Short upgrade guide for changes to jAgE related to AGE-65

The new documentation related to this description can be found on Confluence:

- `https://caribou.iisg.agh.edu.pl/confluence/display/jagedocs/Addressing+scheme`
- `https://caribou.iisg.agh.edu.pl/confluence/display/jagedocs/Addressing+scheme+implementation`
- `https://caribou.iisg.agh.edu.pl/confluence/display/jagedocs/Messages`

1 XML configuration files

1. Remove all definitions of addresses for agents, aggregates and workplaces. Optionally, replace them with name initialisation values.

   The code below:

```
<agent name="scully" class="org.jage.examples.helloworld.HelloWorldSimpleAgent">
    <component name="Scully" class="org.jage.address.AgentAddress"
        isSingleton="false">
        <constructor-arg>
            <value value="Scully" class="String" />
        </constructor-arg>
    </component>
    <property name="address">
        <reference target="Scully" />
    </property>
</agent>
```

   Should be converted to:

```
<agent name="scully" class="org.jage.examples.helloworld.HelloWorldSimpleAgent">
    <property name="nameInitializer">
        <value value="Scully" class="String" />
    </property>
</agent>
```

   Or even:

```
<agent name="scully" class="org.jage.examples.helloworld.HelloWorldSimpleAgent"/>
```

In the second version a call to the `toString()` method of the `AgentAddress` class will produce a little less human-readable output.

2. Remove all declarations of node addresses.

   All declarations like the one below should be removed:

   ```
   <property name="nodeUserFriendlyAddress">
   <value value="HelloWorldNode" class="String" />
   </property>
   ```

3. Change names of all node address provider classes.

   Old code uses classes like:

   - `org.jage.address.node.provider.SimpleNodeAddressProvider`,
   - `org.jage.address.node.provider.UIDAgENodeAddressProvider`.

   They have to be changed to:
   `org.jage.address.node.provider.DefaultNodeAddressProvider`.

4. Move a node address provider definition to the highest level in the configuration file.

   Previously, node address providers were defined as an inner component of the address register component. In the new version of the platform its declaration should be moved to the highest level of the configuration.

5. Add an agent address provider component as an inner component of the workplace manager.

   A new component for generation of agent addresses is required. It should be defined as an inner component of the workplace manager, like this:

   ```
   <component name="workplaceManager" class="org.jage.pico.PicoWorkplaceManager"
       isSingleton="true">
       <component name="addressProvider"
           class="org.jage.address.provider.DefaultAgentAddressProvider"
           isSingleton="true"/>
       ...
   </component>
   ```

6. Remove all old declarations of an address register.

   The declaration of the address register and all its inner components should be removed:

   ```
   <component name="addressRegister"
       class="org.jage.address.provider.DefaultAddressRegister" isSingleton="true">
       ...
   </component>
   ```

2 Java code

1. Remove all uses of `IAddress.getAddress()`.

   This method has been deleted. Standard call to `toString()` has a similiar effect.

2. Remove any usage of:

   - `IAgentAddress.getAddressPrefix()`,
   - `IAgentAddress.getClone()`,
   - `IAgentAddress.register()`.

   These methods have been deleted and do not have any replacement.

3. Modify your code to reflect changes to the lifecycle of the agent.

   If you have used methods: `init()` or `finish()`, you need to modify your code to reflect changes to their semantics.

   a) In the old lifecycle the `init()` method was called by the aggregate after adding an agent. It is no longer true. `init()` method is called by the container and, because of this, the agent does not know its environment during this call. In short: you should move all initialisation code that depends on the agent environment to the `setAgentEnvironment()` method.

   b) If you override the `init()` method from the `AbstractAgent`, you should call it before your code or initialise an address on your own. The reason behind this is the fact, that standard implementation in `AbstractAgent` obtains an address for the agent. To sum up, the code should look like this:

   ```
   public void init() throws ComponentException {
       super.init();
       // Your code
   }
   ```

   c) Change the signature of the `finish()` method to:

   ```
   public boolean finish()
   ```

   It should return true if the finalisation of the agent was performed correctly.

4. Modify your code to reflect changes to messages.

   a) All imports from `org.jage.message` need to be converted to `org.jage.communication.message`.

   b) Completely new interfaces and classes for messages have been introduced. For their description see the documentation on jAgE Confluence.

   c) Following classes were removed: `MessageForward`, `PriorityMessage`, `TextMessage`. Remove all references to them.

d) All message classes are immutable. Use only constructors for creating messages (or use your own implementations).

e) All messages and headers are now generic types. You may want to replace all uses of `TextMessage` with e.g. `Message<IAgentAddress, String>`.