



JAS

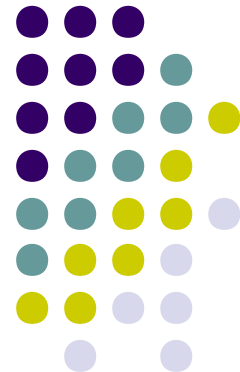
Java Agent-based Simulation library

Michele Sonnessa
(sonnessa@di.unito.it)

Department of Computer Science
University of Torino, Italy

April, 13-15 2003
SwarmFest, Notre Dame

<http://jaslibrary.sourceforge.net>





<http://jaslibrary.sourceforge.net>



What is JAS

- An open source collection of tools to build ABMs
 - Based on the Swarm paradigm
 - Written using Java and XML
 - Founded on 3rd party well-tested open-source libraries
- An application that loads, executes and controls simulation experiments
 - XML project files contain information like list of models to run, classpaths, seed number, windows layout, etc.



<http://jaslibrary.sourceforge.net>

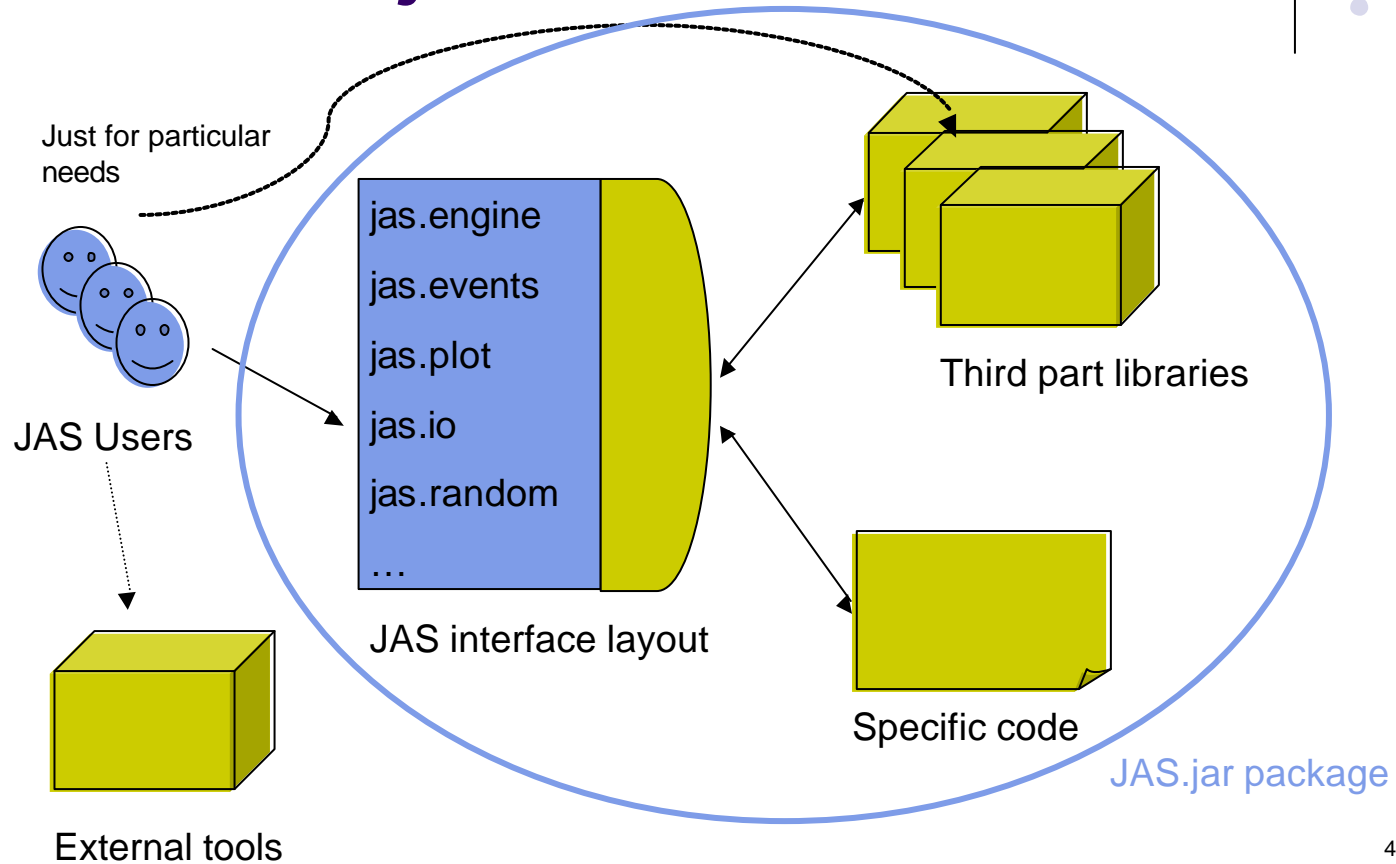


The principles of the project

- Screen technicalities within the package (using abstract template classes)
- “Do not reinvent the wheel”: importing ready to use packages ensures tool reliability (hopefully :-)
 - For each linked package it is important to create an interface layer reducing original complexity of the tool and integrating it with other JAS’ features



