Erlang Front-end Framework - Bug #21492

When dev_registry restarts all drivers because one driver crashes too much, settings don't get downloaded

12/06/2018 08:37 AM - Dennis Nicklaus

Status:	New	**Start date:**	12/06/2018
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	ACSys/FE Framework	**Estimated time:**	0.00 hour
Target version:			

Description

If one driver continually crashes too much (10x, 1 second), the dev_registry supervisor tree gives up, dies, and gets restarted by its supervisor, daq_sup. This causes all the device drivers to restart, but they are running in a semi-uninitialized state because their settings never get downloaded (after this type of restart), so any actions dependent on settings download won't be running.

Example: clx57 Dec 5, 2018, 18:36.

This had happened a couple days earlier also, with the main symptom being that it appeared T:CDLPID device's settings weren't downloaded. Since its dpmclient readings depend on the downloaded settings, the copyset app wasn't doing its job properly.

It looks like this in the log:

```
=SUPERVISOR REPORT==== 5-Dec-2018::18:36:32.944666 ===
    supervisor: {<0.145.0>,daq_sup}
    errorContext: child_terminated
    reason: shutdown
    offender: [{pid,<0.149.0>},
               {id,dev_registry},
               {mfargs,{supervisor,start_link,
                                   [{local,dev_registry},dev_registry,[]]}},
               {restart_type,permanent},
               {shutdown,5000},
               {child_type,supervisor}]
```

History

#1 - 12/06/2018 11:36 AM - Richard Neswold

- Description updated

- Category set to ACSys/FE Framework

The dev_registry process is pretty simple and has been working reliably. Drivers, on the other hand, are of varying stability because they may be testing new features. We should probably configure dev_registry to never give up restarting a driver that crashes. This prevents dev_registry from going down and then restarting all drivers.

Re-downloading settings for the driver is an interesting problem which needs to be fixed.

#2 - 12/06/2018 03:18 PM - Dennis Nicklaus

A few thoughts:

1. Our current model is definitely not working because it allows one bad driver to affect all others.
2. We might be able to wrap the init method of our driver behaviour with something which delays before allowing the restart. One of the reasons we built in the current 10x/1s restarts is that we don't want the driver to completely thrash re-starting. (Even the current 10 times is pretty excessive when you see it in the log.)
3. Even if a crashing driver is restarted successfully, it isn't going to have its settings downloaded after the restart. We can look into ways of changing the current download behaviour to allow us to download settings only for one OID (which is all that one driver affects). Maybe we can use a method similar to how Camac requests a download for a single crate to download our single OID. (either modifying dnldd or the database, or re-writing dnldd).

#3 - 12/06/2018 05:04 PM - Dennis Nicklaus

It appears to be trivial to "wrap the init method of our driver behaviour with something which delays before allowing the restart. "
We just need to really understand what we're trying to accomplish.