



COLLABNET

**ORGANIZING AND
CONTROLLING
ACCESS TO CEE
PROJECTS**

June 2008

COLLABNET®

8000 Marina Boulevard, Suite 600
Brisbane, California 94005-1865 U.S.A.

888.778.9793 toll free
650.228.2500 voice
650.228.2501 fax

www.collab.net
E-mail info@collab.net

Table Of Contents

Introduction	3
CEE Users and Roles.....	3
A Simple Example.....	3
Multiple Users in a Project.....	4
Roles.....	5
Internally Partitioning Projects.....	6
Resource Patterns.....	8
Organizing Multiple Projects.....	9
Multiple Independent Projects.....	10
User Access to Multiple Projects.....	10
A Project with Subprojects	11
Using Inherited Roles.....	12
Further Customizing Access in Subprojects.....	12
User Groups and Project Groups.....	13
User Groups.....	13
Multiple User Groups.....	15
Project Groups.....	15
Public Projects	18
The Registered User Role	19
Private Projects vs. Public Projects	19
“Semi-Public” Projects.....	20
Public CEE Sites.....	21
Project Categories.....	21
Categorizing Public Projects.....	22
Multiple Categories and Subcategories.....	22
Categorizing Private Projects.....	23
Creating Private Categories.....	24
Conclusion.....	25
Appendix A: Standard CEE Roles.....	27
Anonymous Guest	27
Registered User	28
Click-through Viewer.....	28
Observer.....	28
Content Developer.....	29
Developer.....	30
Project Owner.....	30
Domain Admin.....	31

Introduction

The CEE™ collaborative development environment offers a variety of ways in which you can organize CEE projects and users; given this variety it can sometimes be difficult to decide exactly what CEE options one should use for a given project or set of projects. In this paper we provide a guide to using the following CEE features:

- Roles
- Resources
- Multiple Projects
- Subprojects
- User groups
- Project Groups
- Private vs. Public Projects
- Project Categories

We begin with simple configurations and then explore ways to implement more complicated configurations. We assume that you are a CEE administrator with sufficient access to create new projects, add users, grant roles, etc. (In other words, you have been granted the Domain Admin role on your CEE system.)

Note that this document describes CEE version 5.1, earlier versions of CEE may not have the features described here, while later versions may have additional features, or slight changes in these features. For more information on the particular version of CEE you're using, please read the CEE on-line help documentation.

CEE Users and Roles

A Simple Example

We start with the simplest possible CEE configuration: a single project with only a single person using the project. We assume that no one else has access (or should have access) to the project. In CEE you can create this configuration as follows:

- Create the project and specify that it be created as a top-level project and as a private project. (In other words, select the "New top-level" item for the "Parent project" field, and leave unchecked the "Public project" checkbox.)

* Project name	<input type="text" value="new-project"/>	Identify the project with a one-word name. Use characters a-z (mixed case will be converted to all lowercase), 0-9, and dash (though not as first or last character). No spaces, underscores, or other punctuation are allowed. The project name cannot be changed, so choose wisely.
* Summary	<input type="text" value="Demo project"/>	Enter a short description of your project in plain text. HTML and formatted text are not supported.
Parent project	<input type="text" value="New top-level"/>	Select this project's parent, or "New top-level" to make a new top-level project.
Project categories	<input type="text" value="departments"/> pcs-offerings (in pcs) pcs (in departments) pcs-commercial (in pcs-offerings) engagements products	Select one or more categories to determine where your project appears in project listings. You can find descriptions on the category home page. You may also change categories at any time.
Public project	<input type="checkbox"/>	Check this box to make this project publicly viewable to the site at large.

- Create a single CEE user-id and assign the user-id to the person to be using the project.
- Add that user-id to the project and grant the user-id the Project Owner role (or another appropriate role) within the project.