The JAS.jar architecture





<http://jaslibrary.sourceforge.net>



Libraries JAS is based on

Library	Author	Use
<i>ptPlot</i>	<i>Berkley University</i>	Plotting
http://ptolemy.eecs.berkeley.edu/java/ptplot/index.htm		
<i>COLT</i>	<i>CERN</i>	Random generation and statistics
http://nicewww.cern.ch/~hosc hek/colt/index.htm		
<i>jExcelApi</i>	<i>Andy Khan</i>	Microsoft Excel format I/O
http://www.andykhan.com/jexcelapi/index.html		
<i>SVG-Batik</i>	<i>Apache</i>	SVG image generation
http://xml.apache.org/batik/index.html		
<i>XML-RPC</i>	<i>Apache</i>	Sim2Web's remote calls
http://xml.apache.org/xmlrpc		



<http://jaslibrary.sourceforge.net>



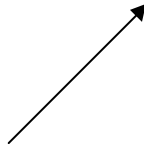
The JAS application

JAS models are stand alone applications based on the JAS library but they can be managed as they were documents thanks to the XML project document.

JAS is able to load models and drop them from memory without shutting down the JVM

How to create a model

1. Type code
2. Compile it
3. Generate XML project



The HeatBugs project file:

```
<?xml version="1.0" encoding="UTF-8" ?>
<JAS projectName="ObsHeatbugs">
  <ProjectParameters>
    <TimeUnit>7</TimeUnit>
    <MajorVersion>0</MajorVersion>
    <Seed randomSeed="true">1020968840339</Seed>
    <ProjectDescription>Model with observer
      example.</ProjectDescription>
  </ProjectParameters>
  <Model className="HeatBugsModel">
    <Window title="Heatbugs">5,130,400,400</Window>
  </Model>
  <Model className="HeatBugsObserver">
    <Window title="Unhappiness">
      504,380,500,300</Window>
    <Window title="Space viewer">
      626,23,320,320</Window>
  </Model>
  <ClassPath>
    <Path>.\examples\HeatBugs</Path>
  </ClassPath>
</JAS>
```



<http://jaslibrary.sourceforge.net>



JAS' highlights

- Real time execution mode
- Sim2Web: a Jas-Zope bridge architecture to publish simulations on the web
- *jas.engine.AgentList* allows asynchronous method execution
- GA, ANN, CS native packages (under construction)
- A multi-run template class for automatic parameter calibration
- Some of the turtles' instructions from Starlogo
- Desktop GUI application mode



<http://jaslibrary.sourceforge.net>



JAS' highlights: MultiRun

- `jas.engine.MultiRun`
 - public void **go()**
 - public abstract [ISimModel](#)[] **startModel()**
 - public abstract boolean **nextModel()**
- An example by Matteo Richiardi
 - FirmDynamics



<http://jaslibrary.sourceforge.net>



JAS' highlights: Turtle

- `jas.space.Turtle`

Instructions from StarLogo

- `public void forward(int steps)`
- `public boolean leap(int steps)`
- `public void setXY(int x, int y)`
- `public int getNextX(int steps)`
- `public void setHeading(int heading)`
- `public void turnCardinalLeft(int steps)`
- `public void turnRight(int degrees)`
- ...

