Added by Ark Xu, last edited by Tammo van Lessen on Jun 04, 2009

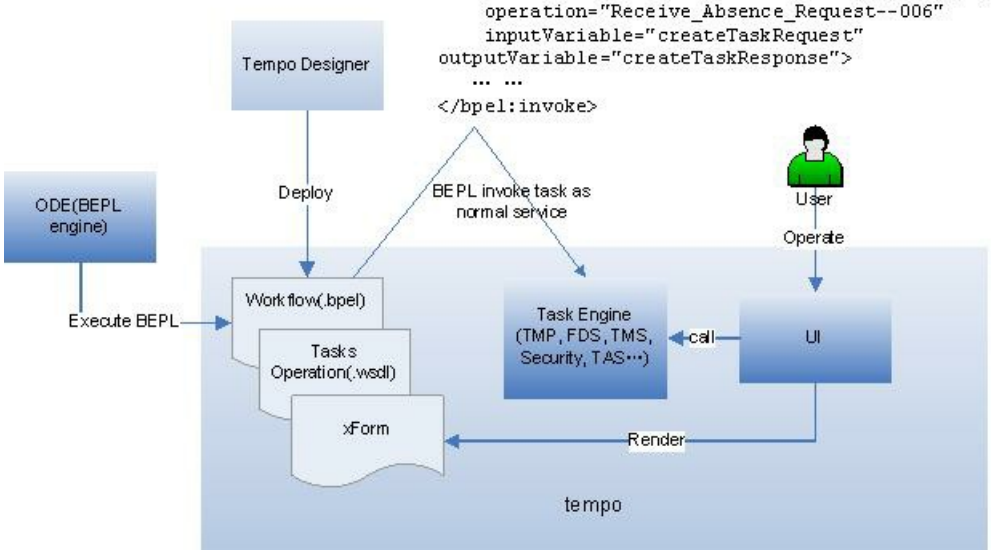
System Analysis

What Tempo currently do

Currently, Tempoal already works as "Human Task Engine", but very different with B4P spec.

Current Tempo way to implement human task - in general
"Tempo Human Tasks" will be designed and deployed by Tempo designer, and called by BPEL workflow, which is executed by ODE. But this implementation way is not exactly compatible with new B4P spec. Following diagram shows the current tempo way.

```
<bpel:invoke
    name="Request_Manager_Approval-Invoke"
    partnerLink="AbsenceRequest_And_Manager_PLink"
    portType="Manager:AbsenceRequest_To_Manager_PortType"
    operation="Receive_Absence_Request--006"
    inputVariable="createTaskRequest"
outputVariable="createTaskResponse">
   ... ...
</bpel:invoke>
```



1. User design new workflow, will generate:
BPEL file - define the whole workflow, which may invoke human task
xform file - define the form displayed on ui, and it's action
wsdl file - define task operation, which may be called by xform action

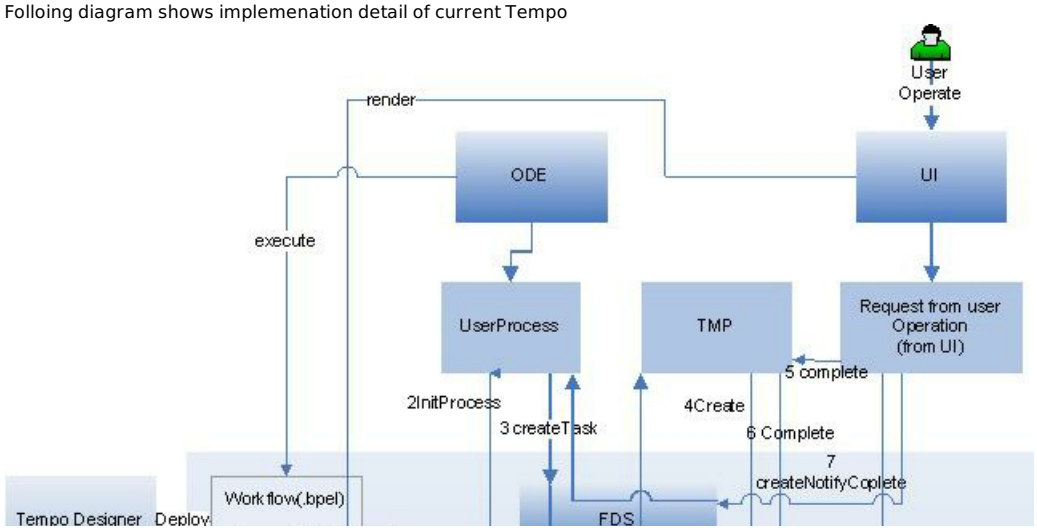
2. In current way, tempo human task will be invoke as general web-service by workflow engine(ODE).

