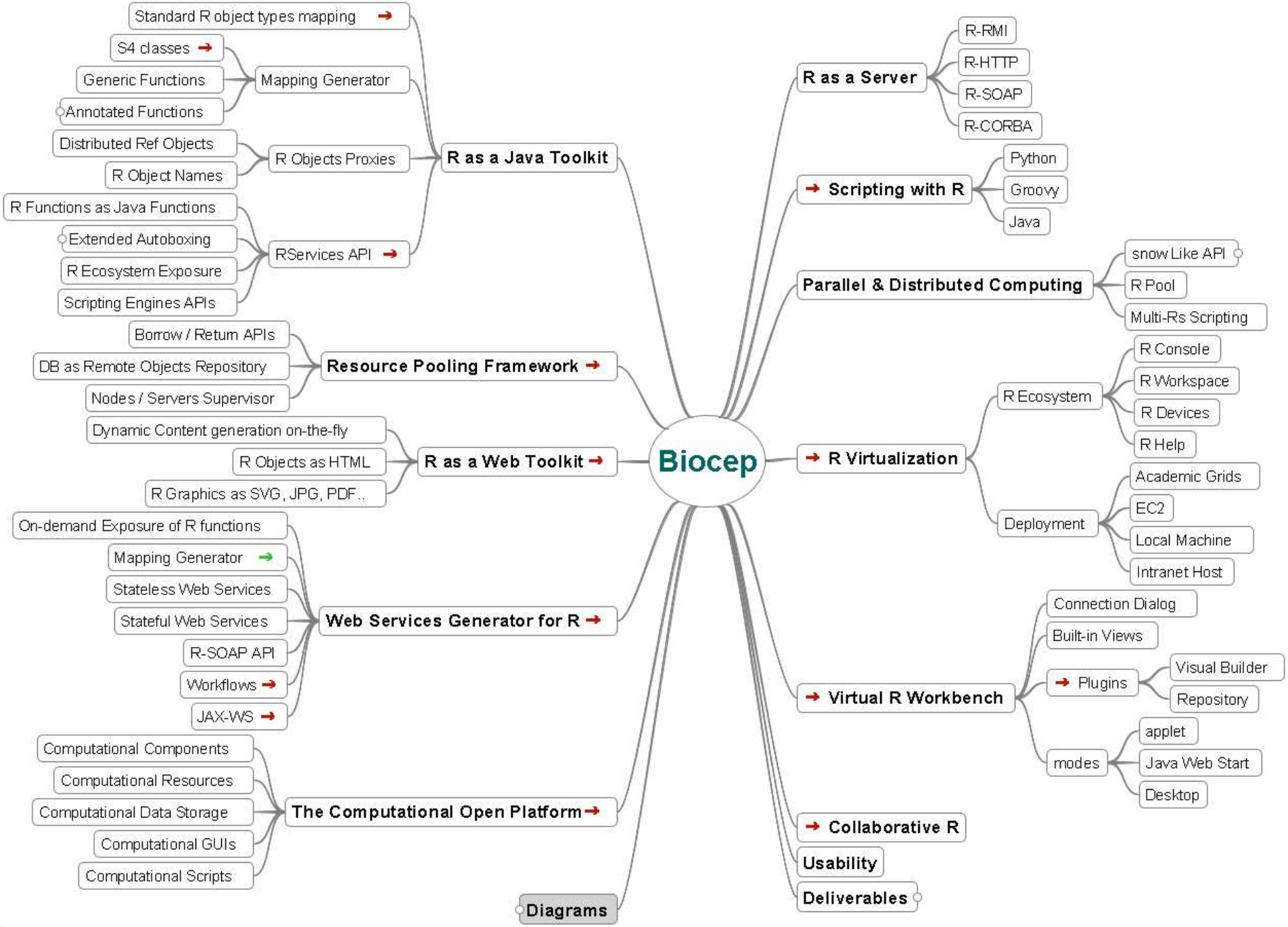
Biocep-R

Towards a federative user-centric and grid-enabled computational open platform

Karim Chine

karim.chine@m4x.org



Interactive Biocep mind map here : http://biocep-distrib.r-forge.r-project.org/freemindbrowser.html

Biocep Computational Open Platform Ecosystem

Computational Components

R packages : CRAN, Bioconductor,..

Wrapped C,C++,Fortran,.. code

open source or commercial

Computational Resources

R servers on local machine or on remote hosts

R servers on clusters, grids

free: academic grids, NGS,.. or pay-per-use: EC2, brokers,..

Computational GUIs

Virtual workbench within the browser

Built-in views (console, graphics, spreadsheets,editors..)

Plugins views (open source or commercial)

Collaborative data analysis views

Computational Data Storage

Shared file systems, FTP servers, S3

Computational Scripts

Python or Groovy

On client side (interactivity..)

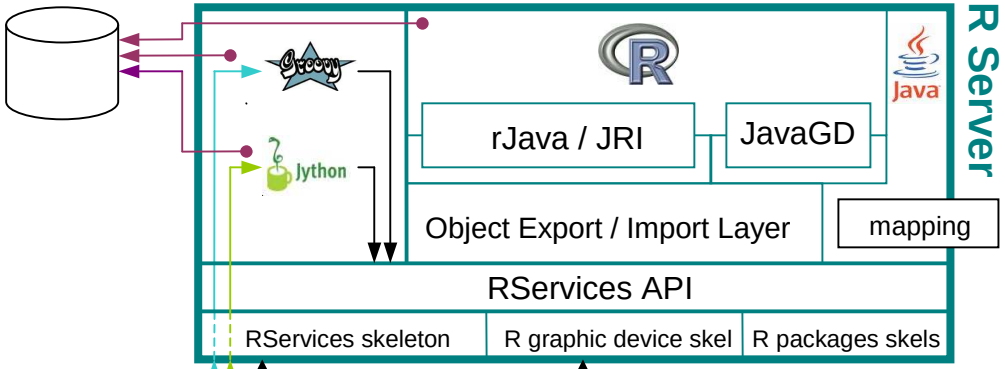
On server side (data transfer ..)

