for creation and generic API for invocation of services
- Special arguments manipulation :
  - Filename concatenations
  - Result filenames as regular expressions
  - Implicit (hidden) arguments

Integration in the middleware

- **Client independence from NeuroLOG middleware in order to create a service**
- **Hot deployment on different servers using the same service package (no exstra configuration is needed)**
- **Invocation of tools**
  - Generic libraries to read service's descriptions and consume service's operations in two ways : workflow composition and independent tool execution
  - Restriction of service executions by roles
  - Control of service creation for privileged users
- **Simple service catalog (WS directory)**
- **Data transfers to the local server or registering new elements on the grid file catalog**

TODO

- Full integration of complex data flows
  - Nested arrays outputs
  - Data types outputs
- Manipulation of special arguments (Analyze, DICOM, directories)
- Data transfers using Grid interfaces
  - Using standard Grid user interfaces
  - Using external WS
- Management of grid configuration
  - General environment (middleware setup variables)
  - Specific application requirements (short time jobs)
  - Personal user configuration (proxy certificates, storage elements)
- Submission policies implementation for fault tolerant executions
- Dedicated mechanism for submitting and monitoring grid jobs
- Pipelining selected data sets into workflows

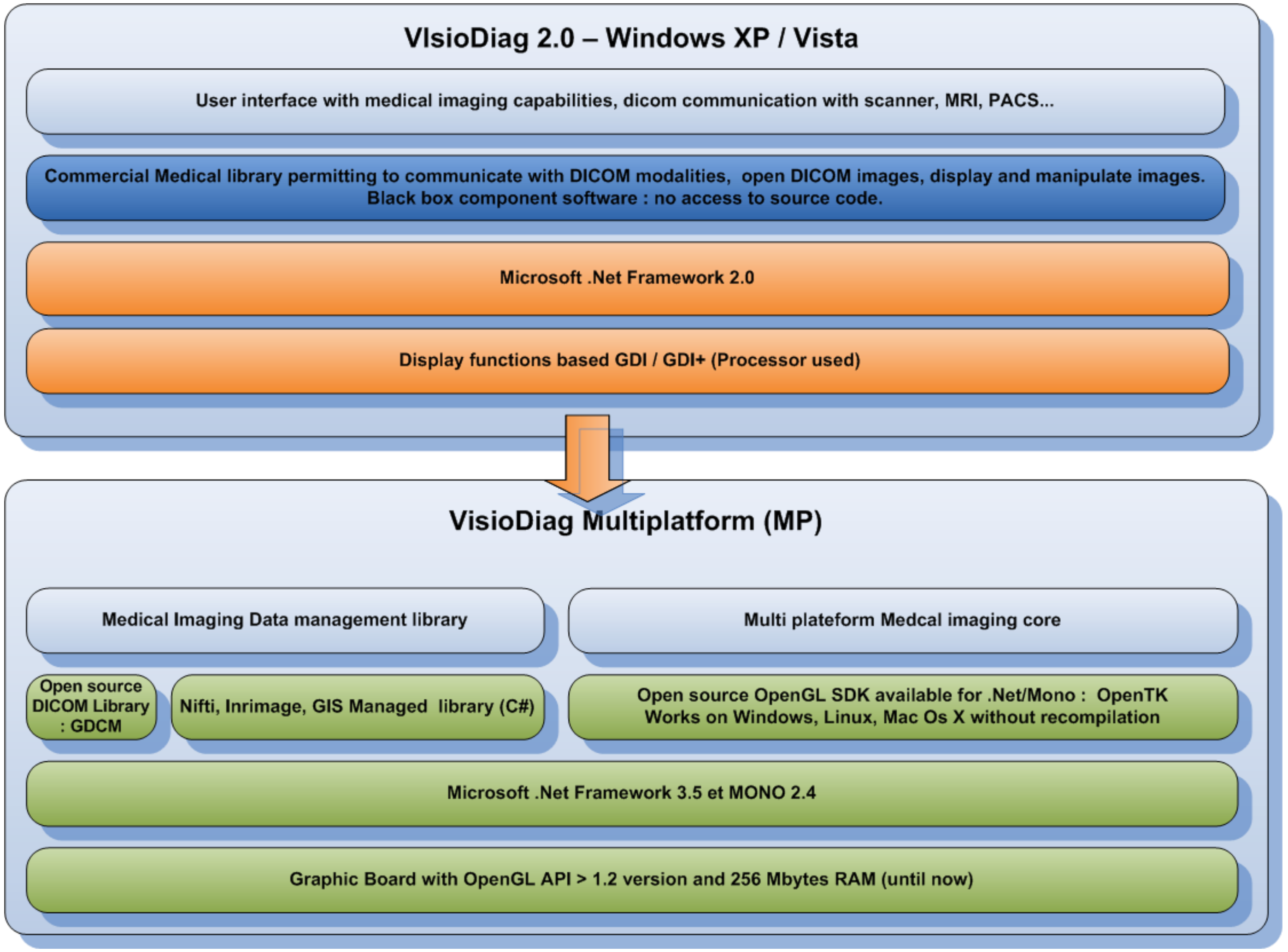
Processing tools

- **Done**
  - jGASW wrapper (Creation of services)
  - Integration on the NeuroLOG middleware (Deployment)
  - Generic API to consume WS (Invokation of tools)
  - Simple service catalog (WS directory)
- **Todo**
  - Full integration of complex data flows
  - Manipulation of special arguments like Analyze or DICOM files
  - Data transfers using Grid interfaces
  - Management of grid configuration (proxy certs, general environment variables + specific app requirements)
  - Pipelining selected data sets into workflows

Applications

NeuroLOG Viewer 1/4

Technical architecture


VIsioDiag 2.0 – Windows XP / Vista
