Problem

Why should components not run GUIs?

Solution

Subsystems that mainly write non-GUI software in form of ACS components sometimes have to display data to the operator. It is a tempting idea to just open a GUI from a components, e.g. a Java Swing GUI, or something with Casapy Python.

However this is highly discouraged and should be considered at most for temporary quick and dirty solutions:

- Our container/components model is server-side only, and generally server-side software should never create graphical displays.
  - Breaking the layered structure makes the software more difficult to maintain and more difficult to test.
  - We want to be free to deploy the component on any machine, and not rely on X forwarding (outside of the container model!) to actually show the GUI to an operator.
- Java swing creates threads that are not daemon threads. The safe thread factory from the container services can not be imposed on swing. Therefore an unclosed GUI will prevent clean shutdown of the component and the entire container. Perhaps similar problems exist also in Python.
- In some cases such as the archive logger reusing code from jlog, residual GUI code caused JVM crashes when X was not installed.
- If there are problems, ACS can only give minimal support.

Then what is the recommended way to present a GUI to the operator?

- A GUI should be written as an application (i.e. with a main() method in java) which is started independently of containers and components.
  - The OMC is an example for this.
  - The GUI application is started and stopped by the operator
- The OMC can also be seen as a GUI container, the components of which are the OMC plugins. An OMC plugin resembles an ACS component (it has a life-cycle, and it has ZLegacy/ACS.ContainerServices), but is allowed to display a GUI panel. It may well be easier to write a plugin for the OMC instead of writing your own application. Of course, this depends on whether an integration into the OMC is desired from a usability point of view.
- Whether separate application or OMC plugin, the GUI must initiate the connection to the components it interacts with. It can use the SimpleClient / ComponentClient classes or the OMC API for this.
- Once the GUI has contacted a component, it can pass a callback object to the component so that the GUI can be notified of data updates etc without polling the component. Alternatively a notification channel can be set up for this, which mostly makes sense if the component and GUI should not know each other directly.
- As far as integrating 3rd party GUI code into Alma code, we don't know any advantage that integration into a component would have over the separate GUI applications or OMC plugins described here.

Related articles

- How can more people do development with ACS on the same machine without disturbing each other?
- Which ports are used by ACS?
- Problems connecting to ACS servers on a remote machine: bad /etc/hosts
- Why does the getComponent method of ZLegacy/ACS.ContainerServices return an object of type None?
- Why are some of my print statements not showing up in the container output section of acscommandcenter?