This presentation will focus on developing Human Tasks with WebSphere Integration Developer Version 6 for WebSphere Process Server Version 6.

Goals

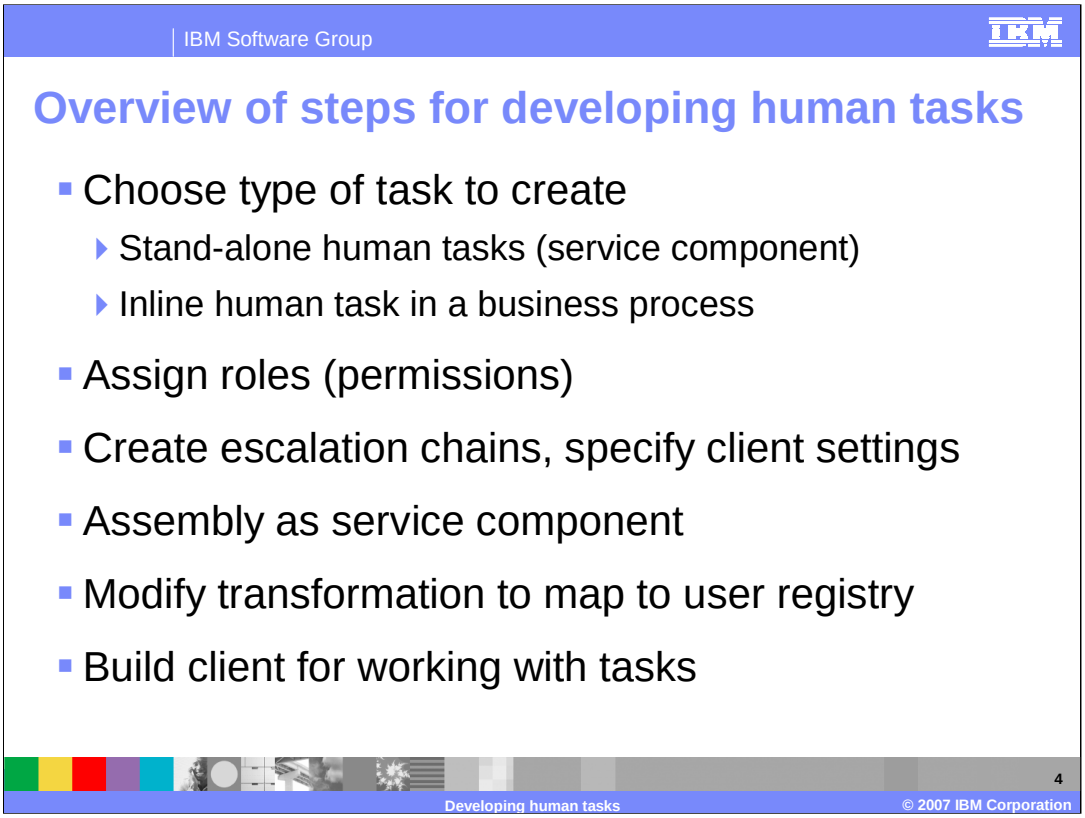
- Describe the architecture for support of human tasks
- Detail the development capabilities for human tasks
- Explain the assembly tasks for human tasks

The goals of this presentation are to start with an overview of the different types of tasks and how they are used and made available as service components. Then the different development capabilities will be discussed. Finally the assembly of Human Task as service components will be explained.

Section

Development overview

This section will provide an overview of the development of Human Tasks.



You can create Human tasks easily with WebSphere Integration Developer Version 6. Six steps are required; some are optional depending on the environment and requirements. When you begin to develop a Human Task, decide which human task is best for the particular situation. There are two categories of task to choose from. A task can either be a stand-alone service component or it can be a human task used inside a business process. The primary difference is in how the task will be used, although there are other differences to be discussed in more detail later in the presentation. If the task will be reused in different situations, a stand-alone task should be used. An in-line task is more appropriate when a business process is part of the solution.

After you select and create the task of either type, you can configure that task. You can specify roles or permission to perform different actions on a task. You can set Escalation chains and client settings. At this point the definition of the human task should be complete, and if it is a stand-alone component, it is ready for you to assemble as a service component. Inline tasks do not need a separate assembly step because they are part of business processes and will be assembled as part of the business process. Before you deploy, you can perform an optional step to map the roles or permissions to the enterprise user registry in order to use specific attributes in determining who should be assigned a role. The final optional step is to create the client for working with and accessing the work items for the task.

Different type of human tasks

- Different type of human tasks available for different integration situations
- Participating task

  - Service creates a work item for human interaction
  - Available separate or inline a business process
- Originating task

  - Human interaction invokes a service
- Pure human task

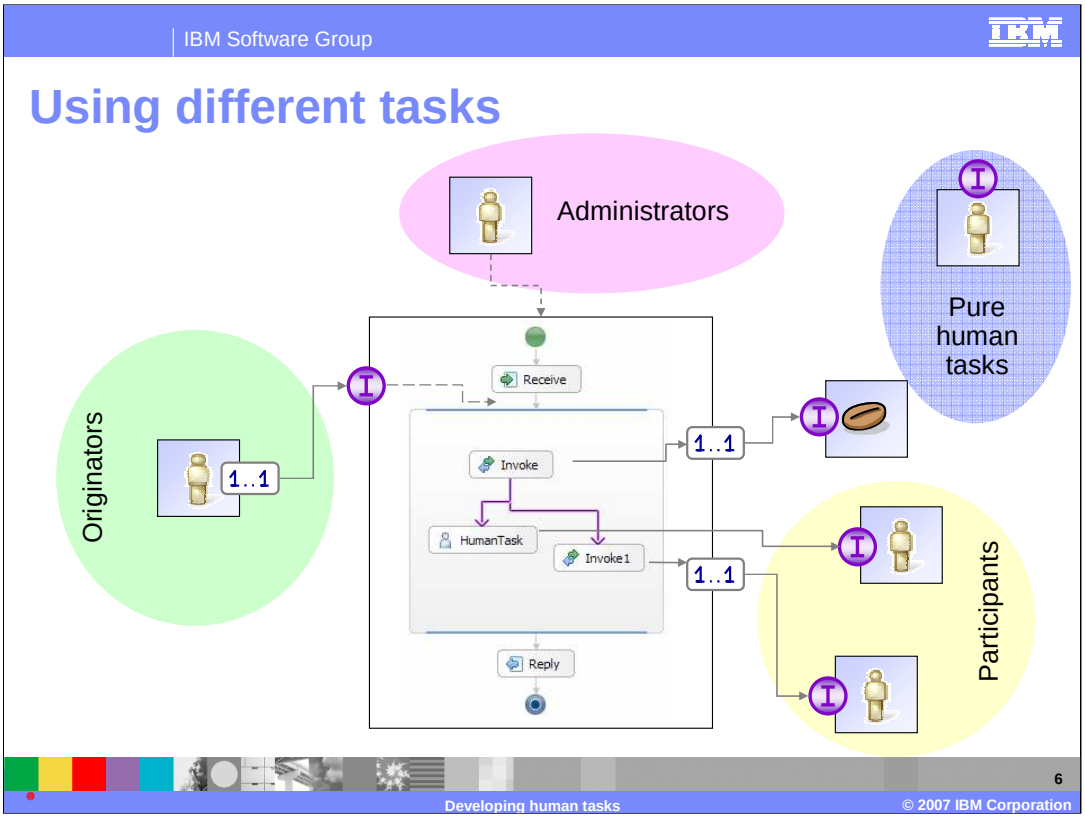
  - Human interaction invokes a service which creates a work item for another human

