



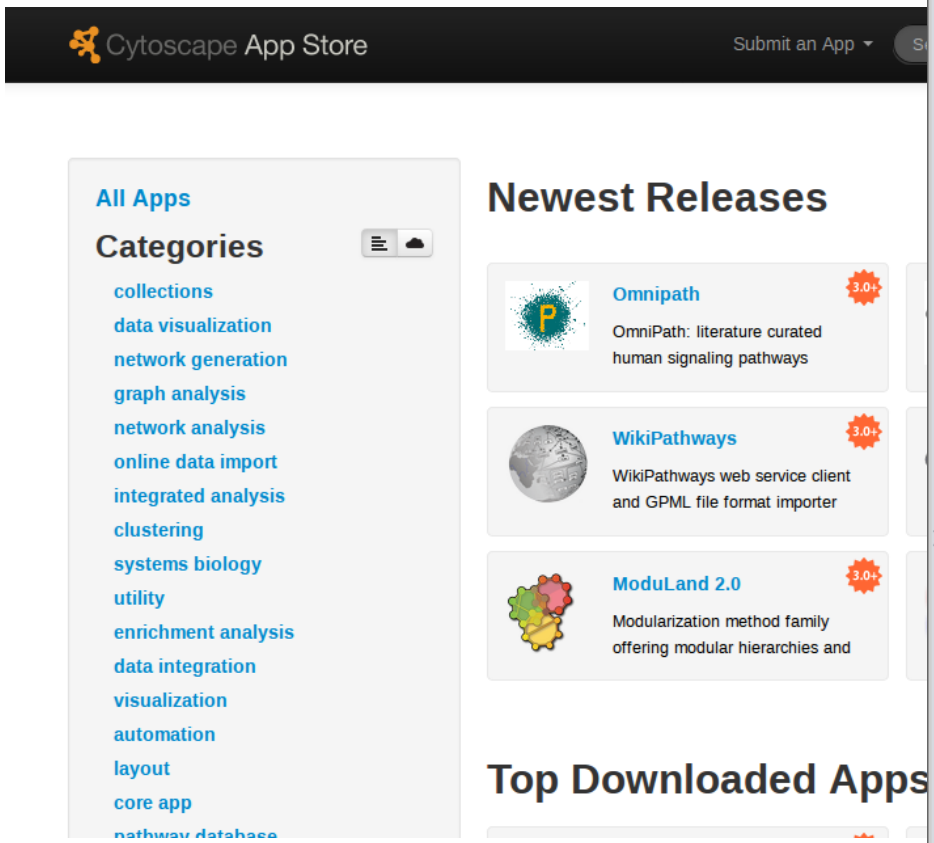
# Developing apps for Cytoscape

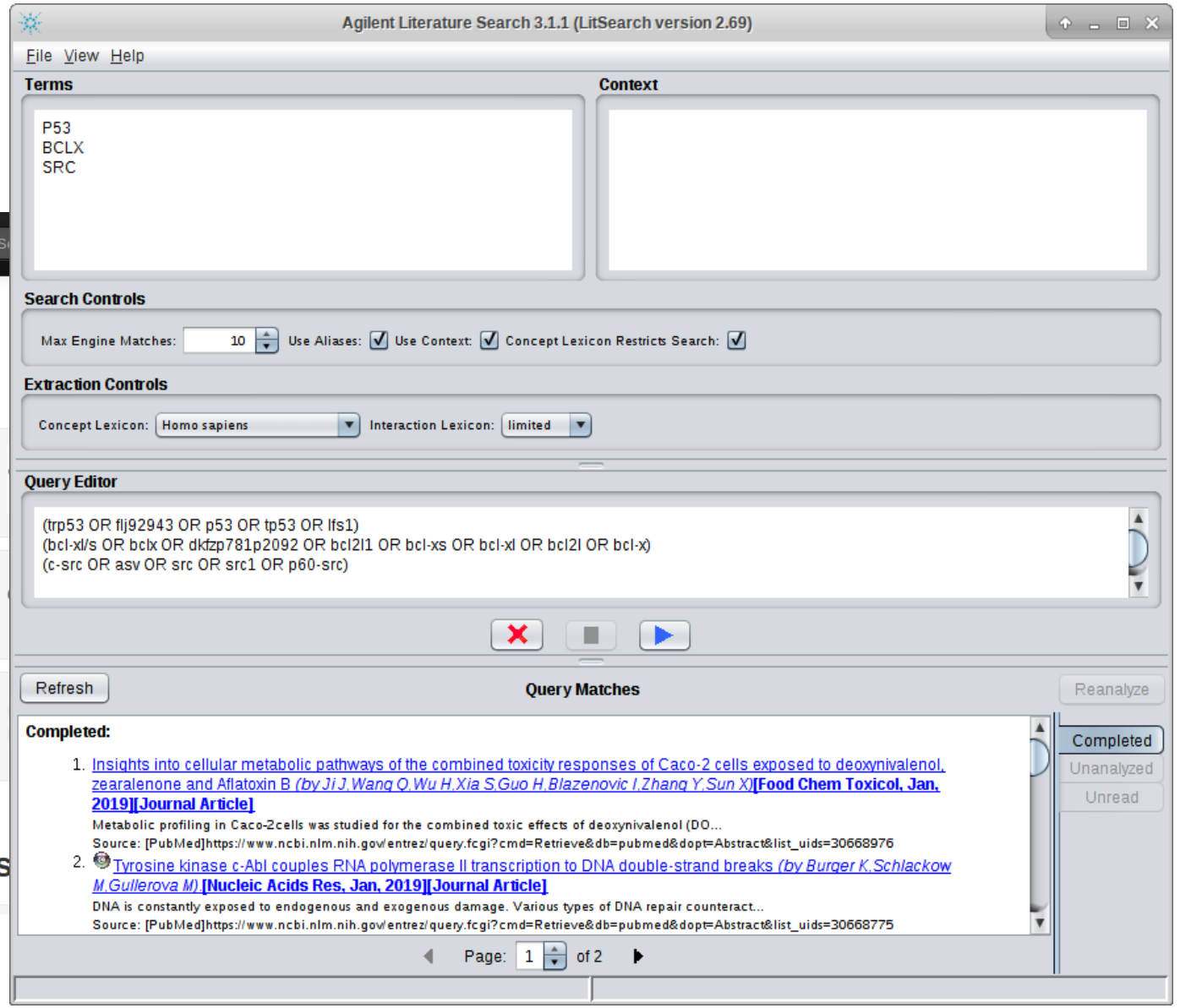
**Dr Tomasz Kurowski**
**t.j.kurowski@cranfield.ac.uk**

