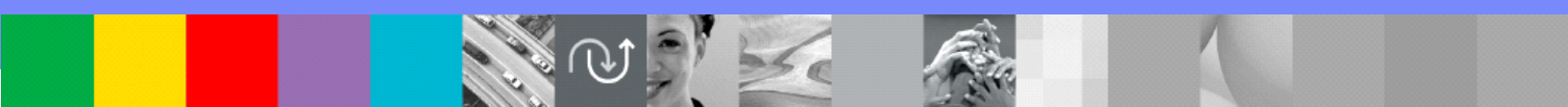
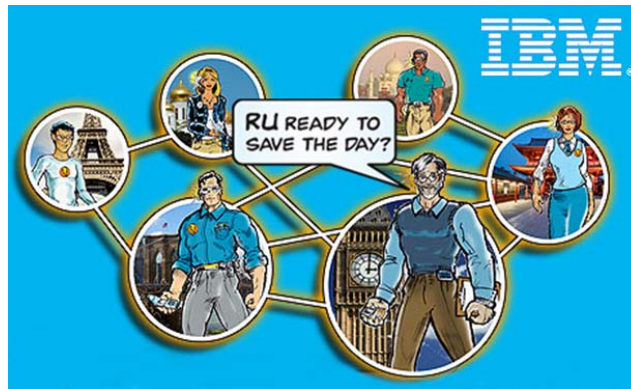


# Introduction to Application Lifecycle Management on ClearQuest

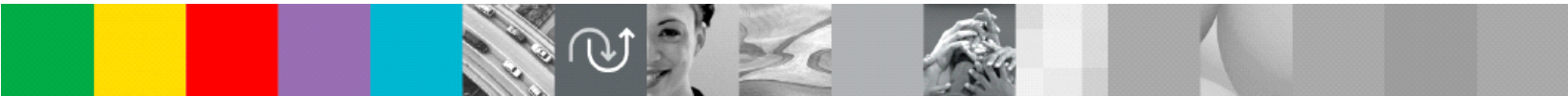
## *Lab #2: Working with Projects*





# Table of Contents

1	Overview	3
2	Explore the Sample Projects	4
3	Copy Project	15
4	Create Project	18



## Lab #2 Working with Projects

### 1 Overview

It is important to understand that projects are a structure comprised of many record types. While there is an ALMProject record, project creation is more of a process than a single record. There are steps to take before creating the project record, and steps to take after you have created the project record. The following exercise will take you through the process, to ensure a full understanding of project structure and features.

#### Projects provide a role-based context for work

All work in the ALM schema is organized by an **ALMProject**. The ALMProject provides the context, access control, and role-based security model for your work. The Project Management Body of Knowledge (PMBOK) defines "Project" as a temporary endeavor undertaken to create a unique product, service, or result. In an ALM system, it is the context within which work is done, and provides traceability for the work completed during the software project's lifecycle. Figure 1 illustrates the architecture of an ALM Project, including the objects that comprise a Project definition, system-wide settings, and existing ClearQuest User and Group administration.

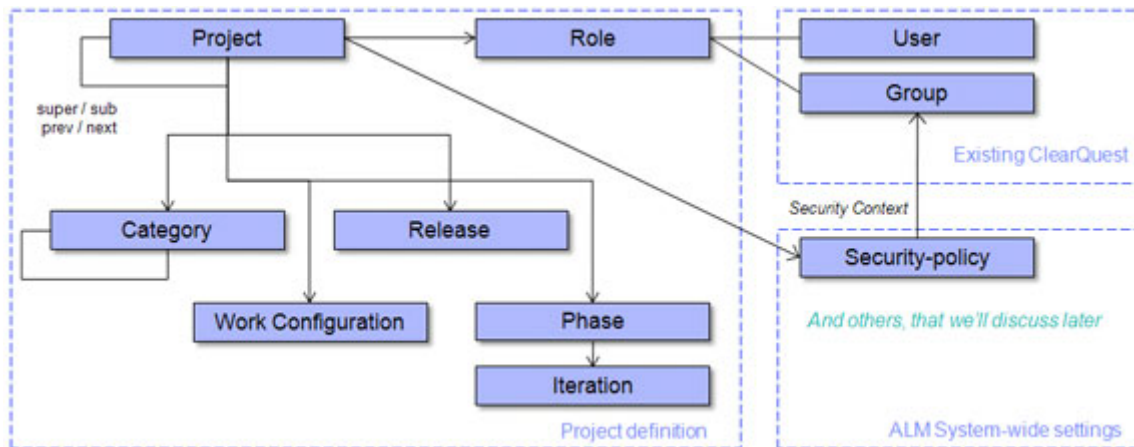


Figure 1. ClearQuest ALM: Conceptual overview of projects. All work in the ALM schema is organized by a project. The project provides both the context and role-based security model for your work.

In this lab you will:

- Explore the configurations of 3 sample projects
- Copy an existing Project
- Create a Project along with the roles, work configurations, phases and iterations for that project



## 2 Explore the Sample Projects

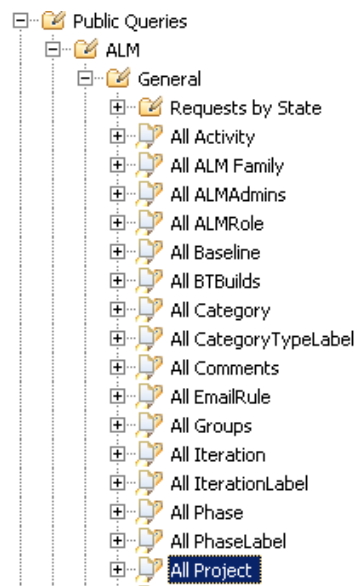
This lesson explores three example projects. You will learn about:

- The Project Record
- The features of projects
- See examples of how projects can be configured differently

### Locate Projects

\_\_\_1. Run the **All Project** query to locate the current projects.

\_\_\_a **Public Queries > ALM > General > All Project**



\_\_\_b The query returns three sample projects.

All Project - ClearQuest Query Results			
ClearQuest Query Results (admin,CQALM@CQALM)			
Name	Category	Release	SecurityPolicy
+ OpenUP Sample Project	Our OpenUP Solution	OpenUP Release 1	OpenUP Security
+ Simple Agile Example	Agile example	1.1.0.0	OpenUP Security
+ Triage Sample Project	Our Triage Product	Triage Release 1	Triage Security



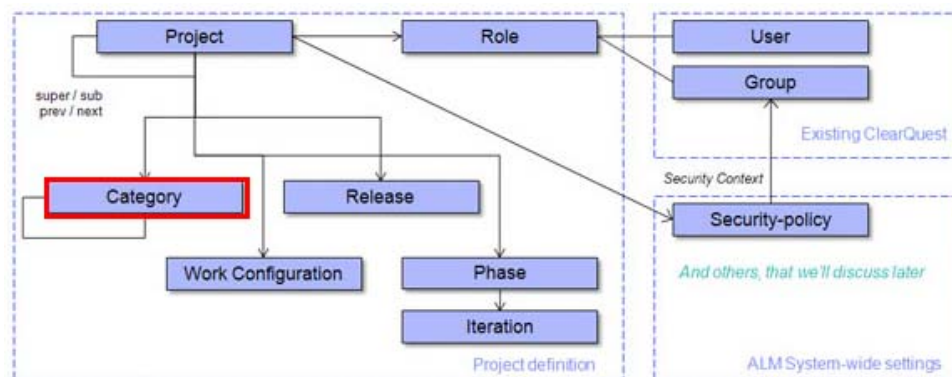
## Explore the Triage Project

- \_\_\_1. Double click on the **Triage Sample Project** to open the record dialog. Note the tabs across the top of the dialog. We will explore each of these tabs in this lesson.
- \_\_\_2. Click **Modify** to explore this record.



The modifiable fields are now enabled and easier to view.

- \_\_\_3. Explore the **Project** tab which contains basic information about the project. Projects are identified by a **Category**, which helps to classify the product, feature, or component that the project delivers.



- \_\_\_4. Click on the **Category** drop-down box to view the available list of categories.