Add new member/role
[Invite new members](#) | [Add new member/role](#)

Filter this list:

Add	User	Full name
<input checked="" type="checkbox"/>	jreynolds	Jeff Reynolds

* = Required fields

Mass add

If you know the usernames of the users to whom you want to assign roles in this project, you may enter them here to do a mass add (one username per line, or multiple usernames if a comma, colon, or semicolon is used to separate them).

* **Grant these roles to all designated users**

- IZ Administration
- IZ only
- Observer
- Observer in Briefing Center and Marketing Extranet
- Project Manager
- Project Owner**
- Project Tracker Administrator
- ProjectDocumentView
- PT Query
- ViewOnly

Done granting roles

When this box is checked, you will return to the project membership page after clicking the "Grant roles" button. If you possibly want to add more members, remove the checkmark to return to the Add new members page.

The person using the project could then login to CEE under their assigned user-id, after which they would have all access to all resources available as part of the project: the version control repository, the file sharing area, the issue tracking system, and so on.

(If you grant the person the Project Owner role then they will also be able to reconfigure the project itself in various ways. If you wish to prevent the person from doing this, assign the person the Developer role instead.)

Also note that the creator of the project is automatically assigned the project roles that are configured to 'Grant role on subproject creation' in the Administration → Roles area.

Multiple Users in a Project

Continuing with our example of the single-person project, let's suppose that a second person wants to use the project resources, and that we wish to permit this person exactly the same level of access as the first person who was granted access.

To do this, assign the second person a second CEE user-id, and then let the first and second persons each login to CEE using their own respective user-ids. Using separate user-ids has the following advantages (among others):

- Using separate user-ids allows better tracking of people's actions within the project. For example, if the two people use different user-ids and make changes to the version control repository then it is possible to determine who checked in a particular change.
- Using separate user-ids allows people to file issues against each other. For example, if the two people in our example divide up the project tasks between each other, one person can use the project or issue tracking system to assign the other person a task relating to that person's area of responsibility.

Create the new user-id and add the new user to the project as described above, and grant the user-id the same CEE role as the first person, e.g., Project Owner in our example.)

Roles

To continue with our example, suppose that we want a third person to have access to the project, and a fourth person, and so on. If each person is to have the same access to the project as the first person, then doing this is simple: We create a new CEE user-id for each new person and add that user-id to the project, exactly as described above for the second person.

This presumes, however, that all the people working in the project are to have the exact same access. What if we want some people to have different access than other people? For example, we might want some people to be able to modify the version control repository (e.g., to “check in” changes), and other people to be able only to read from the repository (e.g., to “check out” code).

To do this we can use CEE roles, and assign different roles to different users. A *role* is essentially a set of permissions specifying what a user can do within a project; the permissions are expressed in terms of *actions* that the user can perform with respect to each of the resources within a project.¹ For example, a person needing the ability to change the contents of the version repository would need permission to add new files to the repository (the “VersionControl – Add” action in CEE terms), permission to make changes to existing files in the repository (“VersionControl – Modify”), and so on.

Permissions
Action
Discussions - Discussion Add
Discussions - Discussion Use Private
Discussions - Discussion View
Discussions - Message Post
Private Project Document - View
Project - View
Project Content - View
Project Dashboard - View
Project Document - Edit
Project Document - Reserve
Project Document - Reserve Mine
Project Document - Suggest
Project Document - View
Project Issue Tracking - Add Comment
Project Issue Tracking - Assignable

Rather than requiring each of these permissions to be granted separately, CEE bundles all the necessary permissions into a role, which can then be granted to a particular user. For example, the standard CEE Developer role includes all the permissions necessary for a person who makes

¹ Roles can also define permissions corresponding to actions taken at the CEE domain level, such as adding new users. However in this document we primarily consider permissions for actions taken within projects, since our goal is to describe ways to organize projects and users within projects.

changes to a project’s version control repository. The Developer role also includes other permissions useful for people who do code development, for example, permission to upload files to the file sharing area or to make changes to issues previously submitted to the issue or project tracking system. By granting to a particular user the Developer role within a project, we ensure that they have all the necessary permissions they will need, and do not need to specify those permissions individually.

