Microarray Informatics at the EBI

Alvis Brazma Group

BIOCEP Project

BBSRC (UK) grant BB/E001653/1

Karim Chine

Principal Investigators: Wolfgang Huber, Misha Kapushesky

What do we need ?

- Provide tools and frameworks for Bioconductor packages automatic exposure as Web Services

- Provide a scalable, robust architecture for R integration within ArrayExpress Warehouse and Expression Profiler

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

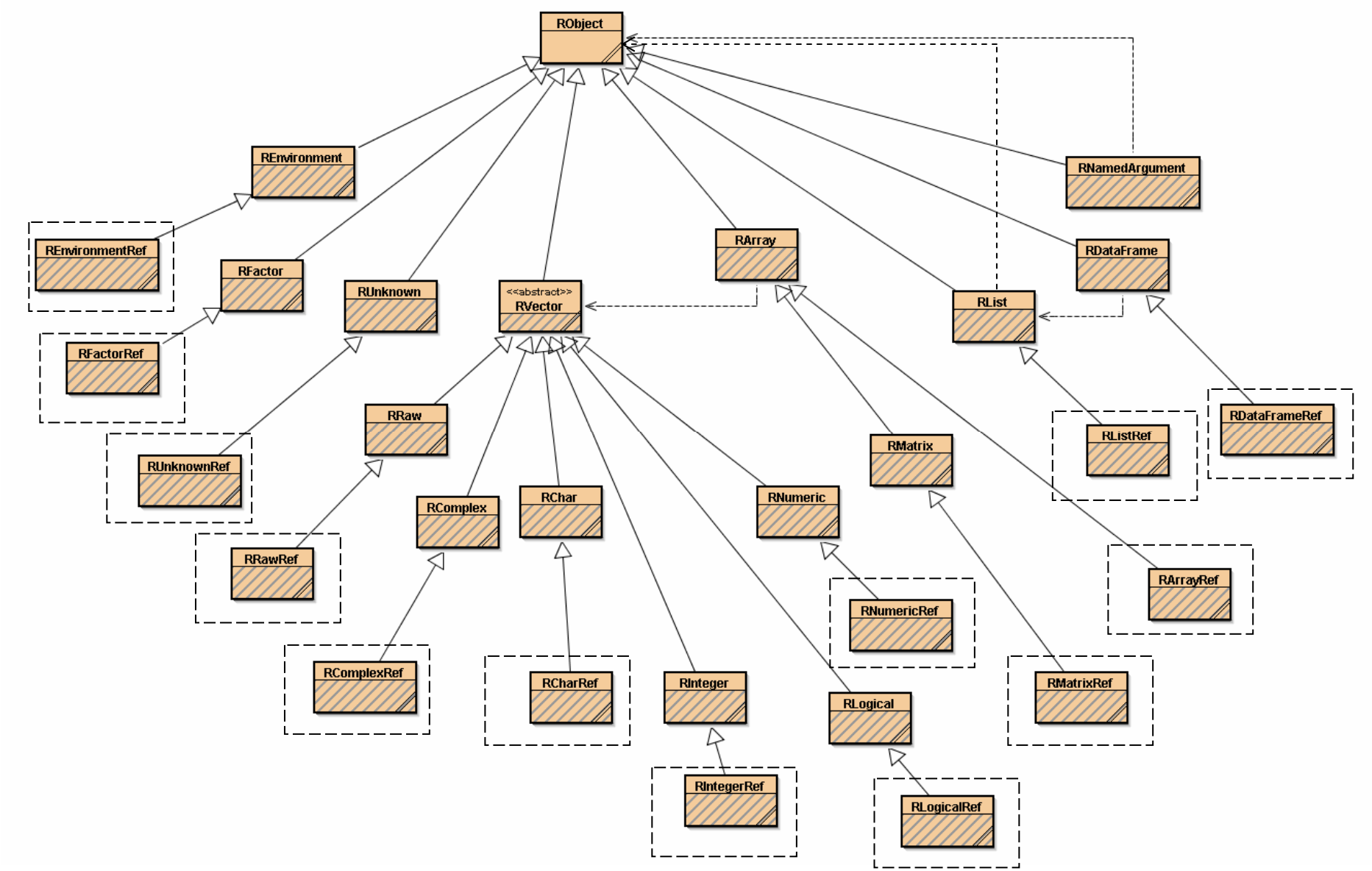
What is missing ?