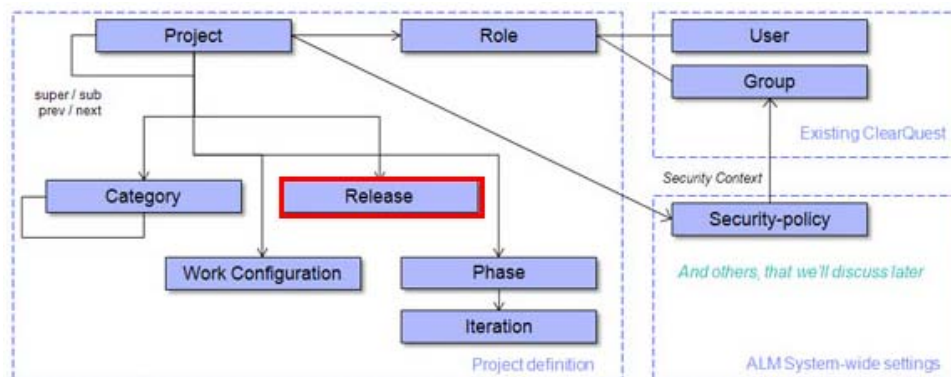
- \_\_\_a Note the **New** Button next to the field. This can be used to create a new category if it does not already exist on the list.

- \_\_\_b Click the **New** Button. The **Create ALMCategory** dialog appears.

This is an example of how the system can be configured without changing the schema. If you completed the required fields on this dialog and submitted the record, a new Category would be entered into the system and available on the **Category** drop-down list on the ALMProject record.

- \_\_\_c We will cover Category creation in a subsequent lab, so click **Cancel** on the dialog.

A **Release** identifies the "version" of the software. Many organizations standardize on a nomenclature for release labels. The ALM solution supports this need by providing the **ALMReleaseLabel** record.



- \_\_\_5. Click the **Release** drop-down. Notice, there are three choices available. Do not change the value.

- \_\_\_a Notice the **Release** field also has a **New** button. Click it.

- \_\_\_b This opens the **Create ALMReleaseLabel** dialog. This is an example of how the system can be configured without changing the schema. By filling in and submitting this record, a new **Release** option would appear on the list.

- \_\_\_c We will cover release creation in a subsequent lab, so click **Cancel** on the dialog.

**✦** Over time, the number of projects produced by an organization can be large. Project uniqueness and identifying features are needed to identify and discern one project from another. Additionally, large projects may be subdivided into smaller projects and share the same Release version. Categories are used to classify a project, and the release is used to identify the version of the software the project will deliver.

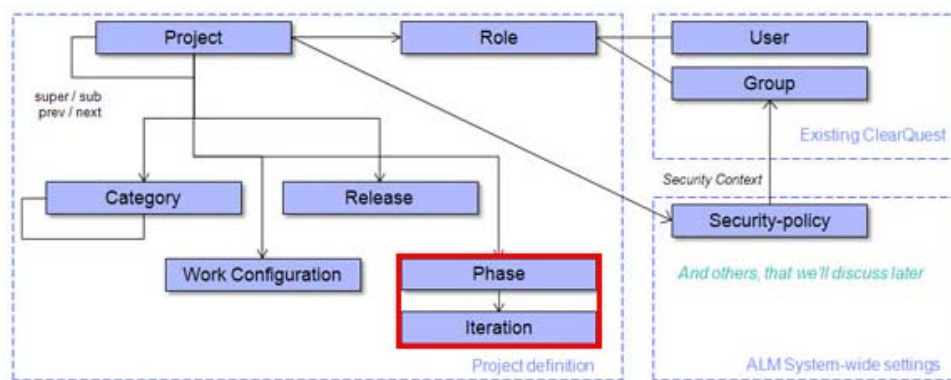


- \_\_\_6. Click the **Current Phase** drop-down. Notice this project has one phase which is named "Iteration." Do not select it.

Current Phase :

- \_\_\_7. The **Start** and **End Date** fields can be used to communicate the dates to the team. These are for informational and tracking purposes and do not provide any further project management function.

The successful creation and delivery of software projects involves plans for completing work. Iterative development techniques have proved to be successful in planning and delivering software projects. Projects can be divided into **Phases** and **Iterations**, which are planned, time-boxed intervals typically measured in weeks. For example, the IBM Rational Unified Process®, or RUP®, defines four phases of Inception, Elaboration, Construction, and Transition.



- \_\_\_1. Explore the **Plans** Tab by clicking it. This tab displays the phases and iterations that have been defined for a project.

View ALMProject CQALM00000002 (admin,CQALM@CQALM)

Project | Plans | Team Members | Related Projects | ALMWork Configurations | History

Phases and Iterations:

Phases	Iterations	Phase Start Date	Phase End Date
Iteration	01		
Iteration	02		

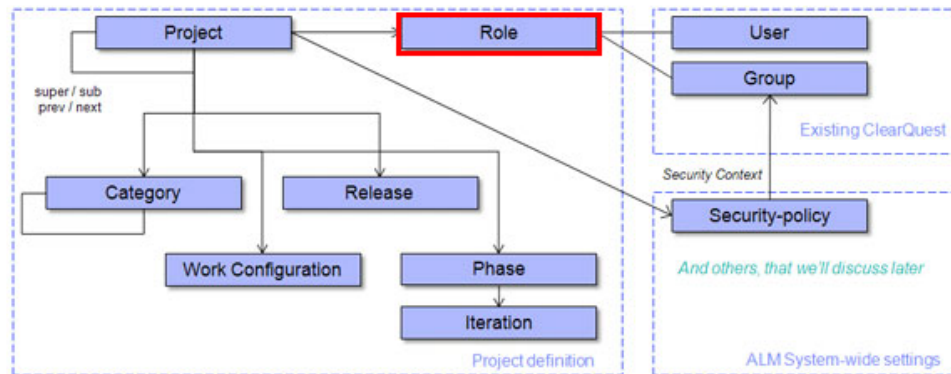
- \_\_\_a This project uses a single **Phase** named "Iteration"
- \_\_\_b This phase has two **Iterations** called "01" and "02"

This is an example for how a team that only works in "iterations" can configure the system to suit their vocabulary.

- While the use of phases and iterations is considered a best practice in software development, they are optional in this system.
- If you have more than 9 iterations, include the zero to ensure proper sorting once you reach 10



Roles are used to define which team members are allowed to perform specific actions. The **ALMRole** record type exists to create the role definition and bind it to a project. In this exercise you view roles that have already been defined for the sample project.

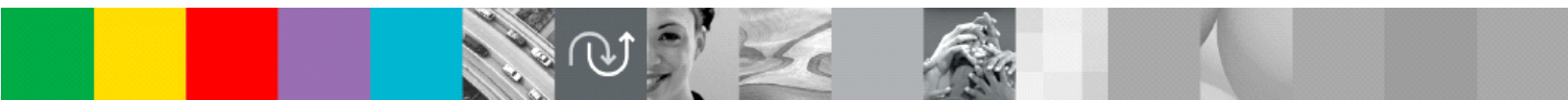


\_\_\_1. Click on the **Team Members tab** to explore its contents. This project has 8 roles defined.

Project	Plans	Team Members	Related Projects	AI
Roles and Members:				
Role	Primary			
Dev Lead	admin			
Developer	admin			
Release Engineer	admin			
Stakeholder	admin			
Test Lead	admin			
Tester	admin			
Triage	admin			
Any Role	admin			

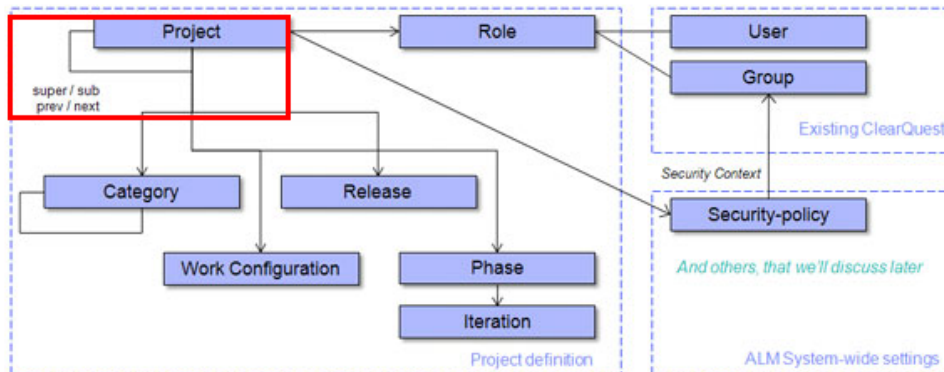
*For simplicity in training, the primary has been set to admin for all roles. This is not normal use. In normal use the primary is set to the person who is the lead for all people who perform that role. We will discuss roles further later in the lab.*

- \_\_\_a Double-click on any of the roles in the list. The dialog for that role opens. A role is defined using an **ALMRole** record that we discuss in depth later in this workshop. The key for this lesson is to know that you can easily access the role record from the project.
- \_\_\_b Take a moment to explore the tabs on the record and then dismiss the record by clicking **Cancel**.