- An example: Termites

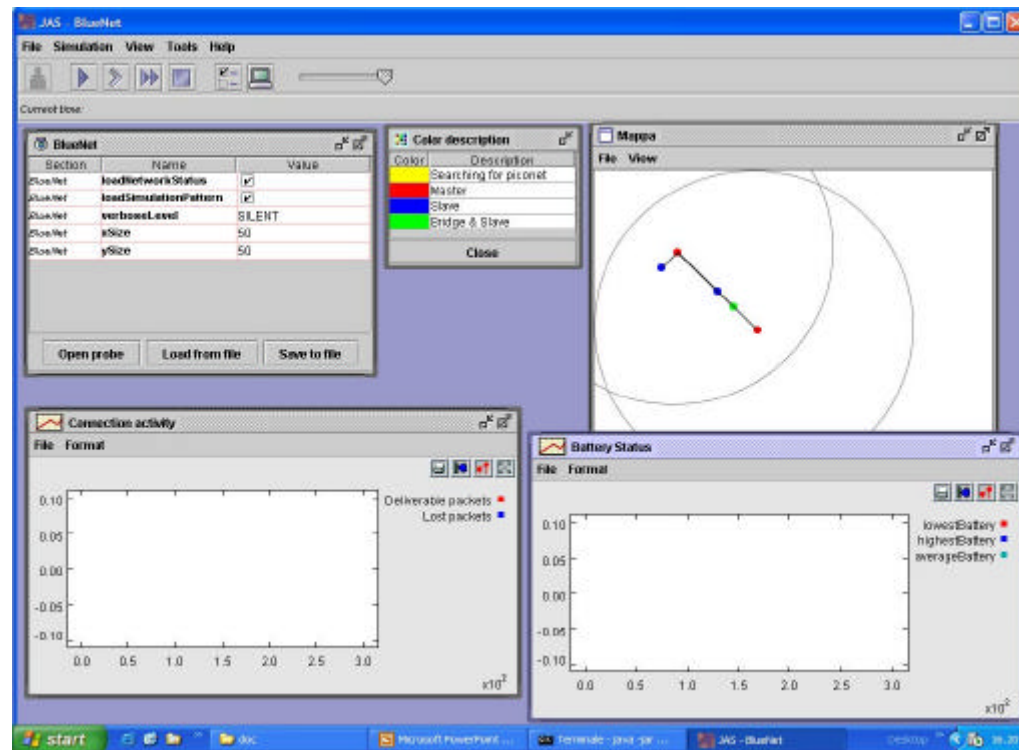


<http://jaslibrary.sourceforge.net>



JAS' highlights: Desktop mode

- > java -jar JAS.jar -d





<http://jaslibrary.sourceforge.net>



The current JAS' structure

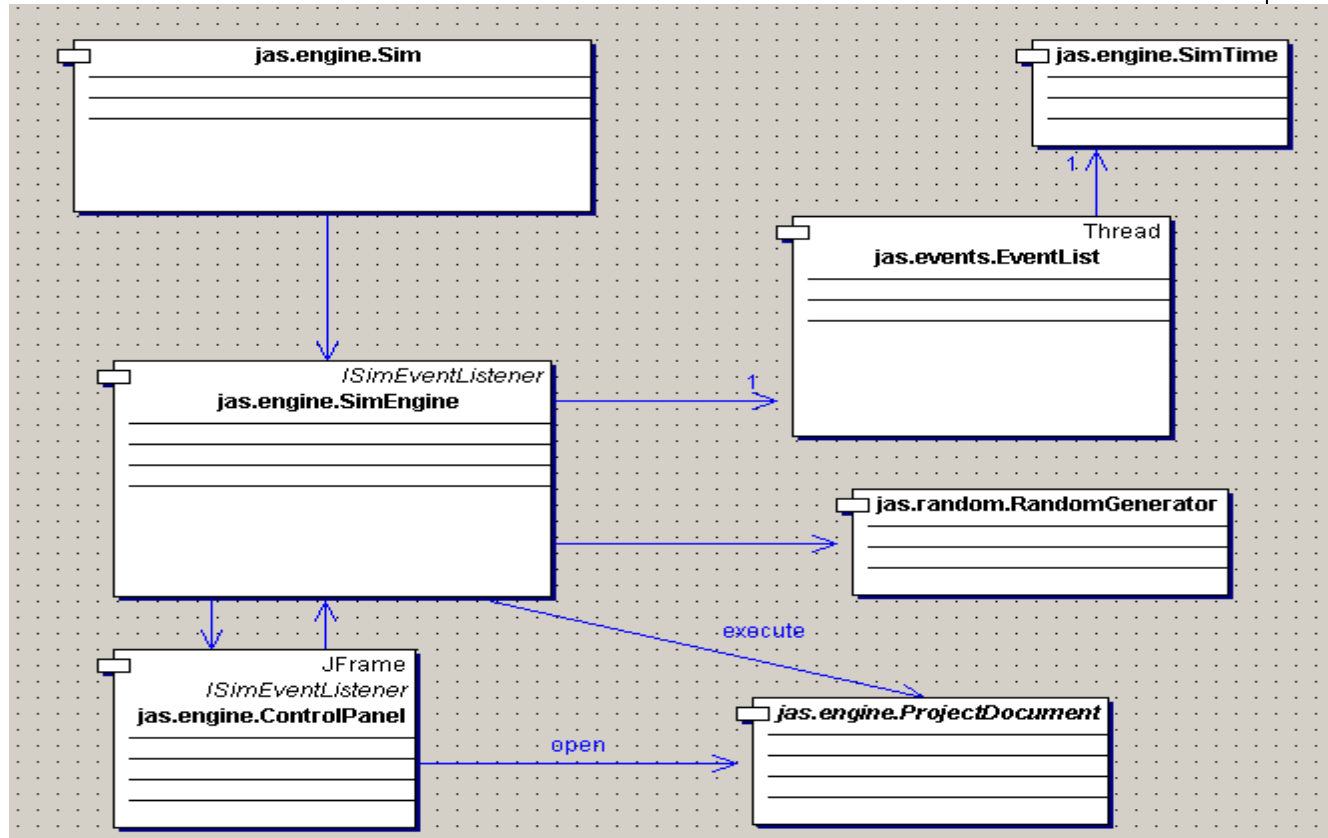
<i>jas.engine</i>	Simulation engine, time manager, GUI layer.
<i>jas.events</i>	The event architecture
<i>jas.io</i>	General purpose I/O classes supporting CSV, Excel and XML formats
<i>jas.net</i>	Network tools (Sim2Web, remote controlling, ...)
<i>jas.plot</i>	The plotting tools (mostly based on <i>ptPlot</i> library)
<i>jas.probe</i>	The probe library (similar to the Swarm's probes)
<i>jas.random</i>	A rich pseudo-random generation library (mostly based on the COLT library)
<i>jas.space</i>	Bi-dimensional grids, cellular automata, etc.
<i>jas.stats</i>	Statistical probes