- Generated light-weight Java proxies for R Types / S4 Classes
- On-demand mapping and deployment of R pacakges 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

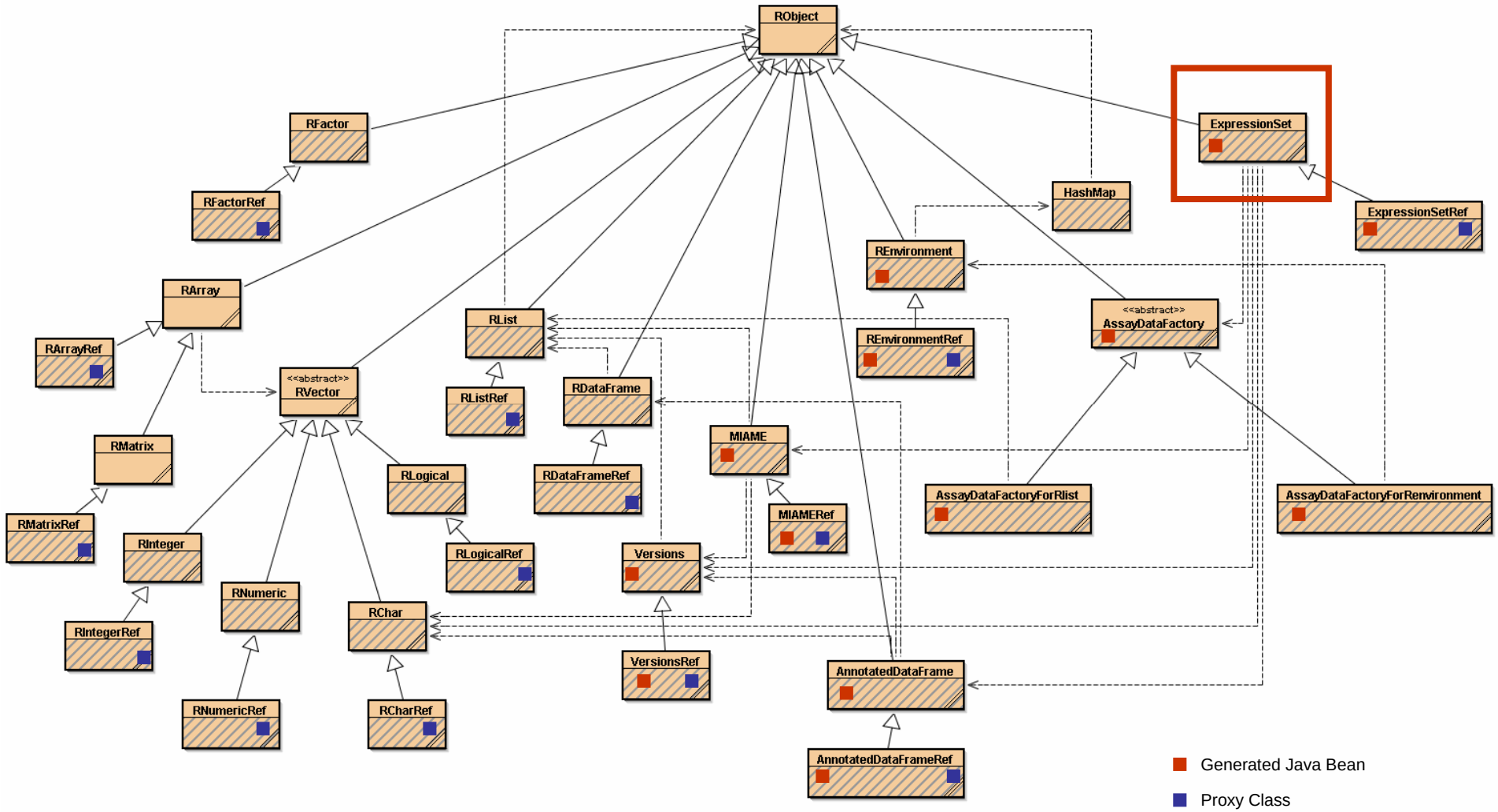