There are several types of human tasks. The type of human that you use depends on your integration situation and how you want human interaction to be used within your application. The types of human tasks are:

•Participating Task – used when an application or machine is going to assign work to an individual. This is very similar to the Staff Activity from V5.1 where you have an automated work flow process and at some point you assign work to an individual. A Participating Task can be packaged as a separate service component or part of a business process. In both situations the functionality is the same, however a stand-alone Participating task can be reused outside of a business process. After the person completes the task, control returns to the caller or business process and processing continues.

•Originating Task – used when a person is going to assign work to some type of machine or service. When the task is complete, a response is returned providing a record of the task and the completion status. This could be used when you are calling a business state machine and updating the state. It might also be used with a BPEL process where you are assigning some type of action to occur. A human initiates the process and the machine or service called is a BPEL process or any SCA component.

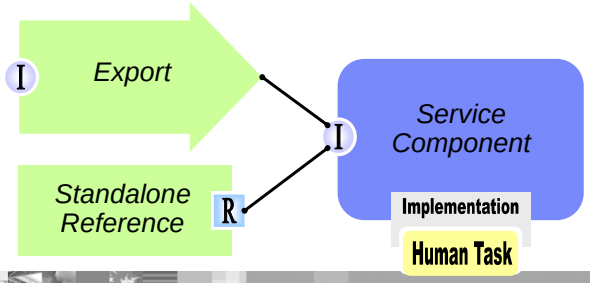
•Human Task – there is no SCA invocation going out or coming in to this task, but you can still take advantage of the human task manager and other escalation and notification capabilities. Using this type of task, you could quickly and easily create an interface and generate work items for specific individuals using escalation and notification support to create an application for a manager to generate “to dos” for his or her department. After tasks are completed, the manager can view the results for each task.



This slide illustrates how the different types of tasks can be used. There are two participating tasks. One is an inline task and directly wired within the business process. The other participating task is a stand-alone task which is packaged as a service component and is wired to the business process through the partner link on an invoke activity and reference on the component. You can also see an Originating task which is used to start the business process. The Originator task will call the business process as a service and when the process completes, a response from the business process will be returned to the originator task. You can also see a Pure Human task which is separate and not used in any way with other service components although it does have an interface. [3] There is also a task for administrators. These tasks are specific for business process and can be specified on the definition of activities in the business process. They are not considered application related tasks.

Utilizing SCA and business objects

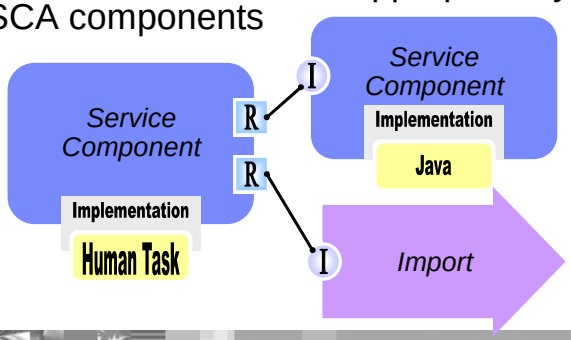
- Human tasks (participating tasks) can be invoked as SCA components
  - Allows for easy invocation from SCA components (WSBPEL business processes) and non SCA components (session bean)
- Business objects provides a standard data format for invoking human task messages
  - Business object messages sent and received



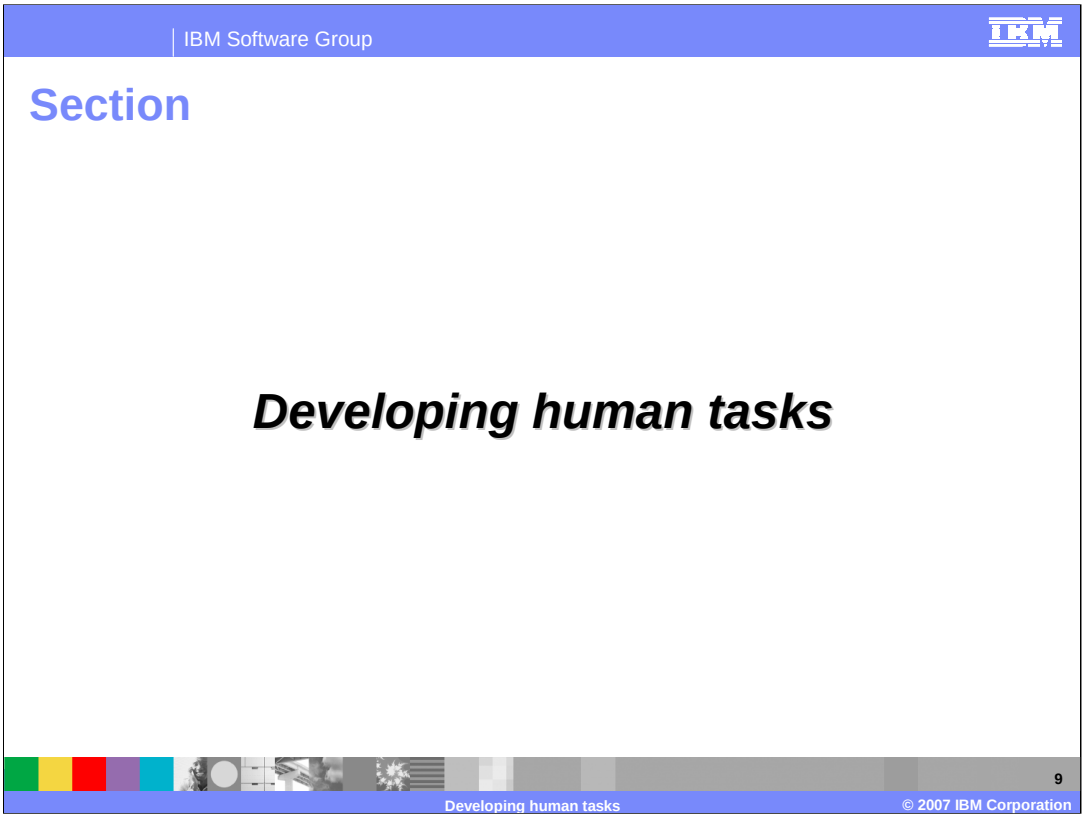
It is helpful to understand how human tasks utilize the SCA programming model and Business Objects. The human task implementation utilizes SCA and SDO so for certain participating tasks, you can expose an SCA interface, allowing other components to call this human task to generate work for a specific individual. With those SCA components, you will be utilizing Business Objects passed in to the service component and then to the human task that the person will work with. This functionality is made available through the use of an export. If you would like to use a human task and generate work items using Java™, you can do so by creating and using a stand-alone reference instead, in which case you would use the SCA APIs to call the component and generate work for an individual.