User interface with medical imaging capabilities, dicom communication with scanner, MRI, PACS...
Commercial Medical library permitting to communicate with DICOM modalities, open DICOM images, display and manipulate images. Black box component software : no access to source code.
Microsoft .Net Framework 2.0
Display functions based GDI / GDI+ (Processor used)
VisioDiag Multiplatform (MP)
Medical Imaging Data management library
Multi plateform Medcal imaging core
Open source DICOM Library : GDCM
Nifti, Inrimage, GIS Managed library (C#)
Open source OpenGL SDK available for .Net/Mono : OpenTK Works on Windows, Linux, Mac Os X without recompilation
Microsoft .Net Framework 3.5 et MONO 2.4
Graphic Board with OpenGL API > 1.2 version and 256 Mbytes RAM (until now)

NeuroLOG Viewer 2/4

- **NeuroLOG Viewer development:**
  - Based on Visioscopie medical imaging application,
  - Technological changes to support multiplatform execution,
  - Rewriting or use of open source components to swap commercial libraries,
  - More effective graphic core because of OpenGL use.

- **Problems :**
  - Open source component not really finalized or stable (Mono 2.4.3.2, OpenTK 0.9.8.2), no forward compatibility, need constant code update,
  - Several Linux distributions and versions, Hardware requirements and drivers.

NeuroLOG

NeuroLOG Viewer 3/4

Software technologies for integration of process, data and knowledge in medical imaging

Software integration

- Viewer based on .Net / Mono framework, not Java,
- Exchange with NeuroLOG client using script file containing images series files list to open

```
LABEL=MR Dataset IRISA-SS-355, expression IRISA-SS-401, Nifti-single-file, 2009-04-17
TYPE=Nifti-single-file
file:///C:/tmp/neurolog/client/downloaded/nl_27342.data
ENDSERIE
LABEL=MR Dataset IRISA-SS-356, expression IRISA-SS-402, Nifti-single-file, 2009-04-18
TYPE=Nifti-single-file
file:///C:/tmp/neurolog/client/downloaded/nl_27343.data
ENDSERIE
```

Deployment

- System requirements :
  - Mono Framework 2.4.3.2 with Mono-Winform package,
  - OpenGL graphic board supporting OpenGL >= 1.2 and OpenGL Drivers
- Copy / Paste deployment, will be included in NeuroLOG Client deployment