Remote Resource Pool Framework

- 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

Standard R objects mapping to Java



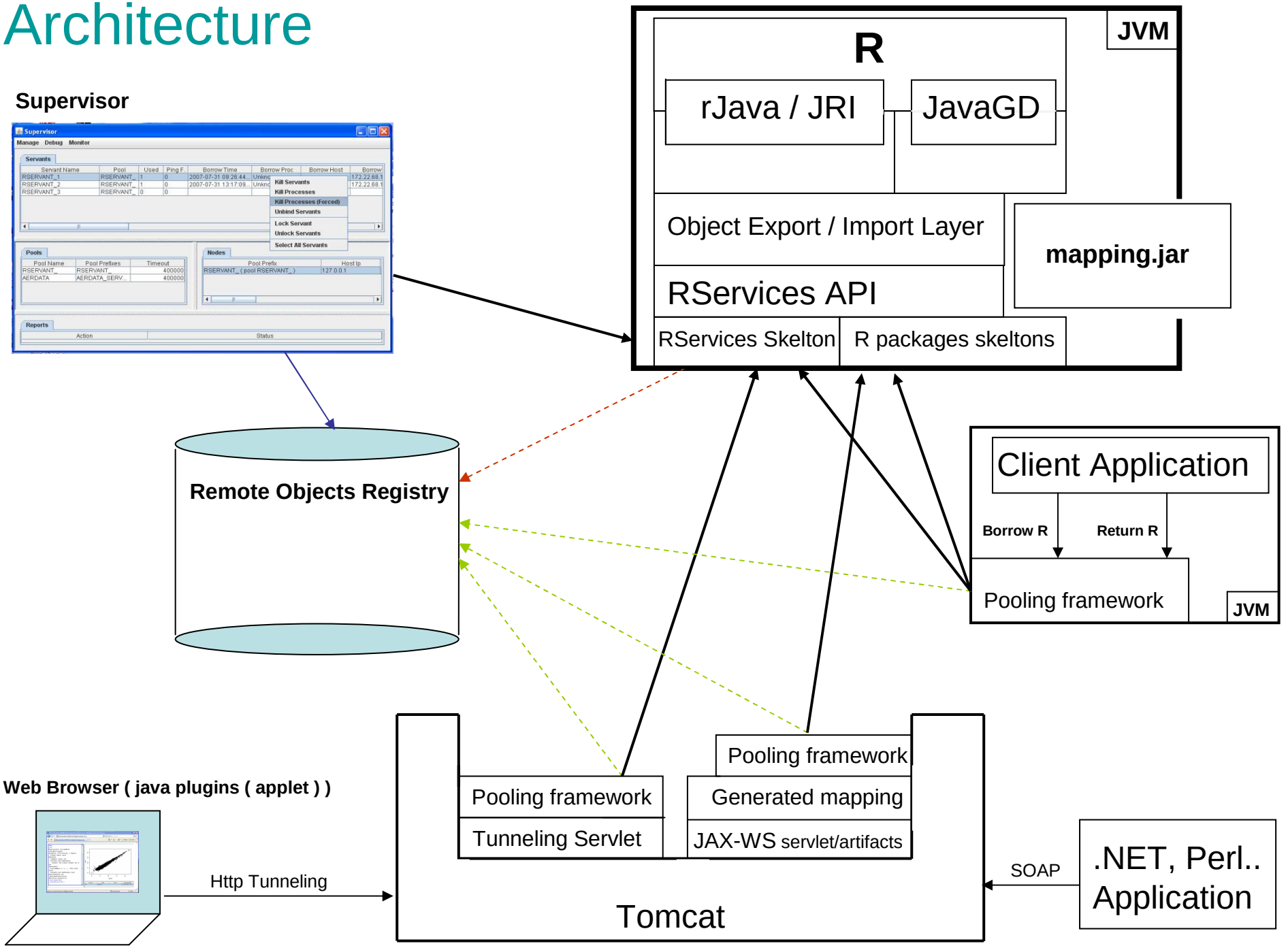
Generated java beans for the R S4 class ExpressionSet