Utilizing SCA and business objects (cont.)

- Human tasks (originating tasks) can invoke SCA components
  - Provides invocation point for a person to invoke SCA components
  - Escalation and notification available while waiting for implementation to complete even though it is non human
- Messages with business objects passed to other SCA components or transformed to the appropriate type when calling non SCA components



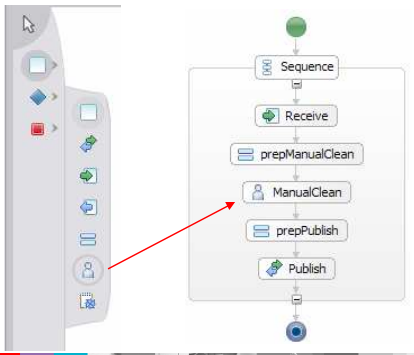
Originating tasks also utilize SCA and Business Objects. Through the use of the human task interface and this component, you can call other components that are defined in the same module using a direct wiring, or in another module using an import. This is similar in many respects to creating a Web interface and calling a servlet or placing a message on a JMS queue. However, by making this an SCA component and using human task support, you can take advantage of additional capabilities such as tracking how long a request takes to complete, which allows you to take action, such as initiating an escalation chain, if the work is not done within a specified period of time.



This section will cover the options for developing human tasks.

Creating human tasks

- Specific wizard for creating participating, originating, and pure human tasks
  - Directly uses SCA and business objects

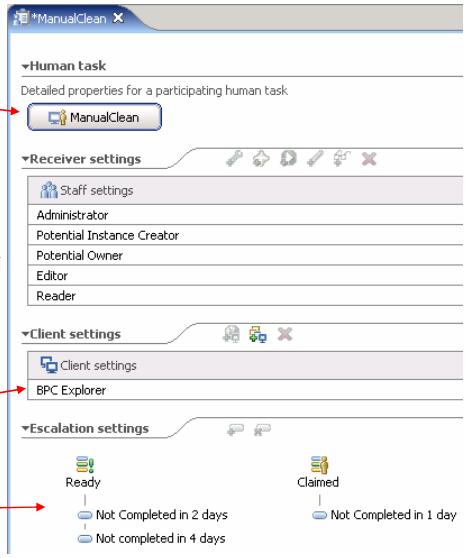


- Special activity for adding a human task to a WSBPEL business process (inline task)
  - More capabilities over calling a participating task

There is now a specialized wizard for creating human tasks outside the Business Process Editor. All three types of human tasks can be created using this editor. When you create human tasks, you are creating a component separate from BPEL. The tight integration between BPEL processes and human tasks can be maintained by using the human task activity in the business process editor. The same staff icon which was used in WebSphere Studio Application Developer Integration Edition V5.1, is used for creating the inline processes. There are benefits and drawbacks in using an inline human task rather than a separate service component task. The benefits include some additional behaviors and management of the task life-cycle from the management of the process lifecycle. The drawback is that it is not flexible and should the task need to be change, the process will need to be updated.

Human task editor

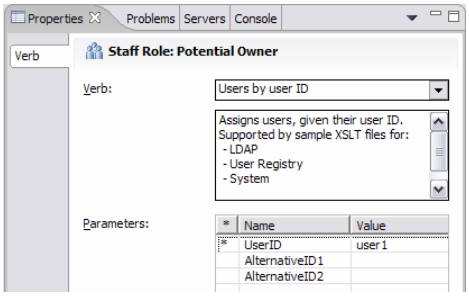
- General properties
  - Staff plug-in provider
  - Expiration and priority settings
  - Calendar
- Permission settings
  - Specify verb and parameters for roles
- Client interface
- Escalations and notifications



When you create a task, whether it is a participating, originating, human, or inline task in your BPEL process, you use the Human Task Editor. Clicking the button at the top of the editor will display general settings in the Properties view. Details such as the duration for the task, starting priority, and JNDI name of the staff plug-in provider can be specified here as well. There is also a calendar setting that can be specified, which is used to track the durations for Escalations. In the second section, you can specify permissions for individuals on the work items for the task. There is a variety of permissions to select from and they will vary depending on the task. Most tasks have administrator, potential owner, editor, and reader as available permissions. For each permission, a verb and group of parameters is specified to indicate the individuals to be assigned the appropriate permission. The client interface section allows you to specify custom JSP or JSF files that you have created for working with these items and displaying them in the BPC explorer. The Escalations settings section allows for different escalation chains to be specified based on the state of the task.

Assigning permissions for human tasks

- Tasks have different roles allowing different permissions to be assigned to a user or group of users for tasks
  - Potential instance creator – may create tasks
  - Potential owner - may claim and complete tasks
  - Administrator – may override owner, delete tasks
  - Editor - may view and updated task; can not complete task
  - Reader - may view tasks
- Potential owner must be specified for a human task
  - Editor and Reader not required
- Each role must have a **verb** and **parameters** specified



The most important part of configuring a human task is setting permissions for the task. It is important that each individual has the proper permissions to work with or perform a work item. Permissions vary slightly depending on the task, but the ones most commonly available are:

•Potential Instance Creator – indicates who is capable of creating tasks. For example, for a pure human task, the potential instance creator would be limited to the managers.

•Potential Owner – the person or persons who can claim a work item. When a work item is created, it is listed as open and available to the potential owners. Any potential owner can claim the work item and it becomes their responsibility to complete the task. Only the person who claims the work item can complete it.

•Administrator - can override restrictions and complete work items or transfer them to another user.

•Editor – can view and update any work item or task. They do not have the ability to complete the work item. An editor might be useful for example, in a loan approval process where a credit department is gathering information and updating a loan application, but final approval and completion must be performed by a loan officer.

•Reader – can view data that has been updated in the work item before or after it has been completed.

While not all these roles might not be used for a given work item, every work item must have a potential owner who will complete the work item. These roles are enforced at runtime, similar to the J2EE roles based model, however there is no relationship between J2EE roles and Business Process roles.