Generated Computational Web Services

Stateful or stateless, automatic mapping of R data objects and functions

Computational Engine API: R as a stateful web service

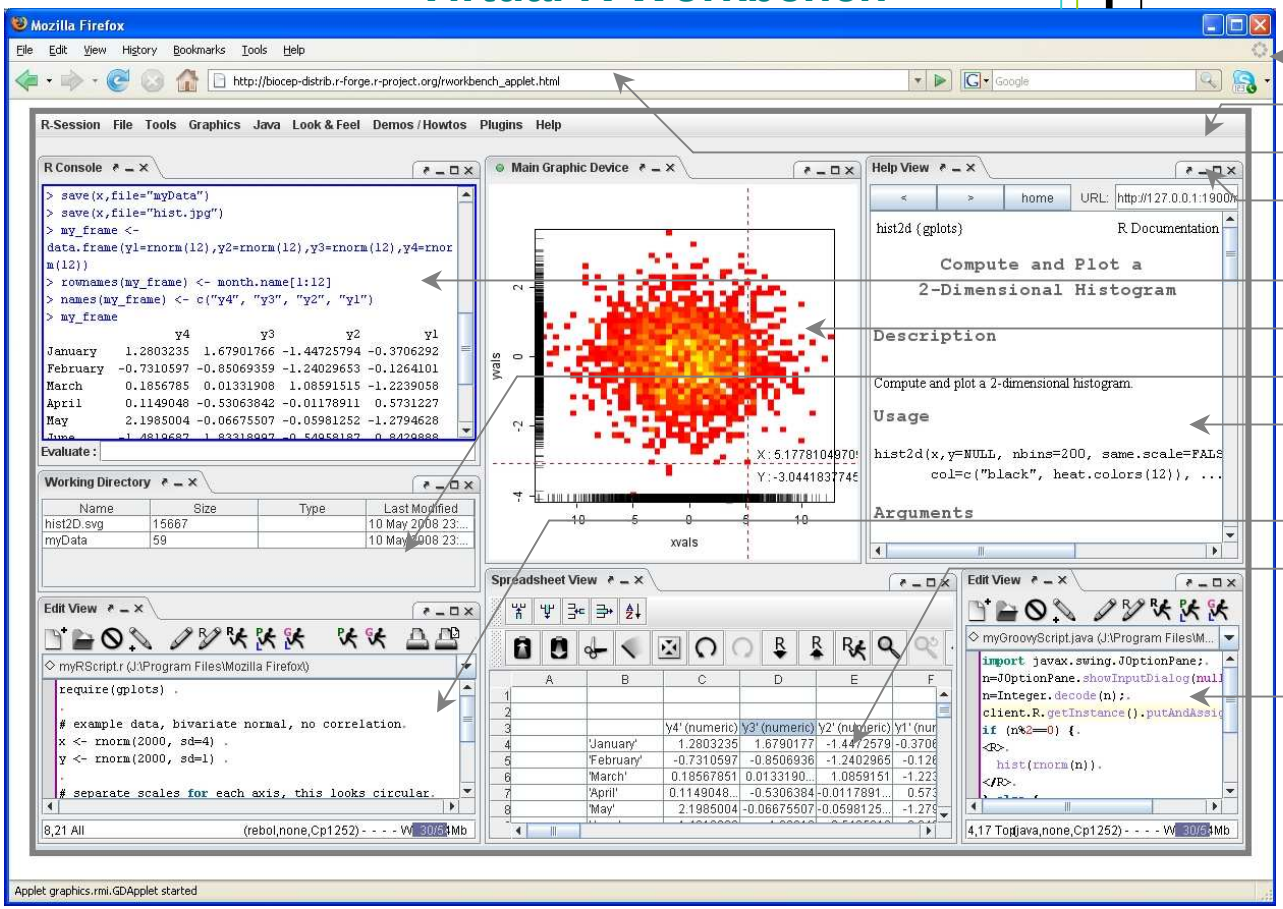
R Virtualization



SERVER SIDE - ACADEMIC GRIDS, NGS, EC2, INTRANET HOST..

CLIENT SIDE - INTERNET

Virtual R Workbench



Internet Browser

Java Applet

Virtual R Workbench URL

Docking Framework

R Console

R Graphic Device+Interactors

R Workspace

R Help Browser

R Script Editor

R Spreadsheet

Groovy / Jython Script Editor

Integrating R - State of the art

- **SJava and rJava/JRI**
  - Basic mapping via JNI of the R C API
- **TypeInfo**
  - Plug meta descriptions to R functions
- **RWebservices**
  - Generated Java Beans for basic R Types / S4 Classes
  - Axis Web Services based on SJava and ActiveMQ
- **JavaGD**
  - R devices connection to Java (JGR)
- **Rserve**
  - TCP/IP interface to R

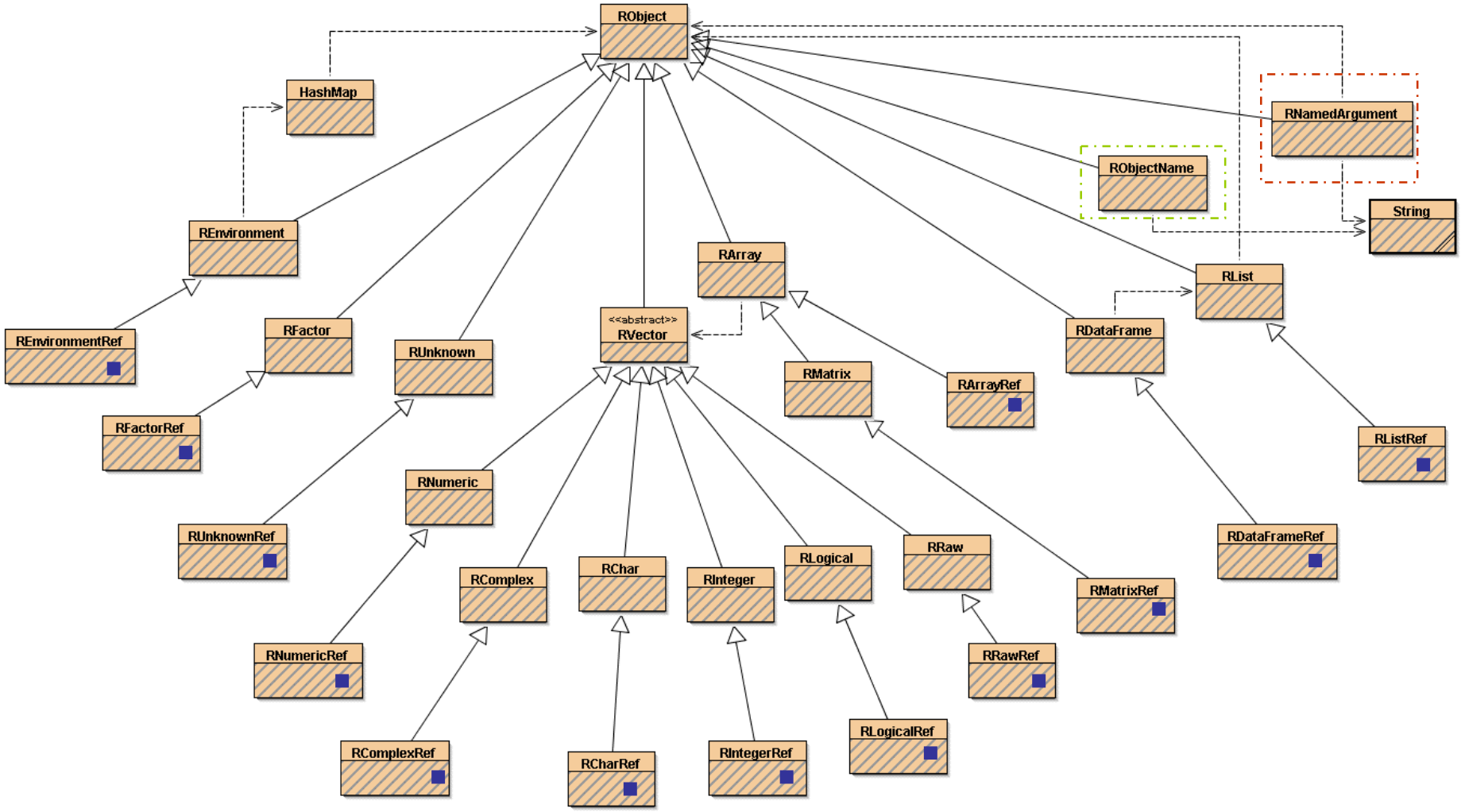
What is missing ?

- High Level Java API for Accessing R
- Stateful, Resuable, Remotable R Components
- Scalable, Distributed, R Based Infrastructure
- Safe multiple clients framework for components usage as a pool of indistinguishable Remote Resources
- User friendly Interface for the remote resources creation, tracking and debugging