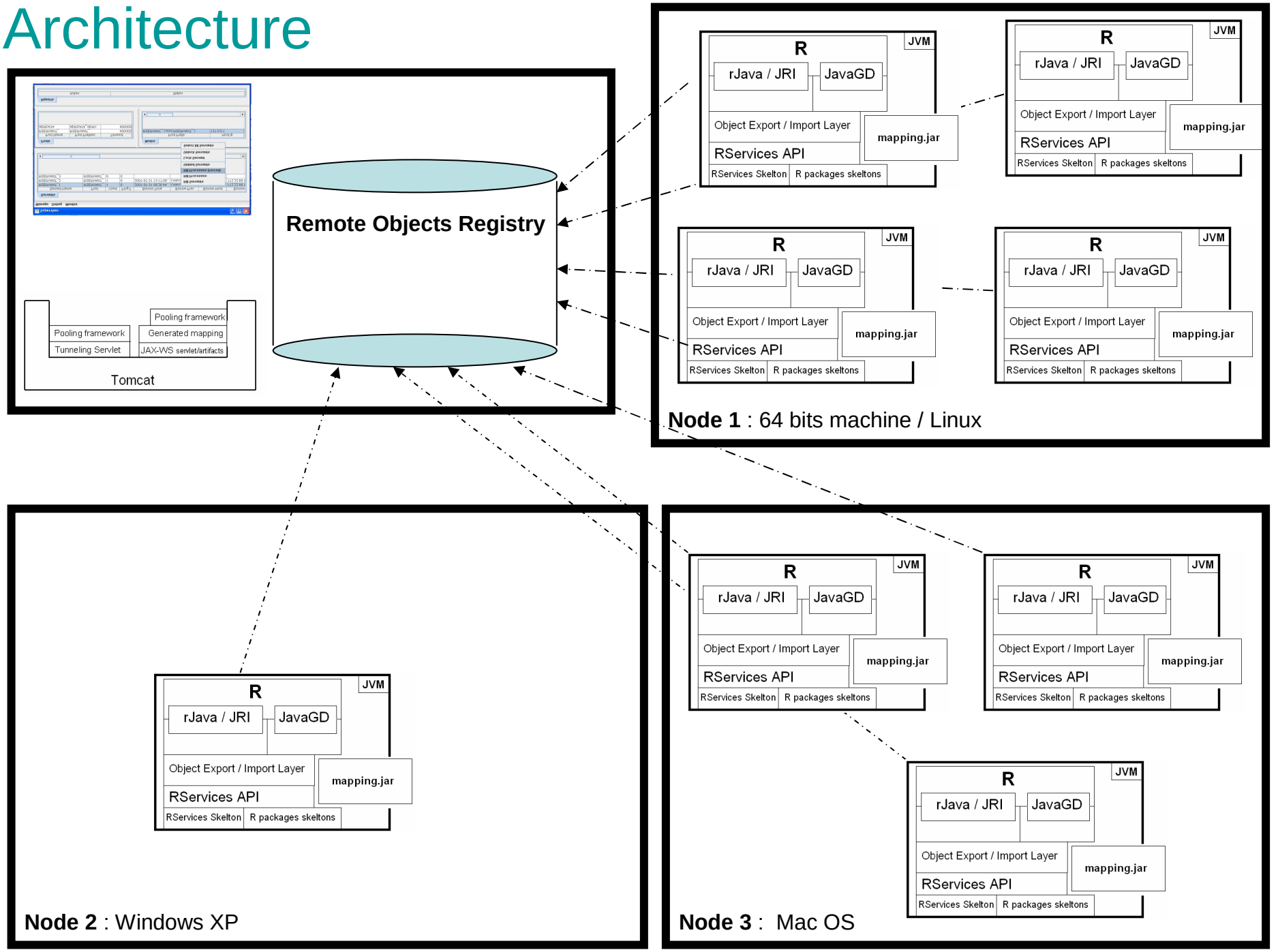
High Level API for Accessing R

```
public interface RServices extends ManagedServant {

        public String Evaluate(String expression) throws RemoteException;
        ...
        public RObject call(String methodName, RObject... args) throws RemoteException;
        ...
        public RObject callAsReference(String methodName, RObject... args) throws RemoteException;
        ...
        public RObject evalAndGetObject(String expression) throws RemoteException;
        ...
        public RObject evalAndGetObjectAsReference(String expression) throws RemoteException;
        ...
        public RObject putObjectAndGetReference(RObject obj) throws RemoteException;
        ...
        public void putObjectAndAssignName(RObject obj, String name) throws RemoteException;
        ...
        public RPackage getPackage(String packageName) throws RemoteException;
        ...
        public void setCallBack(RCallback callback) throws RemoteException;
        ...
        public GDDevice newDevice(int w, int h) throws java.rmi.RemoteException;
        ...
        public byte[] readWorkingDirectoryFileBlock(String fileName, long offset, int blocksize) ..
        ...
        public String getStatus() throws RemoteException;
        ...

}
```