The roles for the human tasks are assigned using queries composed of Verbs and Parameters. The different verbs and parameters will be explained in the next few slides.

Verbs for roles

- 14 verbs and associated mappings are included
  - Some verbs are only available to certain user registries
  - Certain attributes or relationships to exist within the user registries
  - Additional verbs may be added for specific transformations.

- Role: Potential owner
- Verb: Group members
- Parameter: Approvers

↓

- item = $Verb
- p1 = $Parameter

Mapping (Transformation.xsl)

Verb	Parameter Values
Users	User name
Users by User ID	User ID of User
Group Members	Group name
Group Members without Named Users	Group name, NamedUsers
Group Members without Filtered Users	Group name, Filter Attribute, Filter Value
Department Members	Department name
Role Members	Role name
Manager of Employee	User name of employee (not Manager)
Manager of Employee by user ID	User ID of employee (not Manager)
Person Search	Search by person attribute
Group Search	Search by group attribute
Native Query	Search with a specific search string
Everybody	All authenticated users
Nobody	Only Administrators

By default, fourteen verbs and associated parameters are included by WebSphere Integration Developer Version 6 and supported by WebSphere Process Server Version6. Mappings allow the verb and parameter to be transformed into a query statement that can be interpreted by the particular user registry in use. These mappings are specified in a transformation XSL file. Shown here is the list of verbs in a table, covering most of the situations you are likely to encounter in your enterprise. The Users and Users by User ID Verbs are very similar. The difference is that the Users Verb will try to resolve a fully qualified name (cn: Jane Baxter) to their user ID (uid: jbaxter). The user ID value is typically the value an individual will enter when authenticating to the server and will have to be located to match the user up with the appropriate Work Item. The Users by User ID Verb expects that the user ID will be specified and then assigned to the Work Item without resolving it to another value. A work item could also be assigned to group without a specific user. There is also the ability to assign work item to the manager of a specific user ID. In this case, the staff plug-in provider will query the user registry to retrieve the manager of a specific employee identified by user ID. To utilize the Manager of Employee, Department Members, and Role Members verbs, the user registry must support the employee-manager relationship, which is typically limited to LDAP user registries. The Native Query verb allows you to specify a specific search string as the parameter. This search string will be run against the user registry. While it may be necessary to use a specific search string to find individuals who could not be found using one of the other Verbs, using this Verb does expose runtime and implementation specifics about the user registry within the human task definition. Changes to the implementation or the user registry could force changes to be made within the human task. Instead of using a Native Query Verb, consider creating a new Verb and Parameters and a new mapping or transformation, maintaining a layer of abstraction between your design and implementation. The Everybody role can be used to grant multiple individuals a particular role without needing to create a large number of Work Items. For example, if everyone needs the ability to read a travel request task, rather than assigning the reader role to a particular group which contains all individuals, Everybody can be used, avoiding the overhead of creating and managing a large number of Work Items by the Human Task Manager. A Verb setting of "Nobody" can only be used for the Reader role. You can not assign "Nobody" to the Editor or PotentialOwner role. The value "<not assigned>" can be used to prohibit assign the Editor role to any user and no Work Items will be created with this role. The PotentialOwner role can not have a value of “<not assigned>” and must have a valid Verb and Parameters specified. All verbs are not available for use with all user registries. When you select the verb you would like to use, there is information in the development environment that will indicate which user registries the verb can be used with. For example, if you are using LocalOS, you cannot use manager of employee, because LocalOS does not support this type of relationship indicator. If the included verbs do not meet your needs, you can create additional verbs and a matching transformation.

Parameters for staff queries

- Most verbs also require one or more parameters
- Two type of parameters
  - Early binding
    - Early binding uses specific hard-coded values
      - Examples: John Doe, jsmith, Administrators, WestCoastManagers
    - Values set in deployed query

```
select all
where Group = “Approvers”
```

  - Late binding
    - Late binding uses values from business process or task instance context
      - Context variables
      - Custom attributes/properties
      - Process or task variables
    - Deployed query is parameterized to accept values at run time
    - Very flexible

```
select all
where Group = %myValue%
```


There are two types of parameters and most verbs require at least one parameter.

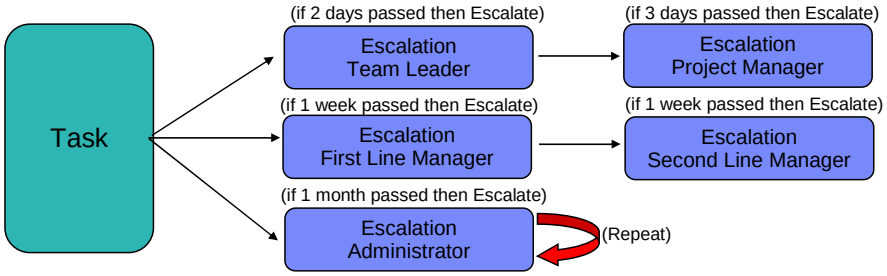
Parameters for the queries can be Early binding values or Late binding values. The Early binding values are specific names which are hard-coded as part of the Role definition. Examples of these would be John Doe or jsmith for a Verb of Users or Users by User ID, or Administrators or WestCoastManagers for the Group Members Verb. These values are not only hard-coded into the definition of your Business Process, but also when the Business Process is deployed into your WebSphere Process Server environment. The Early binding values can seriously limit the flexibility of your Business Process in production as the different roles would be confined to these specific values. If you use groups over specific users, you can have some flexibility and manage the group members to help determine who will be assigned Work Items with that permission.

The Late binding values are a great alternative to the Early binding values. The Late binding values are resolved to values based on the current context of the human task or a business process. The Late Binding values can be resolved to individuals who where assigned to previous activities, completed previous activities, or assigned permissions to the human task or a business process in general. With these values, you can design a very flexible human task and business process allowing for situations where the same activity in two different business process instances can be assigned to different people based upon who completed a previous activity. While late binding values are very flexible, this flexibility has a price. Because the values must be computed at runtime, this can cause extra overhead on task assignment operations.

When a human task is deployed, queries with Late binding values are parameterized to accept these values at runtime. When the human task is run, the Human Task Manager will retrieve the deployed query and resolve the Late binding values and place those into the query before it is passed to the Staff Plug-in Provider to be run against the user registry.