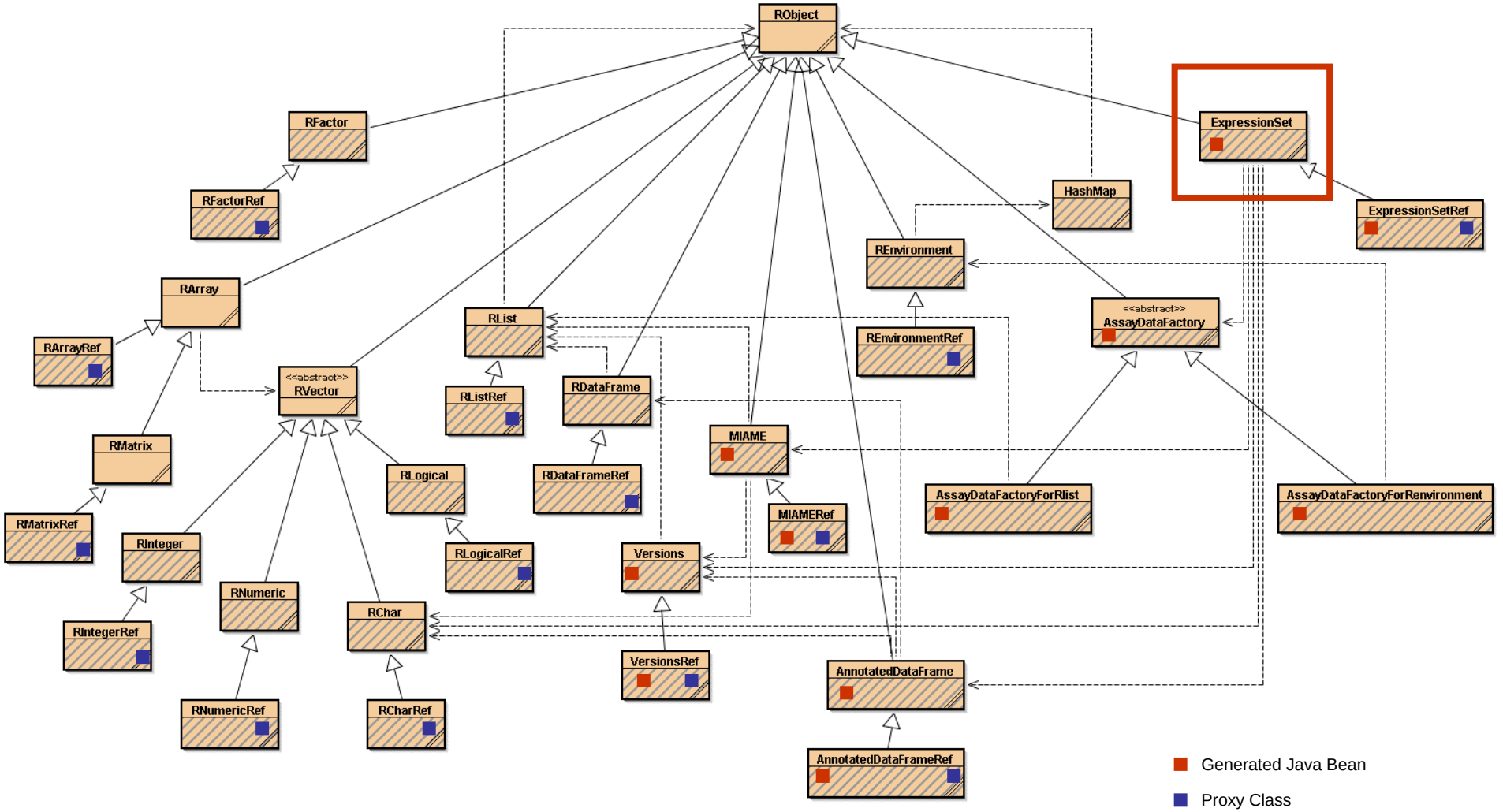
What is missing ?

- Generated light-weight Java proxies for R Types / S4 Classes
- On-demand mapping and deployment of R packages as RMI Components or as JAX-WS Web Services
- Remotable R Graphics / Swing Components for R
- Remote R components files exchange API
- Semi-thick client (applet) for web based tools using R

Standard R objects mapping to Java



Generated beans for ExpressionSet



RServices API - I

```
public interface RServices extends ManagedServant {
-----------------------------------------------------------------------
   public String  consoleSubmit(String expression) throws …
   public String  evaluate(String expression) throws …
-----------------------------------------------------------------------
   public RObject getObject(String expression) throws …
   public Object  getObjectConverted(String expression) throws …
   public RObject getReference(String expression) throws …
   public RObject getObjectName(String expression) throws …
-----------------------------------------------------------------------
   public void    putAndAssign(Object obj, String name) throws …
   public RObject putAndGetReference(Object obj) throws RemoteException;
-----------------------------------------------------------------------
   public RObject call(String methodName, Object... args) throws …
   public RObject callAndConvert(String methodName, Object... args) throws …
   public RObject callAndGetReference(String methodName, Object... args) throws …
   public RObject callAndGetObjectName(String methodName, Object... args) throws …
   public void    callAndAssign(String varName,String methodName,Object...args)throws .
-----------------------------------------------------------------------
   public RObject realizeObjectName(RObject objectName) throws …
   public Object  realizeObjectNameConverted(RObject objectName) throws …
   public RObject referenceToObject(RObject refObj) throws …
-----------------------------------------------------------------------
   public boolean isReference(RObject obj) throws …
   public void    assignReference(String name, RObject refObj) throws …
-----------------------------------------------------------------------
}
```

RServices API - II

```
public interface RServices extends ManagedServant {
-----------------------------------------------------------------------
   public String[] listPackages() throws …
   public RPackage getPackage(String packageName) throws …
-----------------------------------------------------------------------
   public GDDevice newDevice(int w, int h) throws …
   public GDDevice[] listDevices() throws …
```

```
public interface GDDevice extends Remote {
    public Vector<GDObject> popAllGraphicObjects() throws …
    public void fireSizeChangedEvent(int w, int h) throws …
    public void dispose() throws …
    …
}
```

```
-----------------------------------------------------------------------
   public String[] getWorkingDirectoryFileNames() throws …
   public FileDescription getWorkingDirectoryFileDescription(String fileName) throws…
   public void     createWorkingDirectoryFile(String fileName) throws …
   public void     removeWorkingDirectoryFile(String fileName) throws …
   public byte[]   readWorkingDirectoryFileBlock(String name,long off,int size)throws…
   public void     appendBlockToWorkingDirectoryFile(String name, byte[] block)throws…
-----------------------------------------------------------------------
   public String   getRHelpFileUri(String topic, String pack) throws …
   public byte[]   getRHelpFile(String uri) throws …
-----------------------------------------------------------------------
   public Vector<RAction> popRActions() throws …
-----------------------------------------------------------------------
}
```

RServices API - III

```
public interface RServices extends ManagedServant {
-------------------------------------------------------------------------
   public void    startHttpServer(int port) throws …
   public void    stopHttpServer() throws …
-------------------------------------------------------------------------
   public String  pythonExec(String pythonCommand) throws …
   public RObject pythonEval(String pythonCommand) throws …
   public void    pythonSet(String name, Object Value) throws …
-------------------------------------------------------------------------
   public String  groovyExec(String groovyCommand) throws …
   public Object  groovyEval(String expression) throws …
   public void    groovySet(String name, Object Value) throws …
-------------------------------------------------------------------------
   public void    setCallBack(RCallback callback) throws …
-------------------------------------------------------------------------
   public String  getStatus() throws …
   public void    stop() throws …
   public void    freeReference(RObject refObj) throws …
   public void    freeAllReferences() throws …
   public String  print(String expression) throws …
   public String  sourceFromResource(String resource) throws …
   public String  sourceFromBuffer(StringBuffer buffer) throws …
   public RNI     getRNI() throws …
-------------------------------------------------------------------------
   …
}
```