**14 February 2025**

www.cranfield.ac.uk

# Apps/plugins



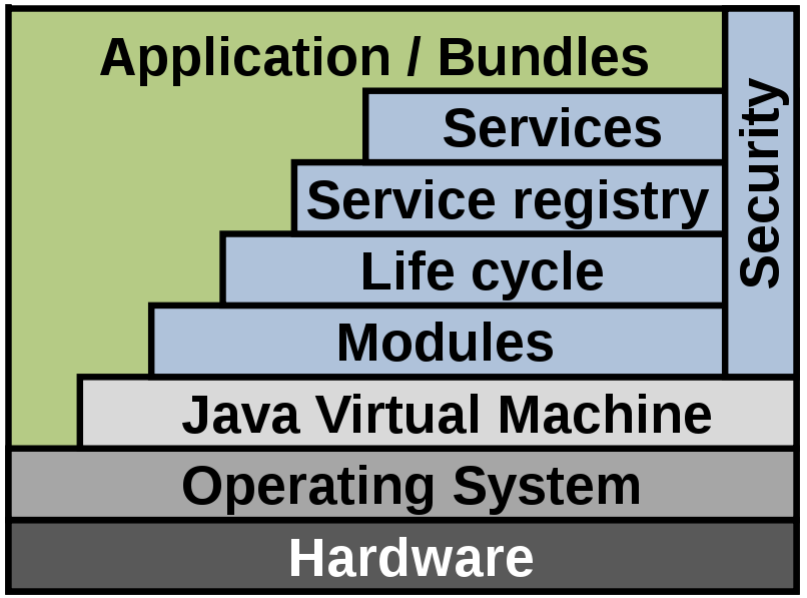



# OSGi

Open Services Gateway initiative.

Dynamic module system / service platform for Java applications. Bundles can be deployed, installed, started, stopped, modified, removed all without needing to restart.

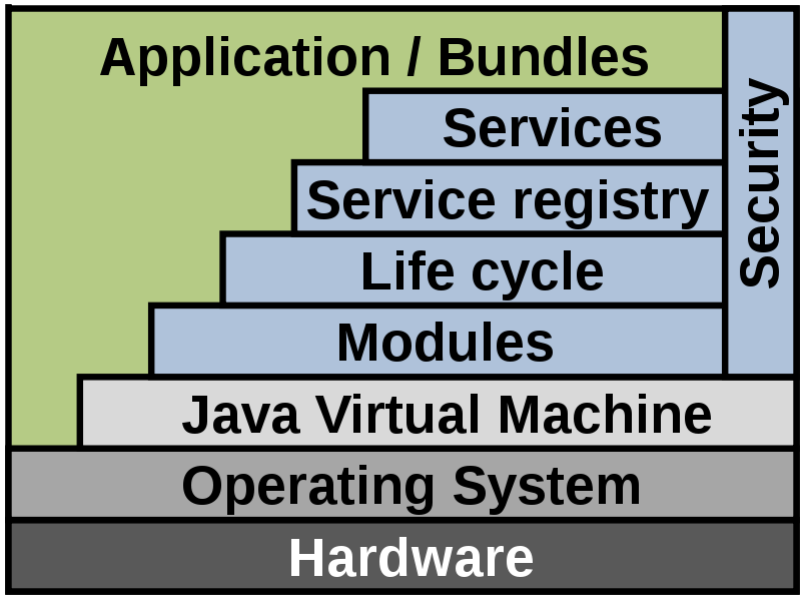
OSGi “bundles” are normal JAR files with extra manifest headers.



# Activator

Instead of a "main class", the entry point for an OSGi bundle is an "Activator class", which implements BundleActivator:

```
public class Activator implements BundleActivator {
    public void start(BundleContext context) { ... }
    public void stop(BundleContext context) {... }
}
```

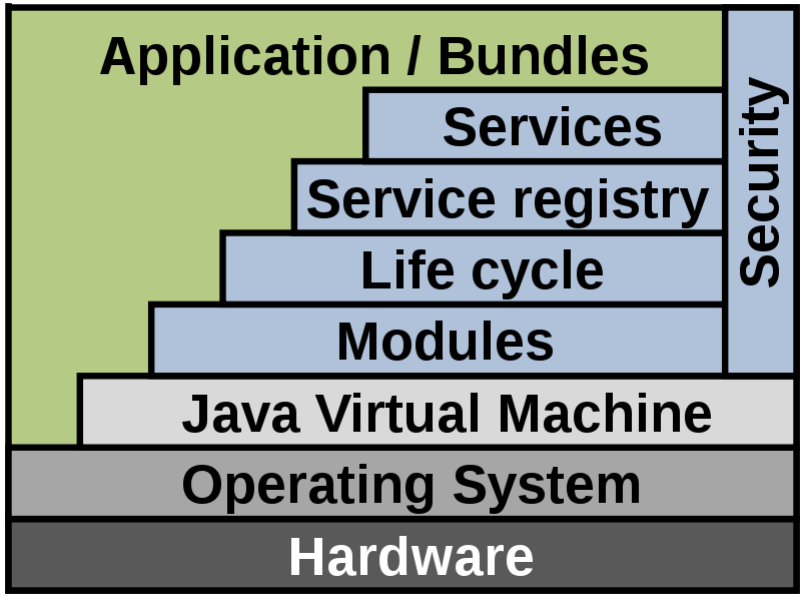


# CyActivator



For Cytoscape Apps, the entry point is a CyActivator:

```
public class CyActivator extends AbstractCyActivator {
    public CyActivator() {
        super();
    }
    @Override
    public void start(BundleContext bc) {
        // TODO
    }
}
```



