**UCD Deployment using REST**

We need to write our own integration to UCD from some other app. This example shows how to use the REST interface to request and track a deployment.

**An Example Execution (using curl)**

# ./execApp.sh

Run the Deploy Process on App SMOKE\_TEST

Request ID: '5b8b4c91-abb6-462b-8847-2ae3b19d2e53'

Check Status Right Away

{"status":"PENDING","result":"SCHEDULED FOR FUTURE"}

Check again after 20 seconds...

{"status":"CLOSED","result":"SUCCEEDED"}

Run the Deploy Process on App SMOKE\_TEST Again but this time trick it to fail

Request ID: 'defc4f40-c487-4531-a399-08b04af27bc2'

Wait 20 seconds before checking for results..

{"status":"CLOSED","result":"FAULTED"}

Here is a link to the deployment log in UCD:

https://zeus:8443/#applicationProcessRequest/defc4f40-c487-4531-a399-08b04af27bc2

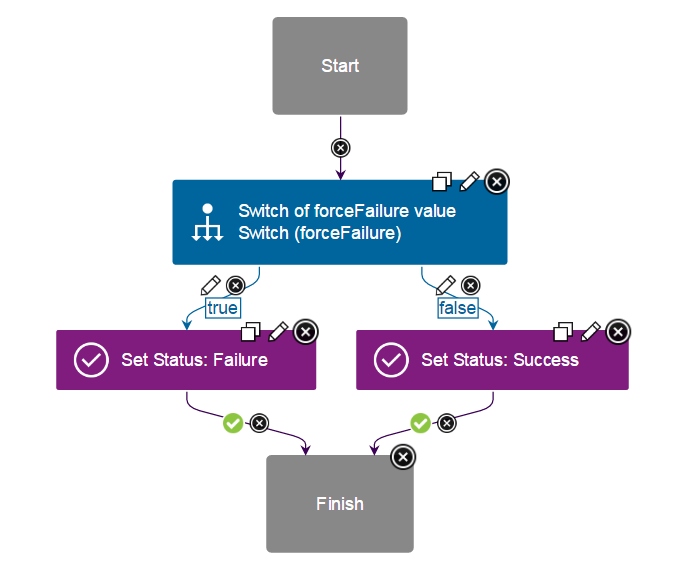
**REST commands (see the UCD KnowledgeBase)**

|  |  |
| --- | --- |
| [cli/applicationProcessRequest/request](https://www.ibm.com/support/knowledgecenter/SS4GSP_6.2.1/com.ibm.udeploy.api.doc/topics/rest_cli_applicationprocessrequest_request_put.html) | Input – json request  Output – request ID  Note: it returns as soon as the process has been initiated /assigned |
| [cli/applicationProcessRequest/requestStatus/request=<*requestID*>](https://www.ibm.com/support/knowledgecenter/SS4GSP_6.2.1/com.ibm.udeploy.api.doc/topics/rest_cli_applicationprocessrequest_requeststatus_get.html) | STATUS and RESULT |
| <*UCD\_URI*>/#applicationProcessRequest/<*requestID*> | URL to Execution Log |

**Appendix**

**Test Application**

Set up a Test app that you can force success and failure. For this test I set up a property on the app process called forceFailure which can be set to either true or false at runtime. Here is the fake deployment process:



**Sample JSON (01-requestAppProcess-Succeed.json)**

{

"application": "SMOKE\_TEST",

"applicationProcess": "Deploy",

"environment": "DEV",

"description": "Run Fake Deploy Process in DEV environment",

"properties": {"forceFailure":"false"},

"onlyChanged": "false",

"versions": [ {

"component": "AAA\_SD\_WIN",

"version": "latest"

}],

}

**Sample Script**

#!/bin/sh

#

# show using REST interface to run a process and fetch results

#

USER="PasswordIsAuthToken:de973f7c-6246-4d19-b294-d58c18d9465f"

URI="https://zeus:8443"

CURL="curl -s -k -u $USER"

CURLAPR="$CURL $URI/cli/applicationProcessRequest/request"

##

## Case 1 - run an app process that (eventually) returns success

## Note the request process only returns the request ID

##

echo ""

echo "Run the Deploy Process on App SMOKE\_TEST"

#

# returns: {"requestId": "79904093-79d5-43b1-b6e2-3d89ad362bcd"}

#

requestID=`$CURLAPR -X PUT -d @01-requestAppProcess-Succeed.json|grep -Po 'requestId":"\K[^"]\*'`

echo "Request ID: '$requestID'"

CURLSTAT="$CURL $URI/cli/applicationProcessRequest/requestStatus?request=$requestID"

echo ""

echo "Check Status Right Away"

result=`$CURLSTAT`

echo "$result"

echo ""

##

## Note - if real life you would set up a loop to sleep and check...

##

echo "Check again after 20 seconds..."

sleep 20

result=`$CURLSTAT`

echo "$result"

##

## Case 2 - run an app process that runs and fails

##

echo ""

echo "Run the Deploy Process on App SMOKE\_TEST Again but this time trick it to fail"

requestID=`$CURLAPR -X PUT -d @02-requestAppProcess-Failed.json|grep -Po 'requestId":"\K[^"]\*'`

echo "Request ID: '$requestID'"

echo ""

echo "Wait 20 seconds before checking for results.."

CURLSTAT="$CURL $URI/cli/applicationProcessRequest/requestStatus?request=$requestID"

sleep 20

result=`$CURLSTAT`

echo "$result"

LOGFILE="${URI}/#applicationProcessRequest/${requestID}"

echo ""

echo "Here is a link to the deployment log in UCD:"

echo "${LOGFILE}"