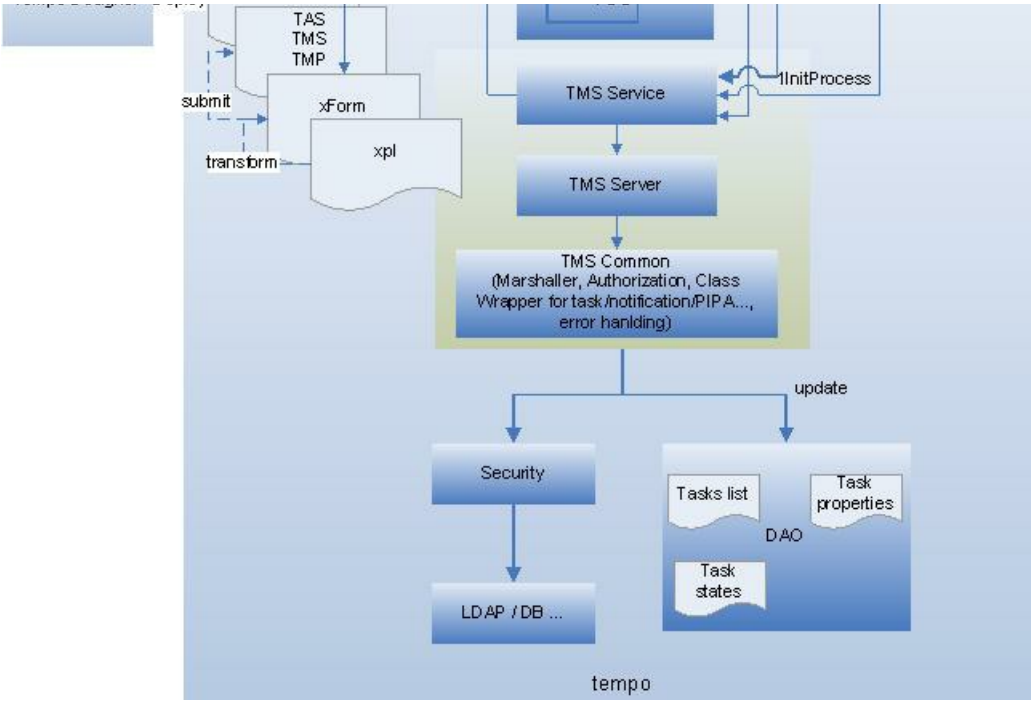
3. Tempo task engine accept the call from ODE,and create specific task, and persist it.

4. UI client can list tasks, display the xform.

5. When User do some operation on UI, corresponding web-service call will be fired, Tempo accept the calling, and do operation on tasks or call back workflow engine.

Current Tempo way to implement human task - in detail
Folloing diagram shows implemenation detail of current Tempo





1.Almost all request to Tempo task engine, will go throught FDS(Form Despatcher Service).

2.The request from workflow engine will be translated to request to call TMS service.TMS Service will create task.

