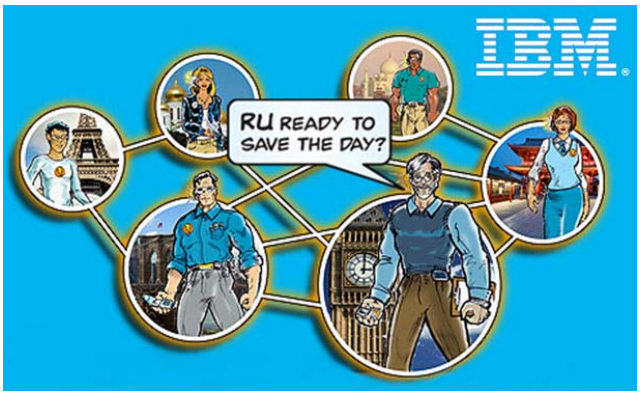
# Introduction to Application Lifecycle Management on ClearQuest

*Lab #1: Responding to a Request*



# Table of Contents





# Lab #1 Responding to a Request

## 1 Overview

This lab exemplifies how a team responds to a new request. Triaging and acting on requests are the primary activities in the CQALM system. The purpose of this lab is to provide an 'end-user' view of the system, and as such, multiple roles come into play in this exercise.

### Roles Involved in this exercise

The roles involved in this reference scenario are pictured in Figure 1.

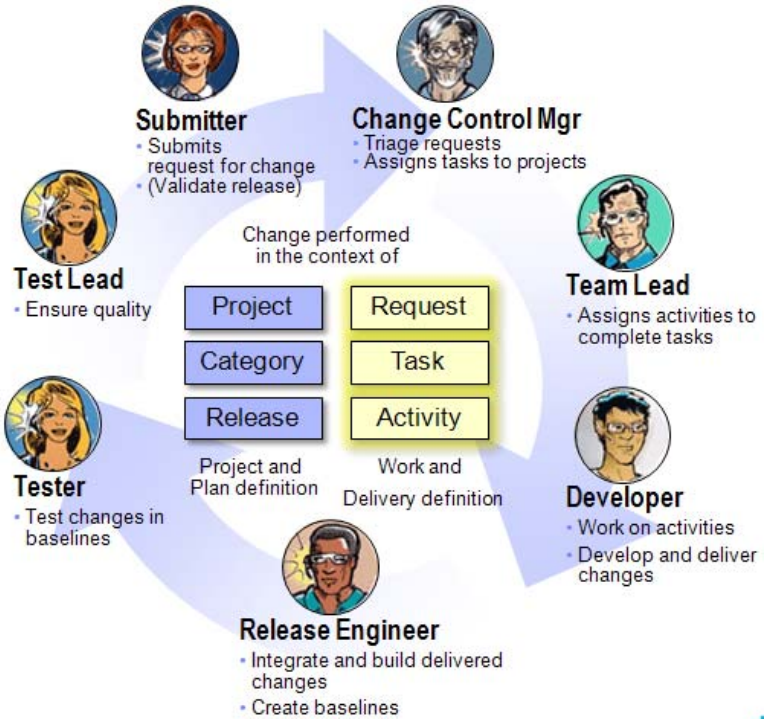



*Figure 1. Roles used the "Triage" scenario*

In this lab, you will play the role of the:

- Submitter. Any one on the team can submit a request.
- Change Control Manager. Triages incoming requests, assigns tasks.
- Team Lead. Reviews Tasks, assigns activities.
- Developer. Completes an activity.
- Release Engineer. Conducts the build.
- Tester. Completes an activity.
- Test Lead. Reviews and Completes the Task.

> *Rather than login and logout repeatedly, the 'admin' user will be used for all roles.*

### The Flow of events for this exercise

The flow of this lab is illustrated in Figure 2.

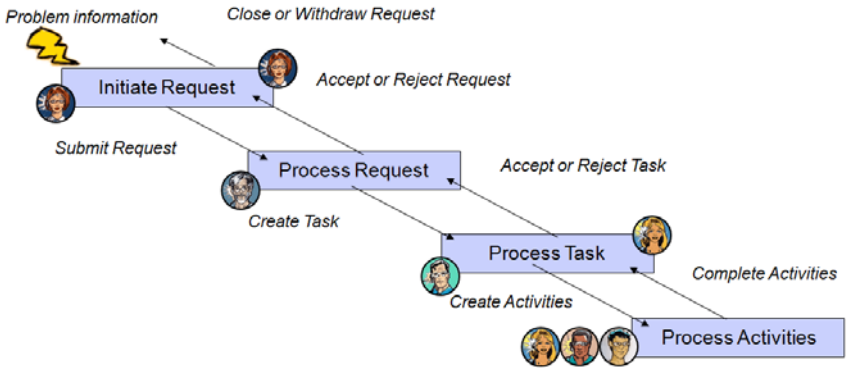



*Figure 2. The Flow of Events for this lab*

- An *ALMRequest* is submitted by the "submitter" who represents anyone on the team.
- Someone acting as the Change Control Manager will triage the request and determine that it will be addressed in the current project. An *ALMTask* record is created to represent the commitment to fulfill the request for a project. The ALMTask is assigned to the "Team Lead."
- The Team Lead creates and assigns *ALMActivity* records to complete the work associated with the task.
- The Developer completes the *Implement ALMActivity*.
- The Release Engineer creates an *ALMBaseline* and *BTBuild* record to represent the build.
- The Tester completes the *Test ALMActivity*.
- The Test Lead processes the *ALMTask* to determine its completeness. The Task is *Completed*.
- The Submitter reviews the completeness of the request. The *ALMRequest* is *Accepted*.

> *A set of queries is provided in the example database provided for this lab. These queries are not shipped with the product.*

## 2 Submit a Request

The scenario starts with the submission of a new *ALMRequest*. In this exercise you will play the role of the "Submitter" to learn about the *ALMRequest* record type.


*Figure 3. The Scenario begins with the Submission of an ALMRequest*

### Login to ClearQuest

___1. Launch the IBM Rational ClearQuest Eclipse User Interface

___a Locate the ClearQuest icon on your desktop. 

___b Double click on the icon to Launch IBM ClearQuest

___2. Login to IBM Rational ClearQuest

___a From the ClearQuest **Navigator** tab, click on the **Repository** connection icon. The database is set up to automatically login as 'admin' 


> *For simplicity, you will remain logged in as the 'admin' throughout this tutorial.*

___b *Alternatively* – if the automatic login does not work, use the following steps

- Click the arrow next to the **Repository** icon.
- On the list of repository connections choose 'admin,CQALM@CQALM'


- The Connection dialog appears with the user name and password field blank. Click **OK** on the Connection dialog. ('admin' does not have a password.)

### Set the Default Project

The ALMProject has a SetDefault Utility action. Running this action sets the project context for subsequent work. Any subsequent records that are created that require a Project field will have this project set as the default value. While not required to be used, the SetDefault action will save mouse clicks when a series of related records are being created.