Architecture



Architecture



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());
```

Web Services Generation

Script / globals.r

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

Script / rjmap.xml

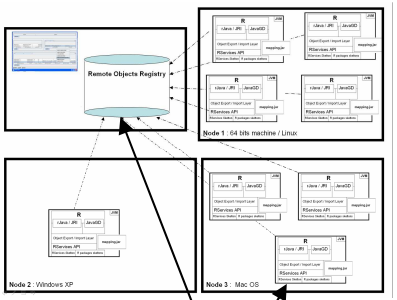
```
<rj>
  <publish>
    <functions> <function name="square" forWeb="true"/> </functions>
  </publish>
  <scripts>  <initScript name="globals.r" embed="true"/> </scripts>
</rj>
```

ant demogen

mapping.war

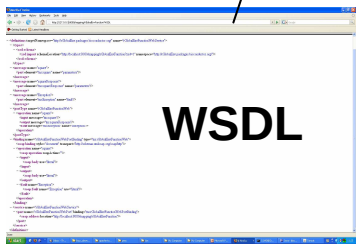
+ mapping.jar
+ pooling framework
+ R Java Bridge
+ **JAX-WS**
  - Servlets
  - Generated artifacts

Deploy



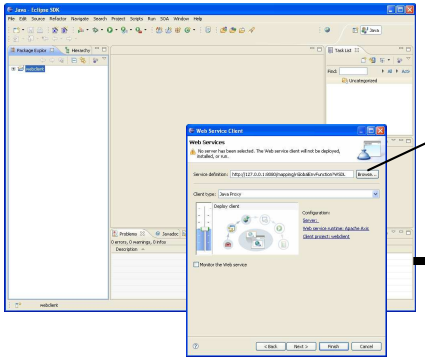
mapping.war

tomcat

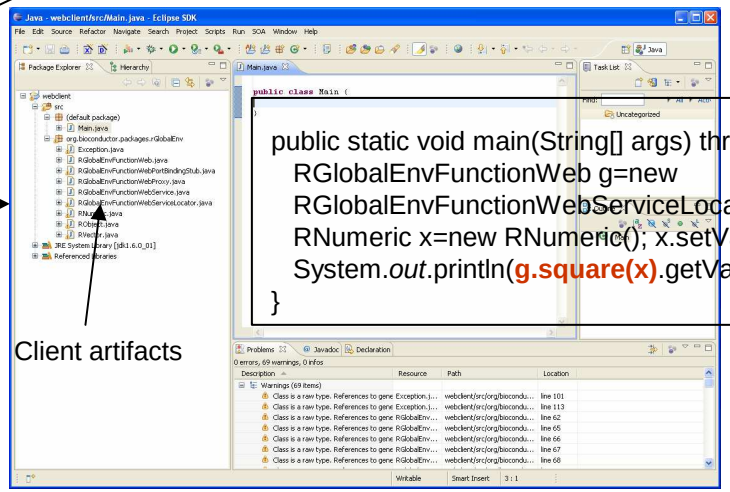


WSDL

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



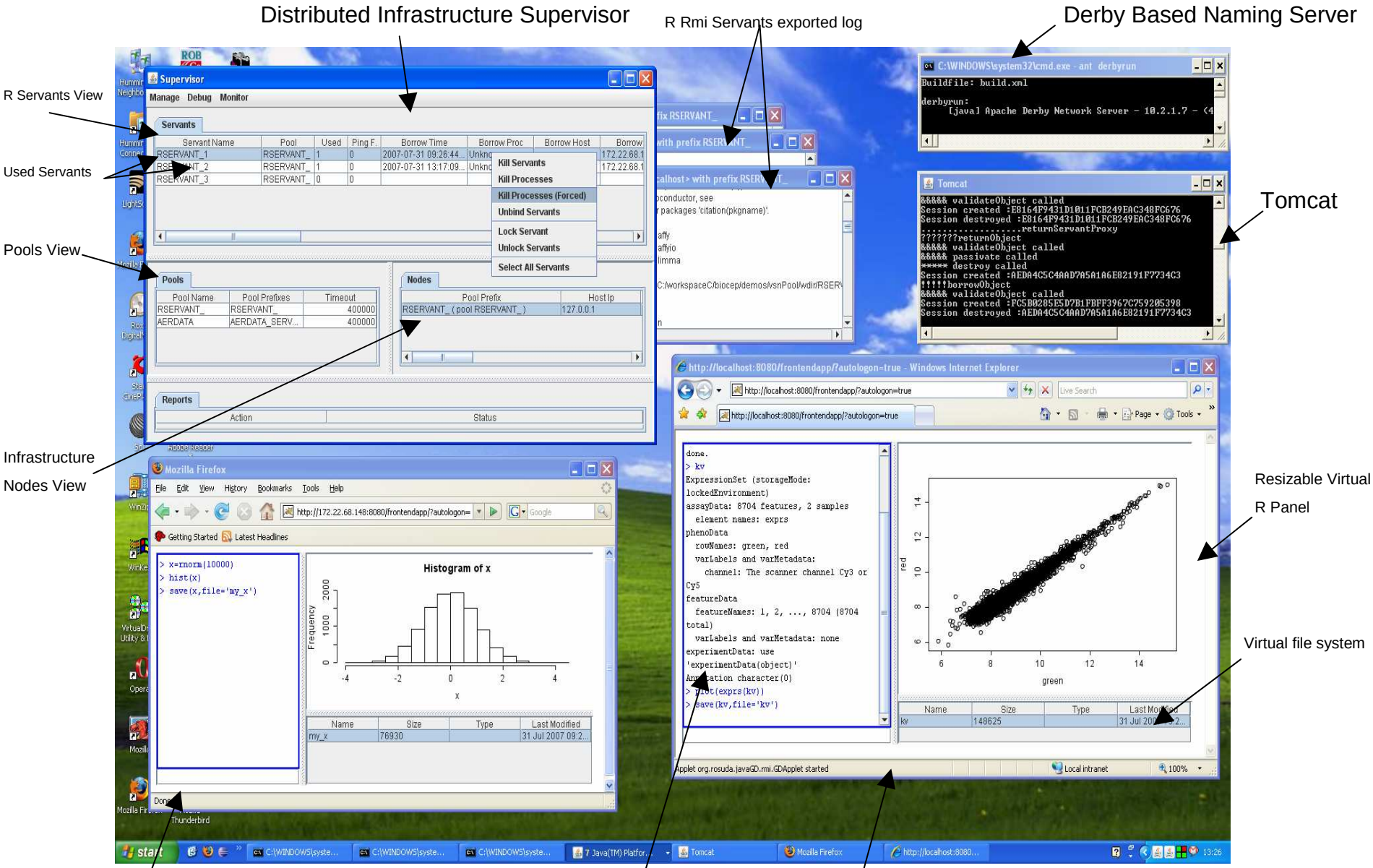
Eclipse Web Service Client Generator



Client artifacts