# Services and Actions

```
public class CyActivator extends AbstractCyActivator {
    public void start(BundleContext context) {
        Service s = getService(context, Service.class);
        Action someAction = new Action(s);
        registerService(context, someAction, . . .);
     }
}
```

```
public class Action extends AbstractCyAction {
       Action(Service s) {
              // store Service S
       }
       public void actionPerformed(ActionEvent e) {
              // Do something with Service s
        }
}
```

# CyNetworkView, CyNetwork

Our two most important classes.

- `CyNetwork` contains all the information on the loaded Cytoscape network;
- `CyNetworkView` contains all the graphical information on the network.

```
CyApplicationManager cyAppManager = getService(bc,
CyApplicationManager.class);

CyNetworkView netView = cyAppManager.getCurrentNetworkView();
CyNetwork net = netView.getModel();
```

# Important classes

**CyNetwork**

e.g. `CyNetwork.getNodeList()`
`CyNetwork.getRow(someNode)`
`CyNetwork.getNeighborList(someNode, CyEdge.Type.ANY)`

**CyNode**

**CyEdge**

e.g. `CyEdge.getSource()`
`CyEdge.getTarget()`

**CyRow, CyTable**

e.g. `CyRow.set("selected", true)`

**CyTableUtil**

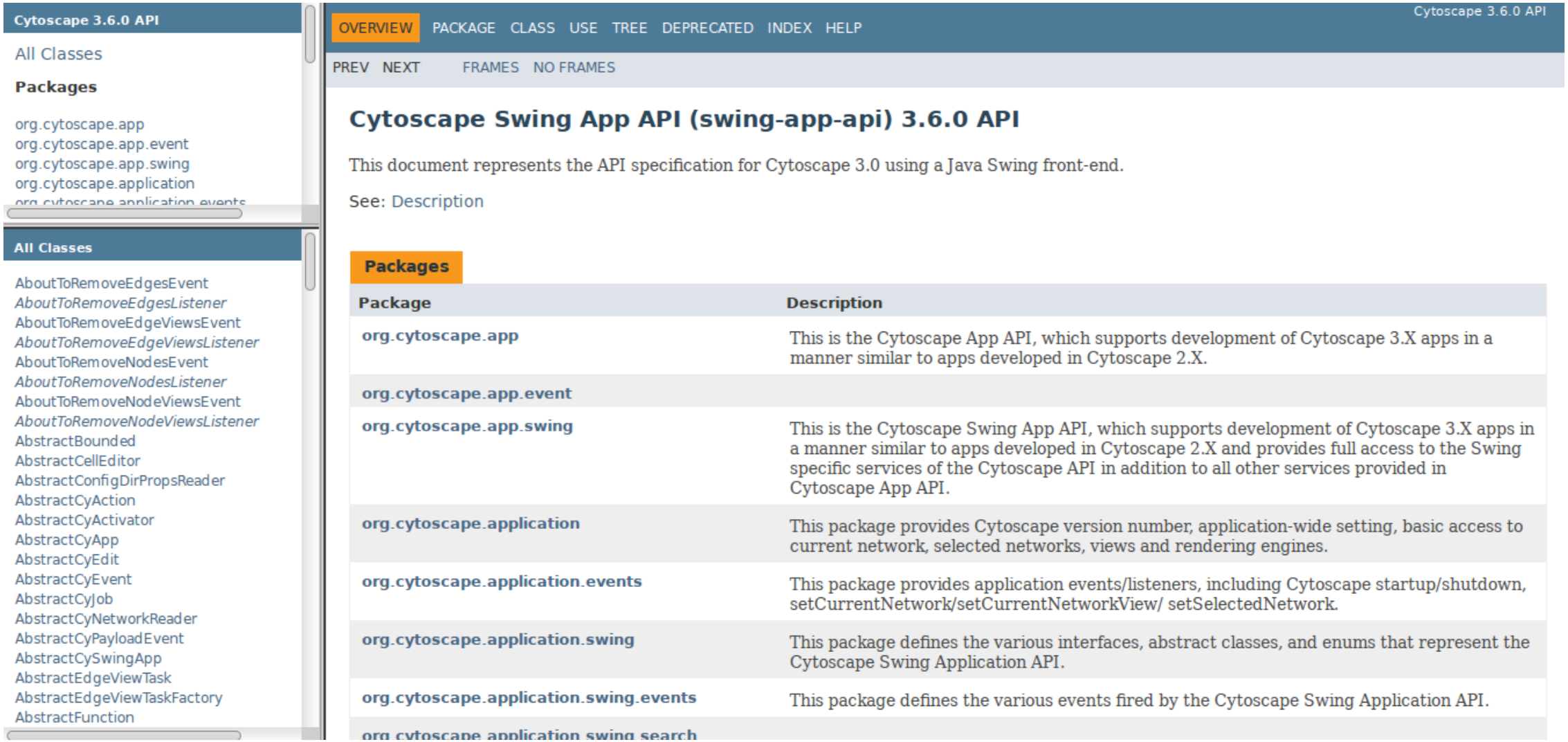
e.g. `CyTableUtil.getNodesInState(net, "selected", true)`