___**1. Public Queries->ALM->General->All Projects**

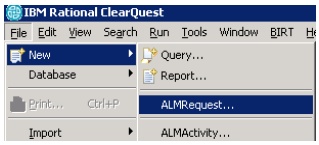
___2. Select `Triage Sample Project`

___3. Select the **Utilities > SetDefault...** action.


### Submit a Request

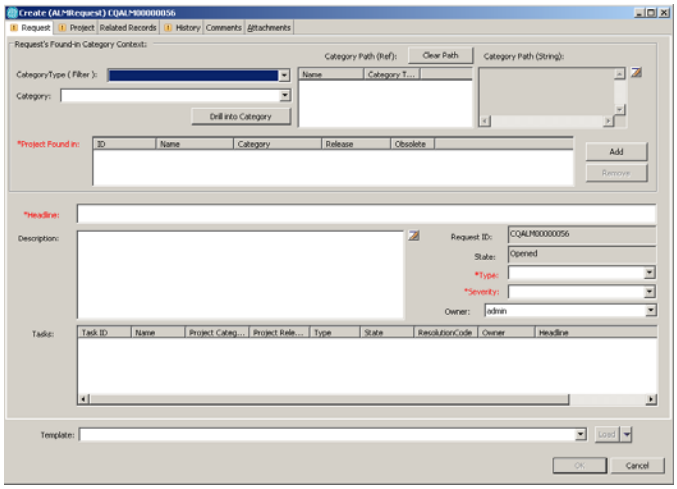
___1. Create an **ALMRequest** record.

___a Choose **File > New >ALMRequest**...




___b Alternative: click the **New** button.  The ALMRequest is the default record type.

___c The ALMRequest record opens



___2. Explore the required fields.

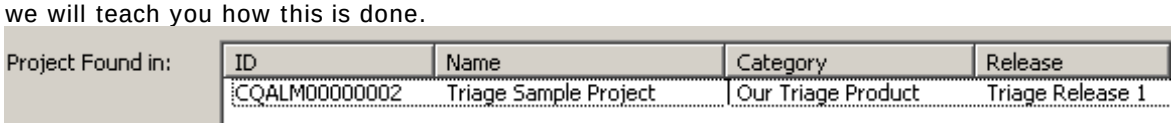
___a Click on the **Type** drop-down. Note that it is empty.

___b Click on the **Severity** drop-down. It is pre-populated. Do not select one.

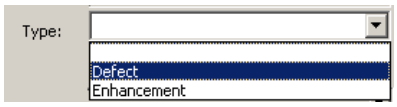
___c Click on the **History** tab. Note that **Security Policy** is required, and that the drop-down is pre-populated with choices. Do not select one.

___3. Click on the **Category** drop-down. Select 'Our Triage Product'


Note that the **Project Found in** field is automatically set to a project. In a subsequent lab we will teach you how this is done.




___4. Click on the **Type** drop-down.




___a Notice that the drop-down is populated. The types are inherited from the project that is defined in the **Project Found in** field. Setting a project establishes the list of choices on the **Type** drop down.

___b Choose Defect from the list of choices.

___5. Click on the **Severity** drop-down and select 'Sev 1 - Critical' from the list of choices.

___6. Type a meaningful description in the **Headline** field, such as "Incorrect logo used in the Login screen"

At this point all required fields are completed.

___7. Click on the **History** tab, and note that the **Security Policy** is now set.


> *Security Policy is inherited from Project. When you set the 'Project Found in' field, the Security Policy field is automatically set. This will be discussed further in another lab.*

You can explore the record, but no additional data is needed.

___8. Click **OK** to submit the request.

Congratulations! You have completed the first exercise by submitting an ALMRequest.

# 3 Triage Request & Create Task

In this exercise you will play the role of the "Change Control Manager" to learn about triaging the **ALMRequest** record and how to create an **ALMTask** with the **CreateTask** utility.

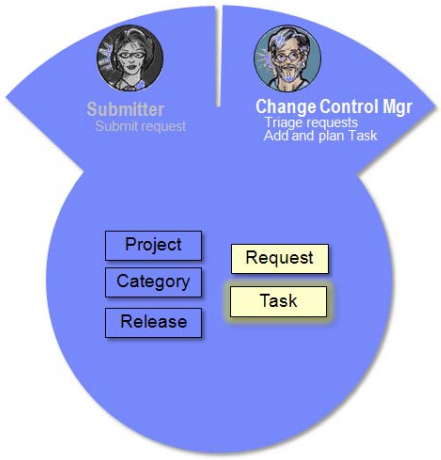


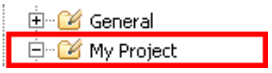

*Figure 4. The Change Control Manager triages the ALMRequest, creates an ALMTask*

> *Normally this task is performed by a Change Control Manager, Project Lead, or Development lead. For simplicity, remain logged in as "admin."*

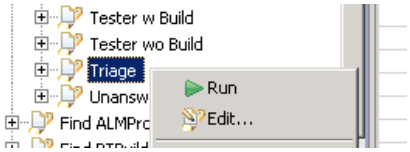
### Locate Requests in need of Triage

___1. Run the **Triage** query

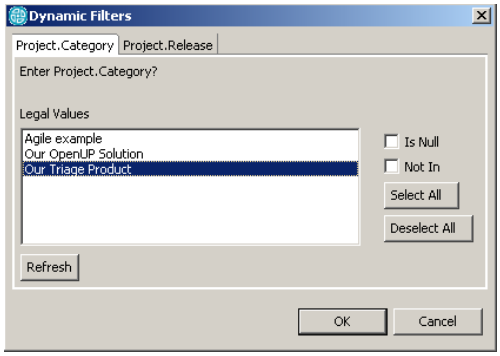
___c In the Navigator, open the **ALM** folder. 


___d Open the "**My Project**" Folder. 

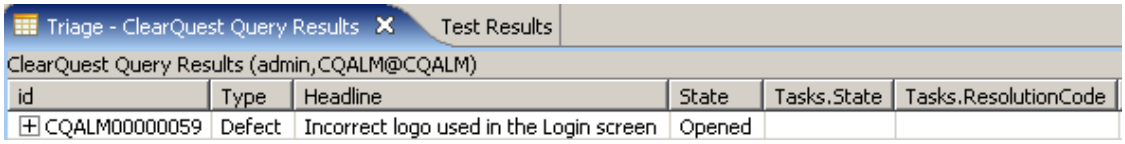

___e **Run** the **Triage** query by double clicking on it, or right click and choose Run from the context menu.



___f On the **Project.Category** tab, choose '`Our Triage project`' and click **OK**.




___2. Select the Request in the results list.




The record details are visible in the Eclipse window. If not, double click on the record to open the record details window.

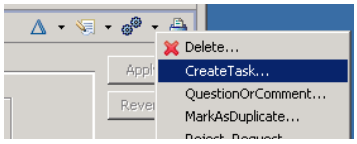
### Commit to the Request by Creating a Task

The act of 'committing' to a request is performed by creating an **ALMTask** that references the **ALMRequest** record. The ALMTask may be completed in the same project that the ALMRequest was 'found in' or it may be completed in a different project. Additionally, there may be multiple projects with ALMTask records that complete the same ALMRequest. For this exercise, the Project Found in for the ALMRequest is the same project that the ALMTask is assigned to.