Escalation support

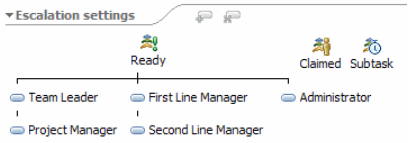
- Escalation may be set on tasks for different states of the task
- Single or multiple chains of escalations can be defined
- Time and state of task are checked before escalation
- Used to create notifications for overdue tasks

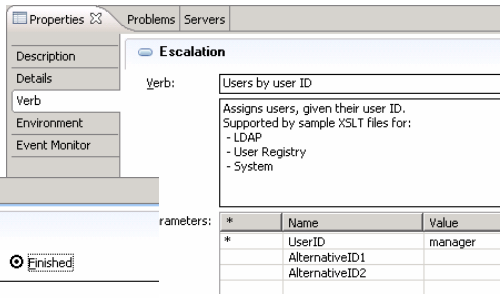


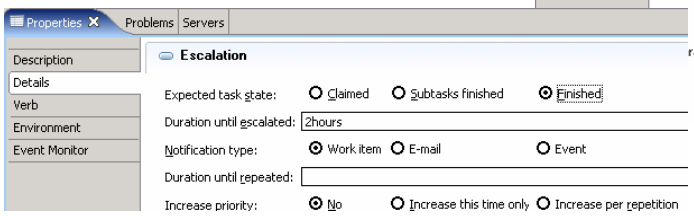
Escalation support provides for a human task to pass through various states as it progresses towards completion. At each of these states, you can specify an escalation chain to begin. When the specified duration expires, a check is performed to determine if the expected state has been met. If it has not a notification is performed and the escalation can continue on for another duration. If the expected state has still not been met after that duration, another notification will occur. The example shown here consists of a human task with three escalation chains. The first chain checks to see if the task is completed after two days, if it has not it will be escalated to the team leader and notification will be performed. The next check is performed three days after the first duration expires, or five days after task creation. If the task has not completed, the task is escalated to the project manager. The second chain performs a check after one week. If the task has not completed, it is escalated to the First Line Manager. Another check is performed one week after the first duration expires and if the task has not completed it is escalated to the Second Line Manager. The third chain checks for completion after one month. If the task has not completed, the administrator is notified. This chain is set up to repeat.

Setting escalations and notifications

- Assignment of escalation notification uses verb and parameter format
- Notification occurs if condition has not been met
- If duration has exceeded and state not reach then escalate



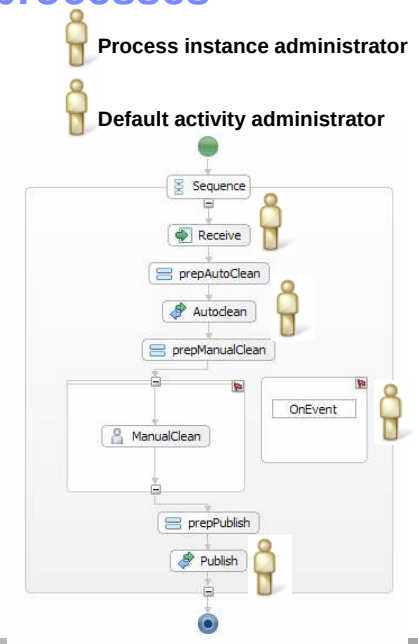




Shown here is the lower part of the editor, where you can set up escalations. You can set up multiple chains to run in parallel. You can see the various states, indicating when the chain of escalations should begin. As soon as the human task is created, it enters the ready state and timers begin for the escalation chains. When the human task is claimed, timers will start for any escalation chains that begin at the time the task is claimed. The third state is if any sub tasks are created, escalation chains can be created there as well. The person who should receive notification is set in the same verb and parameter style used to establish the original permissions. The same verbs are potentially available and you can specify parameters. You can perform various types of escalations, including assign a work item to a specific person, such as project manager or team leader, send e-mail to a person or group, or fire a custom event that an event handler reacts to. You can also increase the priority when that escalation occurs.

Human tasks in business processes

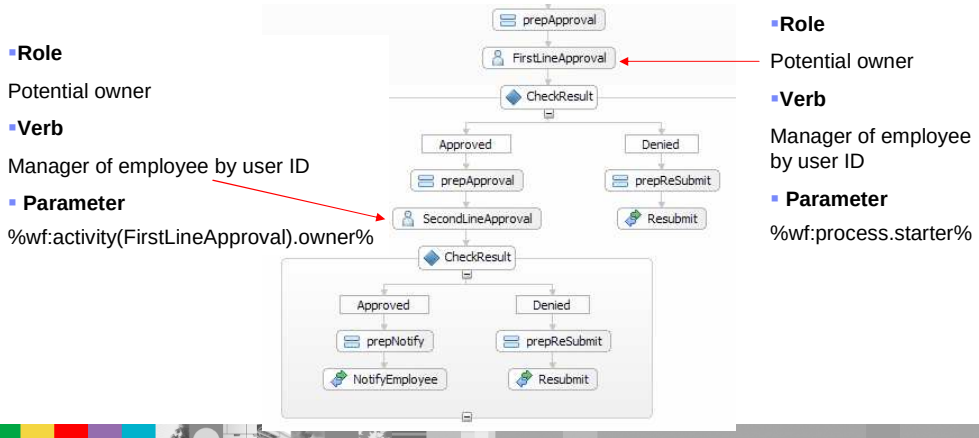
- Tight integration between human tasks and WSBPEL processes
  - Specific human task (inline) activity
  - Special scenarios supported beyond stand-alone tasks
- Human task can be used for restricting incoming operations and indicating administrative capabilities
- Secure and restrict who can call incoming operations
  - Receive, ReceiveChoice, events
- Administrative capabilities
  - For the process, all activities, each activity



There is still the tight integration found in WebSphere Business Integration Server Foundation V5.1 allowing you to use human tasks within a BPEL process as a step within that set of activities. Human tasks are also used for various administrative options as well as setting permissions on different steps. The permissions capability provided by the human task manager can be reused within the BPEL processes to indicate permissions on a particular activity to restrict who can call or administer it. Within the business process editor, you will find a number of places where you can specify human tasks. The first one shown at the top here is the process instance administrator. For any instances created from this BPEL process, there is a participating task that can be created for a specific individual. This individual will have the capability to stop, terminate, or delete a business process. They can also retry events, if a more granular level is not set to initiate compensation on different instances, and they can view all instance information. You can also specify a default activity administrator, who might be separate from the process instance administrator. For example, the process instance administrator might be a technical person, while the activity administrator has the appropriate permissions to view confidential information within the business process and the business objects. You can also establish permissions on the activities to determine who can invoke the receive activity, defined through the originating task for that activity. On invokes, you have the ability to specify a human task. This is the person who has administrative rights if a default activity administrator is not specified, or is specified but for this particular activity you want someone else with authority to view confidential information, you can specify that here. An event is an interface allowing information to come in and you can restrict who has the ability to send that information in as an event. The Publish activity is very similar to the AutoClean activity and would have the same options for restricting its use through a human task definition.