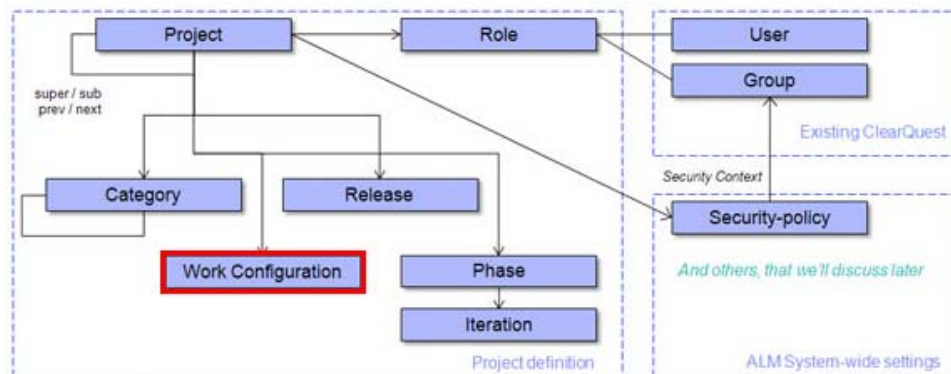
Projects often have relationships to other projects. A common example is a project which has sub-projects. Also, projects can be linked to define a series of projects over time, such as release 1, release 2 and so forth. The **Related Project** tab is used to define and view the relationships for the current project. Last, projects often share the same structure. A clone project feature is provided to copy the structure of the current project into a new project. This not only saves time, but ensures consistency in project definition for similar project types.



\_\_\_1. Click the **Related Projects Tab**. You may need to resize the window to view its contents.

- \_\_\_a The **Clone Current Project** fields allow you to copy the structure of an existing project into a new project. This copies the structure, such as phases & iterations, roles, and work configurations. It does not copy data related to requests, tasks, and activities.
- \_\_\_b Projects can contain other projects. Use the **SuperProjects** field to identify the 'parent' project of which this is a child of.
- \_\_\_c The **SubProjects** is populated with other projects that reference this project as its parent.
- \_\_\_d The **Prior Project** field is used to identify the pre-cursor to the current project. For example, ClearQuest 7.0.1 is the Prior Project to ClearQuest 7.1.
- \_\_\_e The **Next Project** field is populated with other projects that reference this project as its predecessor.

The **ALMWorkConfiguration** record is used to bind request, task, activity, and/or build types to a project. In this exercise you view workconfigurations that have already been defined for the sample project.

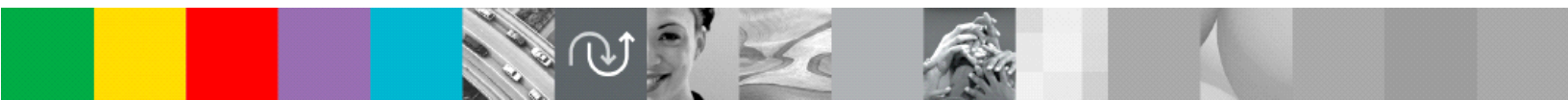


\_\_\_1. Click the **ALMWorkConfigurations** tab to explore its contents.

The screenshot shows the **View ALMProject CQALM00000002 (admin,CQALM00000002)** dialog box. The **ALMWork Configurations** tab is selected. The **UCM\_Project** field is set to a dropdown menu. Below it are sections for **Default Request** and **Default Task**, each with a table for adding, removing, or creating new entries. The **WorkConfigurations** section shows a list of records with columns for Record Type, Type Label, and Obsolete.

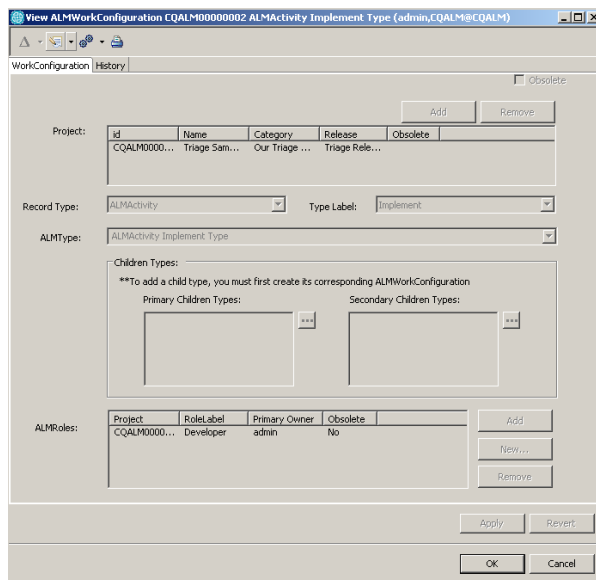
Record Type	Type Label	Obsolete
ALMActivity	Implement	NO
ALMActivity	Test	NO
ALMTask	Defect	NO
ALMRequest	Defect	NO
ALMActivity	Design Imple...	NO
ALMActivity	Implement D...	NO
ALMActivity	Implement T...	NO
ALMActivity	Create Test ...	NO
ALMActivity	RunDev Test	NO
ALMTask	Develop	NO
ALMTask	Test	NO

- \_\_\_a **UCM\_Project** field is used to link this project with an existing UCM Project. This is for identification purposes only.
- \_\_\_b The **Default Request** field is used when there is only one Request for the project. When this is set all Tasks created for this project will automatically reference this Request. We will review a project that makes use of this in a future lesson.
- \_\_\_c The **Default Task** field is used when you want to hide the 3-tiered nature of Request-Task-Activity. By setting the default Task, all new Activities will automatically reference the default Task specified in this field. This is handy for Agile teams that want to use "work-item" as the method of managing work. We will review a project that makes use of this in a future lesson.
- \_\_\_d The **WorkConfigurations** list shows which Request, Task and Activity types are bound to this project. To accommodate each project team's workflow, you configure the default set of Activities and Tasks for the Project. You can also bind Activity types to a Task Type, and a Task Type to a Requirement type. Project and/or Process managers will be most interested in understanding the power of this record.

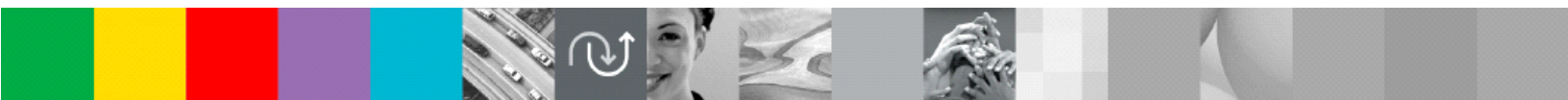


Behind the scenes, the WorkConfigurations came into play all through the previous lesson. Let's take a look at how.

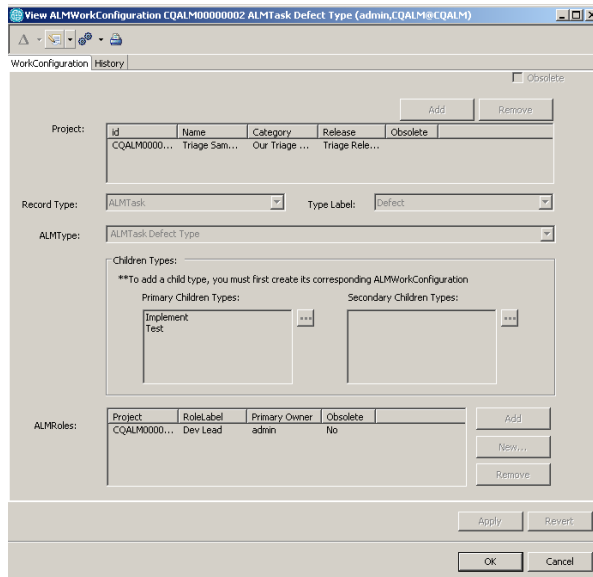
- \_\_\_2. On the current WorkConfiguration tab, in the **WorkConfigurations** list locate the **ALMActivity** of **Type Label** "Implement" and double click on it.
  - \_\_\_a The WorkConfiguration record opens in a new window.
  - \_\_\_b Notice that the project is set to the Triage Sample Project.
  - \_\_\_c The **Record Type** is ALMActivity and the **Type Label** is Implement. This states that the Triage Sample Project will have an ALMActivity of type Implement.
  - \_\_\_d There are no Children Types for Activities. This is correct usage.
  - \_\_\_e Also note the **ALMRole** is set to Developer. This field defines which roles on the team will interact with this Activity type. In this example, only one role is set, but you can set as many as you want. Recall in the previous lesson when you assigned the Implement ALMActivity and the **Owner** field was filtered. The filter is created by the roles selected on the **ALMWorkConfiguration** record.