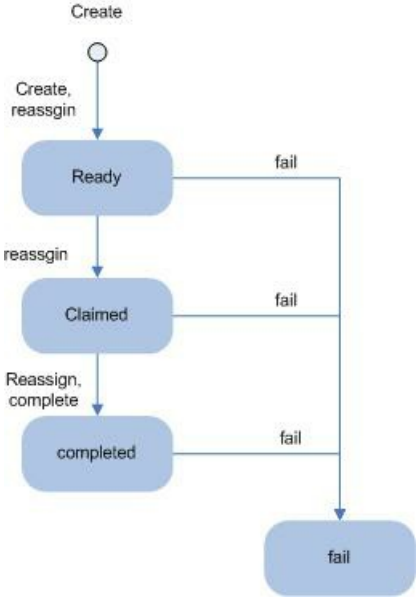
3.The request from UI will be parsed by FDS,the notify request will be translated and sent to TMS, TMS will create notification. Then user can receieve the notification on UI.

4.TMS (Task Management Service) is exposed as web service,which includes all possible operation on Task.
The TMS service implements the web service interface, unmarshall request message, call TMS server, and marshall response.
The TMS server implements the operation logic by calling TMS common authentication/authrozation and persistence classes.

5.The security component provides API for authentication and authrozation.

6.The DAO implements the persistence layer.

