We provide both the source code and binary version of the Conup project, which is available on the Google code.

Using source code

Before downloading source code, several tools should be available on your machine:

- JDK
- Apache Maven
- SVN

After correctly configuring your machine, you can view or modify the source code with your IDE, run the source code.

Using binary version

For using our project, you can just simply download the binary version, decompress the Conup project into any directory on your computer. Here, we assume that you have decompression directory is /home/nju/conup-0.9.0-DU, it takes several steps before you using our project.

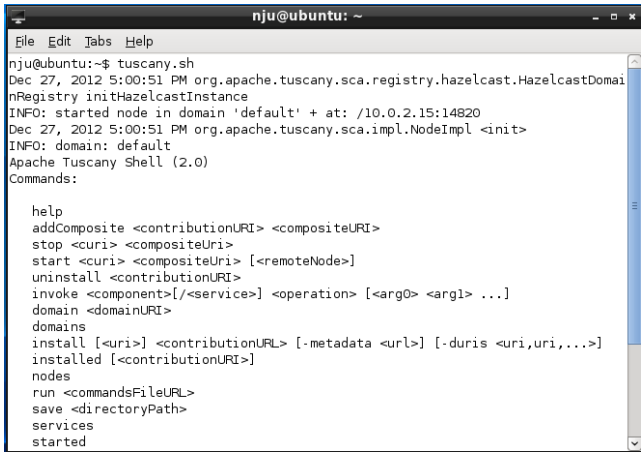
Setup environment path

Setup your environment with the following command:

```
set PATH=$PATH:/home/xx/conup-0.9.0-DU/bin
```

To test whether your environment is correctly configured, running the following command in your terminal, and the result should looks like the figure given below:

```
tuscany.sh
```



Write a Conup.xml for your system

No matter how many components you have in your system, you MUST provide us the global

static configuration in the format of .xml file. Below is a valid Conup.xml file(we put it in `conup/bin`) :

```
<?xml version="1.0" encoding="UTF-8"?>
<conup>
    <configuration>
        <algorithm>TRANQUILLITY_ALGORITHM</algorithm>
        <freenessStrategy>BLOCKING_FOR_FREENESS</freenessStrategy>
        <!—
            Available algorithms:
            TRANQUILLITY_ALGORITHM
            CONSISTENCY_ALGORITHM
            QUIESCENCE_ALGORITHM

            Available freeness strategies:
            CONCURRENT_VERSION_FOR_FREENESS
            BLOCKING_FOR_FREENESS
            WAITING_FOR_FREENESS
        -->
    </configuration>
    <staticDeps>
        <component name="PortalComponent">
            <child>AuthComponent</child>
            <child>ProcComponent</child>
        </component>
        <component name="ProcComponent">
            <parent>PortalComponent</parent>
            <child>AuthComponent</child>
            <child>DBComponent</child>
        </component>
        <component name="AuthComponent">
            <parent>PortalComponent</parent>
            <parent>ProcComponent</parent>
        </component>
        <component name="DBComponent">
            <parent>ProcComponent</parent>
        </component>
    </staticDeps>
</conup>
```

The Conup.xml is composed of two parts: configuration and staticDeps.

- <configuration>: as to the <configuration>, algorithm and freeness strategy must be specified. The <algorithm> means which algorithm you would like to use for dynamic update, while the < freenessStrategy > means which approach you'd like to use to make the component be ready for update.

For the algorithm, three options are available: TRANQUILLITY_ALGORITHM, CONSISTENCY_ALGORITHM, QUIESCENCE_ALGORITHM.

For the freenessStrategy, three options are available: CONCURRENT_VERSION_FOR_FREENESS , BLOCKING_FOR_FREENESS, WAITING_FOR_FREENESS.

Attention: the options you given in your Conup.xml should be exactly the same as the above string.

- <staticDeps:> for each component in your system, the components depends on it and it depends on should be exactly described with <parent> and <child>.

Prepare new version of your component implementation

Temporally, we only support the update for single component one time, and the following rules should be followed:

- the new version of the component should be provided in the format of .class
- the .class file should be put under folders using its package as the name. For example, if your .class file is cn.edu.nju.moon.Sample.class, the Sample.class file should be under the directory cn/edu/nju/moon/

Package your Tuscany application

Packaging your tuscany application into jar files, then executing the following command to install a Tuscany contribution:

tuscany.sh Your*JarFilePath*

If you have multi jars, you can simply run each jar in different terminals.

Invoke your application and Execute dynamic update

As to this part, it has already been described in the documentation on how to run our example.