Robotics Service Bus - Tasks #303
Review De-/Activation Logic

05/19/2011 02:28 PM - S. Wrede

Status:	Resolved	**Start date:**	05/19/2011
Priority:	Normal	**Due date:**	
Assignee:	J. Wienke	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

Check if activate/decativate pattern makes sense and is used correctly in the different implementations.

1. Record for which objects this is needed, e.g., de-activation in Java, resource allocation in C++ constructors
2. Decide on (initially single-threaded) implementation of object lifecycle states: Think about InvalidState checking and corresponding exceptions?
3. Distribute implementation tickets... ;-)

Associated revisions

Revision 32a4cc15 - 06/08/2011 04:53 PM - J. Wienke

- improved deactivation logic of SpreadConnection and clearly documented it
- throw errors in SpreadConnection if sending fails instead of bool return value which can easily be ignored

fixes #308

refs #303

History

#1 - 06/09/2011 03:45 PM - J. Moringen

- Target version changed from 0.3 to rsb-0.10

#2 - 11/18/2011 03:04 PM - S. Wrede

- Assignee deleted (J. Moringen)

- Target version changed from rsb-0.10 to 0.6

#3 - 02/27/2012 03:29 PM - J. Wienke

- Target version deleted (0.6)

#4 - 04/19/2012 05:29 PM - J. Wienke

- Status changed from New to In Progress

- Assignee set to J. Wienke

#5 - 04/19/2012 06:34 PM - J. Wienke

So, first of all some things that we should think of when revising this mechanism:

- We need to be able to control when events start being produced (e.g. by InConnectors) as otherwise we cannot make structured assumptions about this fact in later stages of the processing
- Resetting connections might be a good thing sometimes
- Some parts like connectors probably have more states than just active and inactive. For instance and error state can be common.
- State changes of certain objects should be interceptable (i.e. listening on them), especially if these objects can change state on their own. -> probably a generic pattern

related:

- Short discussion on stackoverflow
- A more lengthy one

#6 - 04/19/2012 06:49 PM - J. Wienke

Assuming we decide that we want to model these different states really with the state pattern, the next question is how the state of a composed object affects the state of the owner. E.g. what happens to a Listener's state of the underlying connector switches to "Error"? Does it help to model an Error state for the listener as well? There are also a lot of concurrency issues to think of which make a "perfect" state switching of the Listener impossible. So clients still need to expect erroneous calls before the Listener actually switches to the next state. I have no idea what the right solution is for this problem right now.

#7 - 09/02/2013 07:02 PM - J. Wienke

- Status changed from In Progress to Resolved

I think we are currently at a quite stable state after the recent refactorings, especially in java. So I will close this for now. Otherwise this will stay open endlessly. We can open a more specific issue in case we find a concrete problem with the current state.