Human tasks in WSBPEL

- Tight integration of human tasks with late binding values in WSBPEL processes allows for a flexible approval process
- Only available if manager attribute is used in the user registry



Human tasks can also be used in BPEL for more complex situations such as multiple levels of approval. This can be done utilizing multiple human task activities and those late binding parameter values. In this example, the business process is initiated and enters a staff activity called FirstLineApproval. That activity will be assigned to the manager of an employee by user ID. The user ID that will be used is the person that started the business process, which is a late binding context value. For example, if you started this business process, your user ID would be passed in as a parameter. The staff plug-in provider would go out to the user registry and find your manager and assign that person a work item. When that work item is completed, you can check the results to see if they approved or denied the request. If it is approved, it will proceed to the second activity for approval from the second line manager. The same verb, Manager of Employee By User ID, will be used, however this time a different late binding value will be passed, specifying the owner that completed the FirstLineApproval staff activity. The SecondLineApproval would then be assigned to the manager of the manager who completed the FirstLineApproval staff activity, making the process very flexible. This chain could be used to traverse the hierarchy of employees for the entire organization, regardless of their reporting chain.

Human tasks in WSBPEL (cont.)

- Multiple approvals by different people in a group (“4-eyes”) can be implemented with human tasks with late bindings values in WSBPEL processes
- Multiple users can be excluded to support more “eyes”

Role

Potential owner

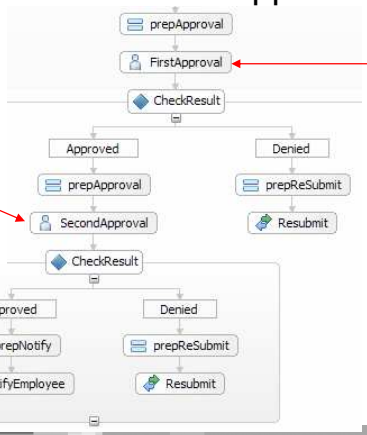
Verb

Group members without named users

Parameter

DocumentApprovers

%wf:activity(FirstApproval).owner%



Role

Potential owner

Verb

Group members

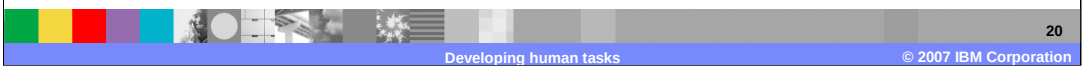
Parameter

DocumentApprovers

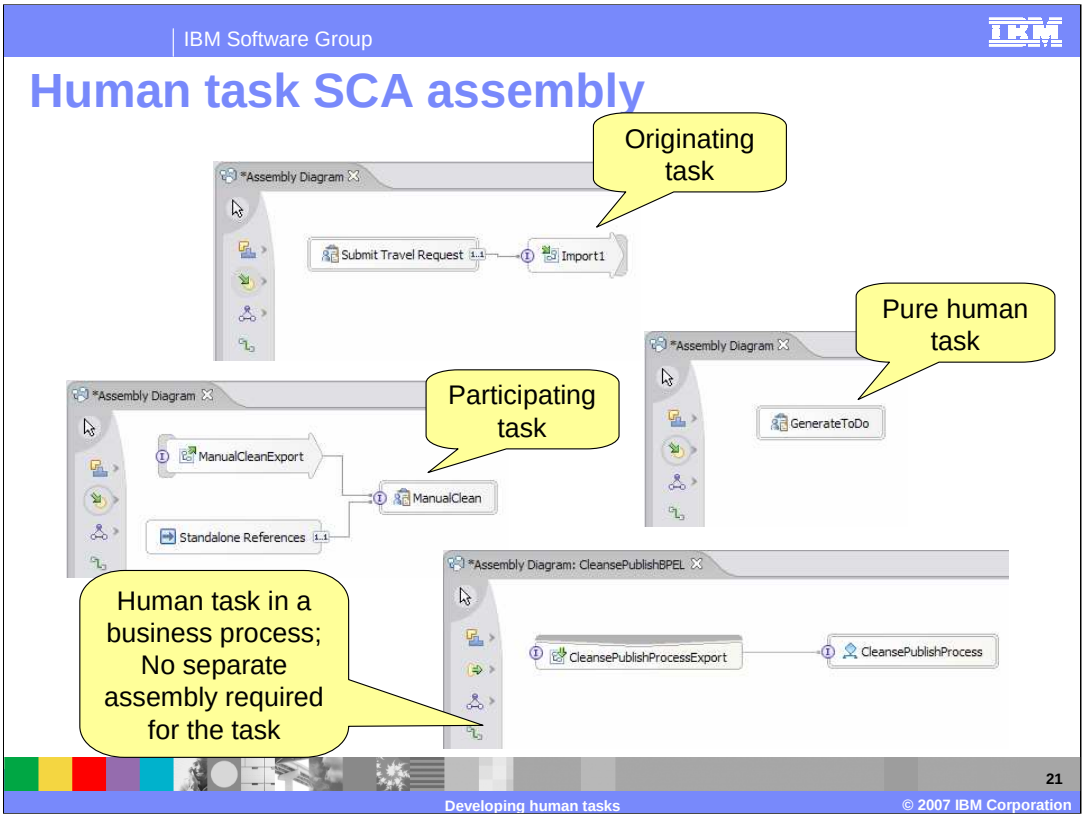
Another way you can use staff activities to do some more robust things is using the Group Members Without Named Users verb. This allows you to implement the “four eyes” concept, which is the ability for multiple users to approve a request while ensuring that the same user does not approve the same request twice. This could be very useful in situations such as insurance or banking applications. In the example shown, there are two staff activities. The first one is assigned to a group of users called Document Approvers. One person from that group will claim the work item and complete it. If they approve the item, it will proceed on to the next staff activity. This time the Group Members Without Named User verb will be used and the name of the first approver will be passed as a parameter, ensuring that the task is assigned to someone other than the first approver. You could extend this concept to follow a chain of approvers, so that each successive request is routed to a different approver. You could also use a while loop to cycle through the group until everyone had approved the request.

Section

Assembling human tasks



This section will provide information on how to assemble Human Tasks prior to deployment.