> *The person conducting the Triage would review the content and determine if there is enough information to work with. For this lab we assume there is.*

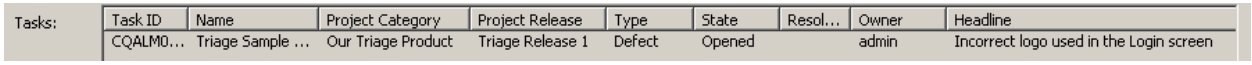
___3. On the *ALMRequest* record, go to the **Utility** drop down and click the down arrow to open the menu. 

___4. Choose "**CreateTask...**"




___5. The utility runs and presents a dialog when complete. Click **OK** to dismiss the dialog box.

___6. On the ALMRequest record, notice that an ALMTask appears in the **Tasks** field. The **Project.Category**, **Project.Release** and **Headline** of the ALMTask default to those of the initiating request. If needed, these can be changed by opening the record and modifying the fields. The CQALM solution automatically creates a trace-ability reference between the request for work, and the assignment to complete the work.

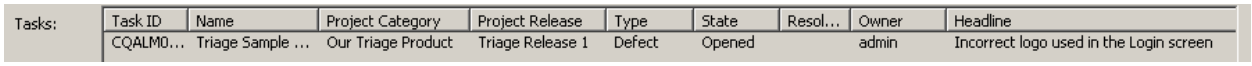


| Tasks: | Task ID | Name | Project Category | Project Release | Type | State | Resol... | Owner | Headline |
|---|---|---|---|---|---|---|---|---|---|
| | CQALM0... | Triage Sample ... | Our Triage Product | Triage Release 1 | Defect | Opened | | admin | Incorrect logo used in the Login screen |

*Notice that the ALMTask is of type "Defect," and that a default owner is assigned (admin). This is configured specifically for this project. In a subsequent lab we will review how to configure these defaults on a project-by-project basis.*

## Assign the Task

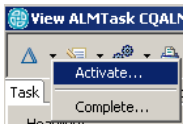
___1. From the ALMRequest record, double click on the ALMTask record listed in the **Tasks** field.



| Tasks: | Task ID | Name | Project Category | Project Release | Type | State | Resol... | Owner | Headline |
|---|---|---|---|---|---|---|---|---|---|
| | CQALM0... | Triage Sample ... | Our Triage Product | Triage Release 1 | Defect | Opened | | admin | Incorrect logo used in the Login screen |

The ALMTask record opens in a new window. Notice the trace-ability from the ALMRequest to the ALMTask & how easily you can find the tasks that are related to this request.

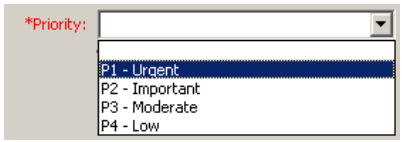
___2. On the Task record, click the arrow on the **Change State** button and choose **Activate**.




Notice the **Priority** and **Owner** fields are now required.

___3. Set the **Owner** to 'admin'


___4. Set the **Priority** to P1-Urgent




Leave all other fields unmodified, including the **Project Assigned To**: and **Headline** fields

___5. Click **OK** to save your changes. The ALMTask dialog is dismissed.

*The Task field is a list box that can contain more than one Task. In fact, a common scenario is to have a single request that is fulfilled by two different projects. In this example, two tasks would appear in the Tasks list, with each Task assigned to a different project.*

# 4 Assign Activities

Now that the ALMTask is related to the ALMRequest, it is time to add ALMActivities that will cumulatively complete the task. In this exercise you will play the role of the "Dev Lead" to learn about adding ALMActivities to an ALMTask with the **CreateActivity** utility.

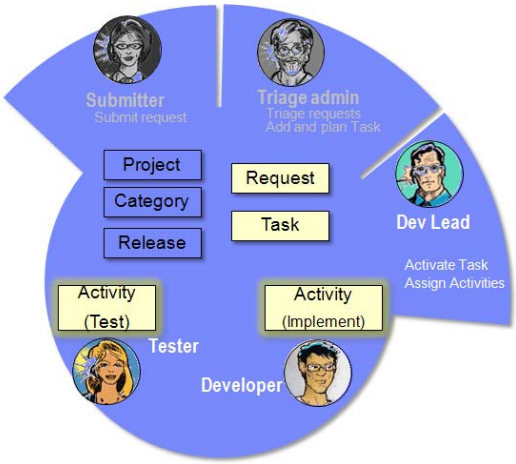


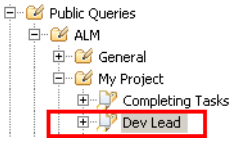

*Figure 5. The Dev Lead role assigns ALMActivities*

Tasks typically require more than one activity to complete. Many times different team members are needed to perform each activity. In this section you add activities to the task and assign the activities to different team members.

> *Normally this task is performed by a Change Control Manager, Project Lead, or Development lead. For simplicity, remain logged in as "admin."*

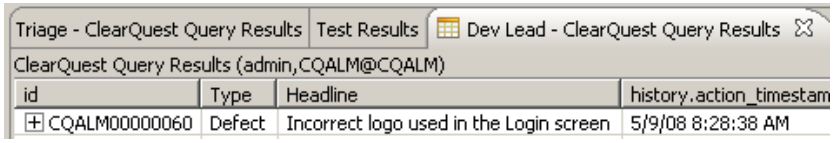
### Create Activities

___1. Run the Dev Lead query. (**Public Queries > My Project > Dev Lead**)

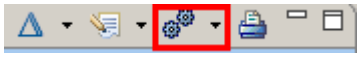



___2. Choose the "`Our Triage Product`" on the Dynamic Filters dialog and run the query.

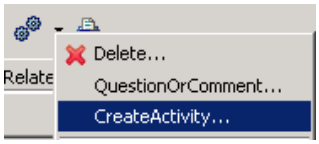
___3. The ALMTask that was just triaged appears in the result set. Select or double click to open it.




___4. On the ALMTask record, go to the **Utility** drop down and click the down **arrow**.

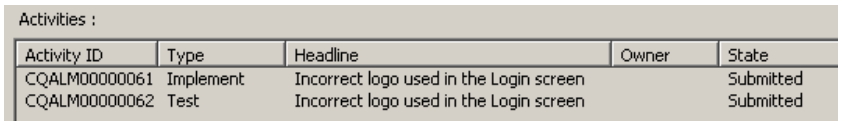


___5. Choose the **CreateActivity…** utility.




___6. The utility runs and presents a dialog when complete. Click **OK** to dismiss the dialog box.

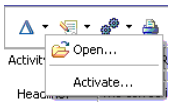

___7. On the ALMTask record, notice the **Activities** field is now populated. The headline of the ALMActivity is the same as the initiating request. The CQALM solution automatically creates a trace-ability reference between the request for work, the task, and the assignment to complete the work.