Remote Resources Pooling Framework

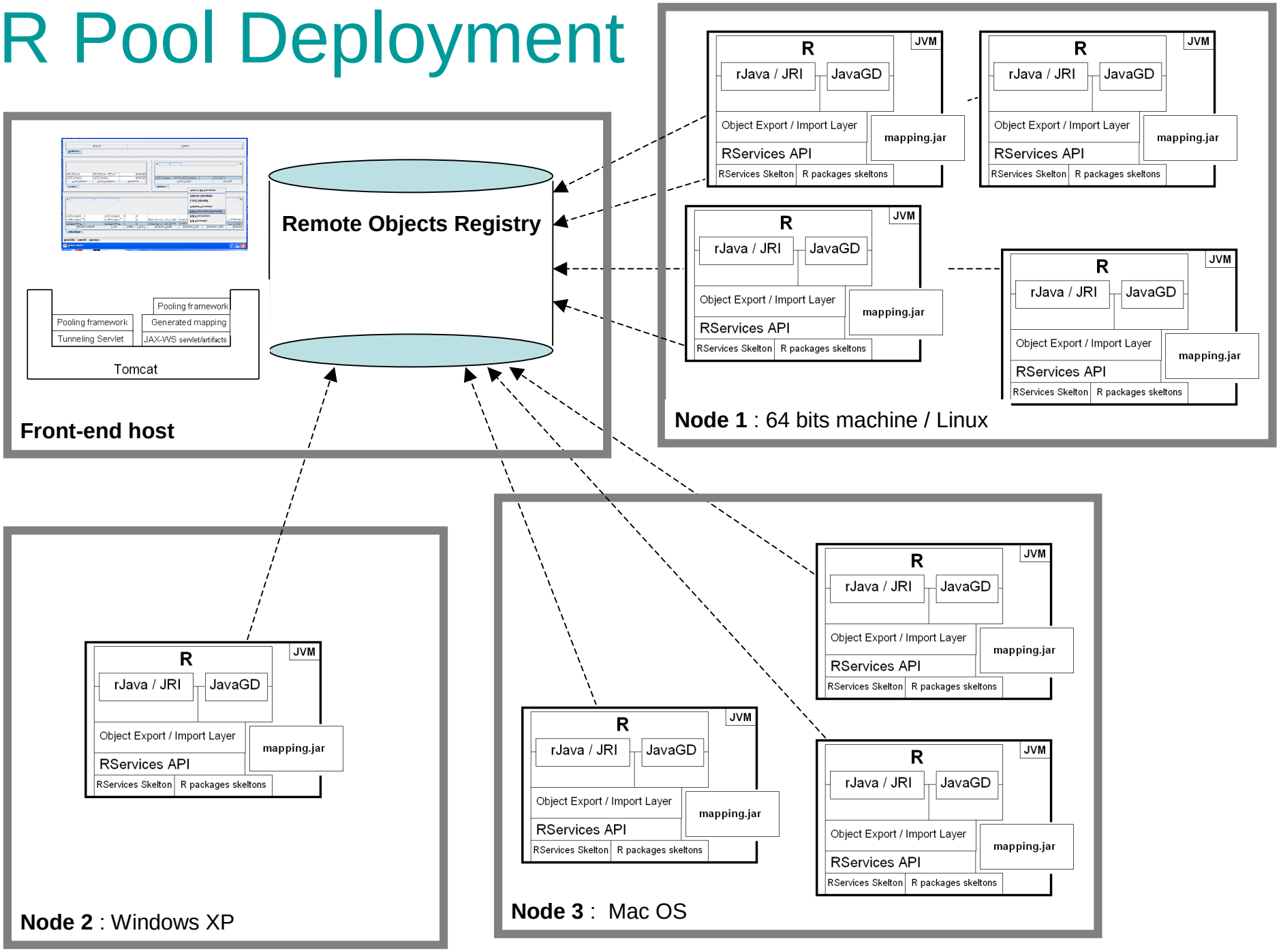
- Generic Standalone framework
- Pooling of any RMI components and if combined with JNI of any library / open architecture

- New Remote Object Registry based on Derby| Oracle| MySQL

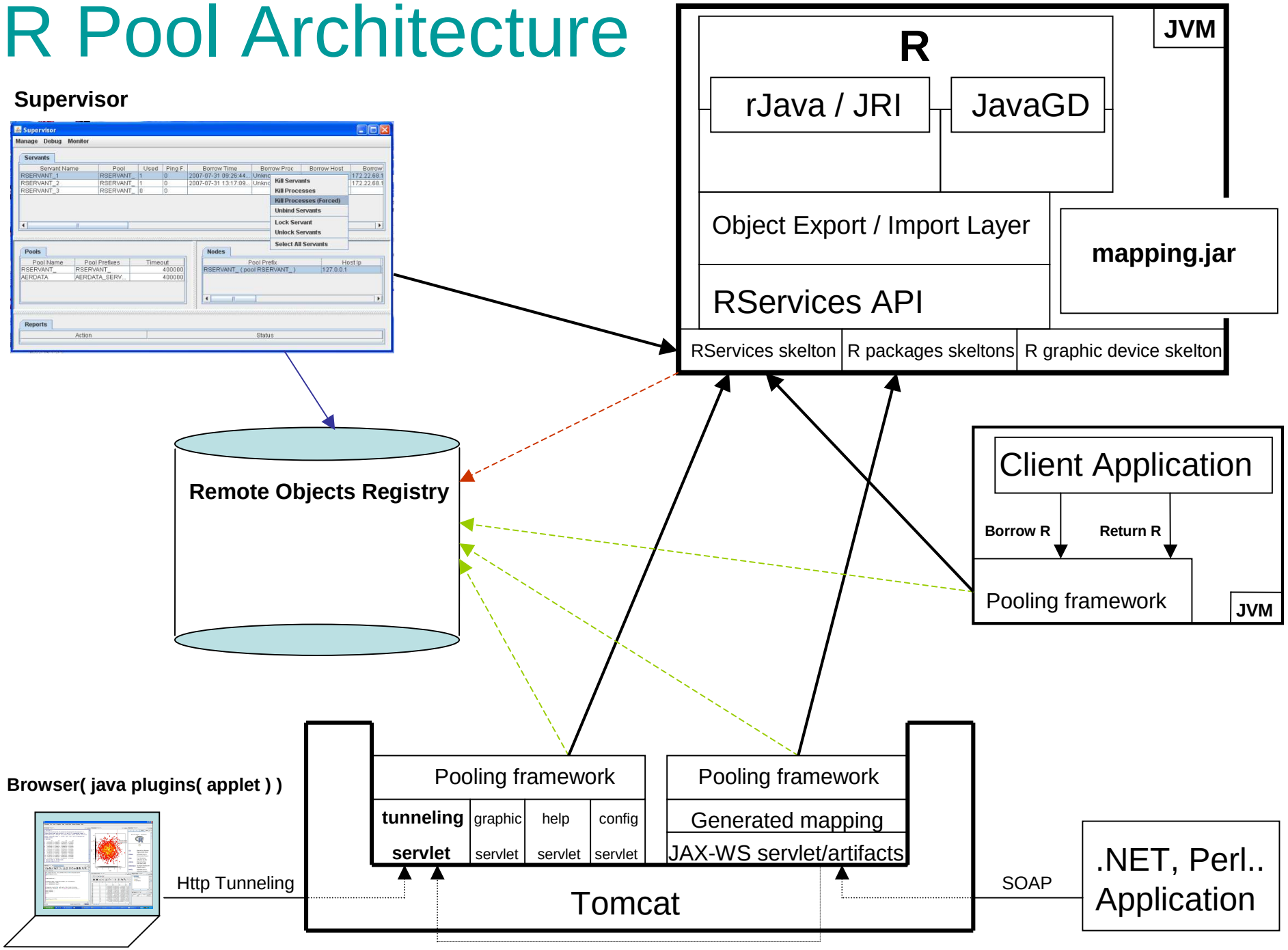
- Three implementations available
  - rmiregistry / mono-node / single client process
  - rmiregistry / multinodes / single client process
  - database ROR / multinodes / multiple client processes

- User friendly interface for the remote resources creation, tracking and debugging, nodes and pools management