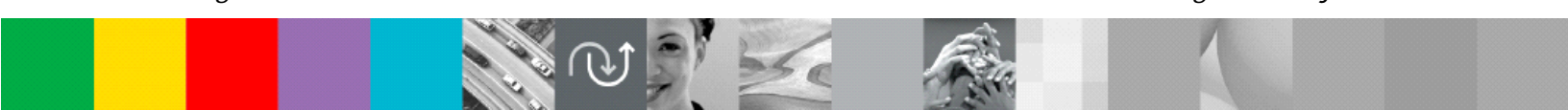
- \_\_\_f **Cancel** to dismiss the dialog.



- \_\_\_3. In the same list, locate the **ALMTask** of Type Label "**Defect**" and double click on it.
- \_\_\_a The ALMWorkConfiguration record opens in a new window.

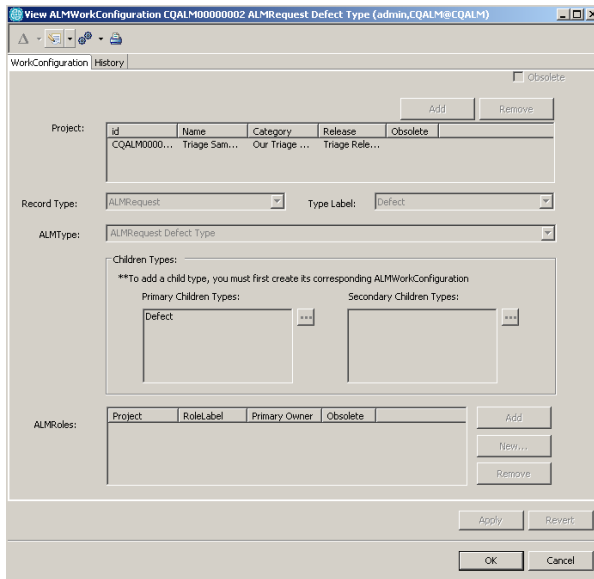


- \_\_\_b Notice that the **Project** is set to the Triage Sample Project.
- \_\_\_c The **Record Type** is ALMTask and the **Type Label** is Defect. This states that the Triage Sample Project will have an ALMTask of type Defect.
- \_\_\_d The **Children Types** list contains activities of type "Implement" and "Test". This defines the behavior of the "**Create Activity**" utility on the Task record. Clicking that utility will automatically create these two Activities. Recall in the previous lesson when you clicked the CreateActivity... utility and two activities were created.
- \_\_\_e Also note the **Role** is set to Dev Lead. This field defines which roles on the team will interact with this Task type. When a Task of type Defect is created, the names of all users acting in this role will appear on the choice list. In this example, only one role is set, but you can set as many as you want.
- \_\_\_f **Cancel** to dismiss the dialog.



\_\_\_4. In the WorkConfigurations list, locate the **ALMRequest** of type `Defect` in the list and double click on it.

\_\_\_a The ALMWorkConfiguration record opens in a new window.



\_\_\_b Notice that the **Project** is set to the Triage Sample Project.

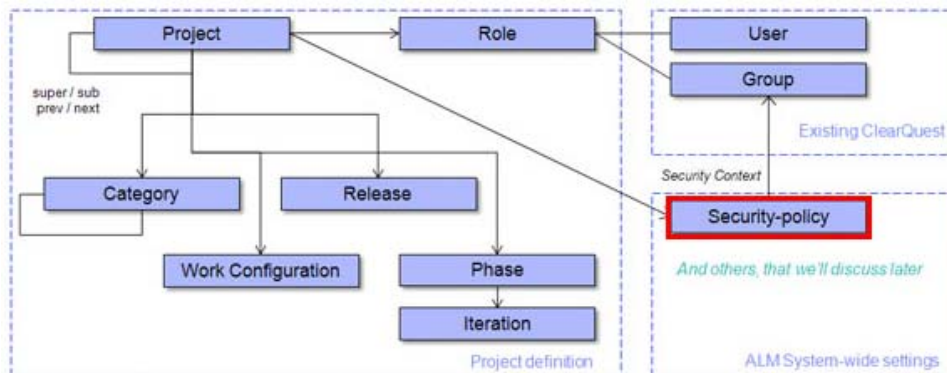
\_\_\_c The **Record Type** is ALMRequest and the **Type Label** is Defect. This states that the Triage Sample Project will have an ALMRequest of type Defect.

\_\_\_d The **Children Types** list contains a **Task** of type "Defect". This defines the behavior for the "Create Task" utility on the Request record. Clicking that utility will automatically create a task of type "Defect"

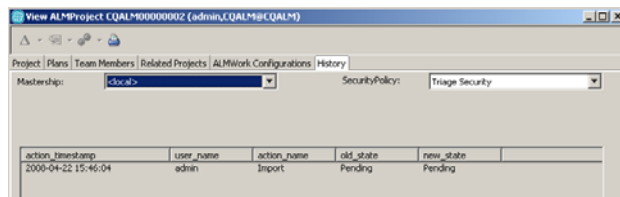
\_\_\_e **Cancel** to dismiss the dialog.



Security is an important aspect of all project-based work. In the ALM Schema project security defines by who has access to the project. The **ALMSecurityPolicy** record is used to create new policies. In this exercise you view Security Policies that have already been defined for the sample project.



\_\_\_1. Click on the **History** tab.



\_\_\_a Make sure the record is in **Modify** mode. Click on the **SecurityPolicy** drop-down to see the available list. Security Policy will be covered in a future lesson.

\_\_\_b Note the history captures an audit trail of changes to the record.

\_\_\_2. **Cancel** the Project record to close it. Discard any changes.

### Explore the other projects

\_\_\_1. Start from the beginning of this lesson and repeat the steps for the *Agile* sample project and note the differences. The Agile project uses the default request and default task. It also has different work configurations.

\_\_\_2. Start from the beginning of this lesson and repeat these steps for the *OpenUp* sample project. The phases and iterations follow the RUP.

### 3 Copy Project

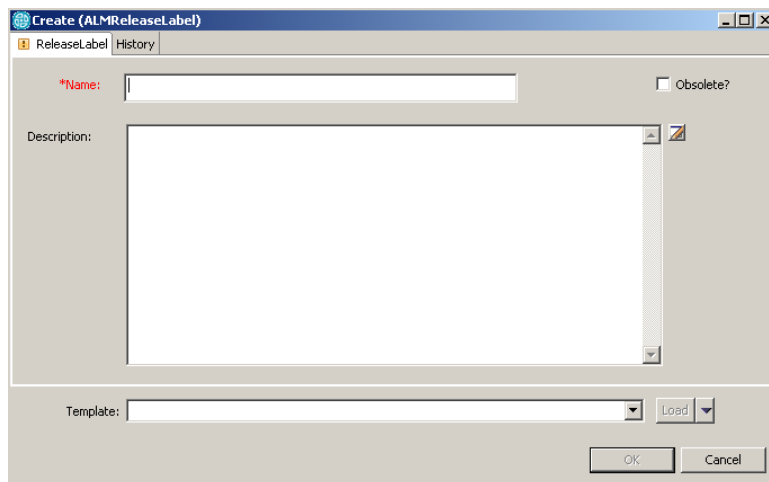
The easiest procedure for creating a project is to start with a copy of an existing project that is configured according to the project governance criteria of your organization. Once a project is copied, the new project can be adjusted for any differences that may be required.

Many times new projects are similar to existing projects. For example, the next version of the same project will have characteristics similar to its predecessor, or sub-projects will share the characteristics of the parent project. The ability to "copy" an existing project is introduced by the ALM Solution. The "Copy Project" command copies the structure of the project, such as the role definitions, phases and iterations, and work configurations. It does not copy the data, such as specific request, task, or activity records. Once you have a copy, you can then make whatever modifications are needed.

You can allow project managers to copy any project or you can establish a best practice where template projects are created. By setting up example projects with all of the expected settings, you can provide guidance to your project managers to copy one of the examples. For example, your organization may have many projects that implement a packaged application such as SAP. On the other hand, your service-oriented architecture (SOA) projects are likely to be different. You can create an example SAP project, and an example SOA project, and the next time one of these project types is funded, you simply copy the example project.

\_\_\_1. Create a new **Release Label**.

\_\_\_a **File-New-ALMRelease** Label.



\_\_\_b Provide a **Name**, such as "Triage Release 1.2", or "OCT 2008"

\_\_\_c Optionally provide a description.

\_\_\_d Click **OK**.

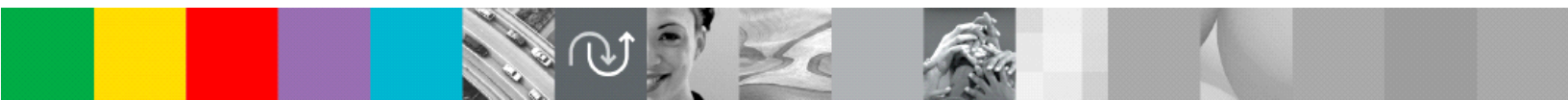
\_\_\_2. Copy a Project.