Task states Transition

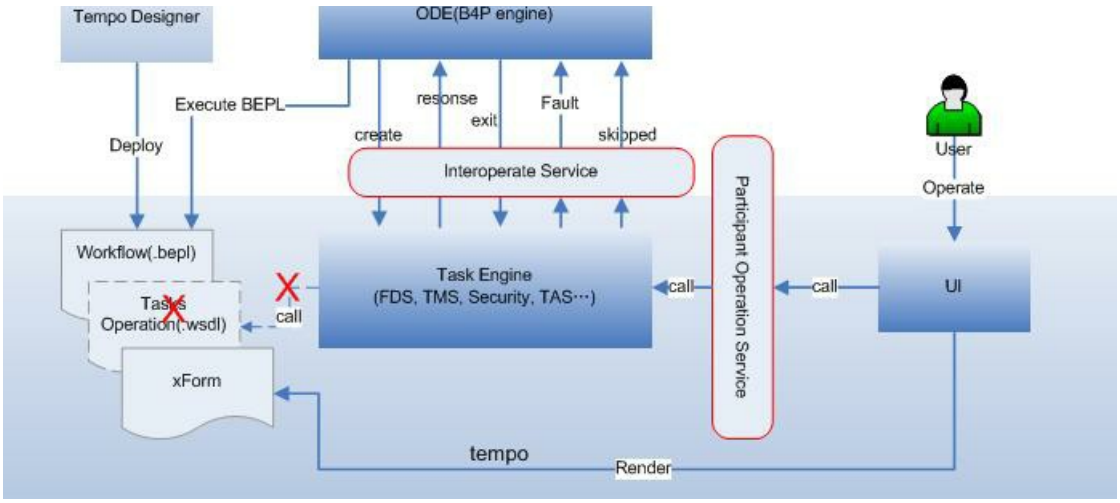


Task properties

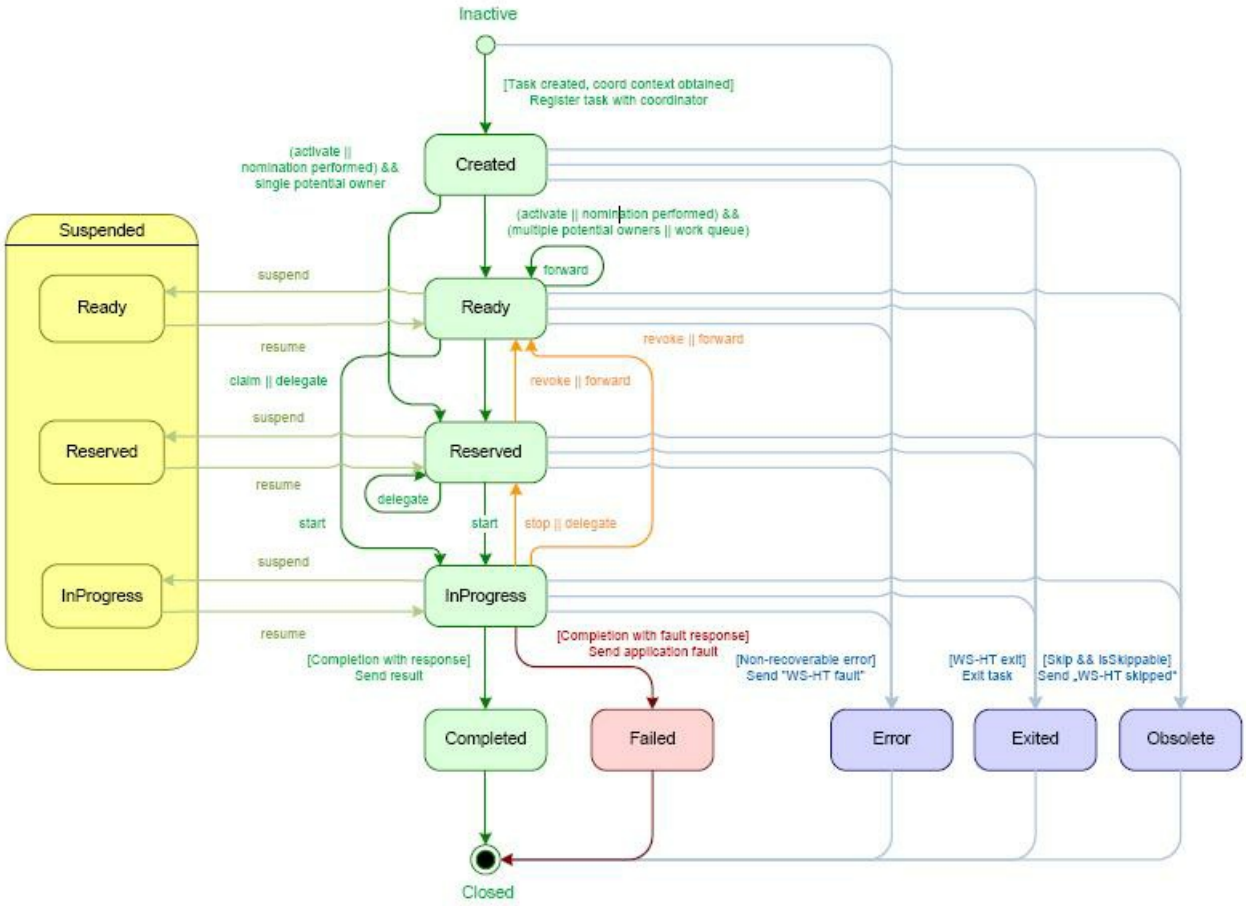
```
int    internalId
String id
String description
Date   creationDate
String formURL
```