> *Notice that there are two Activities of different types. This is configured specifically for this project. In a subsequent lab we will review how to configure the "Create Activity" utility on a project-by-project basis.*

___8. In the **Activities** list, double click on the **ALMActivity** of type "**Implement**" The dialog for the record opens.

___9. Perform the '**Open'** action.




___a Notice the names on the Owner drop-down list. The field has been filtered based on the roles expected to perform this type of activity. Set the **owner** to 'admin'


> *Normally this would be assigned to a developer, such as Dierdre in this example.*

___b Click **OK**. The dialog is dismissed.

___10. Repeat these steps for the remaining activity, assigning the Test activity to 'admin'.

> *Normally the Test activity would be assigned to a tester, such as Tanuj in this example.*

# 5 Complete the Developer Activity

In this exercise you will play the role of the "*Developer*" completing the **ALMActivity**.

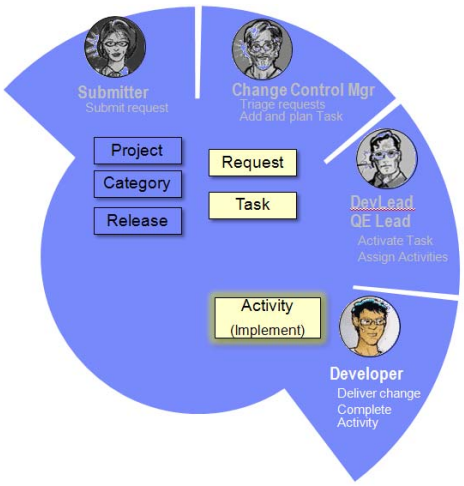


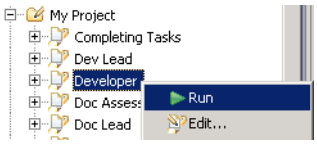

*Figure 6. The Developer completes the ALMActivity*

The completion of activities varies on type. For example an analyst may work with RequisitePro and link the requirements back to ClearQuest. A developer will work in ClearCase or Rational Team concert to implement the solution. The test team can plan for and write the test cases, but will wait until after a build is available to run the tests.

The common denominator in each of these is that the user claims the ALMActivity **Complete**.

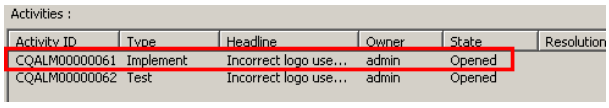
### Complete the Activity

___1. Run the Query for the **Developer** in the **My Project** folder.

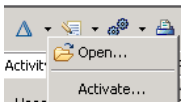



___2. The ALMTask appears in the results list.

___3. On the Task record, locate the Implement activity and open the record by double clicking it.




___4. **Activate** the Activity to indicate that you are working on it.



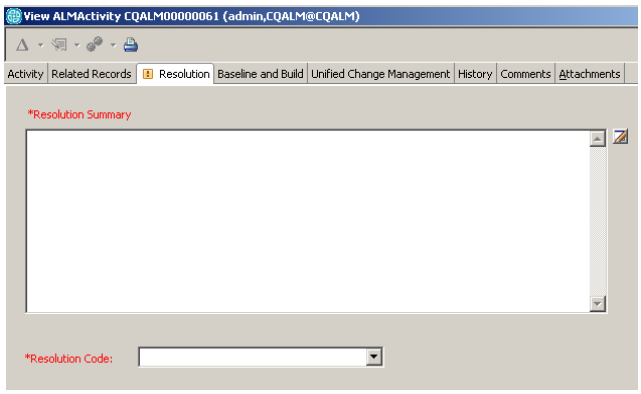
> *Normally development work would occur to complete the work at this time. For UCM users, this work could be completed using the UCM workflow.*

___5. Type something in the description field. Apply your changes by clicking the **Apply** button.

___6. Perform the **Complete** action on the activity.


___7. Note the **Resolution** tab indicates there is a required field. 

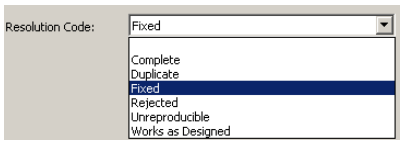

___8. Click on the **Resolution** tab.




___a Provide a **Resolution Summary**, such as "`The Logo is corrected`"

___b Select the **Resolution Code** dropdown.

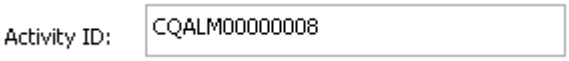
___c Choose '`Fixed`'.




___d Click **OK**.

___9. Write down the **Activity ID** here: _____

You will need this in the upcoming lesson. The ID can be found in title of the dialog, or on the Activity tab for the record:



# 6 Create Baseline & Build Records

Now you will play the role of the Release engineer to learn about the **ALMBaseline** and **BTBuild** Records.

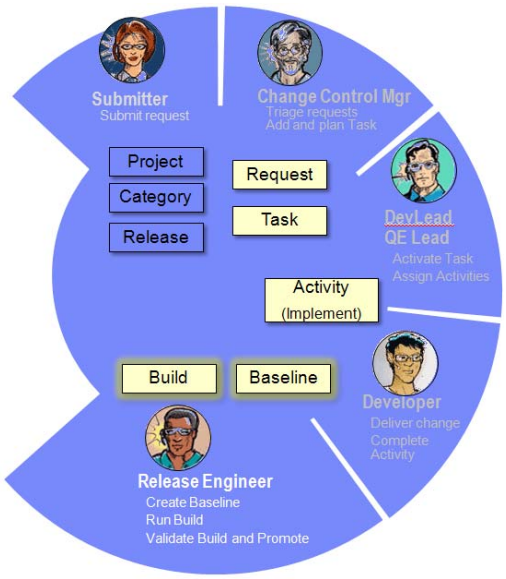



*Figure 7. The Release Engineer conducts the build*

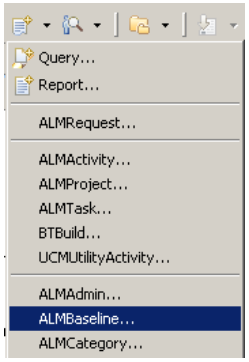
The ALMBaseline record is used to track the activities that are delivered between two builds. The ALMBaseline record can be used to align with UCMBaselines. This provides a deeper level of traceability between UCM and ClearQuest.

The BTBuild record is used to track the status of the build. A reference to the ALMBaseline record is created to establish a link between the build and the activities delivered in that build.