```
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]);
}
```

Virtualized R



Acknowlegments

Alvis Brazma

Martin Morgan

Seth Falcon

Simon Urbanek

R O.O. Programming in a nutshell

- S4 supports : objects / classes / inheritance / polymorhpism

Example

- setClass("Point", representation(x="numeric", y="numeric"))
- setClass("StyledPoint", contains="Point", representation(style="integer"))
- setGeneric("distance", **function**(p1,p2) standardGeneric("distance"))
- setMethod("distance", signature("Point", "Point") ,
  function(p1, p2) { sqrt((p2@x-p1@x)^2 + (p2@y-p1@y)^2) })
- O<-new ("Point", x=12,y=2); P<-new("Point", x=52, y=90)
- distance(O,P)

Introspection functions

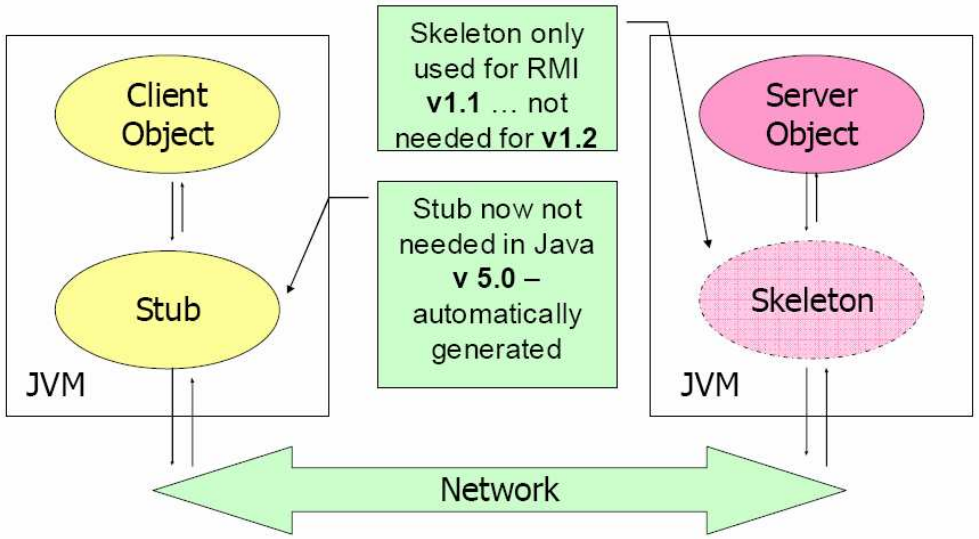
- class(P) / getClass("Point“) / getSlots(“Point“) / ..
- showMethods(class = "Point") / showMethods("distance") / isGeneric ("distance") / ..
- extends("StyledPoint") / ..

Unions

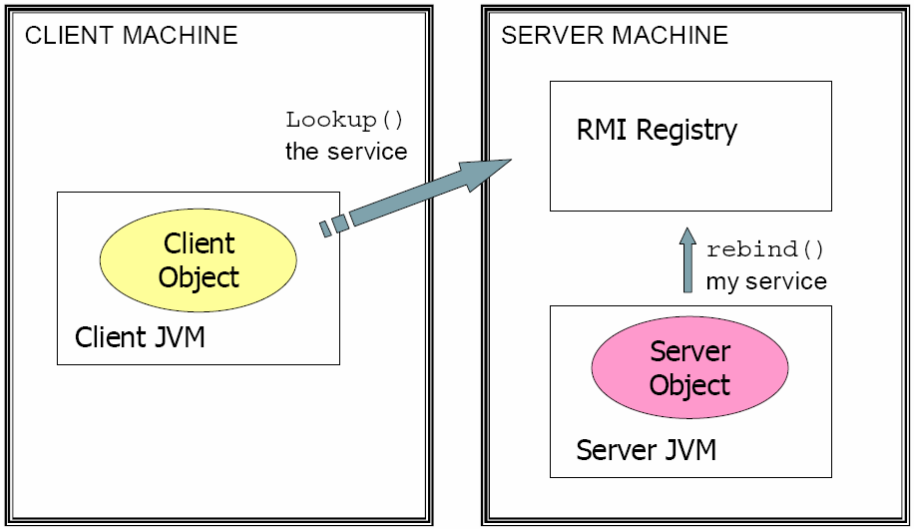
setClass("Egg", representation(color="integer")); setClassUnion("Thing", c("Point","Egg"));

RMI in a nutshell

How does RMI work ? Stubs and Skeletons



RMI Registry



Prerequisites

1. Checkout the **EBI-Biocep** project
   public svn URL : https://svn.ebi.ac.uk/ma/branches/biocep/
   login : ma-guest
   password : CN85JIZ5
   (README.txt contains the instructions to run the demos)

2. **JDK 5 or 6** [Java SE 6 is required for JAX-WS Web Services]
   - JAVA_HOME environment variable set to the installation root folder of the jdk
   - PATH environment variable must include the jdk bin directory

3. **ant >= 1.7.x** installed
   - ANT_HOME environment variable set to the installation root folder of ant
   - PATH environment variable must include the ant bin directory

4. **R >= 2.5.x** installed
   - R_HOME environment variable set to the installation root folder of R
   - Bioconductor packages installed (only vsn & its dependencies for the demos)
   - **TypeInfo** package installed
   - **rJava** package installed

   if you encounter problems installing rJava, check the following : rJava doesn't yet compile with jdk 6,

   if you are installing it from sources , check that R is using a jdk<=5 :

   on Unix like systems : grep JAVA $R_HOME/Makeconf → if jdk=6, set JAVA_HOME to a another jdk → "R CMD javareconf" → install rJava → restore the jdk 6 installation

   - **JavaGD** package installed

5. **apache-tomcat-5.5.x** installed

6. **apache derby 10.x** installed
   - DERBY_HOME environment variable set to the installation root folder of derby