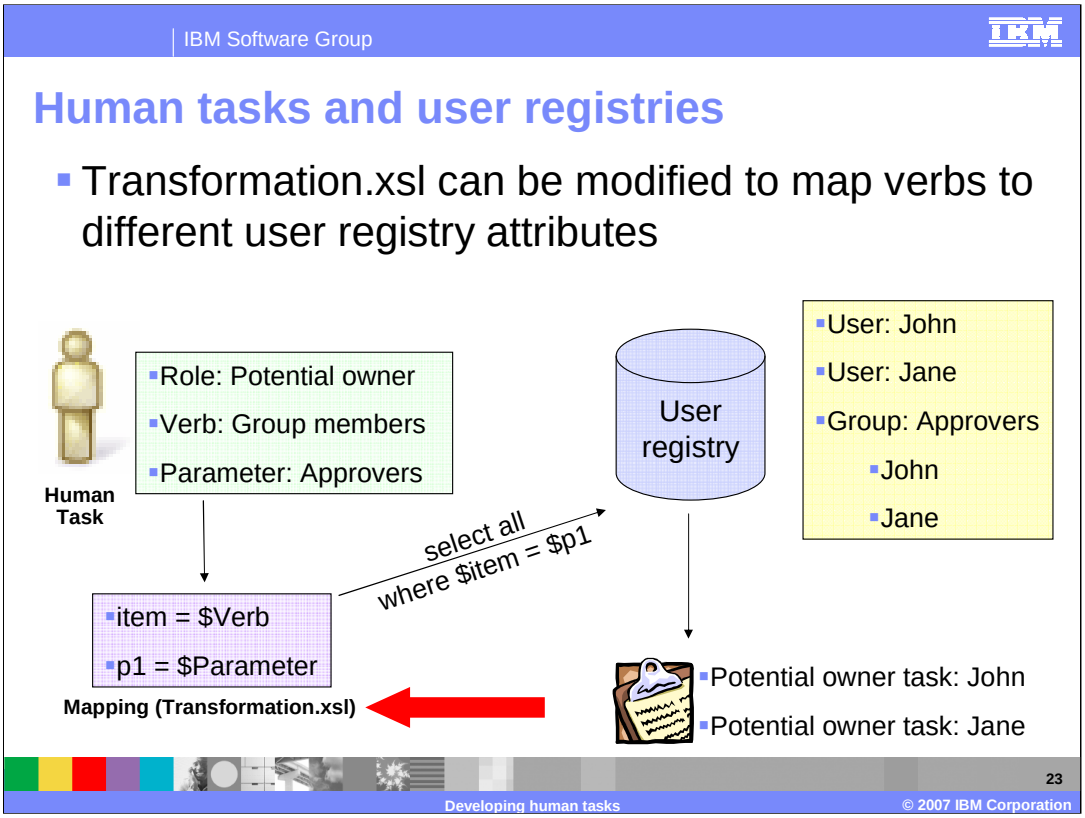


Because a human task in an SCA component, some assembly is required in order to make it available to other SCA components and make it into a deployable artifact for the process server. If you have an originating task, you are creating a work item for some type of machine. You will drop the task on to the assembly editor and wire it to the component you want to call. This could be a component within the same module or outside the module, requiring an import component. If there is a participating task, where some machine or service is going to assign a person work, you can wire that in directly if it is in the same module, or if it is a component in a different module, it can be brought in using an export component. If you will assign work from a Java resource such as a JSP or session bean, you can use a stand-alone reference and use Java SCA APIs. If you have a human to human task, there are no SCA components involved, but you must still add it to the assembly editor because the editor not only provides the means for establishing wiring and binding, but it is also the main mechanism for generating the appropriate deployment code to make it available as an application. Just drop the human task on the assembly editor. If you are using an inline task, there is no additional wiring that you need to do. The wiring that is being done is for partner links and the partner roles are within the BPEL process. These are for invoke activities and there is nothing for you need to wire for an inline task. If there is a participating task such as ManualClean, which is an SCA component because the export has been defined, you could wire the BPEL process to this participating task. However, you might want to avoid this as you would lose the tight integration capabilities, which allow you to use those late binding context values to set multiple approvals or use the “four eyes” concept. Because you introduce an SCA boundary, the context information is not passed between the BPEL process and what is actually an SCA component.

