Robotics Service Bus - Bug #2774
Participant deactivation is blocking forever if the interrupted exception is not recovered.

10/17/2018 03:40 PM - M. Pohling

Status:	Resolved	**Start date:**	10/17/2018
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	100%
Category:	Java	**Estimated time:**	0.00 hour
Target version:	rsb-0.18		

Description

error blocking forever when the interrupted exception is not recovered.

setup spread configured via global config file and locally started.

test code

```
    public static void main(String[] args) {
        LocalServer server = Factory.getInstance().createLocalServer("/test/scope");
        RemoteServer remote = Factory.getInstance().createRemoteServer("/test/scope");

        try {
            server.addMethod("mymethod", new Callback() {
                @Override
                public Event internalInvoke(Event request) throws UserCodeException {
                    System.out.println("process task");
                    try {
                        Thread.sleep(10000);
                    } catch (InterruptedException ex) {
                        // Thread.currentThread().interrupt();
                        System.out.println("interrupt task");
                        throw new UserCodeException(ex);
                    }
                    return request;
                }
            });
        } catch (RSBException e) {
            e.printStackTrace();
        }

        System.out.println("activate");
        try {
            server.activate();
        } catch (RSBException e) {
            e.printStackTrace();
        }
        try {
            remote.activate();
        } catch (RSBException e) {
            e.printStackTrace();
        }

        System.out.println("trigger server task");
```

```
        try {
            remote.callAsync("mymethod");
        } catch (RSBException e) {
            e.printStackTrace();
        }
        try {
            Thread.sleep(5000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println("deactivate");
        try {
            server.deactivate();
        } catch (RSBException | InterruptedException e) {
            e.printStackTrace();
        }
        try {
            remote.deactivate();
        } catch (RSBException | InterruptedException e) {
            e.printStackTrace();
        }
    }
```

output

```
activate
trigger server task
process task
deactivate
interrupt task
Oct 17, 2018 2:27:34 PM rsb.patterns.LocalMethod internalNotify
WARNING: Exception during method invocation in participant: /test/scope/mymethod/. Exception message:
java.lang.InterruptedException: sleep interrupted

// until than its blocking forever when the interrupted exception is not recovered.
```

Associated revisions

Revision 7dcfa051 - 10/25/2018 04:20 PM - J. Wienke

Backport: Make handlers interruptible

Let handlers and callbacks throw InterruptedExceptions to ensure that
participants can be deactivated while a call to handler is running.

fixes #2774

(cherry picked from commit 56db1280e805210314a0defa3300e2512babaaea)

Revision 180525f9 - 10/25/2018 04:24 PM - J. Wienke

Backport: Make handlers interruptible

Let handlers and callbacks throw InterruptedExceptions to ensure that
participants can be deactivated while a call to handler is running.

refs #2774

(cherry picked from commit 56db1280e805210314a0defa3300e2512babaaea)

Revision 20dfdfc8 - 10/25/2018 04:26 PM - J. Wienke

Backport: Make handlers interruptible

Let handlers and callbacks throw InterruptedExceptions to ensure that
participants can be deactivated while a call to handler is running.

refs #2774

(cherry picked from commit 56db1280e805210314a0defa3300e2512babaaea)

History

#1 - 10/23/2018 11:03 AM - J. Wienke

I don't see who interrupts whom at which time in the test code. When should the interruption take place?

#2 - 10/23/2018 11:33 AM - M. Pohling

Its an user code stability issue.
Just execute the example code and you will see that the deactivation method blocks forever.
If the interrupted exception is packed into an UserCodeException:

```
} catch (InterruptedException ex) {
   // Thread.currentThread().interrupt();
   System.out.println("interrupt task");
   throw new UserCodeException(ex);
}
```

and the interruption is not recovered by the usercode like this:

```
Thread.currentThread().interrupt();
```

You should make sure that the

```
server.deactivate();
```

```
remote.deactivate();
```

are never blocking because of any still processing or already canceled tasks.

#3 - 10/23/2018 04:03 PM - J. Wienke

Actually, I don't see how we should solve this without client help. We need interruption to enable a fast deactivation of participants and their internal threads. But if the client code running inside this thread isn't cooperative and swallows the interrupted state, we have no chance to notice it. Adding a second flag in addition to the interrupted flag sounds awkward.

#4 - 10/25/2018 04:29 PM - J. Wienke

- Status changed from New to Resolved
- % Done changed from 0 to 100

Applied in changeset commit:rsb-java|7dcfa051ba2c9fa98104217466084841e2065c1f.

#5 - 11/08/2019 12:38 PM - M. Pohling

Please reopen issue since the test code is still blocking forever.

#6 - 11/08/2019 03:33 PM - J. Moringen

> *Please reopen issue since the test code is still blocking forever.*

It is highly questionable whether such a local method should be supported.

Nevertheless, in the master branch, this example program as well as other variants with respect to re-throwing, wrapping, swallowing the InterruptedException don't cause any hangs.

#7 - 11/08/2019 06:36 PM - M. Pohling

> *It is highly questionable whether such a local method should be supported.*

In my view its really important because
1. there are just a view java developer out there who now how to properly handle InterruptedException
2. once the interruption is not correctly recovered in the internalInvoke block than rsb stucks forever in the participant deactivation method.

I would highly appreciate if you could patch rsb 0.18 as well with the bug fix because there are already so many different cases were rsb java blocks during the deactivation phase so its always a hard challenge to identify the responsible code.

#8 - 11/11/2019 04:41 PM - J. Moringen

|

> *I would highly appreciate if you could patch rsb 0.18 as well with the bug fix because there are already so many different cases were rsb java blocks [...]*

That makes backporting all required changes to make it work reliably less feasible. Given the number of problems and the resources we can exert on this, pplying changes only in the GitHub master branch seems like the best strategy.