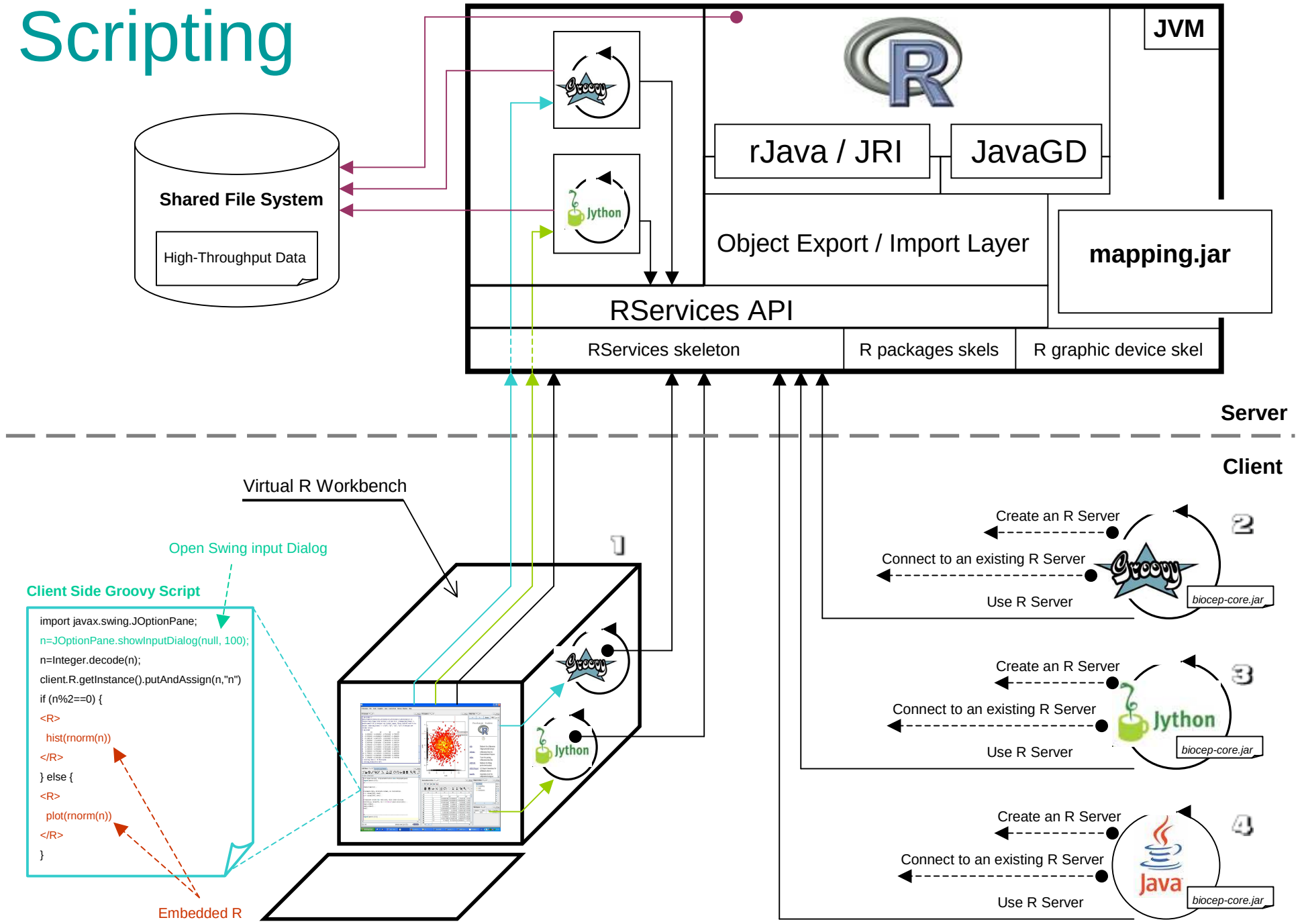
R Pool Deployment



R Pool Architecture



Scripting


JVM
Shared File System
High-Throughput Data
rJava / JRI
JavaGD
Object Export / Import Layer
mapping.jar
RServices API
RServices skeleton
R packages skels
R graphic device skel
Server
Client
Virtual R Workbench
Open Swing input Dialog
Client Side Groovy Script
import javax.swing.JOptionPane;
n=JOptionPane.showInputDialog(null, 100);
n=Integer.decode(n);
client.R.getInstance().putAndAssign(n,"n")
if (n%2==0) {
<R>
hist(rnorm(n))
</R>
} else {
<R>
plot(rnorm(n))
</R>
}
Embedded R
1
Create an R Server
Connect to an existing R Server
Use R Server
2
biocep-core.jar
3
4

Parallel Computing

```
final double[][] m=..;
Future<Double>[] result=new Future[m.length];
ExecutorService exec = Executors.newFixedThreadPool(50);
for (int i=0; i<result.length; ++i) {
                final double[] v=m[i];
                result[i]= exec.submit(
            new Callable<Double>() {
                    public Double call() throws Exception {
                      RServices r=null;
                      try {
                            r=(RServices)ServantProviderFactory.getFactory().getServantProvider().borrowServantProxy();
                            Rnumeric mean=(RNumeric)r.call("mean", new RNumeric(v));
                            return mean.getValue()[0];
                      } finally { ServantProviderFactory.getFactory().getServantProvider().returnServantProxy(r); }
                  }
          });
}
while(true) {
                int count=0; for (int i=0; i<result.length; ++i) if (result[i].isDone()) ++count; if (count==result.length) break;
                Thread.sleep(100);
}
for (int i=0; i<result.length; ++i) System.out.println(result[i].get());
```

Snow with Biocep

From the R Console :

- makeCluster(n,...) stopCluster(cl)
  - → Starting and Stopping clusters
- clusterEvalQ(cl, expr)
  - → The expression is evaluated on the slave nodes.
- clusterApply(cl, seq, fun, ...)
  - → Calls the function with the first element of the list on the first node, with the second element of the list on the second node, and so on.
- clusterExport(cl, list)
  - → Assigns the global values on the master of the variables named in 'list' to variables of the same names in the global environments of each node.

…

Web Services Generation

Script / globals.r

```
square ← function(x) {return(x^2) }
typeInfo(square) ← SimultaneousTypeSpecification(
TypedSignature(x = "numeric"), returnType = "numeric")
```

Script / rjmap.xml

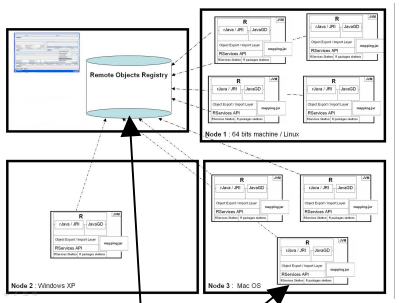
```
<rj>
  <publish>
    <functions> <function name="square" forWeb="true"/> </functions>
  </publish>
  <scripts>  <initScript name="globals.r" embed="true"/> </scripts>
</rj>
```

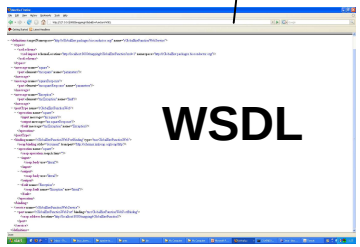
WS generator

rws.war

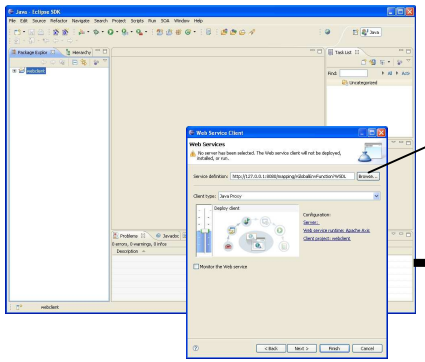
+ mapping.jar
+ pooling framework
+ R Java Bridge
+ **JAX-WS**
  - Servlets
  - Generated artifacts

