Making a free JIRA Scheduler

The main purpose here is to utilize the Linux Cron facility to Transition Issues, rather than using the almost useless JIRA builtin services, or buying an expensive cron/scheduling plugin for JIRA.

The solution is pretty flexible for all issue types, by making the "step Initiate" a parameter instead of hardcoded, all issuetypes can be transitioned.

- **Prerequisites**
  - Linux User
  - jq processor
  - A custom Field for crontab values
  - Location
- **The useful Scripts**
  - Make the crontab script
  - Make the Transition Script
  - The Master Cron
- **Troubleshooting**

**Prerequisites**

**Linux User**

All scripts and the crontab stuff assumes that there is a "jira" user with access to /opt/jira-cron/

**jq processor**

jq command-line JSON processor must be available to the scripts

https://stedolan.github.io/jq/Workflow

My IssueType is called "Repeatable Task" and has a very small workflow:

![Issue Workflow Diagram](image)

Upon the Transition of **Initiate** (id 61), the Post Function:

- Creates a copy of the Issue as a subtask
- Sets the issue back to Status "Frozen"
A custom Field for crontab values

The Issuetype has a custom field called "Cron Scheduling" (Text Field (single line)), as this will hold the schedules for the Issue:

In the script, the field has the JIRA identifier customfield_12821 - You must change this to fit Your field.

Location

I have my stuff in /opt/jira-cron

The usefull Scripts
Make the crontab script

The scripts makes and enables a crontab for the "jira" user. It search for issues that satisfies:

- **Issuetype=Repeatable Task**
- **Status=Frozen**
- **Cron Scheduling has a value**
#!/bin/bash

rm /tmp/crontab > /dev/null 2>&1

cd /opt/jira-cron/

IFS=$(echo -en \n
TODAY=`date +%Y-%m-%d.%H:%M:%S`

TransitionScript="/opt/jira-cron/jiraTransitionIssue.sh"

JIRAFilter="issuetype%20%3D%20%22Repeatable%20Task%22%20AND%20status%20in%20%28Frozen%29%20%20and%22Cron%20Scheduling%22%20%20NOT%20%20%20%20%20maxResults=999"

JIRAUSER=""

JIRAPASS=""

#echo https://jira.server.dk/rest/api/2/search?jql=$JIRAFilter

CURLOPT_HEADER=0
echo CURLOPT_HEADER

IssueTotal=`cat /tmp/issuelist.json | ./jq-linux64 '.total'`

echo "$TODAY Total: $IssueTotal" >> /opt/jira-cron/jiraMakeCrontab.log

Count=0

for IssueId in `cat /tmp/issuelist.json | ./jq-linux64 '.issues[].id'`
do
Count=$(($Count + 1))

IssueId=`echo $IssueId | sed "s/"//g"

curl -D -k -u $JIRAUSER:$JIRAPASS -X GET -H "Content-Type: application/json" https://jira.server.dk/rest/api/2/issue/$IssueId > /tmp/issue.json

IssueKey=`cat /tmp/issue.json | ./jq-linux64 '.key'| sed "s/"//g"

#IssueStatus=`cat /tmp/issue.json | ./jq-linux64 '.fields.status.name'| sed "s/"//g"

cat /tmp/issue.json | ./jq-linux64 '.fields.customfield_12821' | sed "s/"//g" | sed "s/\r//g" | sed "s/\t/ /g" > /tmp/cronfield.json

for CronEntry in `cat /tmp/cronfield.json | tr ";" "\n"
do

echo "$CronEntry $TransitionScript $IssueKey > /dev/null 2>&1" >> /tmp/crontab

echo "$TODAY Added ($Count): $CronEntry $TransitionScript $IssueKey to /tmp/crontab" >> /opt/jira-cron/jiraMakeCrontab.log

done

echo "" >> /tmp/crontab

done

#Replace JIRA Users crontab
if [ ! -f /tmp/crontab ]
then
echo "" > /tmp/crontab
fi

crontab /tmp/crontab

The final line, making the JIRA crontab, can be extended with an error handler.
After running the script; the `/var/spool/cron/crontabs/jira` (on Ubuntu LTS) should look like this:

```bash
#Crontabs for SUPPORT-513
* * 1 * * /opt/jira-cron/jiraTransitionIssue.sh SUPPORT-513 >> /opt/jira-cron/jiraTransitionIssue.log 2>&1

#Crontabs for HOMEPAGE-3846
0 5 2 * * /opt/jira-cron/jiraTransitionIssue.sh HOMEPAGE-3846 >> /opt/jira-cron/jiraTransitionIssue.log 2>&1

#Crontabs for HOMEPAGE-2933
0 0 1 * * /opt/jira-cron/jiraTransitionIssue.sh HOMEPAGE-2933 >> /opt/jira-cron/jiraTransitionIssue.log 2>&1

#Crontabs for PROJECTTOOLS-2467
0 5 5 * * /opt/jira-cron/jiraTransitionIssue.sh PROJECTTOOLS-2467 >> /opt/jira-cron/jiraTransitionIssue.log 2>&1
```

**Make the Transition Script**

Make sure the "jira" user has executeable acess to this. The TRANSITIONJSON defines the **Initiate** Transition (id 61)

```bash
#!/bin/bash

IFS=$(echo -en "\n\b")
TRANSITIONJSON='{"transition": {"id": "61"}}'
JIRAUSER=""
JIRAPASS=""
CURLOPT_HEADER=0
export CURLOPT_HEADER

cd /tmp

IssueKey=$1

if [ $IssueKey != '' ]
then
    curl -D header.txt -u $JIRAUSER:$JIRAPASS -X POST --data $TRANSITIONJSON -H "Content-Type: application/json" https://jira.server.dk/rest/api/2/issue/$IssueKey/transitions
    rm header.txt
else
    echo "No IssueKey as parameter"
fi
```

**The Master Cron**

To make the JIRA users crontab on a regular basis, I use `/etc/crontab` on Ubuntu:

```
0 20 * * * jira /opt/jira-cron/jiraMakeCrontab.sh
```

**Troubleshooting**

Make sure the crontab is functioning, look in `/var/log/syslog` for tings like:
Cron reloading the new crontab for JIRA user:

<table>
<thead>
<tr>
<th>Time</th>
<th>Server</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr 5 13:28:50</td>
<td>myserver</td>
<td>crontab[12857]: (jira) REPLACE (jira)</td>
</tr>
<tr>
<td>Apr 5 13:29:01</td>
<td>myserver</td>
<td>cron[4946]: (jira) RELOAD (crontabs/jira)</td>
</tr>
</tbody>
</table>

Transitions via jira users cron entries:

<table>
<thead>
<tr>
<th>Time</th>
<th>Server</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr 5 05:00:01</td>
<td>myserver</td>
<td>CRON[1563]: (jira) CMD (/opt/jira-cron/jiraTransitionIssue.sh HOMEPAGE-2467 &gt;&gt; /opt/jira-cron/jiraTransitionIssue.log 2&gt;&amp;1)</td>
</tr>
</tbody>
</table>

FIELD and STATE variables in the script files

Remember to change the variables to Your setup