**CyNetworkView**

e.g. `CyNetworkView.updateView()`

# Cytoscape API

http://chianti.ucsd.edu/cytoscape-3.6.0/API/
https://cytoscape.org/javadoc/current_release/

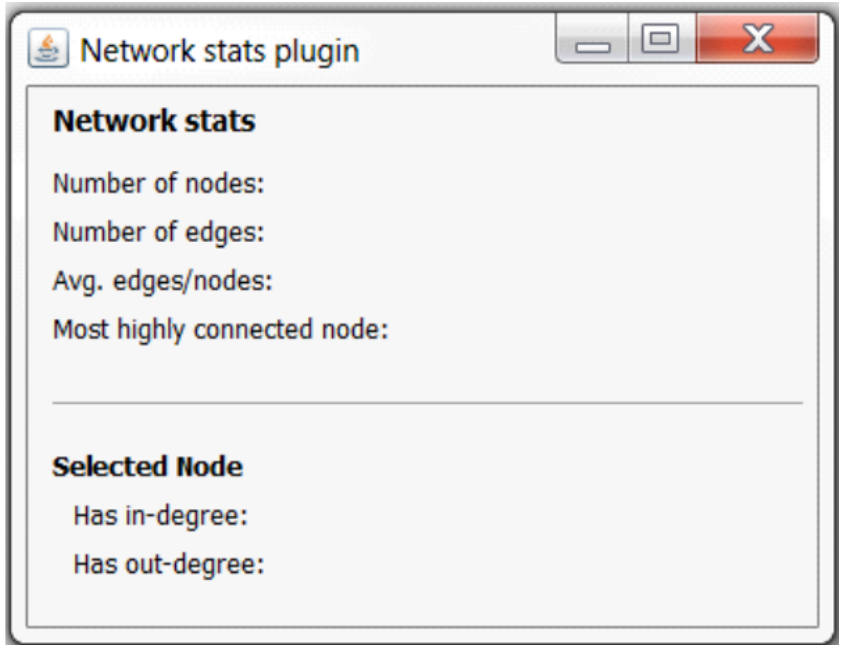


# Our application

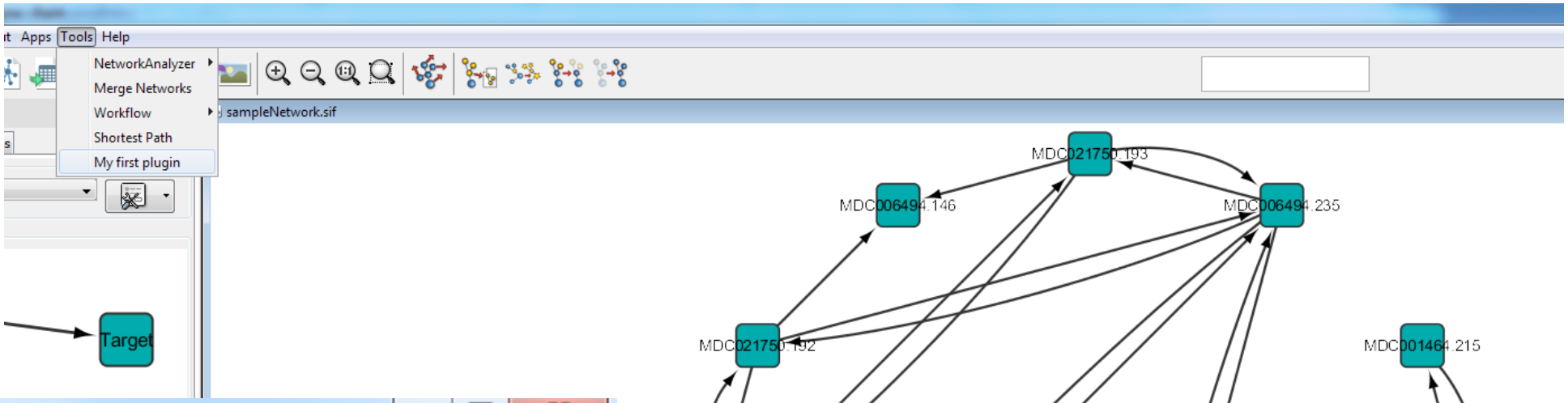
Our plugin will compute and display some simple statistics for the displayed graph and for a selected node.