Deploy

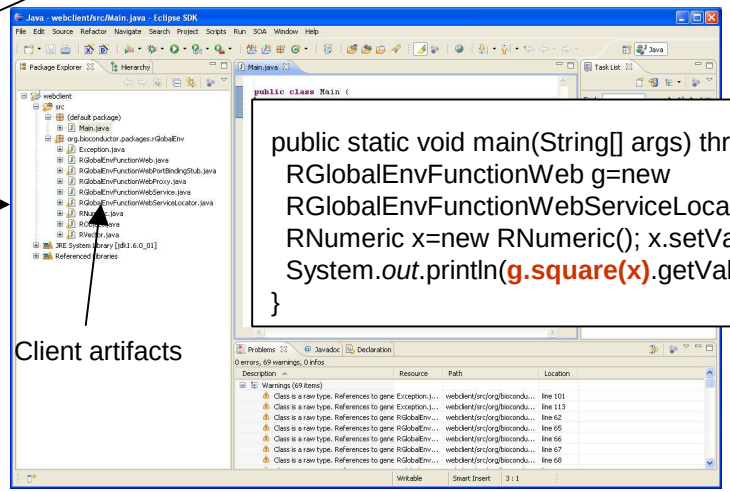




http://127.0.0.1:8080/rws/rGlobalEnvFunction?WSDL

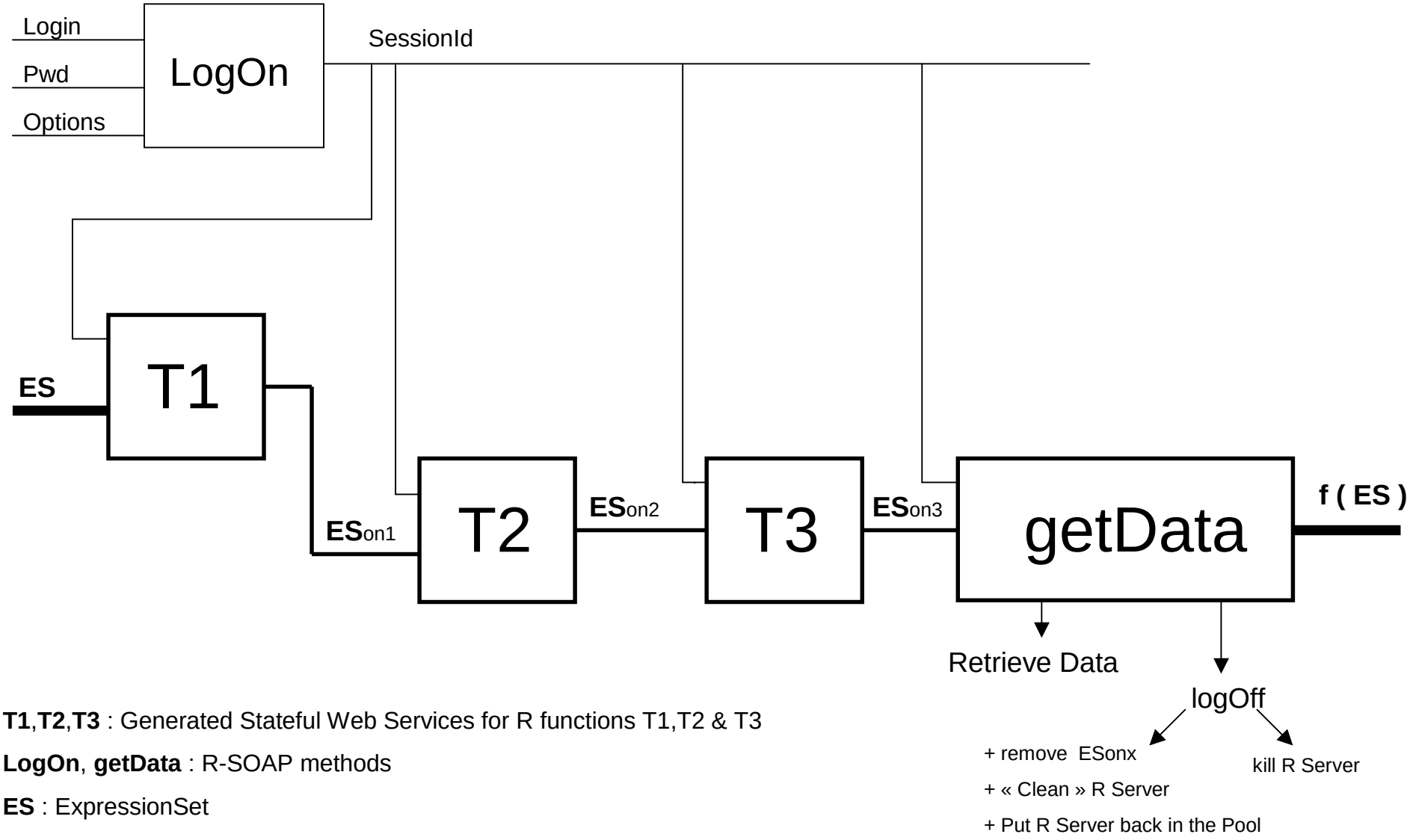


Eclipse Web Service Client Generator




```
public static void main(String[] args) throws Exception {
  RGlobalEnvFunctionWeb g=new
  RGlobalEnvFunctionWebServiceLocator().getrGlobalEnvFunctionWebPort();
  RNumeric x=new RNumeric(); x.setValue(new Double[]{6.0});
  System.out.println(g.square(x).getValue()[0]);
}
```