7. **pslist** tool installed (only required if you are using windows)
   - PSLIST_HOME environment variable set to the location of pslist.exe

Useful links :

Java SE: http://java.sun.com/javase/downloads/index.jsp
ant: http://ant.apache.org/bindownload.cgi
R: http://cran.r-project.org/
BioConductor : source("http://bioconductor.org/biocLite.R");biocLite()
R packages : biocLite(c('TypeInfo', 'rJava', 'JavaGD'))
tomcat: http://tomcat.apache.org/download-55.cgi
derby: http://db.apache.org/derby/derby_downloads.html
pslist: http://www.microsoft.com/technet/sysinternals/utilities/pslist.mspx

Deployement of a Virtualized R infrastructure

cd biocep/demos/vsnPool

ant clean

ant compile

ant demogen

ant webcompile

Copy biocep/demos/vsnPool/frontenapp.war to %tomcat%/webapps

Open a a terminal → start Derby : **%derby%/bin/startNetworkServer**

Open a terminal → start Tomcat : **%tomcat%/bin/startup**

Open a terminal → cd biocep/demos/vsnPool → **ant demoserver**

Open a terminal → cd biocep/demos/vsnPool → **ant demoserver**

Open a terminal → cd biocep/demos/vsnPool → **ant demotop**

Open a browser → type the URL : **http://127.0.0.1/frontendapp/?height=400**

In the console part of the R applet

type **logon**

Create a normal distribution **x<-rnorm(1000)**

Plot it **plot(x)**

Draw the histogram of x **hist(x)**

Save x : **save(x,file='x_data')**

Right-click on the file and save it to your local disk

Use an editor to create an R script on your local disk : 'script.r'

Add any R commands to your script

Right click on the file system area of the R applet and copy the script you've created to that file system

Source your script : **source('script.r')**

Bring the supervisor window up front to have it simultaniously visible with the applet

Type **logoff**, notice the changes on the supervisor

Type **logon your_name**, notice the changes on the supervisor

type **data(kidney)**

type **jk<-justvsn(kidney)**

plot the expression of jk

right-click on the console log area and copy the Java Serialization of jk to your disk (jk.ser) [can be used directly by java code]

right-click on the console area and push the Java Serialization (jk.ser) to R, name the

target variable jk2

plot the expression of jk2 **plot(exprs(jk2))**

Project Home:

http://code.google.com/p/ebi-biocep/

Contacts:

Karim Chine

Email : kchine@ebi.ac.uk

Phone : + 44 (0) 1223 492 597

Misha Kapushesky

Email : ostolop@ebi.ac.uk

Phone : +44 (0)1223 494 647