Continuing with our example, suppose that we wish to have twenty people participate in the project, with five of the people having relatively full access to all project resources, and the other fifteen people having primarily “read only” access to those resources. We can assign the five people a CEE role giving relatively full access (e.g., the Developer role, or perhaps even the Project Owner role), while assigning the other fifteen people a CEE role giving relatively limited access (e.g., the Observer role).

In CEE terms you can do this as follows:

- Create the project as a top-level private project as described previously.
- Create a CEE user-id for each of the twenty people, as described previously.
- Add each user-id to the project. If the user-id corresponds to one of the five people with relatively full access, then grant to the user-id the Developer role (or an equivalent role) within the project. On the other hand, if the user-id is for one of the fifteen people with limited access, then grant to the user-id the Observer role (or an equivalent role) within the project.

As noted above, the default CEE configuration provides a set of standard roles that can be granted to users, including the roles Project Owner, Developer, Content Developer, and Observer. (See Appendix A for more information on all the roles included in the default CEE configuration.) If you need to grant to a group of users a set of permissions that do not exactly correspond to those of the standard CEE roles, you can either modify the permissions for one or more of the standard roles or create one or more new roles. For more information on modifying or adding roles, see the CEE online help documentation.

Membership
[Invite new members](#) | [Add new member/role](#)

Filter this list Username contains

User	Full name	Roles
ben_cn	Ben Analyst	<input type="checkbox"/> Business Analyst
dana_cn	Dana Developer	<input type="checkbox"/> QA/Test
dan_cn	Dan Java	<input type="checkbox"/> Developer
devki_cn	Devki Chopra	<input type="checkbox"/> Developer
jreynolds	Jeff Reynolds	<input type="checkbox"/> Developer <input type="checkbox"/> Project Manager <input type="checkbox"/> Project Owner
pam_cn	Pam Project	<input type="checkbox"/> Project Manager
rwoodfine	Roy Woodfine	<input type="checkbox"/> Project Owner

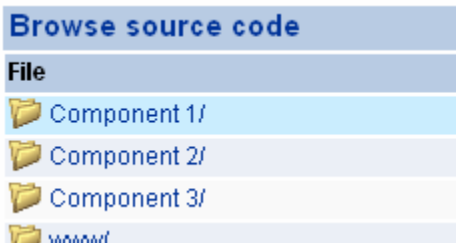
Internally Partitioning Projects

Our example configuration now consists of a single project with multiple users, with each user assigned to one of several possible roles (in the CEE sense) within that project. In the next section we will expand our discussion to include multiple projects. However before doing that we discuss the advantages and disadvantages of centralizing all activities and users in a single project.

One major advantage of using a single project is that this minimizes the number of places users have to go to in order to get their work done. For example, if all code is stored in a single project then users can use a single version control checkout operation to get a copy of all the code.

Even if you are working with multiple different components and multiple areas of activity, you can still consolidate those components and areas within a single project, by doing the following:

- Create separate directories (or subdirectories) within the version control repository, and store separate collections of source code in different directories.






- Create different components (or subcomponents) within the project or issue tracking system, so that new artifacts can be associated with a particular component.



- Create multiple discussions, one for each component and/or area of activity.

Discussions

[Add new discussion](#) | [Manage my subscriptions](#)

Title	
 Component 3 *	Component 3 Development Topics
 Component 2 *	Component 2 Development Topics
 Component 1 *	Component 1 Development Topics

- Create multiple folders (or subfolders) within the file sharing area, so that documents for different components or areas of activity can be stored in different folders.

Documents & files: Component 3

[Add new folder](#) | [Add new file](#) | [Edit folder](#)



Such a scheme allows you to impose a considerable degree of internal organization on a project, while still allowing users easy access to project resources. Note that with one exception (described below) this internal partitioning scheme does *not* support specifying different access controls for different project resources of the same type. Thus, for example, although you may organize the project file sharing area to create different folders and subfolders, corresponding to different components within the project, a given user would have the same access to all documents no matter in which folder or subfolder they were located.