What B4P Human Task Engine should do

Task Engine comply with B4P



Task States Transition



Task properties

```
<xsd:element name="task" type="tTask"/>
<xsd:complexType name="tTask">
<xsd:sequence>
<xsd:element name="id" type="xsd:string"/>
<xsd:element name="taskType" type="xsd:string"/>
<xsd:element name="name" type="xsd:QName"/>
<xsd:element name="status" type="tStatus"/>
<xsd:element name="priority" type="xsd:nonNegativeInteger"
minOccurs="0"/>
<xsd:element name="taskInitiator" type="htd:tUser"
minOccurs="0"/>
<xsd:element name="taskStakeholders" type="htd:tOrganizationalEntity"
minOccurs="0"/>
<xsd:element name="potentialOwners" type="htd:tOrganizationalEntity"
minOccurs="0"/>
<xsd:element name="businessAdministrators" type="htd:tOrganizationalEntity"
minOccurs="0"/>
<xsd:element name="actualOwner" type="htd:tUser" minOccurs="0"/>
<xsd:element name="notificationRecipients"
type="htd:tOrganizationalEntity" minOccurs="0"/>
<xsd:element name="createdOn" type="xsd:dateTime"/>
<xsd:element name="createdBy" type="xsd:string" minOccurs="0"/>
<xsd:element name="activationTime" type="xsd:dateTime" minOccurs="0"/>
<xsd:element name="expirationTime" type="xsd:dateTime" minOccurs="0"/>
```

```
<xsd:element name="expirationTime" type="xsd:dateTime" minOccurs="0" />
<xsd:element name="isSkipable" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="hasPotentialOwners" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="startByExists" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="completeByExists" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="presentationName" type="tPresentationName" minOccurs="0"/>
<xsd:element name="presentationSubject" type="tPresentationSubject" minOccurs="0"/>
<xsd:element name="renderingMethodExists" type="xsd:boolean"/>
<xsd:element name="hasOutput" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="hasFault" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="hasAttachments" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="hasComments" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="escalated" type="xsd:boolean" minOccurs="0"/>
<xsd:element name="primarySearchBy" type="xsd:string" minOccurs="0"/>
<xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
```

Compare Tempo with B4P compliant Task Engine

Feature	Tempo	BPEL4People	Influence upon implemention
Task Properties	Only 5 properties	28 properties(4 is mandatory) and can be extended	big changes, but not huge
Task States	4 States and only 4 operations(create/reassign/complete/fail) will change states	9 states may produced by more than 10 operations	Huge changes...
Task operations	**18** Operations only **9** op may be reused	**37** operations/function needs to be implemented, only **13** operations can resue Tempo OP	will cost a lot of efforts
InterOperate with workflow engine	Will call endpoint of wf engine on create/init/esculate task	use coordinate protocol	a lot of efforts
Permission	using RBAC	use 7 generic human role and logic group	a lot of efforts
Attachment	cannot exchange attachment with wf engine?	will be exchanged by interoperable protocol	seems not big change
UI Side	??	??	??

operation	reuse?=>B4P HT Operation
getTaskList	no relevant HT operstion
getAvailableTasks	=>getMyTasks
reassign	?=>delegate,but no HT-op can set any state like reassign
getAttachments	=>getAttachments
addAttachment	=>addAttachment
removeAttachment	deleteAttachment
getTask	?=>getTaskInfo,getTaskDescption
setOutput	=>setOutput
complete	=>complete
setOutputAndComplete	=>setOupt,complete
fail	=>fail
delete	=>remove
deleteAll	no relevant HT operstion
create	no relevant HT operstion
initProcess	no relevant HT operstion
getPipa	no relevant HT operstion
storePipa	no relevant HT operstion
deletePipa	no relevant HT operstion

Operation	reuse?=>Tempo task Operation
claim	no
start	no
stop	no
release	no
suspend	no
suspendUntil	no
resume	no
complete	=>complete
remove	=>delete
fail	=>fail
setPriority	no
addAttachment	=>addAttachment

addAttachment	=>addAttachment
getAttachmentInfos	?=>getAttachments
getAttachments	?=>getAttachments
deleteAttachments	=>removeAttachment
addComment	no
getComments	no
skip	no
forward	no
delegate	?=>reassign
getRendering	no
getRenderingTypes	no
getTaskInfo	=>getTask
getTaskDescription	=>getTask
setOutput	=>setOutput
deleteOutput	no
setFault	no
deleteFault	no
getInput	no
getOutput	no
getFault	no
getMyTaskAbstracts	no
getMyTasks	=>getAvalaibleTasks
query	no
activate	no
nominate	?=>reassign
setGenericHumanRole	no

Comments

Site powered by a free **Open Source Project / Non-profit License** (more) of **Confluence - the Enterprise wiki**.
Learn more or **evaluate Confluence for your organisation**.

Powered by Atlassian Confluence, the Enterprise Wiki. (Version: 1.4.1 Build:#212 Jun 02, 2005) - Bug/feature request - Contact Administrators