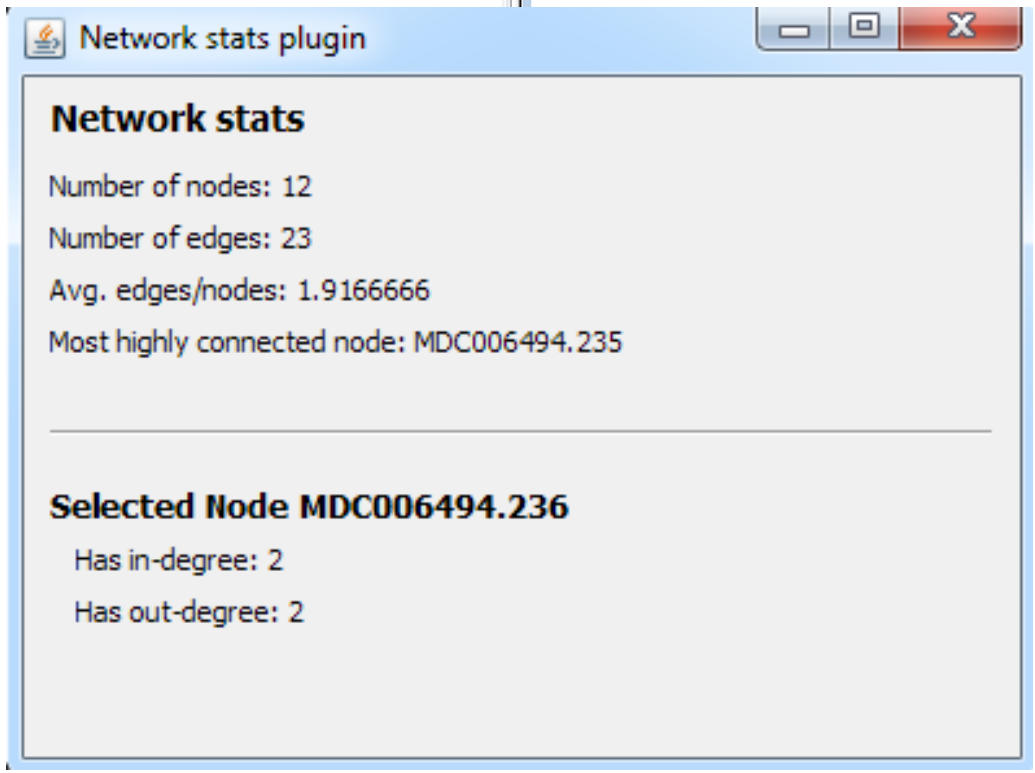
## What kind of stats?

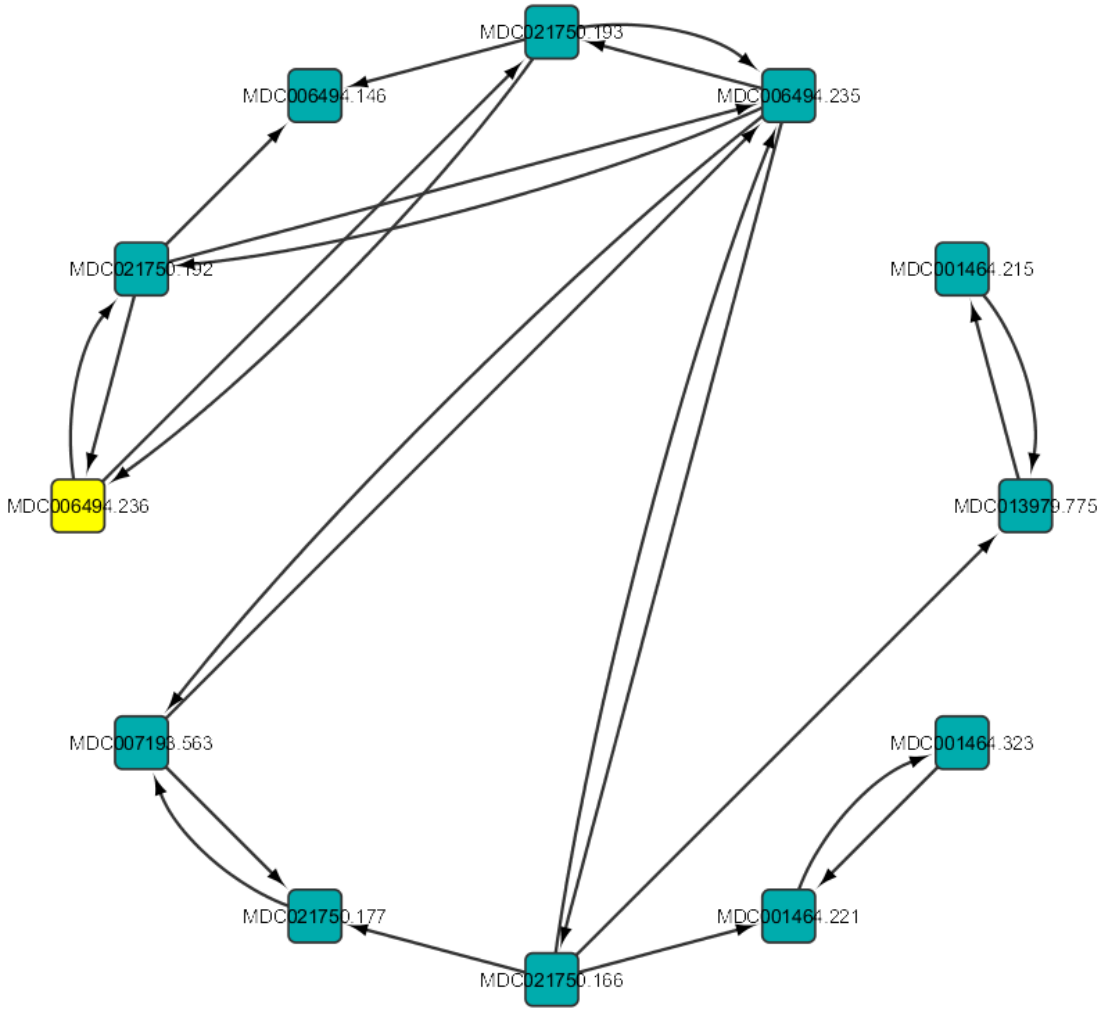
- Number of nodes
- Number of edges
- Average edges/node
- Most highly connected node
- In-degree of a selected node
- Out-degree of a selected node