\_\_\_a Run the All Projects query (**Public Queries > ALM > General > All Projects**).

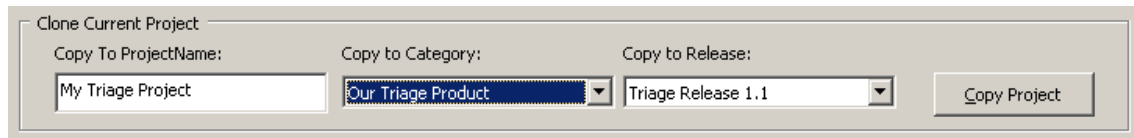
\_\_\_b Select the Triage Sample Project.

\_\_\_c Click **Modify**. 

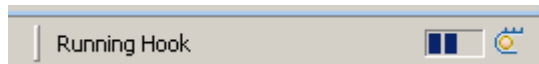
\_\_\_d Click the **Related Projects** tab



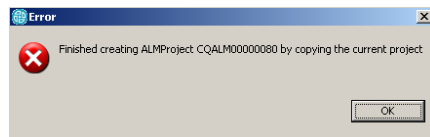





- \_\_\_e In the **Clone Current Project** field, provide a **Copy to ProjectName**, such as "My Triage Project"
- \_\_\_f Select "Our Triage Product" from the **Copy to Category** drop-down list
- \_\_\_g In the **Copy to Release** drop-down, select the Release Label you just created. "Triage Release 1.2"
- \_\_\_h Click the **Copy Project** button. A hook will run which is indicated by a status message in the lower right corner of the eclipse shell.




- \_\_\_i When the project is copied, a dialog appears indicating success or failure. Click **OK** to dismiss the window.

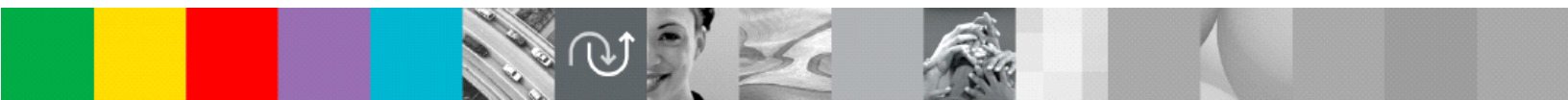


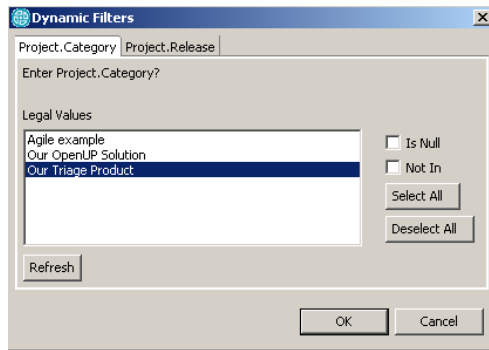
 The "Error" dialog appears even upon success. This is a known defect.

- \_\_\_j Click **Cancel** on the Project Dialog.
  - \_\_\_k Dismiss the Warning about discarding changes.
- \_\_\_3. Locate your new project
- \_\_\_a Run the All Projects query (**Public Queries->ALM->General->All Projects**)
  - \_\_\_b Select your project in the result set to view the record details.
  - \_\_\_c Click on the **Plans** tab – the plans from the Triage Sample Project are copied
  - \_\_\_d Click the **Work Configurations** tab – the Work Configurations are copied.
  - \_\_\_e Click the **Team Members** tab – the roles are copied
- \_\_\_4. Set the security Context for the Roles.
- \_\_\_a On the **Team Members** tab, double click on one of the roles.
  - \_\_\_b Click on the **History** tab. Note the **Security Policy** defaults to "Everyone"

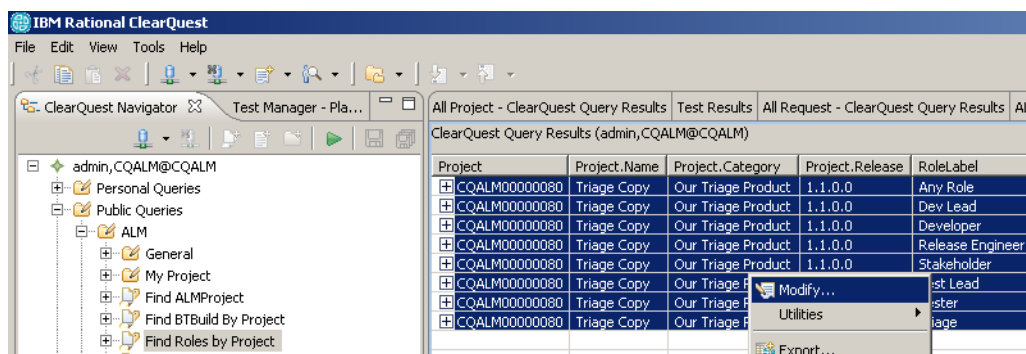
 It's important to update the role security policy field to match the new project. Normally roles inherit the security of the project. However when copying, it's possible that the users or groups assigned to the role are not members of the new security policy. Therefore, when copying a project, always review and update the roles for the project. This involves the Security Policy, the User / Group assignments, and the Allowed Actions.

- \_\_\_c **Cancel** the Role dialog.
- \_\_\_d **Cancel** the Project dialog if it is open.
- \_\_\_e Run the **Find Roles by Project** query. **Public Queries > ALM > Find Roles by Project**

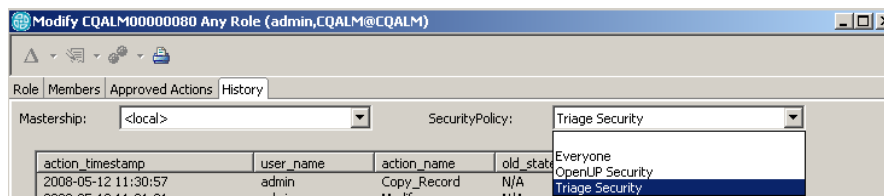




- \_\_\_f On the **Project.Category** tab, select “Our Triage Product”
- \_\_\_g On the **Project.Release** tab choose the Release number you used when you copied the project.
- \_\_\_h Click **OK** to run the query.
- \_\_\_i **Shift-select** all role records in the query, right click and choose the **Modify** action.



- \_\_\_j Click on the **History** tab.
- \_\_\_k Set **Security Policy** to “Triage Security”



- \_\_\_5. Click **Apply All**.

Congratulations! You have just copied a project.

## 4 Create Project

In this exercise you will go through the process of project creation, which involves creating the Project record, creating roles, work configurations, phases and iterations.


### Pre-Requisites to Project Creation

For this exercise, you will choose some items that have already been defined. Exercises for creating the pre-requisites will be covered in a later lab. These will be noted.

- Security context
- Category Type Label
- Category

### Create Project


- \_\_\_1. Choose **File > New > ALMProject**.
  - \_\_\_a The Project Dialog opens.
  - \_\_\_b Note the **Project** and **History** tabs both have required fields.
- \_\_\_2. On the **Project** tab, provide a **Name** 'RSDC Triage Project'
- \_\_\_3. Click on the **Category** drop down and choose the Our Triage Product.

 *Category is a pre-requisite. In a new schema, create Categories before attempting to create a project. Or click the "New" button next to the Category field on the Project record.*

- \_\_\_a Notice the **History** tab no longer requires a field. This is because the Security Policy is inherited from the Category.
  - \_\_\_b Click on the **History** tab. The **Security Policy** is set to Triage Security.
  - \_\_\_c Click on the **Project** tab.
- \_\_\_4. Create a **Release**.
    - \_\_\_a Click on the **Release** drop down. Notice that there are two choices. Let's create a new Release label by clicking on the **New** button .



- \_\_\_a In the **Name** field, type Release 2.0.
- \_\_\_b Click **OK**.

