Kepler - Bug #3207

consolidate command line execution actors

04/04/2008 09:03 AM - Daniel Crawl

Status:	Resolved	**Start date:**	04/04/2008
Priority:	Normal	**Due date:**	
Assignee:	Christopher Brooks	**% Done:**	0%
Category:	actors	**Estimated time:**	0.00 hour
Target version:	1.0.0	**Spent time:**	0.00 hour
Bugzilla-Id:	3207		

Description

Command Line Exec and External Execution have different interfaces
but perform the same basic functionality. They should be combined
into a single actor.

History

#1 - 04/04/2008 09:12 AM - Christopher Brooks

The Ptolemy Exec actor also has duplicate functionality.
I looked at CommandLineExec and it looked like it
really needed a re-design. It appeared that there
had been many small changes to the code.

I'm willing to spend some effort on a redesign.

Perhaps one idea would be to compare and contrast
the feature set of the three actors and decide what
features we wanted.

#2 - 04/08/2008 10:51 AM - Daniel Crawl

Hi Christopher,

Thanks for helping out. The Ptolemy Exec actor is called "External
Execution" in its KAR file, so there are only two actors that need
to be consolidated.

Both Exec and CommandLineExec have the following:

command portparameter command with arguments to execute
directory fileparameter working dir
environment parameter env vars for child process
input(Stream) input port strings to stdin
output(Stream) output port strings from stdout
error/exitCode output port strings from stderr

Exec additionally has:

throwExceptionOnNonZeroReturn parameter if true, throw exception if failure

CommandLineExec additionally has:

arguments input port command arguments
infileHandle input port use file for stdin (e.g., ./run < file)
outputFile fileportparameter use file for stdout (e.g., ./run > file)
outfileHandle output port copy of outputFile
waitForProcess parameter if true, wait for child proc
outputLineByLine parameter not used
hasTrigger parameter if true, create input trigger port

The first three (arguments, infileHandle, and outputFile) are not
strictly necessary since the same information can be provided in
the command string. (This works in CommandLineExec, but not in Exec;
this is probably due to the different ways they tokenize their
arguments). If outputFile is removed, then so should outfileHandle.

A more flexible approach to trigger the actor is subclass it from

LimitedFiringSource, which would give it a trigger port and
firingCountLimit parameter.

The remaining parameters, throwExceptionOnNonZeroReturn and waitForProcess,
seem useful to me.

Thoughts?

#3 - 04/15/2008 04:13 PM - Daniel Crawl

I updated the External Execution actor based on the design in comment #2.

If there are no complaints, I will remove CommandLineExec from the actor library.

#4 - 04/15/2008 07:03 PM - Christopher Brooks

I can fold this in to the ptII devel tree asap, but should this ship in 1.0.0
or wait?

#5 - 04/17/2008 12:57 PM - Daniel Crawl

I committed my remaining updates to Exec.java. Once it has been folded into ptII CVS, this bug can be closed.

#6 - 04/17/2008 01:13 PM - Christopher Brooks

I'll take this bug and fold the Exec actor changes into the ptII release
branch.

#7 - 04/18/2008 08:58 AM - Christopher Brooks

I'm part way through the merge, and had two questions.

1) The new version of the actor always uses adds prepends cmd or
sh to the command line.
This method was added:
/** Get the command list arguments for exec. */
protected List<String> _getCommandList() {
List<String> retval = new LinkedList<String>();

```
String osName = System.getProperty("os.name");
        if (osName.equals("Windows NT") || osName.equals("Windows XP")
        || osName.equals("Windows 2000")) {
            retval.add("cmd.exe");
            retval.add("/C");
        } else if (osName.equals("Windows 95")) {
            retval.add("command.com");
            retval.add("/C");
        } else {
            retval.add("/bin/sh");
            retval.add("-c");
        }
        return retval;
    }
```

which is later called with:
List<String> commandList = getCommandList();
commandList.add(((StringToken) command.getToken()).stringValue());

My question here is: do we always want to preface the command with sh
or cmd? This is definitely a change from the old Ptolemy version, and
worthy of discussion. At a minimum, we should update the class comment.
My concern here is that we are breaking backward compatibility.
What do we gain by invoking bash?
This change does cause $PTII/ptolemy/actor/lib/test/auto/ExecEnvironment.java
to fail because that test uses the "env" command, which is not found
for me under Windows XP with cygwin when Exec uses sh.

For example, if I invoke sh from a cmd window, then the path does not
include /usr/bin:
C:\cxh\ptII>c:/cygwin/bin/sh -i
sh-3.2$ env
sh: env: command not found
sh-3.2$ echo $PATH
/cygdrive/c/Program Files/Java/jdk1.6.0_05/bin:/cygdrive/c/matlab701/bin/win32

This is with a fairly new installation of cygwin.

However, if Exec does not invoke sh, then the test passes, I think
because the path is properly set from the parent process.

Here I propose that either we remove the code that adds the sh or cmd
or else make it optional. I'd consider keeping it non-optional, but would
like some discussion.

2) The second problem is that the new version does not properly handle
command lines that use double quotes to separate command line arguments.

This text was removed:
// tokenizeForExec() handles substrings that start and end
// with a double quote so the substring is considered to
// be a single token and are returned as a single array
// element.
String[] commandArray = StringUtilities
.tokenizeForExec(((StringToken) command.getToken())
.stringValue());

As a result, $PTII/ptolemy/actor/lib/test/auto/Exec.xml fails because
it invokes the command
echo "Hello World"

I think this should be fairly easy to fix in the new version.

A third issue is that I should merge this actor with code in
ptolemy.util.StreamExec. However, I think it would be best to wait on this
change.

#8 - 04/18/2008 04:10 PM - Christopher Brooks

I did the merge.
I added a parameter called "prependPlatformDependentShellCommand",
which, if set to true, will prepend cmd.exe \c or /bin/sh -c to
the path. The default value is false.

I also modified Exec so that it properly handles command lines like
echo "Hello world"

The two Command Line demos worked for me.

I did notice kepler/src/org/sdm/spa/CommandLineExec.java still exists though.

I did not update the release branch though. Can someone else handle that?

#9 - 04/22/2008 02:14 PM - Daniel Crawl

By prepending sh or cmd, the command can use file redirection operators.

I tested this actor on several platforms, and only Cygwin does not correctly copy the path. Additionally, I tried using ProcessBuilder instead of Runtime.exec(), but got the same result. (However, on windows, using ProcessBuilder, the subprocess inherits all the variables, but using Runtime.exec(), it does not).

This bug can be closed once Exec is updated on pt rel-7-0-beta-2.

#10 - 04/22/2008 05:10 PM - Christopher Brooks

Ok, I'll update ptII rel-7-0-beta-2 tomorrow (Wed).
I'm taking this bug pending the update.

#11 - 04/23/2008 12:23 PM - Christopher Brooks

This model writes a random number to a file using the Exec actor, but
the file is not created before the attempt to read the file.

#12 - 04/23/2008 12:24 PM - Christopher Brooks

Ok, I folded these changes into ptII rel-7-0-beta-2.
I spent some time working with the file redirection operators by setting
the prependPlatformDependentShellCommand parameter to true and found
that it does not work very well. I think the problem is more with
Windows than anything.
What I wanted to do was echo a random number to a file and read it

in a workflow. What seems to happen is that the echo command does not create the file.

However, this is a separate problem and not significant.
I'm closing this bug!

#13 - 03/27/2013 02:22 PM - Redmine Admin

Original Bugzilla ID was 3207

Files

ExecFileRedirect.xml	12.7 KB	04/23/2008	Christopher Brooks