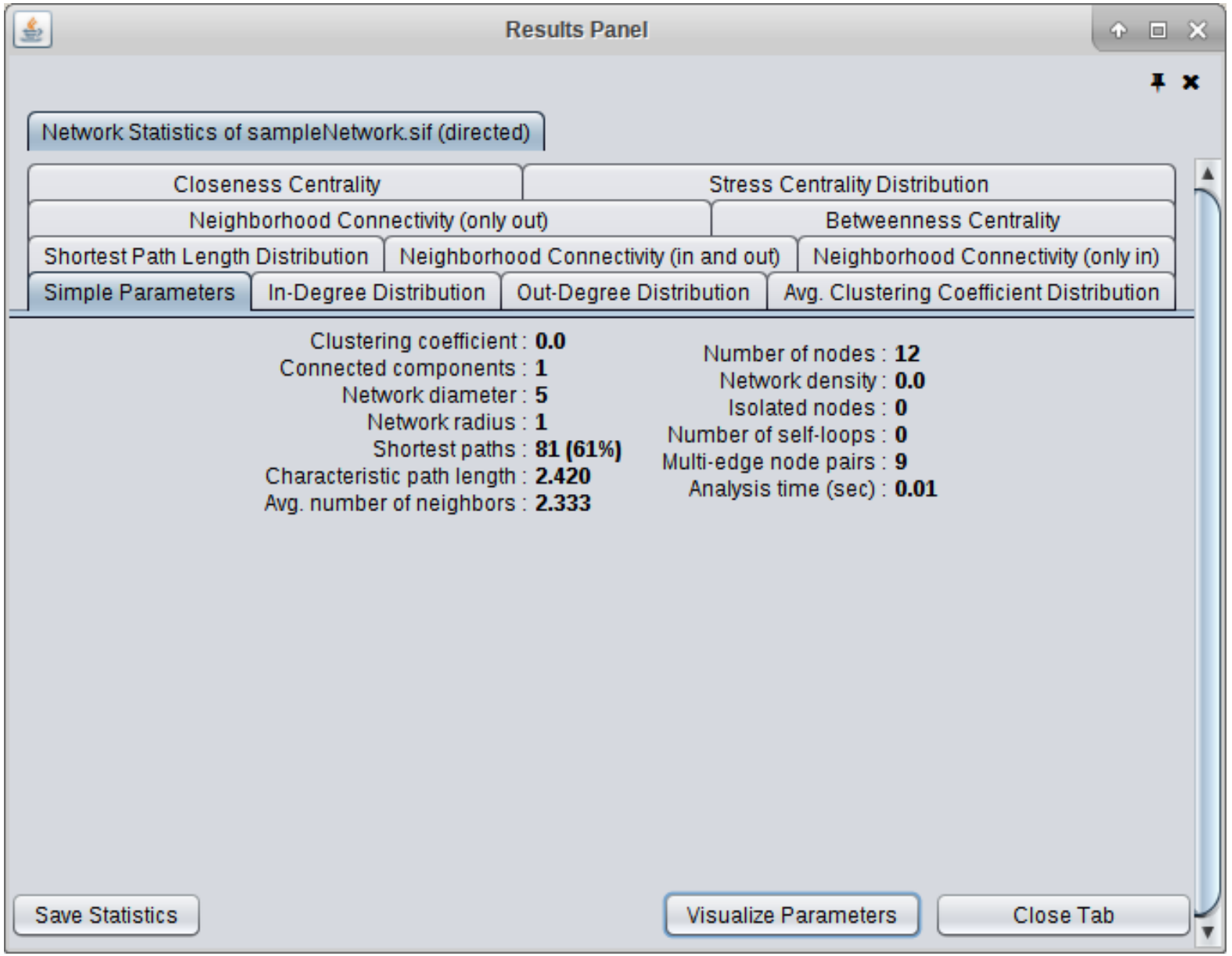
# We can use sampleNetwork.sif from Canvas for testing.