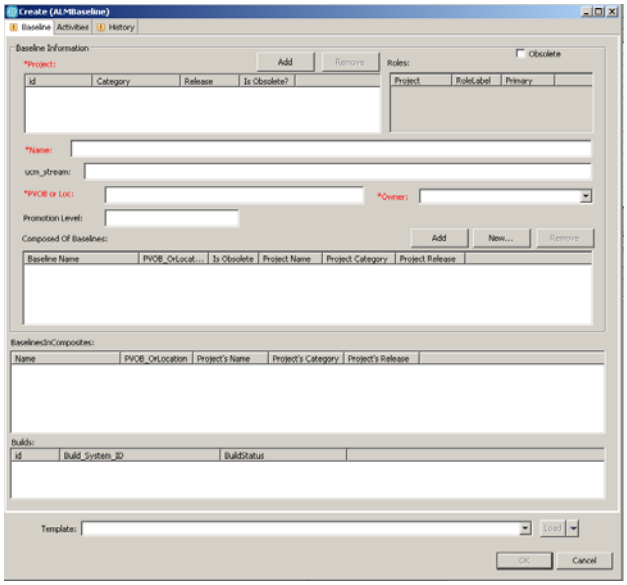
> *Normally these records would be created as part of an automated build process. For the purposes of the lesson we will manually create them so you can learn about each record. A PERL script is provided that will automatically create the ALMBaseline record. The script is called create_baseline_record.pl and is located in the pkg_util folder of the ALMWork package. (C:\Program Files\Rational\ClearQuest\packages\ALMWork\0.4\pkg_util)*

### Manually Create a Baseline record

**___1. File > New > ALM Baseline**



___2. The **ALMBaseline** Record dialog opens

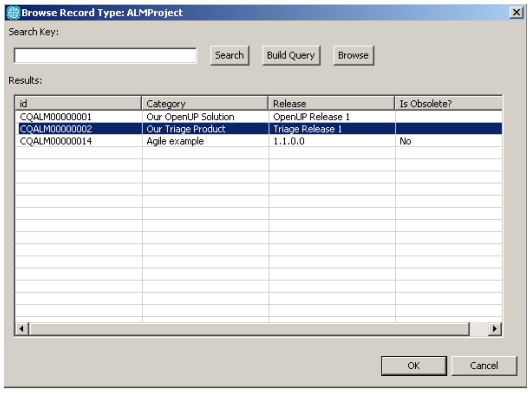



___3. Associate the Baseline record with the Triage Sample Project

___a On the ALMBaseline dialog, click the **Add project** button.

___b The Browse record type dialog appears.

___c Click **Search**




___d Choose `CQALM0000002 / Our Triage Product / Triage Release 1` from the results list.

___e Click **OK.** The dialog is dismissed and you should see the ALMBaseline record with the project field set.

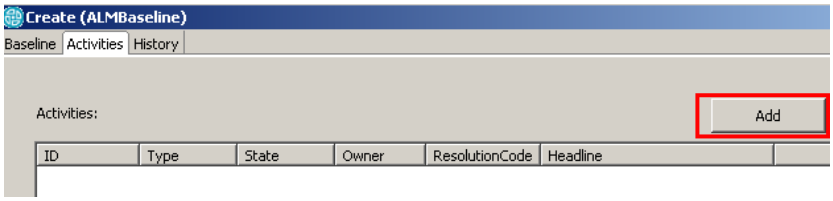
___4. In the required **Name** field, type "Updated UI"

___5. In the **PVOB or Loc:** required field, provide a location of where the source code is stored such as C:\Sources.

> *The location is the directory where the sources are. When using UCM this is the PVOB. For non-UCM users it is the directory where the code has been checked in. This creates a traceability link between the ClearQuest record and the source code.*

___6. Click on the **Activities** tab.

___a Click the **Add** button

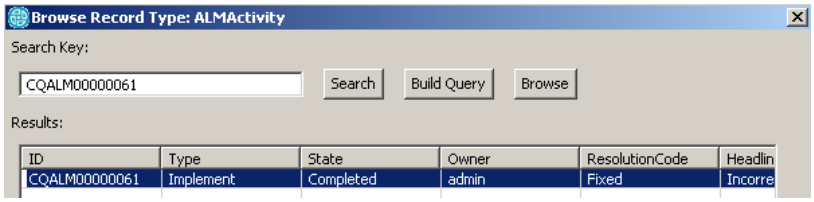



___b Type the Activity ID copied from the previous lesson in the **Search Key** field

___c Click **Search**.

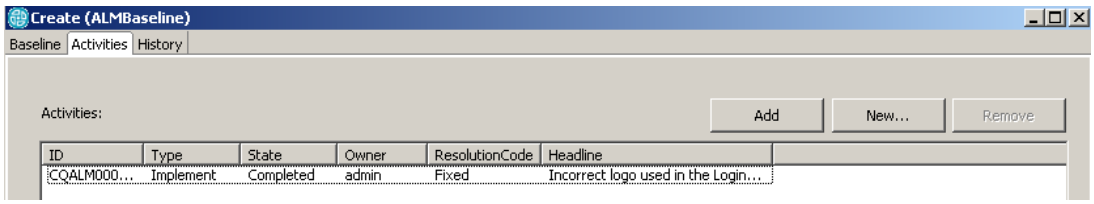
___d The activity appears in the result set.

___e Select the activity.




___f Click **OK**. The dialog is dismissed.

The activity appears in the list on the **Activity** tab of the ALMBaseline record.




___7. Click **OK** to save the Baseline Record

> *The create_baseline_record.pl PERL script will automatically find all activities since the last baseline and populate this list.*

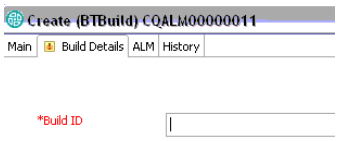
## Manually Create a Build Record

> *Normally this record would be created as part of an automated build process. For the purposes of the lesson we will manually create one. A PERL script is provided that will automatically create the BTBuild record. The script is called `create_build_record.pl` and is located in the pkg_util folder of the ALMWork package. (`C:\Program Files\Rational\ClearQuest\packages\ALMWork\0.4\pkg_util`)*

___1. **File > New > BT Build**

> *The BTBuild record comes from the Build Package that shipped with version 7.0. It has been extended to work with the ALM solution.*

___2. Click on the **Build Details** tab and provide a **Build ID.** Provide a meaningful name that you will recognize.



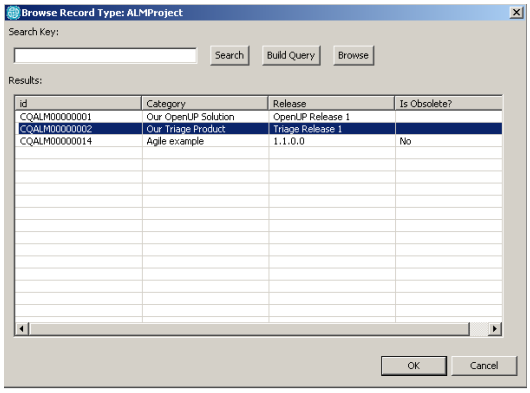

___3. Click on the **ALM** tab to associate the build with an **ALMProject**.

___a Resize the window to see the fields

___b Click the **Add** button next to the Project list.

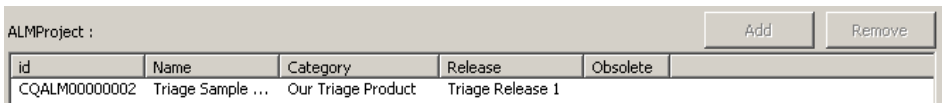
___c The Browse record type dialog appears.