Workflows with Stateful Web Services



T1,**T2**,**T3** : Generated Stateful Web Services for R functions T1,T2 & T3

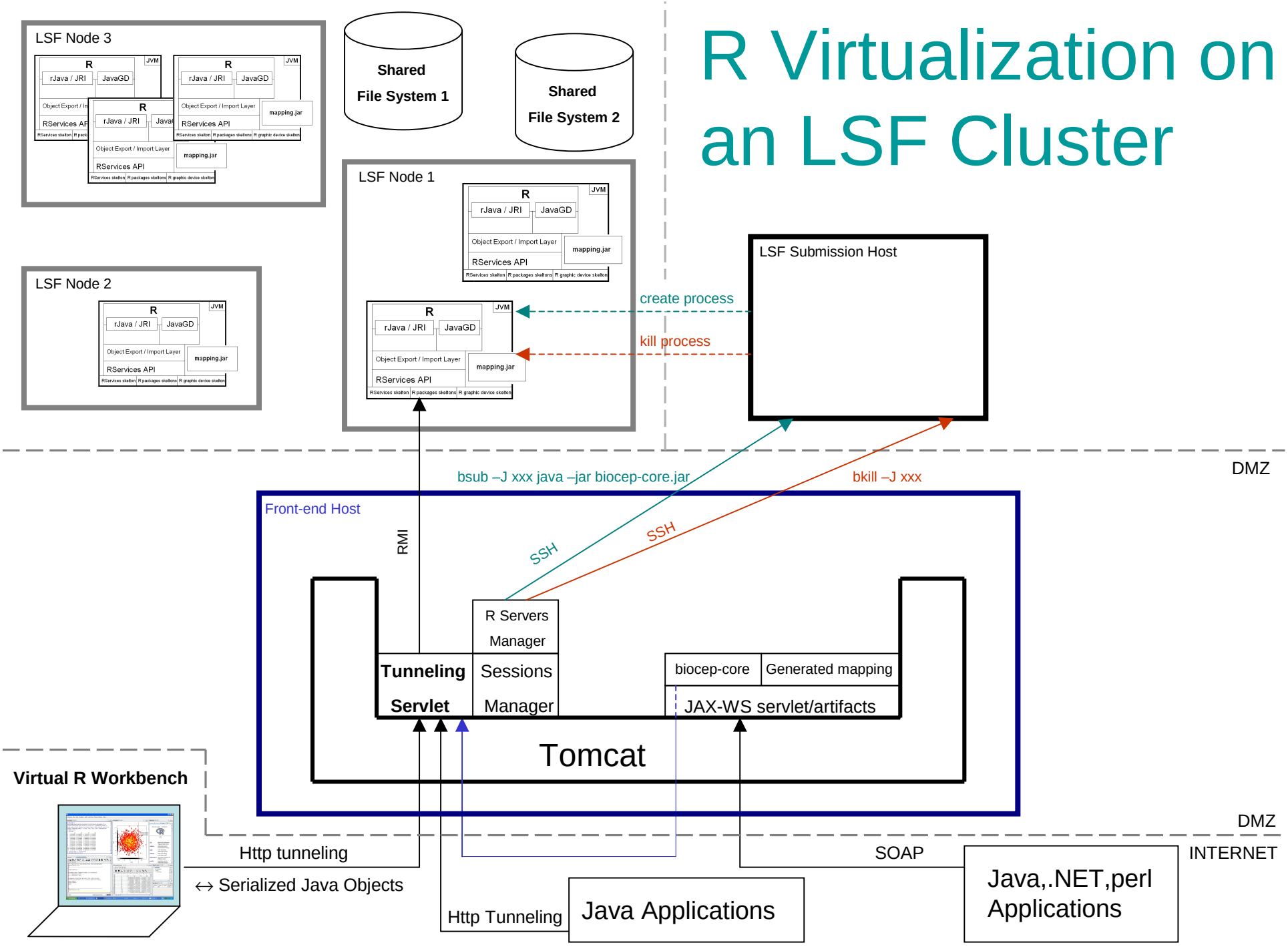
LogOn, **getData** : R-SOAP methods

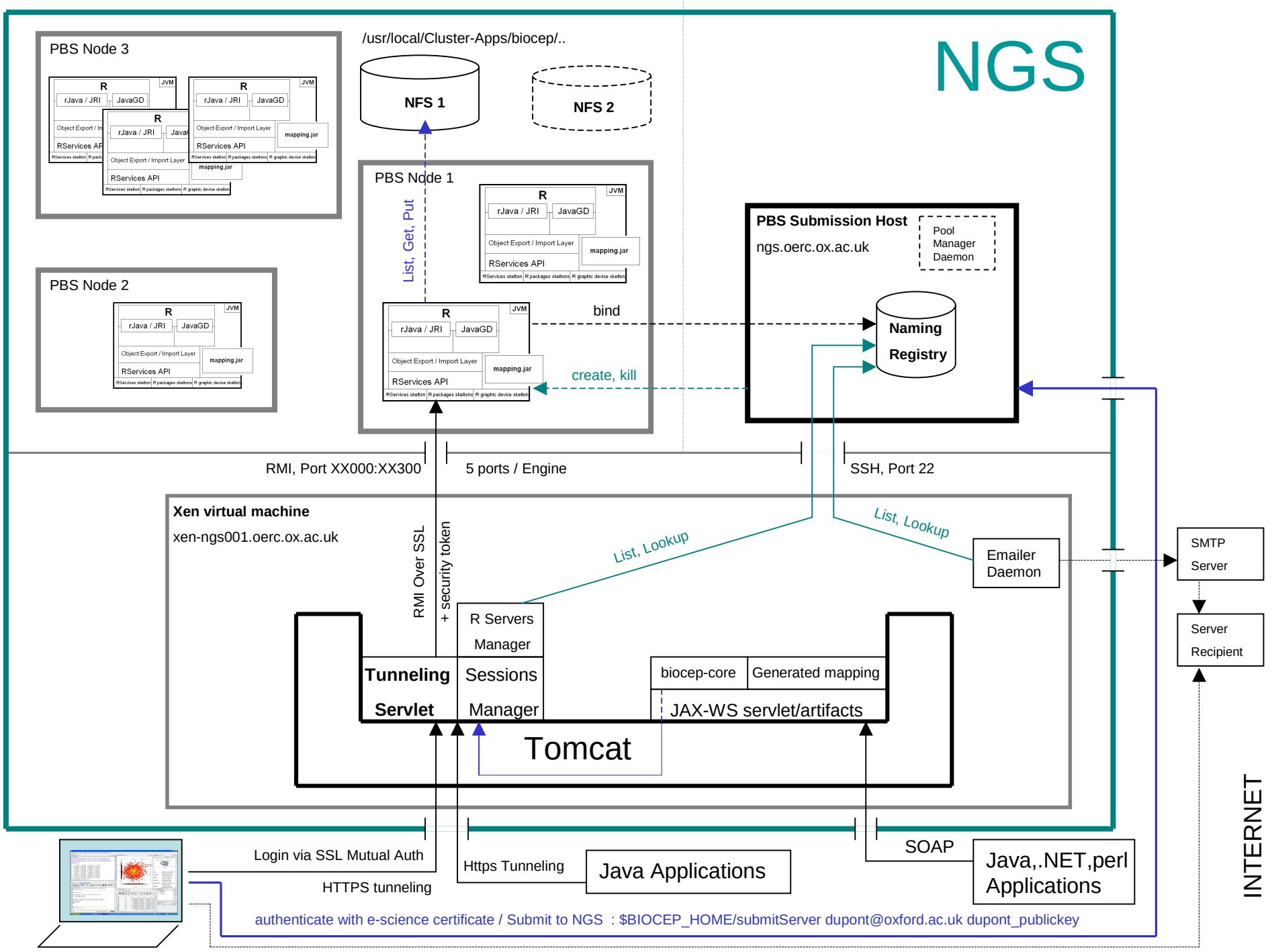
ES : ExpressionSet

ESon1, **ES**on2, **ES**on3 : ExpressionSet Object Names

f = T3 o T2 o T1

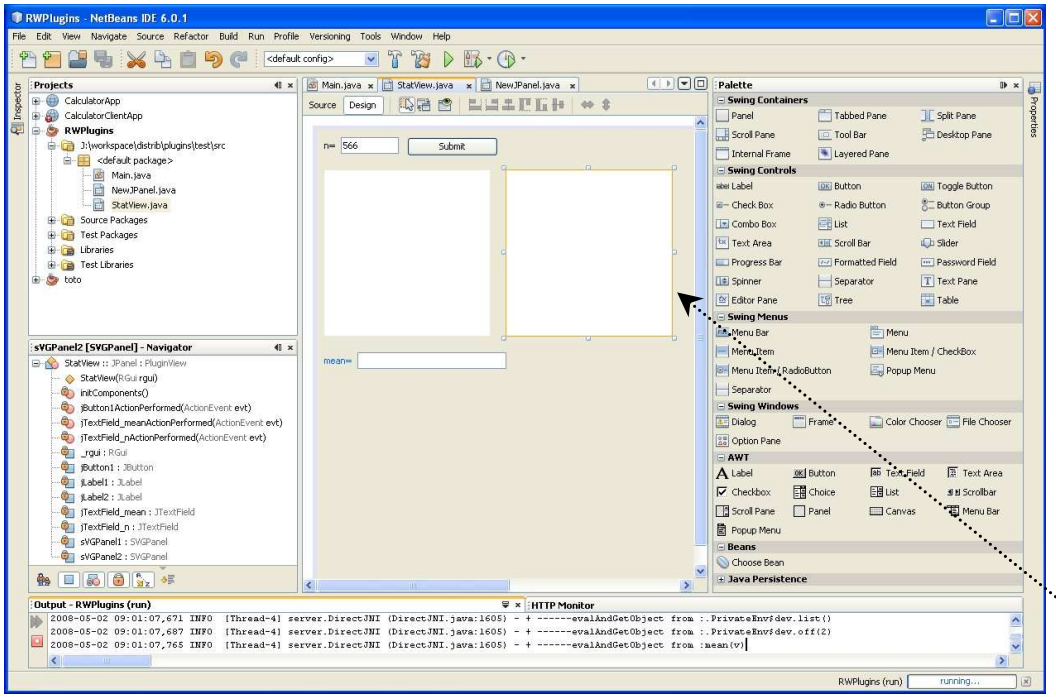
R Virtualization on an LSF Cluster




NGS
PBS Node 3
/usr/local/Cluster-Apps/biocep/..
NFS 1
NFS 2
PBS Node 1
List, Get, Put
PBS Node 2
PBS Submission Host
ngs.oerc.ox.ac.uk
Pool Manager Daemon
Naming Registry
bind
create, kill
RMI, Port XX000:XX300
5 ports / Engine
SSH, Port 22
Xen virtual machine
xen-ngs001.oerc.ox.ac.uk
RMI Over SSL
+ security token
List, Lookup
List, Lookup
Emailer Daemon
SMTP Server
Server Recipient
R Servers Manager
Tunneling Servlet
Sessions Manager
biocep-core
Generated mapping
JAX-WS servlet/artifacts
Tomcat
Login via SSL Mutual Auth
HTTPS tunneling
Https Tunneling
Java Applications
SOAP
Java,.NET,perl Applications
INTERNET
authenticate with e-science certificate / Submit to NGS : $BIOCEP_HOME/submitServer dupont@oxford.ac.uk dupont_publickey