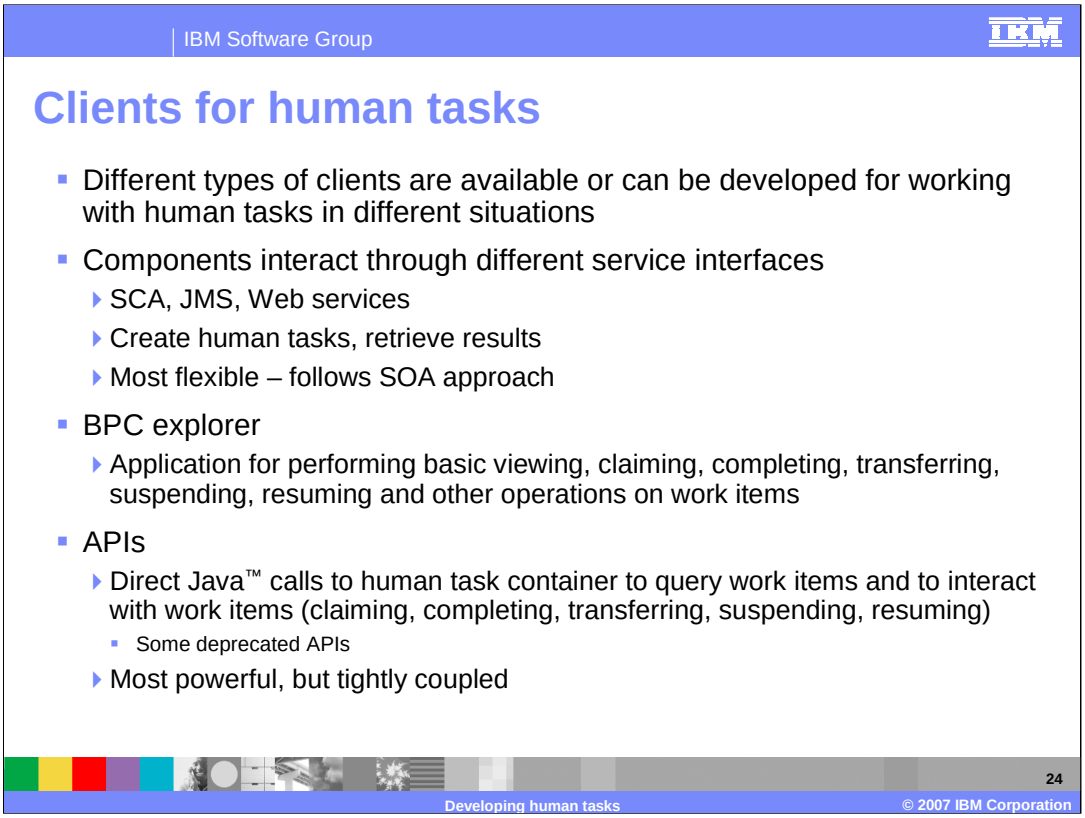
Section

Additional development

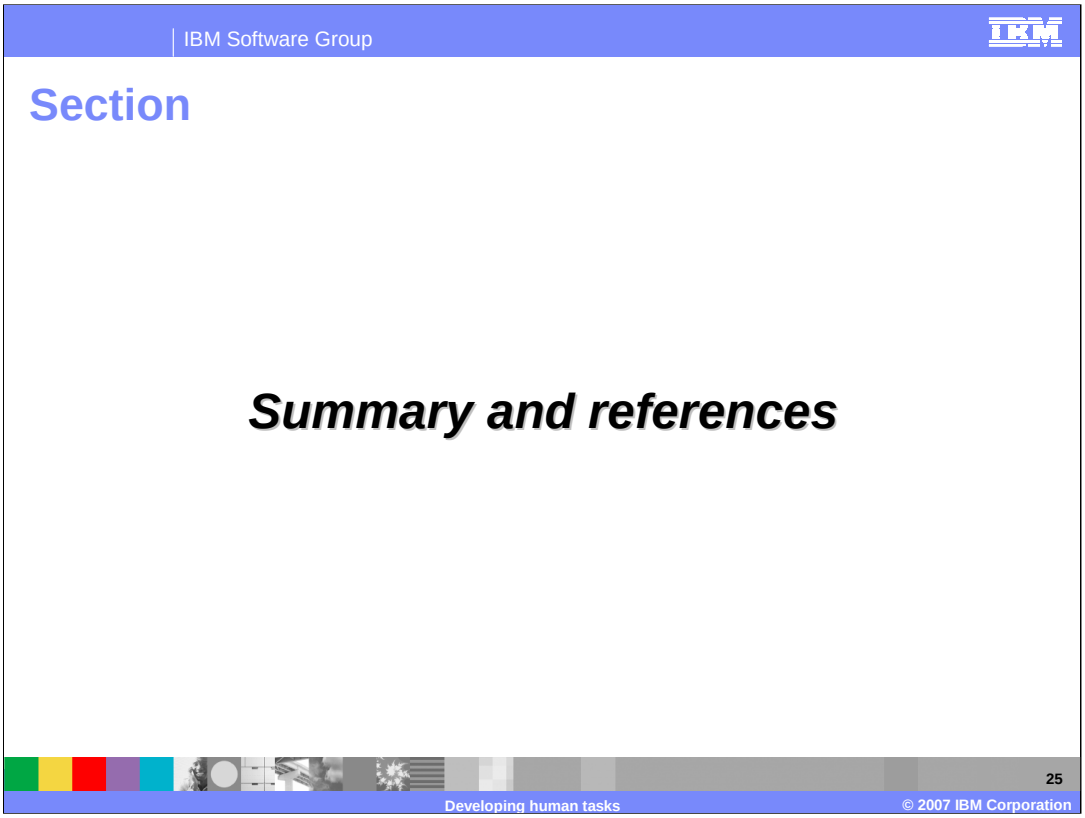
This section will provide information on additional development around human tasks.



Another area for development is around mapping the generic verb to the specific query that will be used to retrieve the user or group of users who will be assigned a work item with a specific permission level. The different verbs which are available in WebSphere Integration Developer are only supported in specific user registries as discussed earlier. If a verb is not available for a certain type of query which you would like to be performed to retrieve the users, you may modify the transformation.xsl file with the appropriate query which can utilize specific attributes. Parameters may be specified which are passed and used in the query as well. Modifying the transformation.xsl file is something that is typically done by someone who has a clear understanding of the schema of the user registry.



A final consideration for development around human tasks. You can use a variety of clients for working with participating human tasks. You can call the participating task using SCA, JMS, and Web Services when the task is exposed as an SCA component. With these interfaces, you have the ability to create human tasks and retrieve the results, which fits in directly with the SOA programming model. The BPC Explorer is a basic application provided with the product that allows you to view, claim, suspend, resume, transfer, and complete work items. Finally, there is a full set of APIs exposed on the Human Task Manager so you can create your own custom client that retrieves a set of work items based on specific attributes. To do this, you could use these APIs along with the query capability. These APIs have changed since V5.1 and some have been deprecated. These APIs provide a robust means of retrieving work items using a very tightly coupled Java implementation.



This section will provide a summary and references.

Summary

- Human tasks use service-oriented (SCA) design to easily integrate into business applications
- Tight integration with WSBPEL business process provides support for more robust scenario
- Enhanced capabilities for escalations, and notifications

Developing human tasks is very easy. Human task support fits tightly within the service oriented architecture, is built upon SCA, and utilizes business objects. You can make human tasks available as SCA components and very quickly and easily wire those into your integration applications in a flexible manner. There is still tight integration within the BPEL processes, allowing you to do things such as the “four eyes” concept and multiple levels of approval. There are new capabilities related to escalation and notifications as well.

References

- WebSphere Application Server Enterprise Process Choreographer: Staff resolution architecture
  - http://www-106.ibm.com/developerworks/websphere/library/techarticles/wasid/WPC_StaffArch/WPC_StaffArch.html
- WebSphere Application Server Enterprise Process Choreographer: Programming model for staff resolution
  - http://www-106.ibm.com/developerworks/websphere/library/techarticles/wasid/WPC_StaffModel/WPC_StaffModel.html
- WebSphere Application Server Enterprise Process Choreographer: Staff resolution parameter reference
  - http://www-106.ibm.com/developerworks/websphere/library/techarticles/wasid/WPCStaffReference.html

Late binding values for human tasks

- Values typically used for assignment of escalations
  - Names must be specified as they appear on the escalation

Task Value	Description
%htm:task.originator%	User who created a non-human task
%htm:task.administrators%	User(s) with the permission to administer tasks
%htm:task.readers%	User(s) with permission to read the tasks
%htm:task.potentialOwners%	User(s) who can claim a task
%htm:task.potentialStarters%	User(s) who can start (create) a human task
%htm:task.owner%	User who has claimed the task
%htm:task.starter%	Users who started a human task
%htm:task.editors%	User(s) who can edit the current task
%htm:task.escalation(<escalation name>).receivers%	User(s) who received a previous escalation

Late binding values for business processes

- Values can be based on individuals who completed previous activities or individuals assigned permissions for the overall business process
  - Activity names must be specified as they appear in the business process editor

Process Value	Description
%wf:process.starter%	User who started the current instance
%wf:process.readers%	User(s) with the permission to read instances
%wf:process.administrators%	User(s) with permission to administer instances
%wf:activity(<activity name>).potentialOwners%	User(s) who could claim a previous Staff activity
%wf:activity(<activity name>).owner%	User who claimed a previous Staff activity
%wf:activity(<activity name>).editors%	User(s) who could edit a previous Staff activity
%wf:activity(<activity name>).readers%	Users who could read a previous Staff activity

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback

You can help improve the quality of IBM Education Assistant content by providing feedback

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM WebSphere

Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.