___d Click **Search**

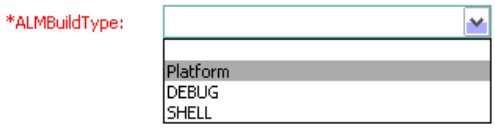



___e Choose `CQALM0000002 / Our Triage Product / Triage Release 1` from the result set.

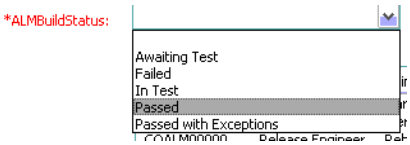
___f Click **OK.** The dialog is dismissed and the **ALMProject** field is set.



___4. Set the **ALMBuildType** to `Platform`

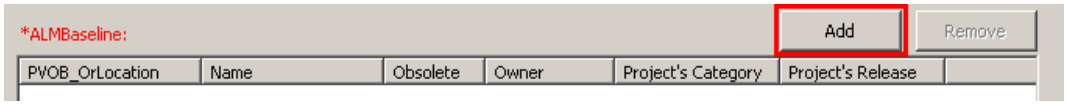



___5. Set the **ALMBuildStatus** to `Passed`.




___6. Associate the **ALMBaseline** record

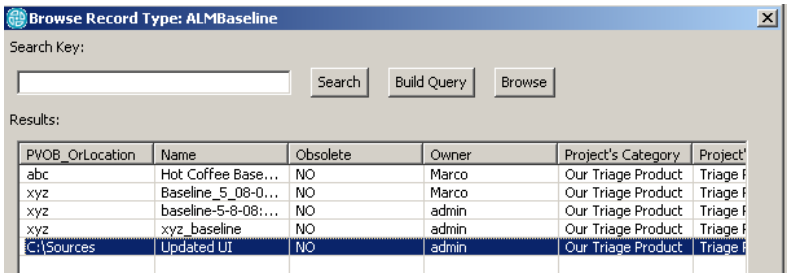
___g Click the **Add** button on the **ALMBaseline** field




___7. The Browse Record Type dialog opens.

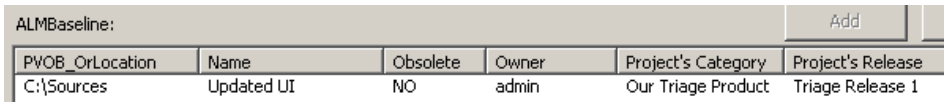
___a Click the **Search** button

___b Choose the Baseline record you just created.

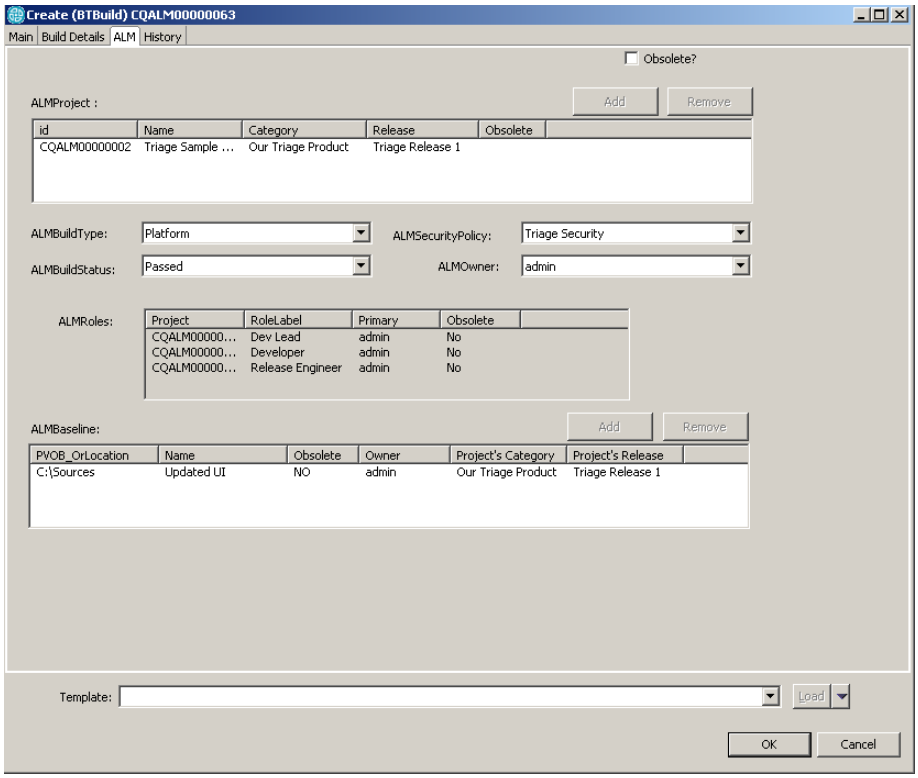



___c Click **OK,** which dismisses this dialog.

___8. The Baseline record appears in the list for the ALMBaseline field on the BTBuild record.



___9. The BTBuild record now has all of the information to proceed.




___10. Click **OK** to submit the record.

Congratulations! You have just created an ALMBaseline and BTBuild record for your project. These records create traceability links between the activities delivered, the source code and the build status. In this exercise the records were manually created to teach about their purpose and fields. It is expected that these would be created as part of the build process. PERL scripts are provided to help automate the creation of these two records.

# 7 Complete the Test Activity

In this exercise you play the role of a tester who must confirm that the defect is fixed.

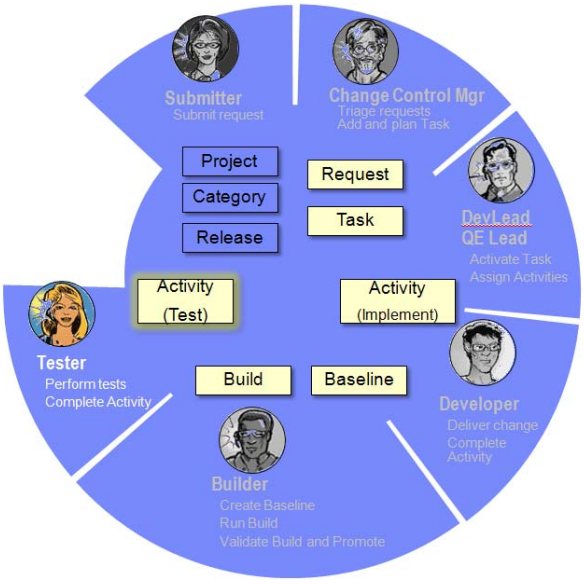
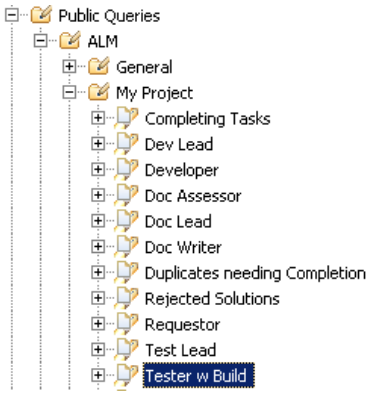



*Figure 8. The Tester completes the activity.*

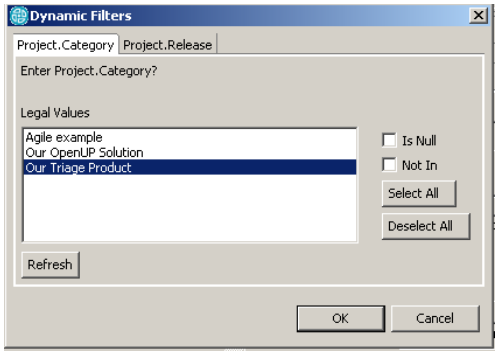
After the Build is complete the test team will conduct tests & report results. For now, assume the testing is completed & no defects were found.