<http://jaslibrary.sourceforge.net>

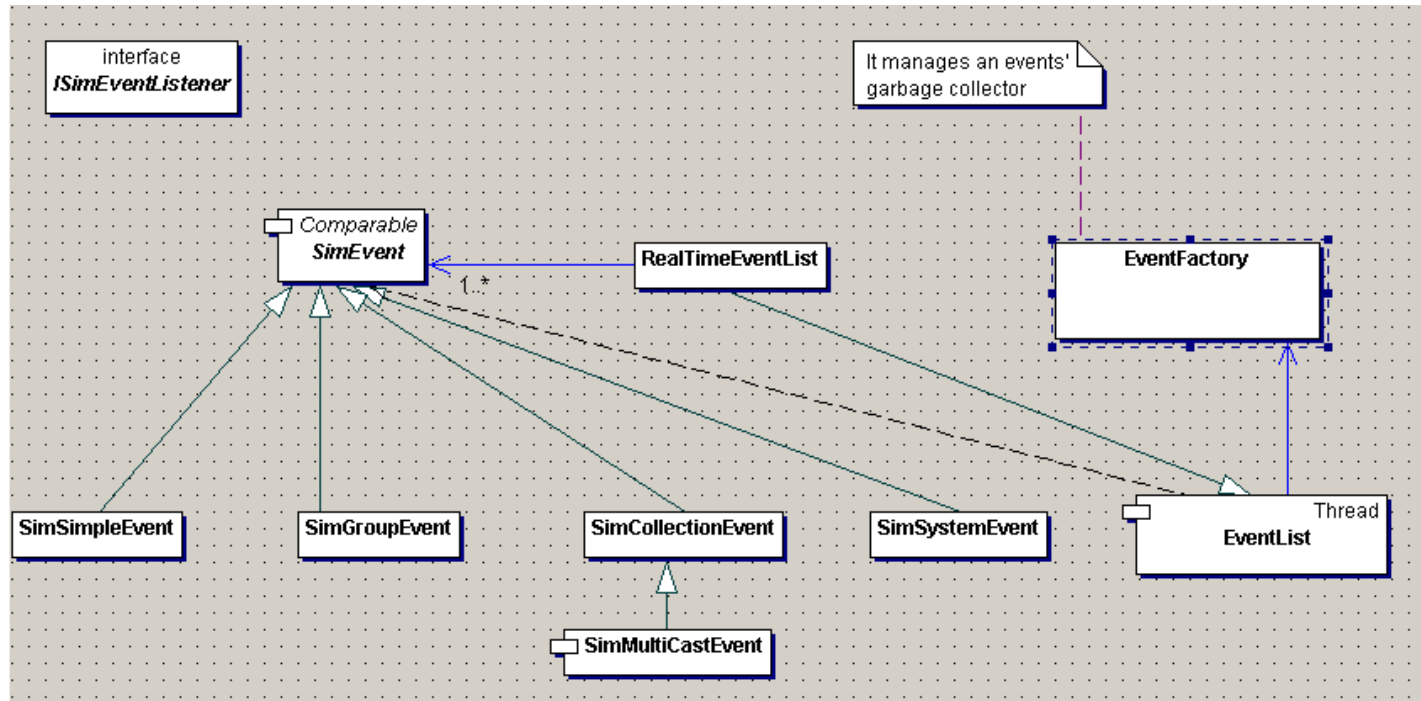


The core package layout





The jas.events package

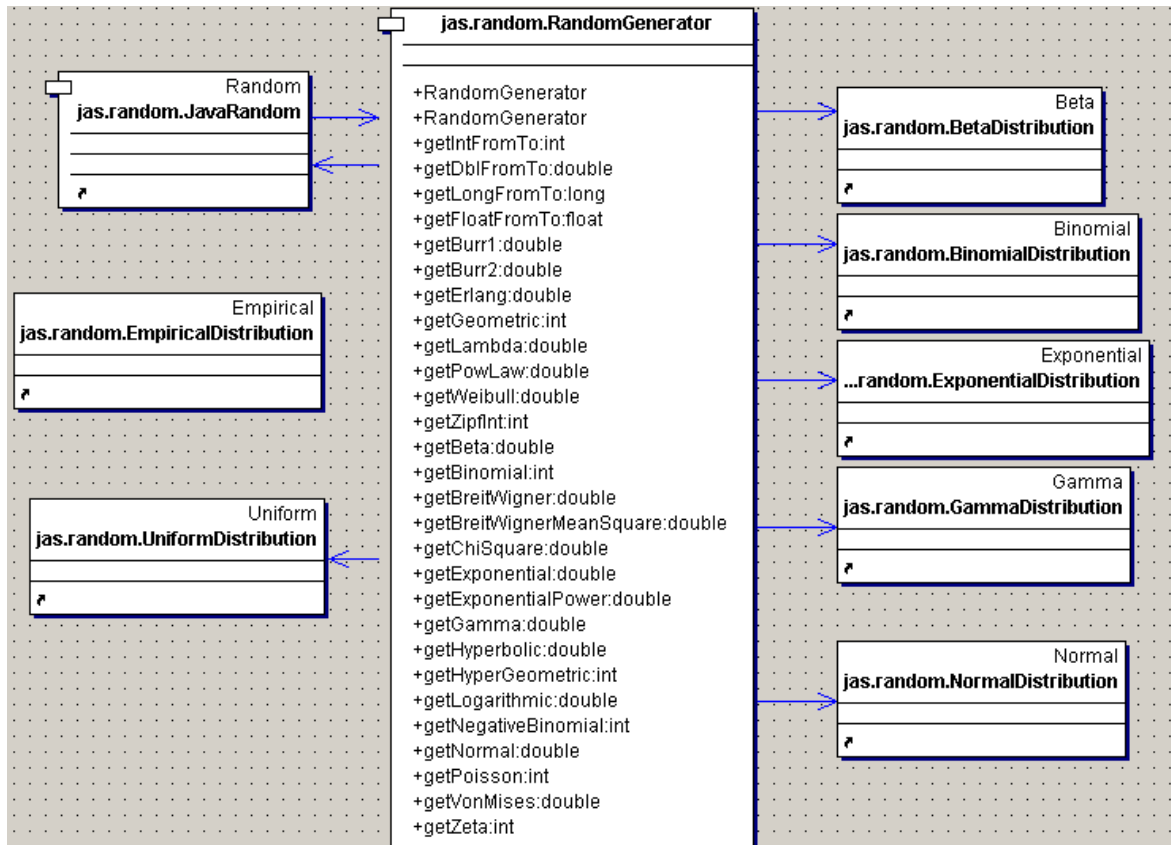




<http://jaslibrary.sourceforge.net>

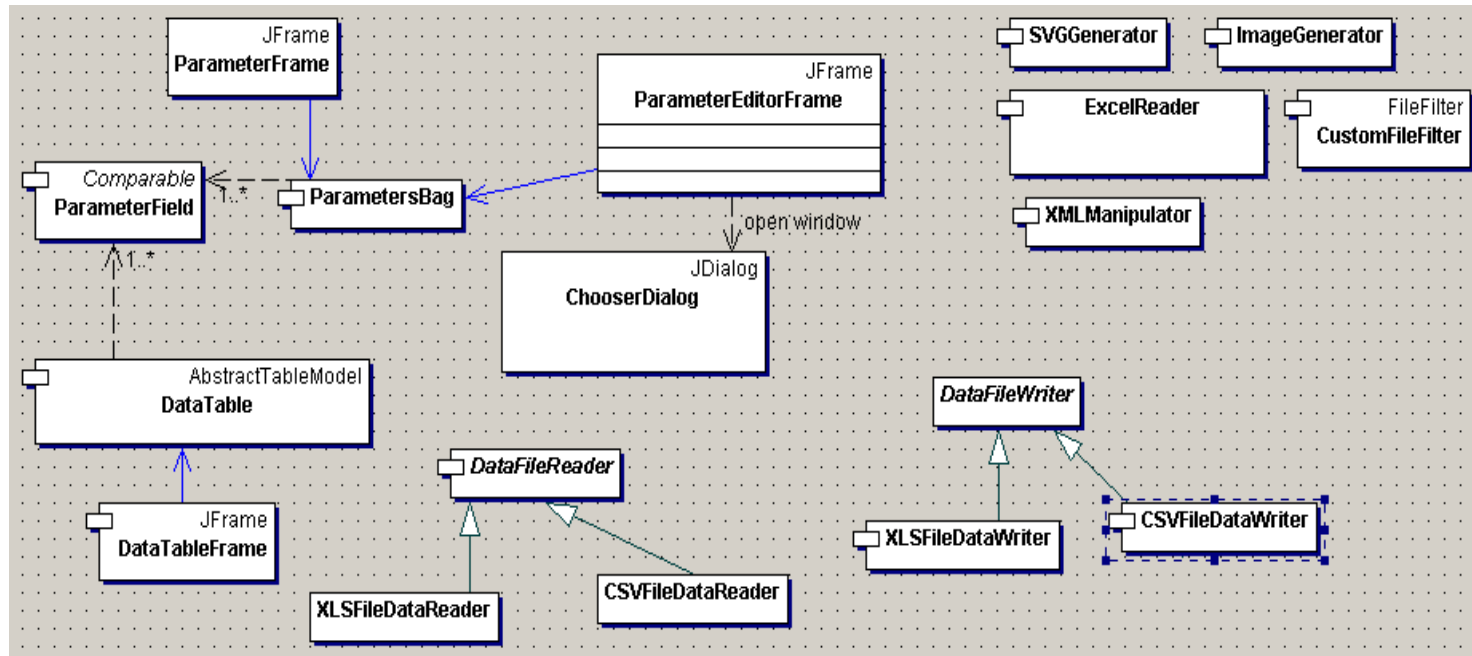


The jas.random package





The jas.io package