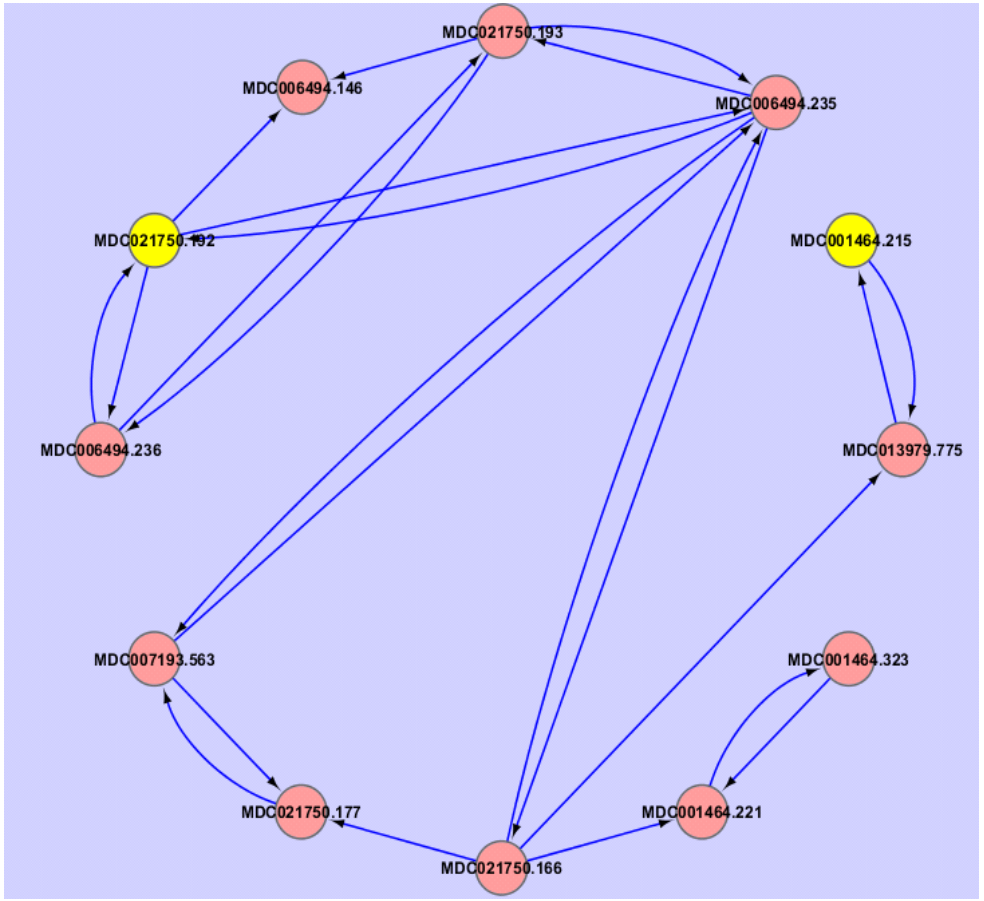



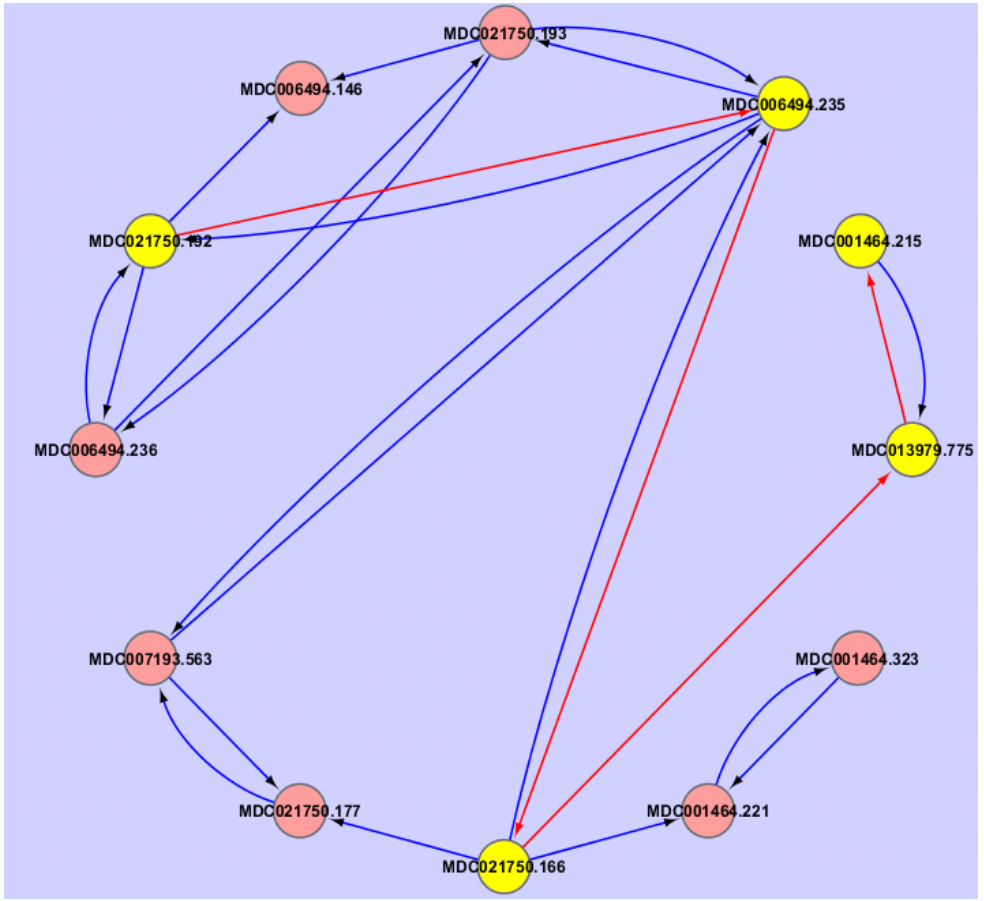



# Network Analyzer



# Extra task – shortest path between two nodes (BFS)



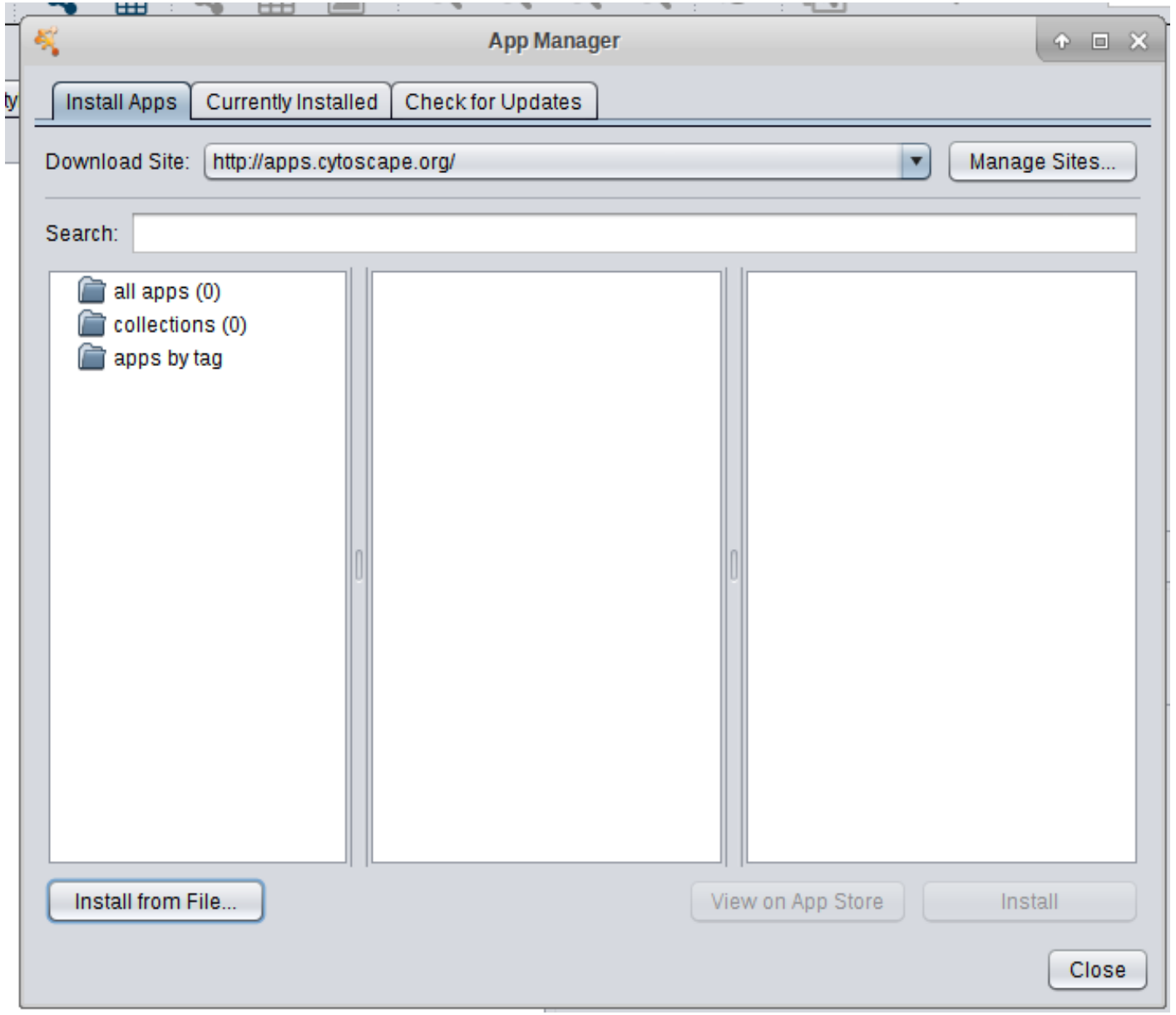



# Installing an app

Once you compile our OSGi bundle (app), the JAR will be in the **target/** directory (NOT dist/).

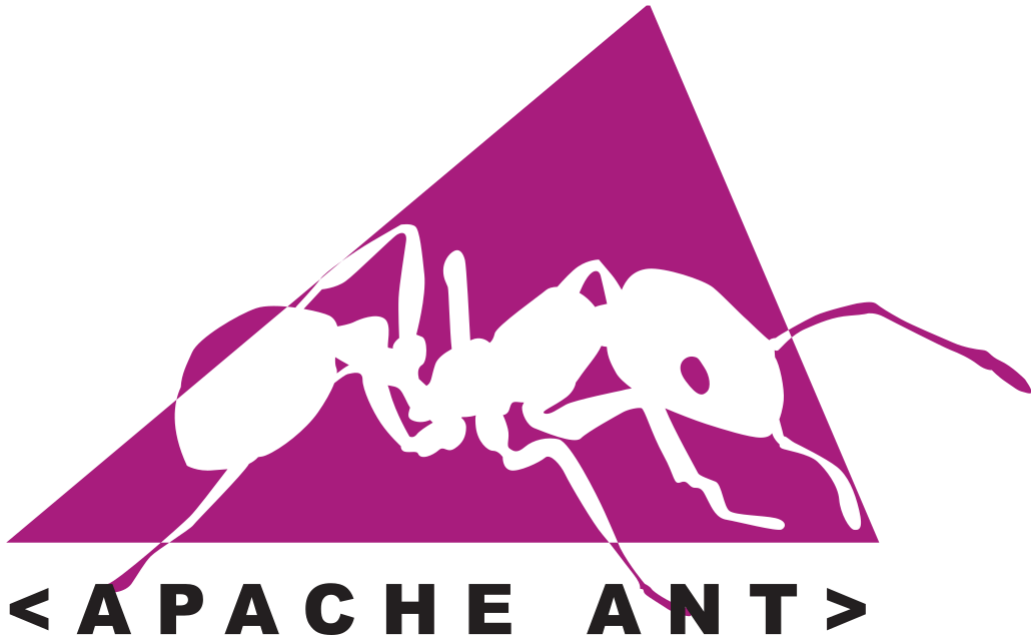
You can install it in the Cytoscape App Manager by clicking the "Install from File..." button.

Make sure to start Cytoscape from the command line – any errors (and log messages) from your app will be printed there!



# Maven


- Build automation system.
- Will download your dependencies for you!
- Will *find* dependencies for you!
- Configuration in **pom.xml / build.xml / settings.xml**



# Maven dependencies

Netbeans gives you the option to locate and add dependencies (**"Search Dependency at Maven Repositories"**) when clicking the lightbulb next to your code, but you can also add them to **pom.xml** manually, which you may need to do for Cytoscape. First, you add the repository:

```
<repositories>
    <repository>
        <id>cytoscape_releases</id>
        <releases>
            <enabled>true</enabled>
        </releases>
        <name>Cytoscape Releases</name>
        <url>
http://code.cytoscape.org/nexus/content/repositories/releases/
        </url>
    </repository>
</repositories>
```

# Maven dependencies

With the repository added (so that Maven can find dependencies), you can add the dependencies themselves:

```
<dependency>
    <groupId>org.cytoscape</groupId>
    <artifactId>swing-application-api</artifactId>
    <version>3.0.0</version>
</dependency>
<dependency>
    <groupId>org.cytoscape</groupId>
    <artifactId>work-api</artifactId>
    <version>3.0.0</version>
</dependency>
<dependency>
    <groupId>org.cytoscape</groupId>
    <artifactId>service-api</artifactId>
    <version>3.0.0</version>
</dependency>
```