### Locate the Task

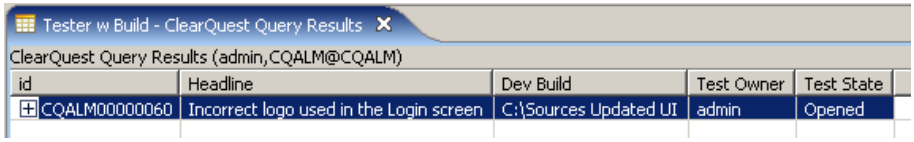
___1. In the ClearQuest Navigator, run the "**Tester w Build**" query (**Public Queries > ALM > My Project >Tester w Build**)



___d The Dynamic Filter dialog appears. Choose **`Our Triage Product`** and Click **OK**




___2. The task appears in the result set.



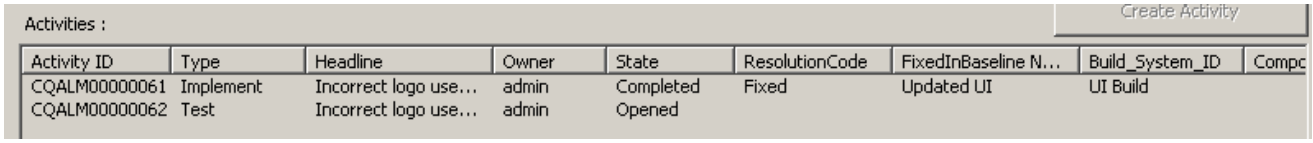

___3. Click to view the record details. Notice the **Activities** list.

___a The Implement activity is in the "Completed" **State**.

___b The **ResolutionCode** is set to Fixed

___c The **FixedInBaseline** shows the name of the Baseline record you just created.

___d The **Build_System_ID** is the name that you just provided on the Build record.

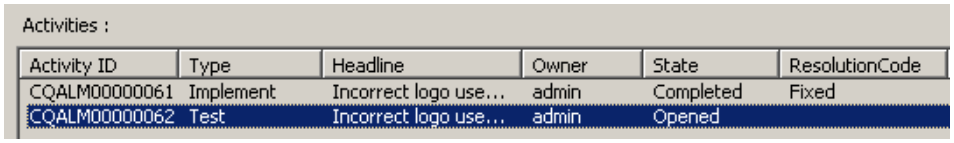



The Tester now knows that what Baseline and Build contains the fix to the Defect is fixed.

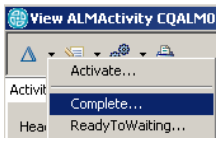
*The tester deploys the build to a test lab & conducts the testing.*

### Complete the Activity

___1. From the Task record, double click on the “Test” activity in the Activities list.

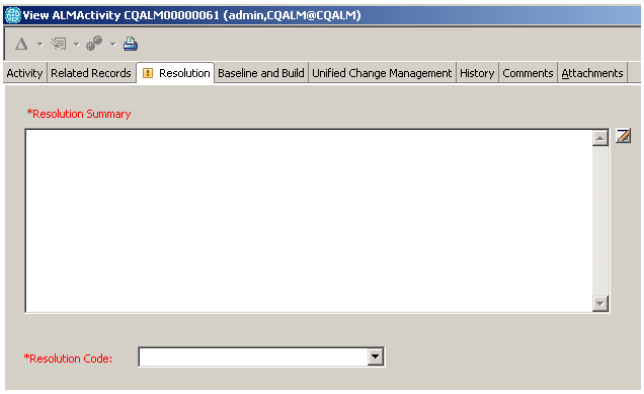



___2. The Activity dialog appears. Perform the **Complete** action on the Activity



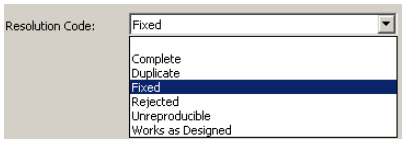

___3. Note the **Resolution Tab** indicates there is a required field. 


___4. Click on the Resolution Tab.




___a Provide a **Resolution Summary**, such as “`The Logo is corrected`”

___b Click the **Resolution Code** dropdown and choose “`Fixed`”.




___5. Click **OK** to complete the activity.

The testers work is now complete. You have now completed all activities for this task.

# 8 Complete Task

You will now play the role of Test Lead whose job is to review all outstanding Tasks and confirm that the work is satisfactory. In this exercise you will review all **ALMActivities** for a single **ALMTask** and use the **Complete** action to transition the state of the **ALMTask**.

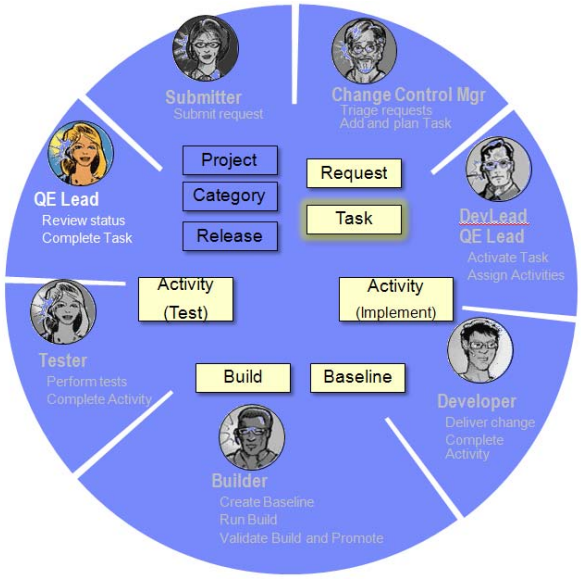
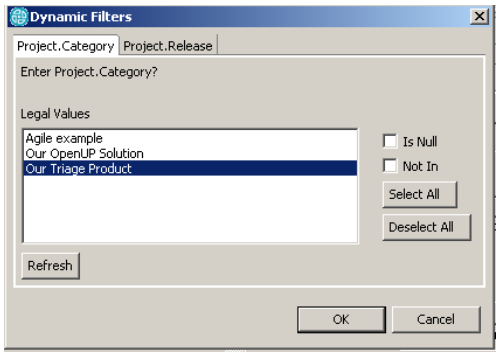