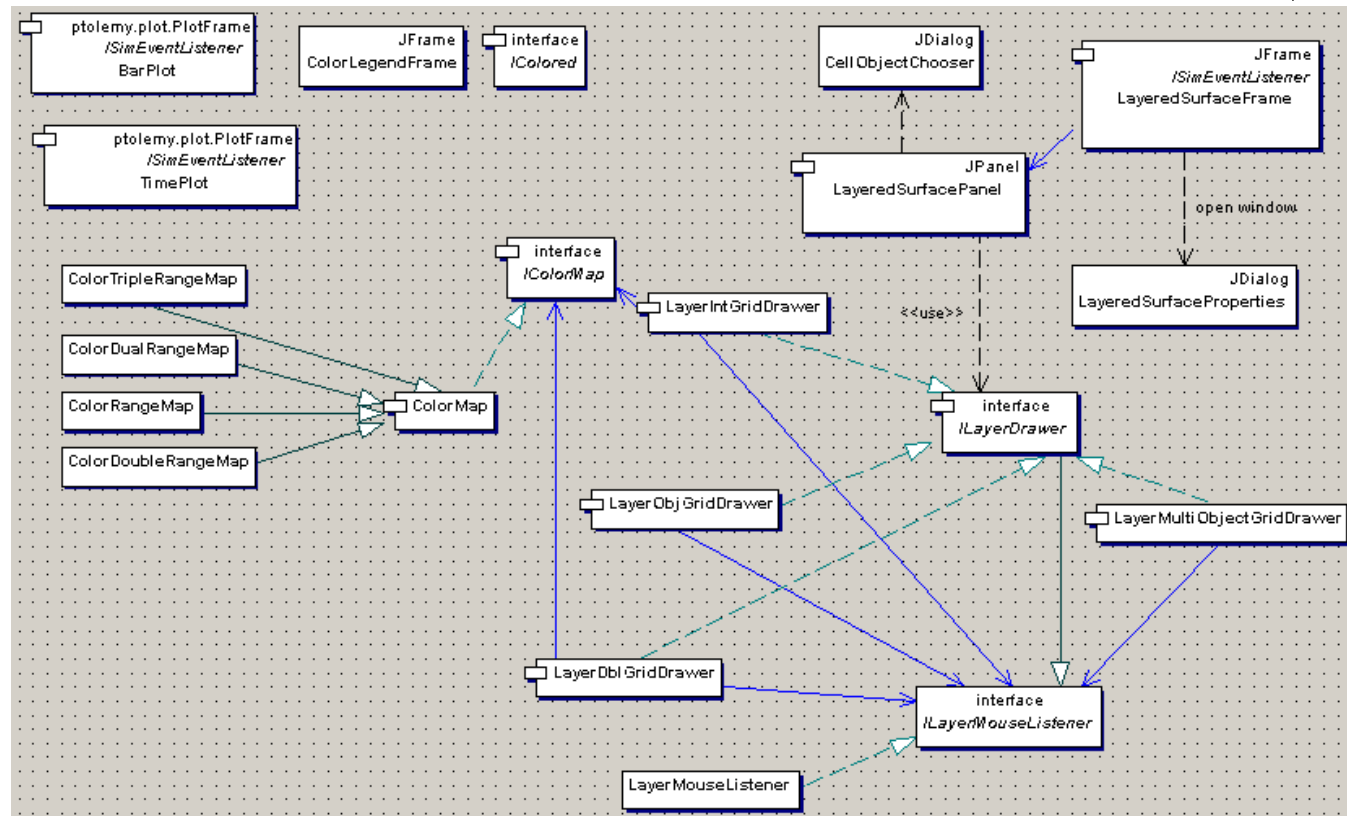




<http://jaslibrary.sourceforge.net>

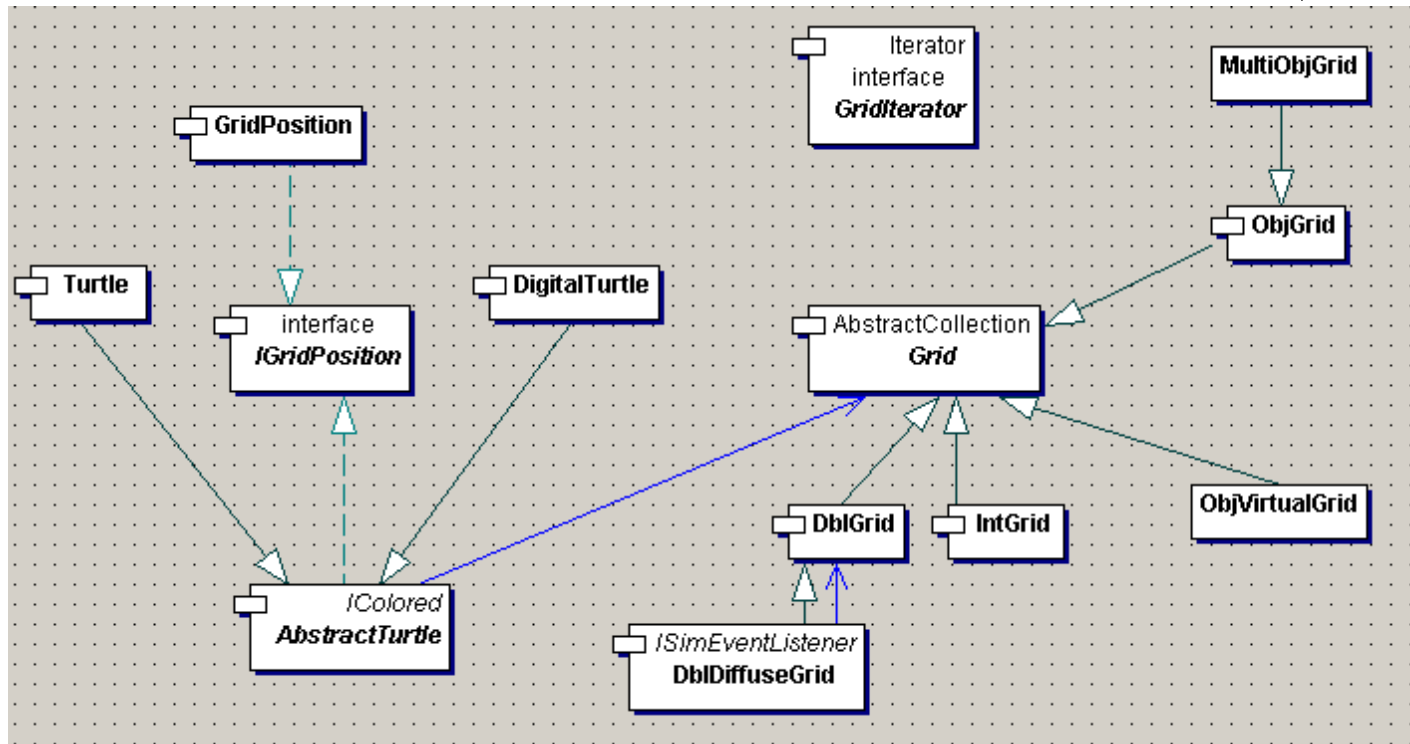


The jas.plot package





The jas.space package

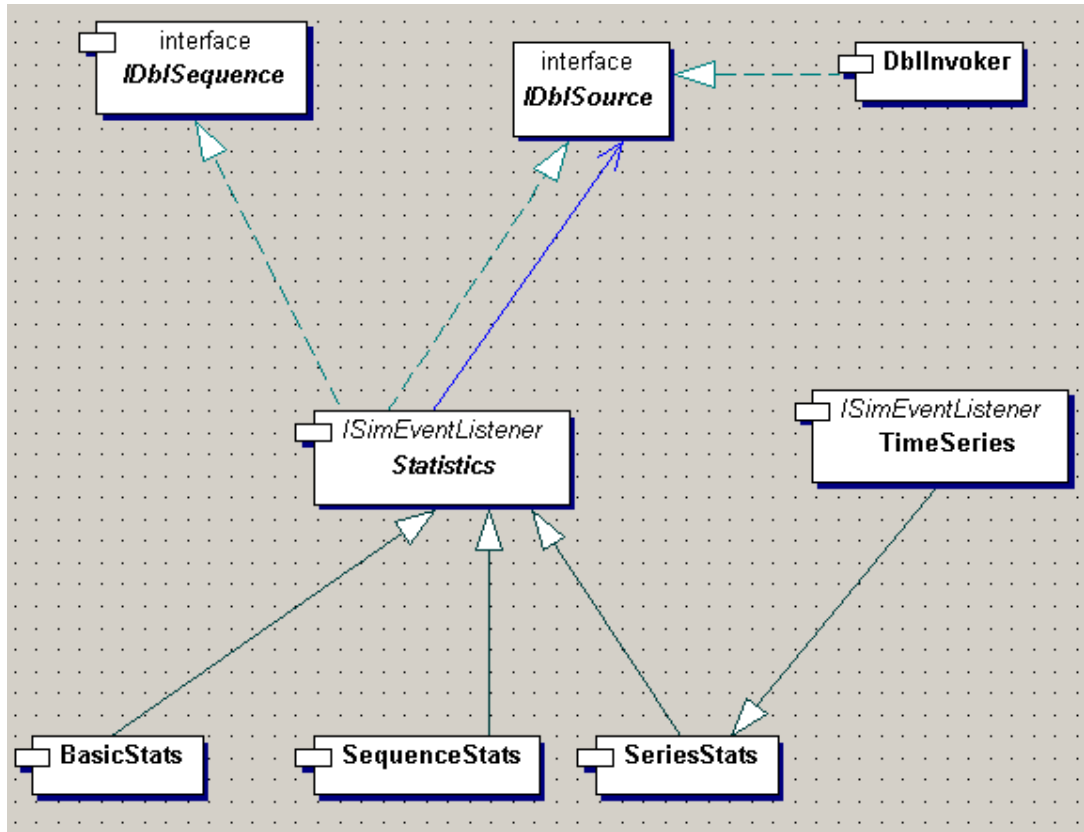




<http://jaslibrary.sourceforge.net>



The jas.stats package





<http://jaslibrary.sourceforge.net>



JAS & Swarm

SWARM	JAS
swarm.activity ScheduleImpl ActionGroup	jas.events EventList SimGroup
swarm.analysis	jas.stats + jas.plot
swarm.collections	(Java standard collections)
swarm.defobj	jas.io
swarm.gui	jas.plot
swarm.objectbase	jas.probe
swarm.simtools	-
swarm.random	jas.random
swarm.simtoolsgui	jas.probe
swarm.space	jas.space



<http://jaslibrary.sourceforge.net>



Limits

- Too few users: little debugged
- Incomplete documentation
- Some features are still missing (XML datatable I/O, histogram plotter, data panels, ...)



<http://jaslibrary.sourceforge.net>



Future developments

- Tutorial, examples and library reference
- *jas.ai* package will be released soon
- Graph/GIS tools integration
- A queue systems package to integrate ABM with *process simulation* (something is already present in *jas.de*)
- What do you need?



<http://jaslibrary.sourceforge.net>

Michele Sonnessa

(sonnessa@di.unito.it)

(sonnessa@users.sourceforge.net)

