Problem

How come string and pattern monitors are implemented differently from other types of monitors?

Solution

A: On-change monitors

While the **Monitorpattern** IDL interface may be defined in exactly the same fashion as other monitor types, it is actually quite different. The most important aspect here is the delta value passed to the **set_value_trigger** method is ignored entirely by the C++ implementation of this monitor interface. You see the concept of a delta change value does not really apply to a fixed array of 64 boolean values (which is identical in principle to a pattern property). The way this monitor really operates is that any change from one period of time to the next in the property's value triggers the monitor.

A: Monitorstring IDL interface is defined but string properties return monitors of type Monitor

string properties return monitors of type **Monitor** instead of the **Monitorstring** IDL interface because the monitor subinterfaces' only purpose is to deal with on-change monitor properties. With a string, it is impossible to change a delta value so there is no reason to create the subclass (aside from consistency).

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?