 *Release is a pre-requisite. In a new schema, create Release(s) before attempting to create a project. Or click the "New" button next to the Release field on the Project record, as we are doing in this lesson.*

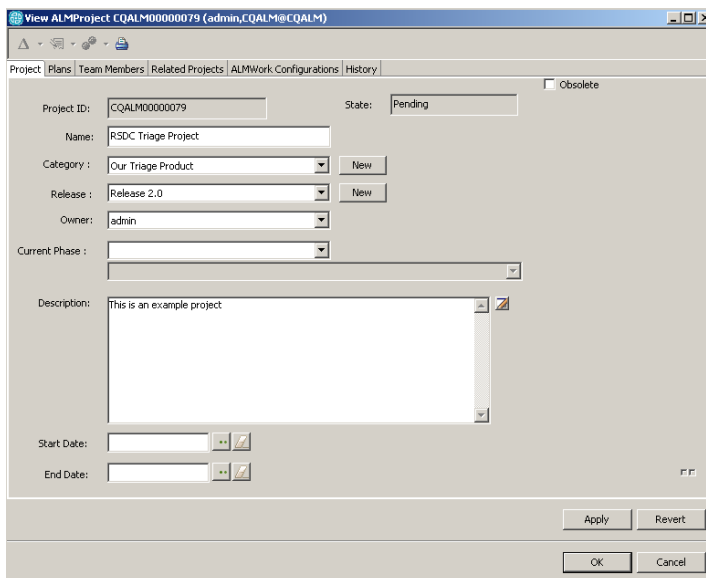
- \_\_\_5. Click on the **Release** drop down list, the Release Label you just created should now appear on the list. Select it.
- \_\_\_6. Set the **Owner** field to 'admin.'
- \_\_\_7. Set any other optional fields on the Project tab, such as **Description**, **Start / End** date.




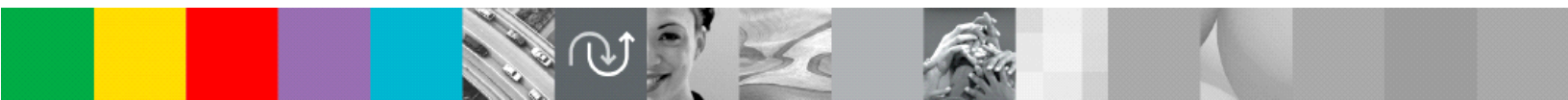
- \_\_\_8. Note **Current Phase** drop-down list is empty. This is correct.
- \_\_\_9. Click on the **Plans** tab. Note the Phases & Iterations list is empty. This is correct, because we have not created any for this project yet.
- \_\_\_10. Click on the **Team Members** tab. Note the list is empty. This is correct, because we have not created any Roles for this project yet.
- \_\_\_11. Click on the **Related Projects** tab.
  - \_\_\_a On the Prior Projects list, click the **Add** button. The Browse Record Type Dialog appears.
  - \_\_\_b Click **Search** on the Browse records dialog.
  - \_\_\_c Select the project you copied in the previous exercise. It should be "Our Triage Product, Release 1.2, My Triage Project."
  - \_\_\_d Click **OK**. Doing this creates a relationship that indicates this is the next version of the Triage Product.

Prior Project		
ID	Category	Release
CQALM00000...	Our Triage Product	Triage Relea...

- \_\_\_e Select the project in the list. Notice the **Remove** button is enabled. The **Add** button is dimmed. Only one project can be designated as the prior project.
- \_\_\_12. Click on the **WorkConfigurations** tab.
  - \_\_\_a If you are using UCM, your UCM projects will appear in the drop-down list. This is part of the UCM integration.
  - \_\_\_b Leave the **Default Request** and **Default Task** fields blank.
  - \_\_\_c Last, note the **WorkConfigurations** list is blank because we have not yet created any.
- \_\_\_13. Ensure that all required fields are completed and click **OK**.



 At this point you have created a Project record, but you are not quite finished defining the project.

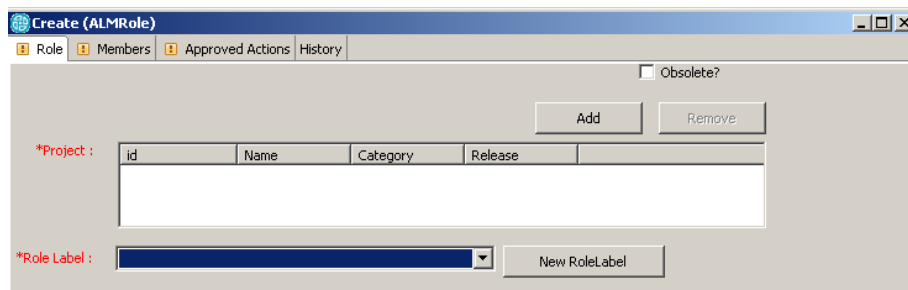


## Create Roles

The next step involves creating roles for the project. A set of Role Labels are already provided which can be used system wide. When creating a new project however, a set of Roles must be created. The role binds users or groups to a specific role definition with approved actions for the project. The role names can be shared across projects, however the user and/or group assignment and the approved actions are likely to vary. This exercise teaches you how to create a role for a project.

The **ALMRole** record is used to define which users are members of the project and what actions they are allowed to perform. Role records are bound to the Project record. Users are assigned to act in the role. Allowed actions are defined for each role. Each project can have a differing set of roles, users and actions, simply by defining the ALMRole records for that project.

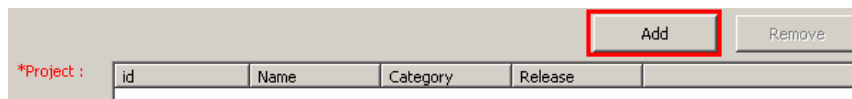
\_\_\_1. **File > New > ALMRole**. The create ALMRole dialog appears.



The 'Create (ALMRole)' dialog box is shown. It has tabs for 'Role', 'Members', 'Approved Actions', and 'History'. The 'Role' tab is active. There is an 'Obsolete?' checkbox. Below it are 'Add' and 'Remove' buttons. A table with columns 'id', 'Name', 'Category', and 'Release' is present. Below the table is a '\*Project :' label. At the bottom, there is a '\*Role Label :' dropdown menu and a 'New RoleLabel' button.

 *Role, Members, and Approved Actions tabs all have required fields.*

\_\_\_2. On the Role tab, Project is required. Click the **Add** button.



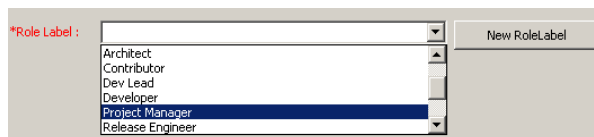
A close-up of the 'Add' button in the 'Create (ALMRole)' dialog, which is highlighted with a red rectangle.

\_\_\_a The Browse Record dialog opens. Click the **Search** button.

\_\_\_b Choose the Project you created in the exercise above: Our Triage Product, Release 2.0, RSDC Triage Project

\_\_\_c Click **OK** to dismiss the dialog.

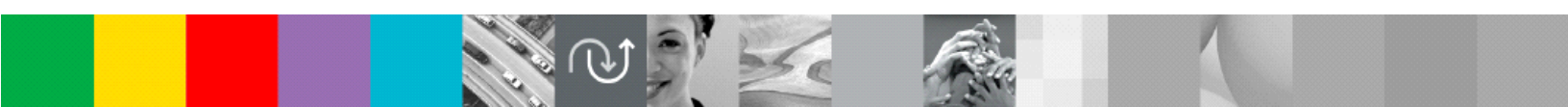
\_\_\_3. Click **Role Label** drop-down list and select Project Manager. The role labels on this list are provided in the Sample data that ships with the product.



The 'Role Label' dropdown menu is open, showing a list of roles: Architect, Contributor, Dev Lead, Developer, Project Manager (highlighted), and Release Engineer. The 'New RoleLabel' button is visible to the right.

Note the **Approved Actions** tab is no longer required. The Approved Actions are inherited from the Role Label. We discuss this in the System-wide settings lab.

Alternatively you can create your own RoleLabel by clicking the NewRoleLabel button, fill out the fields and click OK.



\_\_\_4. Click on the **Members** tab.

\_\_\_a Click on the **Add** button next to the Members list. The Browse Record dialog opens.

**✦** Either User(s) or Group(s) must be added to perform this role. Setting one releases the 'required' nature of the other.

\_\_\_b Click **Search** button on the Browse Record dialog.

\_\_\_c Choose '**Patricia**' and '**admin**' from the list of Members.

\_\_\_d Click **OK**.

\_\_\_5. On the **Members** tab of the Role dialog, set the **Primary** by choosing a name from the drop-down list. The Primary is used as the default by the system when a list of users is presented in the user interface. For simplicity set it to '**admin**.'

\_\_\_6. All required fields should be completed at this point. Click the **OK** button to save your Role.

\_\_\_7. Repeat these steps to create the following roles:

- Analyst = Bob, admin
- Dev Lead = Marco, admin
- Developer = Dierdre, admin
- Tester = Tanuj, admin
- Project Manager = Patricia, admin (if you have not already done so)


**✦** At any point you can check your work by running the All Projects query, selecting your project and clicking the Team Members tab. The roles you've defined will show up on this list.



