Talend User Component tRunTask

cimt
let your work flow

Purpose **http://www.cimt-ag.de**

This component runs job as task in the Talend Administration Center (TAC).
The advantages are:

- It is possible to create execution plans with nearly unlimited flexibility
- It enables the chaining of external job scheduler with the TAC
- It is not necessary to build “monster” jobs by embedding the whole process with tRunJob
- With chaining task with tRunTask it is always possible to monitor all single steps (one task)
- The used API does not need Talend specific libraries and is based on Open Source technology

Talend-Integration

This component can be found in the palette under Management (next to tRunJob)
This component provides several return values.

Parameters

Property	Content
TAC URL	URL of the TAC (it is the same as used in Studio or in the browser) ***required***
TAC login	User login (User need the Administrator role) It is recommended to use an technical user
TAC password	Users password
Use task label	If false you need to know the ID of the task (refer to the information view of a task in the TAC)
Task ID	ID of the task ***requiered (if Use task label is false)***
Task label	Label of the task ***requiered (if Use task label is true)***
Job run by task	The job which will be run by this task. This configuration expect to choose the job which is actually used in the already deployed task. It does NOT change the job for a task. The purpose is to configure the context variables.
Context Parameters	The context parameters and its source.
Run task asynchronously	If true it starts the task and pools for its end. If false it waits for the end in the same http request (in long running task this could lead to broken pipes)
Wait until the end	Waits for the end of the job. Otherwise the job will be started and the component finish and the current job can continue. It is like fire and forget. To prevent calling a task twice at the same time, the component checks at if the task is already running and waits until its end.
Check time cycle until job is running	If Run task asynchronously is true, we have to wait until the TAC has started the job the poll on its end. Set the time in ms.
Check time cycle until job is running	If Run task asynchronously is true, we have to poll on its end. It depends on the experiences about the typical task run duration. It is not recommended to

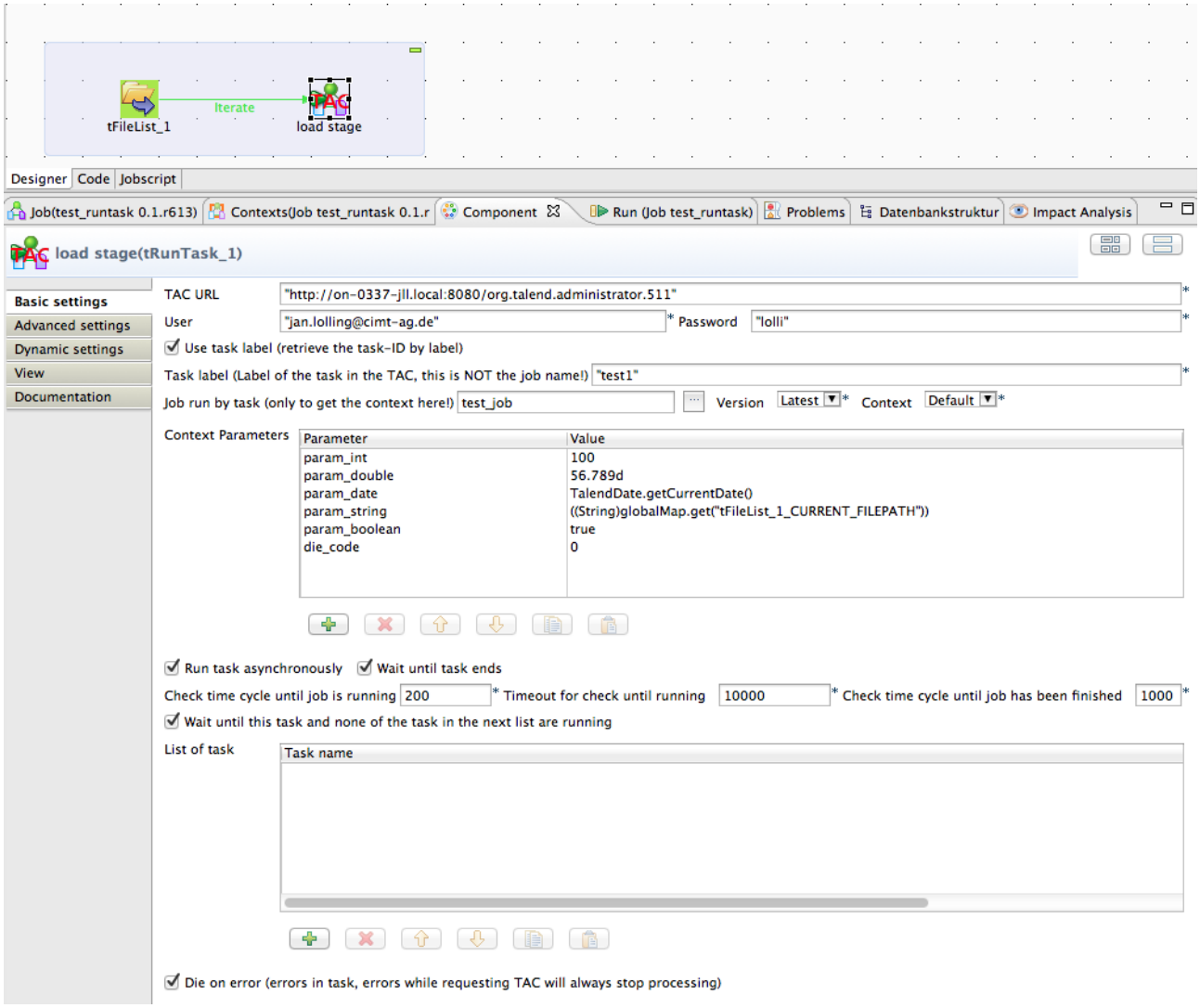
	pool to often, because it could lead to a notable load for the TAC.
Wait until this task and none of the task in the list are running	If true, the component pools on the finish status of it own task and the listed tasks and start its own task after none of these task are currently running.
Die on error	Dies of the started task fails (means the job fails because of any problems within the job). If there are any errors in communication with the TAC, the tRunTask component will always fail.

Return values

Return value	Content
ERROR_MESSAGE	Last error message. Unfortunately this is not the error message from the actually running job. This message is build from the tRunTask component. The current TAC web service does not provide this message.
TASK_ID	The task id retrieved from the TAC by the task label. All commands to the TAC related to task needs the task ID. Thats why the tRunTask component retrieves this ID at first.
RUN_DURATION	The time the task is running measured by the tRunTask component. It is not exactly the time because of the possible polling time delay.
RETURN_CODE	The return code is at the moment not the return code from the job, because of the lack of capabilities of the TAC web service. At the moment the value is 4 if the job fails and 0 if the job succeeded. It is on the way to get the original return code but at the moment there is no TAC release known in which this feature will be available.

Szenario:

Building a job execution plan:



This scenario shows the way to implement a trigger which starts a task in the TAC for every found file. There are a lot of other scenarios possible.
One of the most used scenario is to trigger a task from another Job scheduler because of an company policy about scheduling. In large companies there are typically dedicated schedulers and with this component you can write a job with it self can started as simple script from such kind of schedulers.