Netbeans 6 – Visual GUI builder

GUI Plugins



Compile

myPlugin.jar

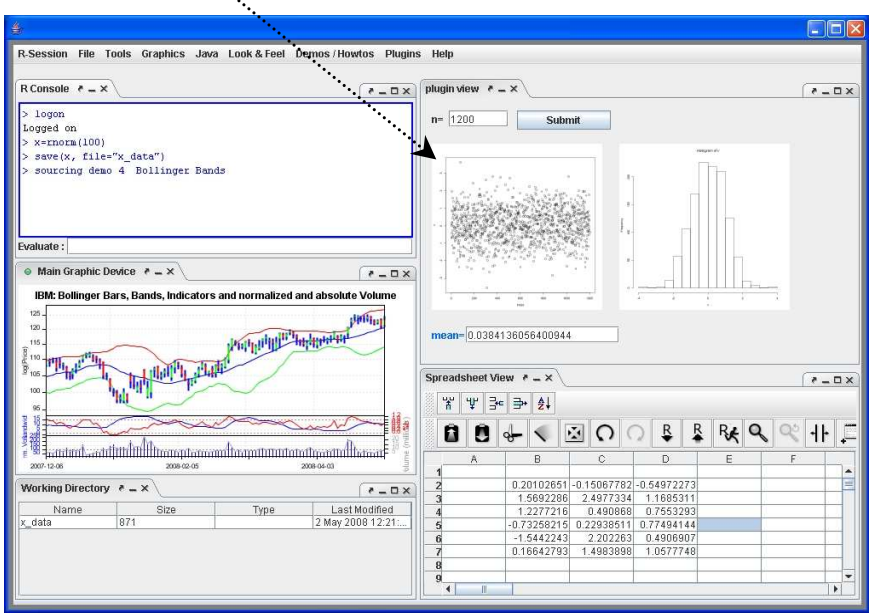
+ myView1

+ myView2

+ descriptor.xml

Import Plugin

Virtual R Workbench



Upload plugin

Browse Repository

Download Plugin

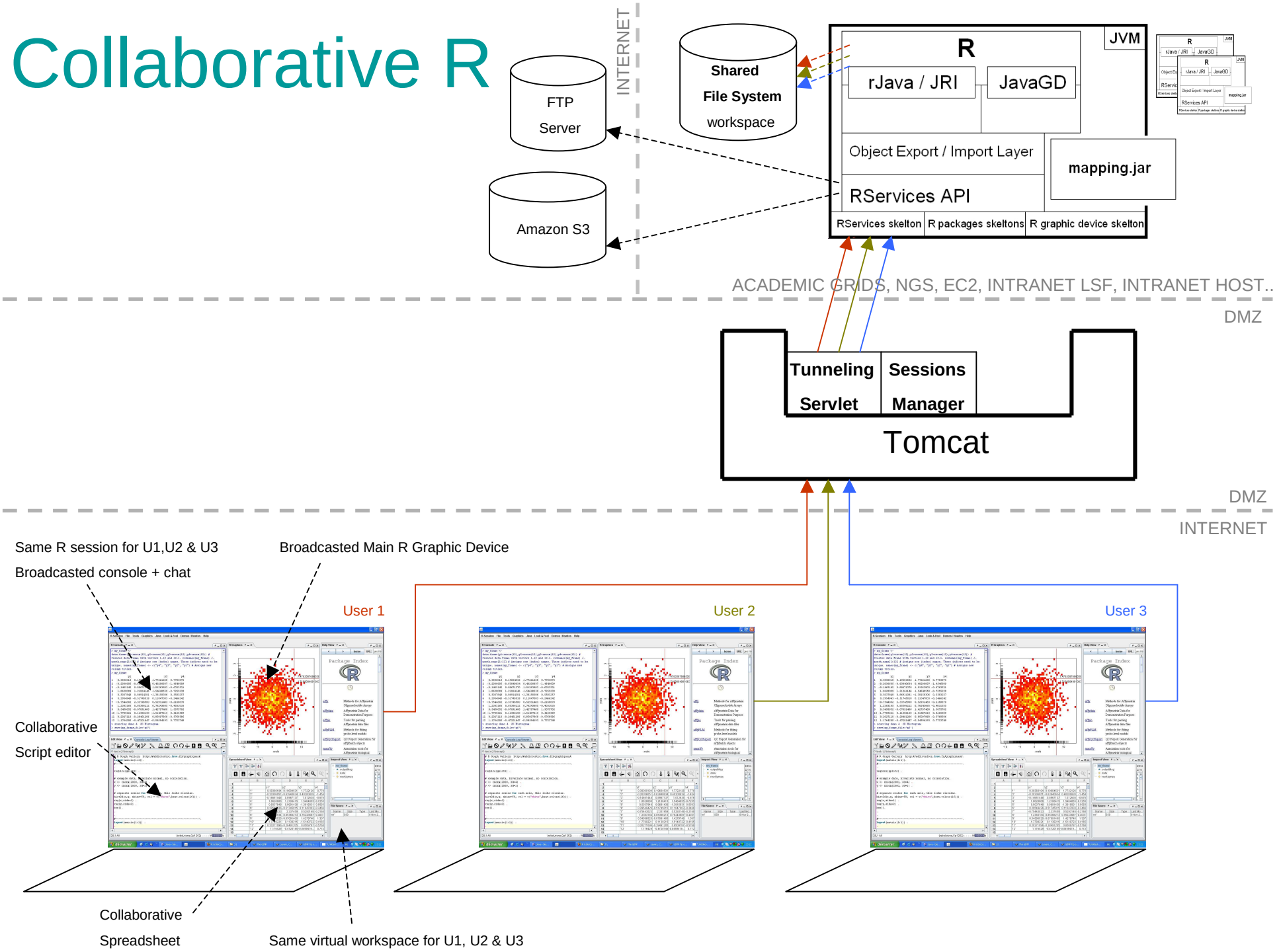
Plugins Repository

* myPlugin *myDashboard

* Klimt * iPlots

* Mondrian *E. Profiler

Collaborative R



Ease Of Use - I

- **Reasonable Pre-requirements**
  - java 5 installed : to run the workbench and connect to R servers on remote hosts
  - java 5 and R>=2.5 accessibles from the command line : to run R servers, generate mappings & Web Services, run the miniature virtualisation and the R-SOAP Web Apps..
- **All-in-one Highly Productive Workbench**
  - Docking framework, spreadsheets, syntax highlighting enabled editors, objects viewer, help browser, storage views, interactive zooming system on R graphics, settings persistence..
- **Easy Computational Resource Acquisition**
  - Provide nothing to run R servers on local machine
  - Provide HOST / PORT / LOGIN / PWD to run R Servers on remote hosts (SSH)
  - Provide URL & (LOGIN/PWD or X.509 Certificate) to Connect to Grid Rs or Cluster Rs
- **Easy Scripting**
  - Simple API for running/connecting to R servers
  - Embeddable R code (<R> </R>) within scripts
  - Automatic conversion from/to R Objects for common data types(standard,arrays,collections)

Ease Of Use -II

- **Easy Plugins Integration**
  - Import local file / Browse Plugins repository and choose a plugin
- **« Push button » Web Services Generation / Web Services Deployment**
  - Add TypeInfo to your function / add your function name to an XML / run biocep-tools
  - Deploy: *java –port=80 –cp biocep-core.jar HttpServer rvirtual.war MyWebServices.war*
- **Self-contained jar & war files distribution :**
  - biocep.jar biocep-core.jar biocep-tools.jar rvirtual.war rws.war
- **Configurationless Parallel Computing from R console :**
  - makeCluster(n,..), stopCluster(cl), clusterEvalQ(cl, expr), clusterApply(cl, seq, fun, ..) ...

Acknowledgements

AT&T Research Labs: Simon Urbanek

EBI: Alvis Brazma, Wolfgang Huber, Misha Kapushesky, Philippe Rocca-Serra, Ugis Sarkans

EPFL: Darlene Goldstein

ETH Zürich: Yohan Chalabi, Diethelm Würtz

FHCRC: Seth Falcon, Martin Morgan

Imperial College London: John Darlington, Brian Fuchs

OeRC: Matteo Turilli, David Wallom, Steven Young

www.biocep.net