Spawning processes from build

Sometimes you'd like to spawn a process from a build that lives longer than the build itself. For example, maybe a part of the build is to launch a new application server with the result of the build. When you do this, you often experience a problem where the build doesn't terminate; you'll see that shell script/ant/maven terminates as expected, but Jenkins just insists on waiting, as if it didn't notice that the build is over.

Starting Jenkins 1.136, Jenkins detects this situation and instead of causing infinite block, it will just print out a warning and let you get going. But you should still understand what's causing this.

Why?

The reason this problem happens is because of file descriptor leak and how they are inherited from one process to another. Jenkins and the child process are connected by three pipes (stdin/stdout/stderr.) This allows Jenkins to capture the output from the child process. Since the child process may write a lot of data to the pipe and quit immediately after that, Jenkins needs to make sure that it drained the pipes before it considers the build to be over. Jenkins does this by waiting for EOF.

When a process terminates for whatever reasons, the operating system closes all the file descriptors it owned. So even if the process didn't close stdout /stderr, Jenkins will nevertheless get EOF.

The complication happens when those file descriptors are inherited to other processes. Let's say the child process forks another process to the background. The background process (AKA daemon) inherits all the file descriptors of the parent, including the writing side of the stdout/stderr pipes that connect the child process and Jenkins. If the daemon forgets to close them, Jenkins won't get EOF for pipes even when the child process exits, because daemon still have those descriptors open. That's how this problem happens.

A good daemon program closes all file descriptors to avoid problems like this, but often there are bad ones that don't follow the rule.

Workarounds

On Unix, you can use a wrapper like this to make the daemon behave. You can call your command like this:

```
daemonize -E BUILD_ID=dontKillMe /path/to/your/command
```

In case of Jenkins Pipeline use `JENKINS_NODE_COOKIE` instead of `BUILD_ID`

Note that this will set the BUILD_ID environment variable for the process being spawned to something other than the current BUILD_ID. Or you can start jenkins with -Dhudson.util.ProcessTree.disable=true - see ProcessTreeKiller for details.

On Windows, the 'at' command can be used to launch a process in the background. See the example below:

```
<scriptdef name="get-next-minute" language="beanshell">
  <attribute name="property" />

  date = new java.text.SimpleDateFormat("HH:mm")
    .format(new Date(System.currentTimeMillis() + 60000));
  project.setProperty(attributes.get("property"), date);
</scriptdef>

<get-next-minute property="next-minute" />
<exec executable="at">
  <arg value="${next-minute}" />
  <arg value="/interactive" />
  <arg value="${jboss.home}\bin\run.bat" />
</exec>
```

Another similar workaround on Windows is to use a wrapper script and launch your program through it.

```
<exec executable="cscript.exe">
   <env key="ANTRUN_TITLE" value="Title for Window" />  <!-- optional -->
   <env key="ANTRUN_OUTPUT" value="output.log" />  <!-- optional -->
   <arg value="//NoLogo" />
   <arg value="antRunAsync.js" />  <!-- this script -->
   <arg value="real executable" />
</exec>
```

Another workaround for Windows XP and later is to shedule permanent task and force running it from the ant script.
Once run the command:

```
C:\>SCHTASKS /Create /RU SYSTEM /SC ONSTART /TN Tomcat /TR "C:\Program Files\Apache Software Foundation\Tomcat
6.0\bin\startup.bat"
```

Note, that ONSTART can be replaced with ONCE if you do not want to keep Tomcat running.
Add the following code to your ant script:

```
<exec executable="SCHTASKS">
    <arg value="/Run"/>
    <arg value="/TN"/>
    <arg value="Tomcat"/>
</exec>
```

Another possibility that we can consider is to do something in Jenkins.