NeuroLOG Viewer 4/4

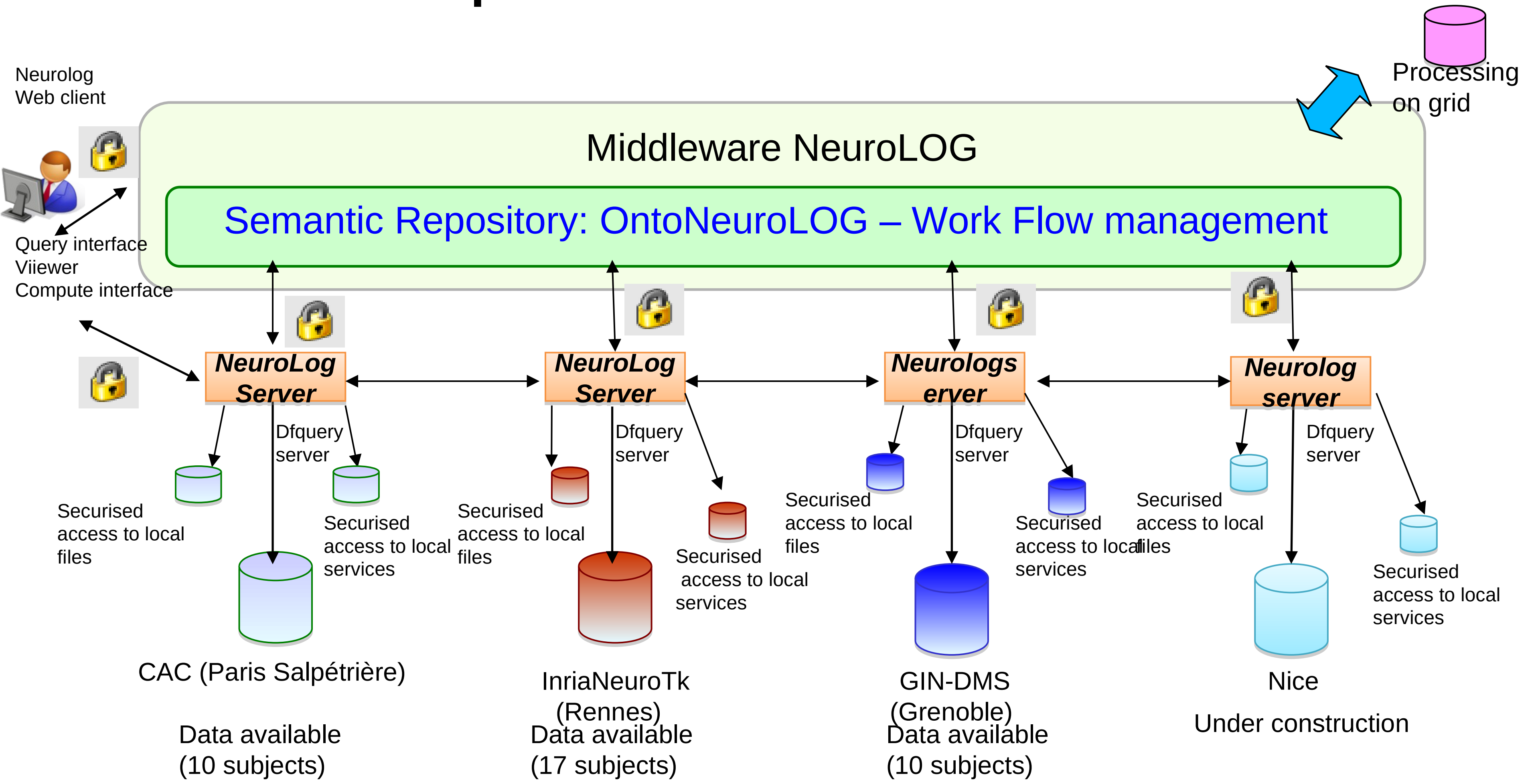
- **Done**
  - Different format integration :Dicom, Nifti, Inrimage, Jpeg, TIF,
  - 8, 10, 12, 16 bits imaging core based on OpenGL,
  - Display functions (Window Level, pan,…).

- **Todo**
  - Analyze, GIS format validation,
  - Stabilization and bug correction on all platform (Windows, Linux, Mac Os X),
  - MPR and compare mode integration,
  - Image processing functions internal to viewer (list to define),
  - Redesign user interface,
  - Deployment with NeuroLOG Client to be finalized,
  - User documentation.

Applications

- **Done**
  - Local imaging data repositories (CAC=10, Visages=17, GIN=10)
  - Scuffl workflows for 4 processing pipelines
- **Todo**
  - Building Nice local database, populating existing databases
  - Insert Neurobehavioral data
  - Preliminary tests Viewer, Query interface, Workflows
  - Evaluation plan: definition of specific scenarios for qualitative and qualitative architecture evaluation
  - Perform deployment and evaluation phase

Toward evaluation phase


Processing on grid
Neurolog Web client
Query interface
Viiewer
Compute interface
Middleware NeuroLOG
Semantic Repository: OntoNeuroLOG – Work Flow management
NeuroLog Server
Dfquery server
Securised access to local files
Securised access to local services
CAC (Paris Salpétrière)
Data available (10 subjects)
NeuroLog Server
Dfquery server
Securised access to local files
Securised access to local services
InriaNeuroTk (Rennes)
Data available (17 subjects)
Neurologs erver
Dfquery server
Securised access to local files
Securised access to local services
GIN-DMS (Grenoble)
Data available (10 subjects)
Neurolog server
Dfquery server
Securised access to local files
Securised access to local services
Nice
Under construction