## Create Work Configurations

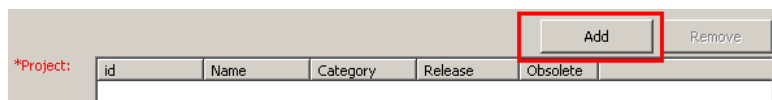
In this exercise you will create **ALMWorkConfigurations** for a new Request type called Defect. The separation of requests, tasks, and activities creates a powerful new model. However, in the prior ClearQuest model a record type was used to define specific state transitions. In effect, processes could be managed by the record type and its states. The question then becomes: *How can I manage such processes in this new model?* A secondary concern involves having to manually create tasks and activities for every request. In the original model, you create a record, and the states come with it. In the new model a single activity represents a state change. That's potentially a lot of activities to create, and how can you guarantee that the same activities are used every time?

Enter the ALMWorkConfiguration record. To accommodate each project team's workflow, you can now configure the default set of activities and tasks each project. You also define the default activity types per task type, and the default task type per request type.


 *Project and/or Process managers will be most interested in understanding the power of this record. Other users on the team need not be aware.*


The first part of this exercise involves binding activity types to your project.

- \_\_\_1. **New > ALMWorkConfiguration.** The **Create ALMWorkConfiguration** dialog opens.
- \_\_\_2. Start by clicking the **Add** button next to the **Project** field.

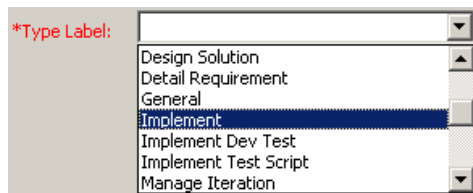


- \_\_\_a On the Browse Record dialog, click **Search**.
- \_\_\_b Choose the project you created from the result list.
- \_\_\_3. Next click the **Record Type** dropdown and choose **ALMActivity**.




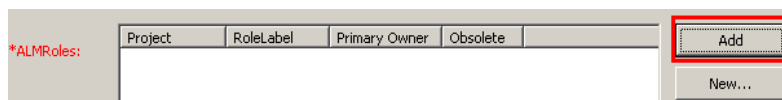
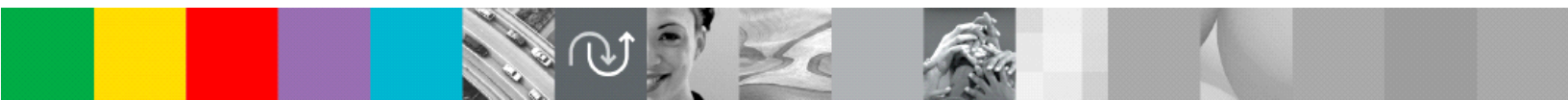
 *For new projects, always start with ALMActivity. The ALMTask and ALMRequest Work Configurations often include existing ALMActivity records. This will become apparent in the next exercise.*

- \_\_\_4. Next click the **Type label** for this Activity. Choose '**Implement**' from the list.



- \_\_\_5. Click on the **Add ALMRoles** button.

 *The **Children Types** section does not apply to Activity*



- \_\_\_a On the Browse Record dialog, Type “**Developer**” in the field and click **Search**.

- \_\_\_b Choose the developer you created in the previous exercise from the result list.

**IMPORTANT:** You will see several entries for “Developer” in this list. Be sure to locate the “Developer” role for *your* project. This can be done by noting the record ID in the left column of the list.

- \_\_\_c Click **OK**.

The developer role you selected appears in the list of ALMRoles.

- \_\_\_d Click **OK** on the Create ALMWorkConfiguration dialog.

- \_\_\_6. Repeat these steps to create an **ALMActivity** of **Type=Test**. This time set the ALMRole to Tester.

This next part of the exercise binds an **ALMTask** to the project and sets a default set of activities for that task.

- \_\_\_1. **New > ALMWorkConfiguration**

- \_\_\_2. Start with by binding the Project by clicking the **Add** button next to the **Project** field.

- \_\_\_a On the Browse Record dialog, click **Search**.

- \_\_\_b Choose the project you just created from the result list. Click **OK**.

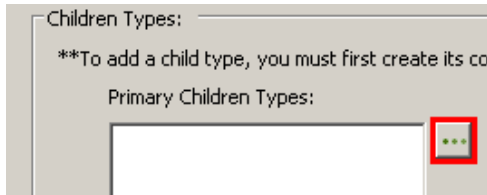
- \_\_\_3. Next click the **Record Type** drop-down. Choose **ALMTask**.

**For new projects, bind the ALMTask records after binding the ALMActivity records, and before binding the ALMRequest records.**

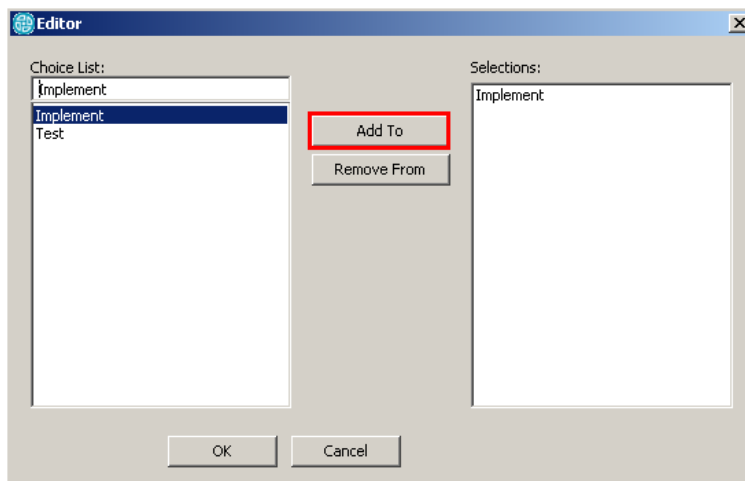
- \_\_\_4. Next choose a **Type label** for this Activity. Choose ‘Defect’ from the list.

- \_\_\_5. The **Children Types** section applies when you want to define a default set of activity types for this Task type. These activities are created when the **CreateActivity...** utility is used on the ALMTask record.

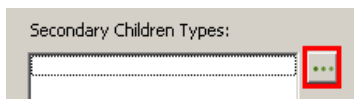
- \_\_\_a Click on the button next to the **Primary Children Types** field.



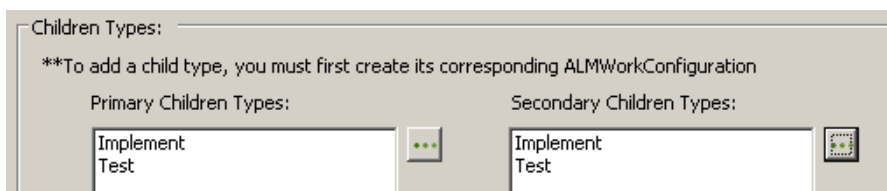
- \_\_\_b In the Editor window, select the **Implement** activity and click the **Add To** button.



- \_\_\_c Do the same for the **Test** activity.
- \_\_\_d Click **OK**.
- \_\_\_e Click the button next to the **Secondary Children Types** field. The Editor window opens.



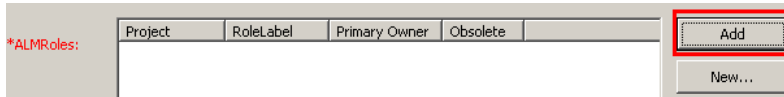
- \_\_\_f Select the **Implement** and **Test** activities and click the **Add To** button.
- \_\_\_g Click **OK**.



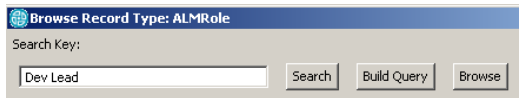
The **Primary Children Types** and **Secondary Children Types** fields have an Implement and Test activity listed.

Now when you create an ALMTask of Type=Defect, the **Create Activity** utility will create these two activities. By defining the same records in the **Secondary Children Types** field, the same two activities will be created for subsequent clicks of the utility.

- \_\_\_6. Click on the **Add** button next to the **ALMRoles** list.



- \_\_\_a On the Browse Record dialog, Type "Dev Lead" in the field and click search.



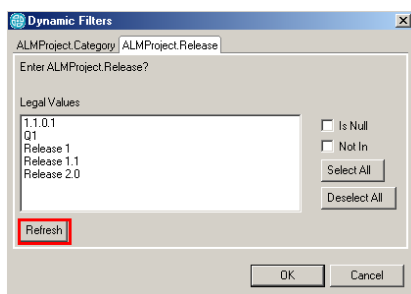
- \_\_\_b Choose the Dev Lead you created in the previous exercise from the result list.