*Figure 9. The Test lead completes the ALMTask*

### Locate Task

___1. Run **Public Queries > ALM > My Project > Completing Tasks** query.

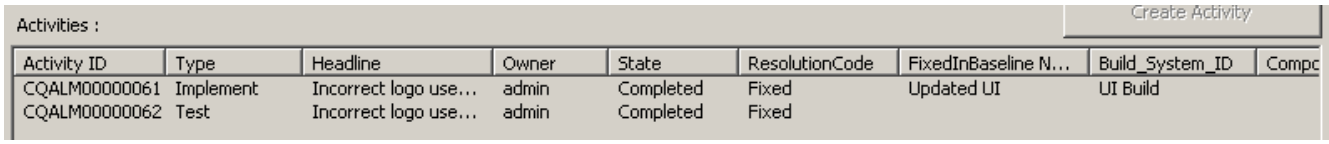

___2. The Dynamic Filter dialog appears. Choose **Our Triage Product** and click **OK**.




___3. The Task appears in the result set. Select the Task, or double click to open the dialog.

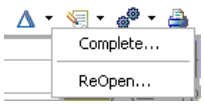
## Complete Task

___1. On the Task record, note the **Activities** field. Review the state and resolution code for the listed Activities.



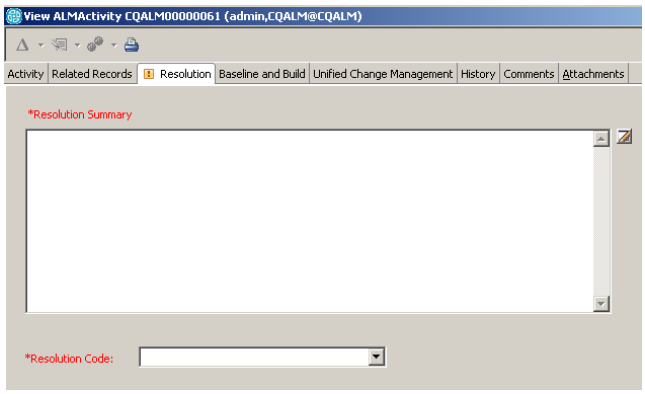

You can double click on each Activity in the list to review the content of the record

___2. If everything is satisfactory, perform the "**Complete**" action on the Task



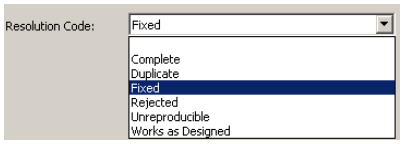

___3. Note the **Resolution Tab** indicates there is a required field. 


___4. Click on the Resolution Tab.




___a Provide a **Resolution Summary**, such as "`The Logo is corrected`"

___b Select the **Resolution Code** dropdown and choose "`Fixed`"




___5. Click **OK** to complete the Task.

Congratulations! You have now completed the Task for the project.

# 9 Complete Request

You will now return to the role of the “submitter.” In this exercise you review the work that was completed for your request.

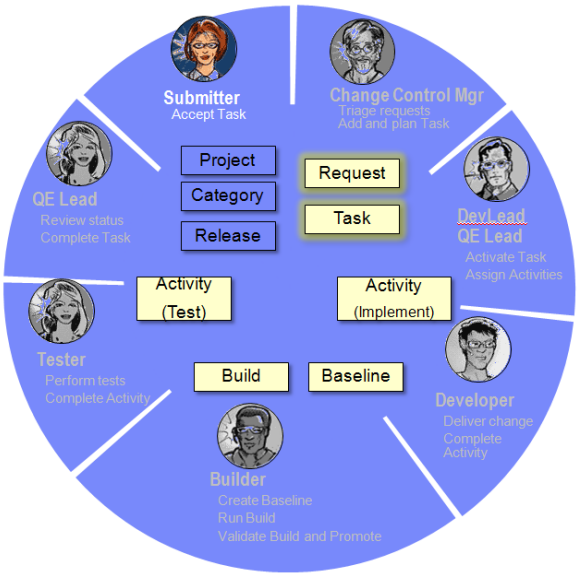


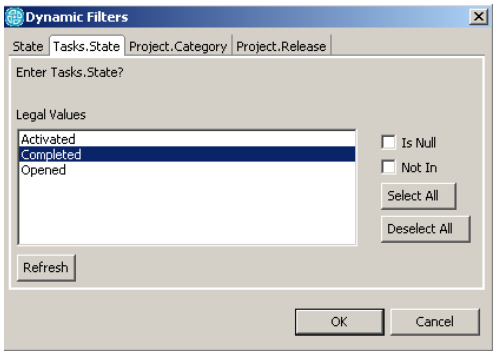

*Figure 10. The submitter reviews the completed task.*

### Locate Request

___1. Run the “Requestor” Query in the My Project folder.

___2. The Dynamic Filters dialog appears.

___a Click on the **Tasks.State** Tab




___b Select `Completed`

___c Click **OK**

___3. Select the Request in the result set. (Or double click to open the dialog)

### Complete the Request

___**1.** With the Request record visible, review the **State** and **Resolution** code for the task listed in the **Tasks** field


___2. Double click on the task to open the record in a separate window.

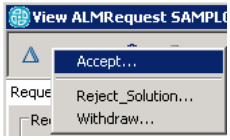
___3. Review the **State** of the activities listed in the **Activities** field.

___a You can double click on an activity and review its detail if needed.

___b Close the Activity window.

___4. Close the Task window.

___5. If everything is satisfactory, return to the Request record and perform the **Accept** action.




___6. Note the record changes to modify mode. Optional detail can be added if needed.

___7. Click **Apply** (or **OK** if the dialog is open).

Congratulations! You have just played the roles of an entire development team responding to a request.

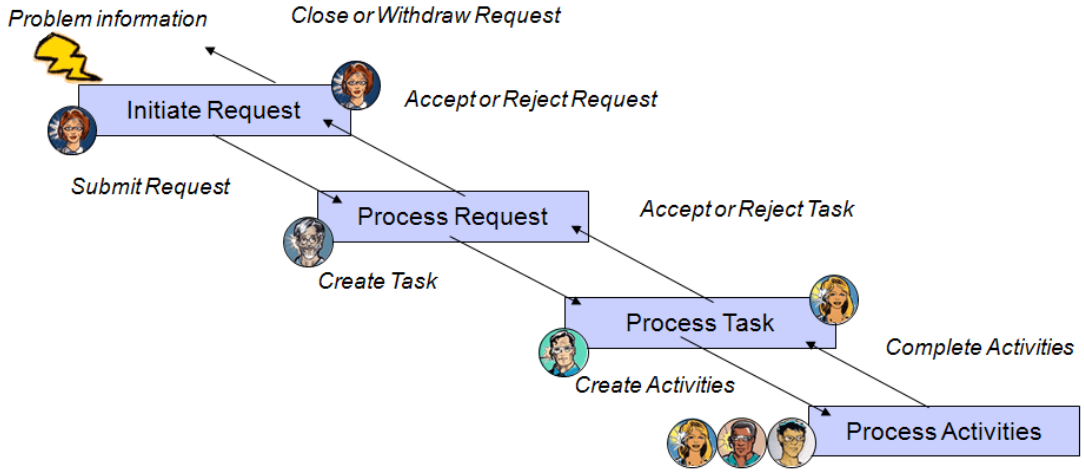



*Figure 11. Completing a Request, Task and Activities*