**IMPORTANT:** You will see several entries for "Dev Lead" in this list. Be sure to locate the "Dev Lead" role for *\*your\** project. This can be done by noting the record ID in the left column of the list.

- \_\_\_c Click **OK** to close the Browse Record Type dialog.
- \_\_\_d Click **OK** on the Create ALMWorkConfiguration dialog.
- \_\_\_7. Repeat these steps for the **ALMRequest** of Type '**Defect**'.
- \_\_\_a Set the Primary and Secondary Children Types to Task=Defect.
- \_\_\_b Role is not applicable to Request. Do not set this field.
- \_\_\_8. Run the query '**Find WorkConfigurations by Project**' (**Public Queries > ALM > General**).



- \_\_\_a On the **Project.Category** tab choose "Our Triage Product."
- \_\_\_b On the **ALMProject. Release** tab, choose the Release you created. If you do not see it, click the Refresh button.

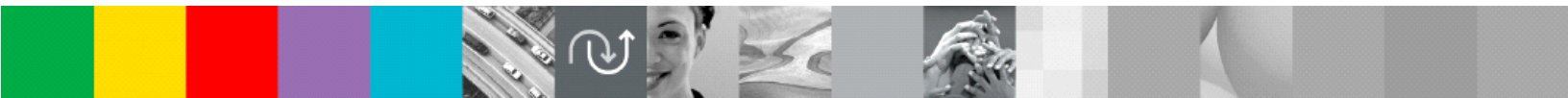


- \_\_\_c Run the Query.

You should see


- o 2 ALMActivity (Implement, Test)
- o 1 ALMTask (Defect)
- o 1 ALMRequest(Defect)

**IMPORTANT:** This same set is listed on the Project record WorkConfigurations tab to indicate the association with the project.

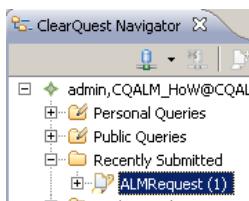


Test your implementation:

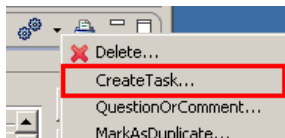
- \_\_\_1. Create an **ALMRequest**.
- \_\_\_2. Choose set the project to the one you just created.
- \_\_\_3. Provide a **Headline**.
- \_\_\_4. Set the **Type** to Defect.

 The choice list is built from the list of WorkConfigurations defined for ALMTask. In the previous exercise, one WorkConfiguration is created for ALMTask = Defect.

- \_\_\_5. Choose a **Severity**.
- \_\_\_6. Click **OK**.
- \_\_\_7. Locate your request record (run the Recently Submitted query for ALMRequest).

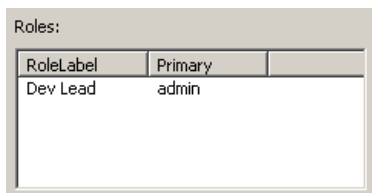


- \_\_\_8. From the Request dialog, click the **CreateTask** utility.

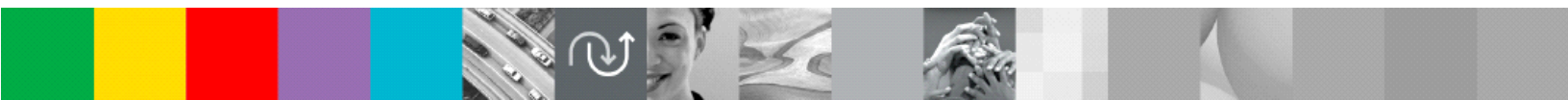


A Task of Type=Defect is created and is listed in the Tasks list on the request record.

- \_\_\_9. View the Task by double clicking the record.
- \_\_\_10. Click **Modify**.
- \_\_\_11. Click on the **Owner** field. The choice list is restricted to the users that act in the role defined for this record. The roles are listed in the Roles field.



- \_\_\_12. Click the **CreateActivity** utility.
- \_\_\_13. The utility runs and the dialog will close. Re-open the Task record and view the activities in the list. They should be the same activities that you just defined.



## Create Phases for your Project

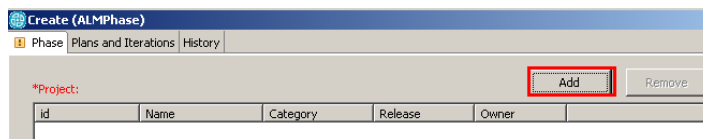
Projects can be divided into Phases and Iterations, which are planned, time-boxed intervals typically measured in weeks. For example, the IBM Rational Unified Process®, or RUP®, defines four phases of Inception, Elaboration, Construction, and Transition.

Users of RUP would create one ALMPhase record to represent each phase of the project: Inception, Elaboration, Construction, and Transition. Iteration records using names such as "I1" for Inception iteration one, and "C1" for "Construction iteration one" would be created to manage the iterations.

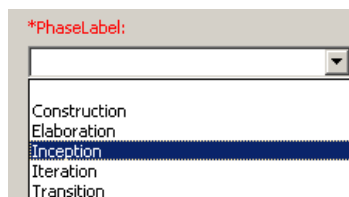
Agile users however, may take a different approach to phase and iteration names. One approach is to create a single phase record named "Iteration." Next create an iteration record with the numeric value of the iteration. For example, create iteration records and name them "1" and "2" and so forth. When using the system, you will have "Iteration 1" "Iteration 2" and so forth.

The system is flexible enough to allow small teams to manage iterations while also scaling up to larger teams who use more formalized phases and iterations. Not all teams practice iterative development, therefore, the use of phases and iterations is optional. Because iterative development is a best practice of software development however, the records are provided as part of the solution.

- \_\_\_1. Create an ALMPhase. **File > New > ALMPhase**
- \_\_\_2. On the **Phase** tab, click the **Add** button next to the project list.



- \_\_\_a On the Browse Record dialog, click the **Search** button.
- \_\_\_b Choose the Project you just created from the results list.
- \_\_\_c Click **OK**.
- \_\_\_3. Click the **PhaseLabel** drop-down and choose "Inception" from the list.

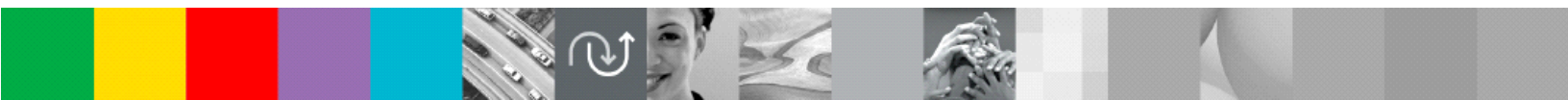


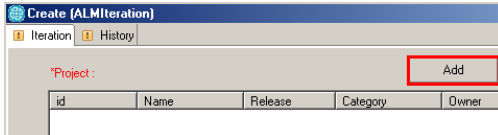
- \_\_\_4. Optionally set the **Status** Label.
- \_\_\_5. Optionally provide a description, **Start / End** date.
- \_\_\_6. Click **OK**.

Repeat steps 1-7 for the rest of the RUP phases (in this order: Elaboration, Construction, Transition). For each of these phases, set the "**Prior Phase Label**" to the previous phase.

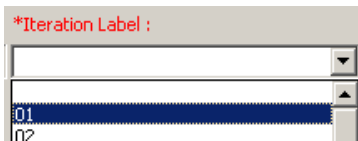
### Next create iterations

- \_\_\_1. Create an ALMIteration: **File > New > ALMIteration**.
- \_\_\_2. On the **Iteration** tab, click the **Add** button next to the project list.





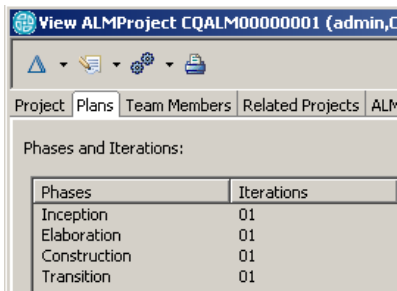
- \_\_\_a On the Browse Record dialog, click the **Search** button.
- \_\_\_b Choose the Project you just created from the results list.
- \_\_\_c Click **OK**.
- \_\_\_3. Set the **PhaseLabel** to Inception.
- \_\_\_4. Set the **IterationLabel** to 01.



- \_\_\_5. Click **OK**.

Repeat these steps for each of the Phases creating at least one iteration per Phase.

- \_\_\_6. Run the **All Projects** query: **Public Queries > ALM > General > All Projects**.
- \_\_\_7. Locate your Project. Click on the **Plans** tab & notice the phases and iterations added.



Congratulations! You have completed this lab. You have explored sample projects and how they are configured, learned how to copy a project, and created a new project.