However it *is* possible to place different access controls on different components of the code in the project's version control repository. For example, you may have multiple development teams and may want to allow any particular team to change the code only for the components for which they are responsible. You can do this in CEE within a single project using CEE resources, as described below.

Resource Patterns

CEE *resources* can be used to specify particular directories and/or types of files within a project's version control repository, and then to customize permissions to refer to those directories and/or files. For example, suppose that within a single project we would like to maintain four separate collections of source code. We assume that the various collections of code are part of a single application, so project developers will be checking out and using all four collections together; however we also assume that the developers have been divided into four teams, one for each collection of code, and we want to ensure that a team can make changes to its own collection, but not to collections associated with the other teams.

You can create such a configuration with CEE as follows:

- Create four top-level directories within the project's version control directory; for this example we'll refer to these directories as "alpha", "bravo", "charlie", and "delta".
- Create four new CEE resource patterns associated with the project, one for each directory: "alpha/*", "bravo/*", "charlie/*", and "delta/*".²
- Create four new CEE roles using the Developer role as a model; in this example we will name the roles "AlphaDeveloper", "BravoDeveloper", "CharlieDeveloper", and "DeltaDeveloper". For each role, set the appropriate version control related permissions to allow commits and other related actions³ only for the corresponding resource patterns. Thus, for example, the AlphaDeveloper role would be allowed to do read-only operations on any part of the repository (represented by the "/*" resource), but allowed to do commits and related actions only for the directory

² In some CEE versions these resource strings should be preceded with a slash character, e.g., "/alpha/*".

³ In CEE terms the permissions in question are for the "Add", "Commit", "Edit", "Import", "Release", "Remove", and "Tag" actions against the project repository, i.e., the actions "VersionControl - Add", "VersionControl - Modify", and so on.

represented by the "alpha/*" resource. Note that resource patterns are additive. Giving a role multiple resource patterns will allow users playing that role to have the corresponding accesses to multiple areas within version control.

- When adding developers in the "Alpha" team to the project, grant to them the AlphaDeveloper role, and similarly when adding developers in the other teams grant to them the role associated with that team. If a developer needs to be able to change code in two collections, grant to them the roles corresponding to both collections and teams. If a developer needs to be able to change code anywhere in the repository, grant to them the standard Developer role.
- The four permissions that are used the most often in conjunction with resource patterns to limit access to the version control directory structure are:
 - VersionControl - Add
 - VersionControl - Delete
 - VersionControl - Modify
 - VersionControl - Read

Note that a similar scheme is used to distinguish the standard CEE roles Content Developer and Developer: As with the Developer role, being granted the Content Developer role allows a user to check out a copy of the entire repository; however the Content Developer role allows a user to commit changes to the repository only for files in the "www" directory used by CEE to store the project's web pages.

<input type="checkbox"/>	VersionControl - Add (Core)	All web pages: (/trunk/branches/[*/]*)?/www(\$ /.*)
<input type="checkbox"/>	VersionControl - Copy (Core)	All web pages: (/trunk/branches/[*/]*)?/www(\$ /.*)
<input type="checkbox"/>	VersionControl - Delete (Core)	All web pages: (/trunk/branches/[*/]*)?/www(\$ /.*)
<input type="checkbox"/>	VersionControl - Modify (Core)	All web pages: (/trunk/branches/[*/]*)?/www(\$ /.*)
<input type="checkbox"/>	VersionControl - Read	All applicable resources: .*

You can also provide finer-grained control over who can read and/or modify web pages associated with the project, by defining different roles to have different access to resources corresponding to subdirectories of the "www" directory in the project's repository.

Organizing Multiple Projects

Previously in this document we discussed organizing users within a single CEE project, using CEE roles and resource patterns to give different users different levels of access to project data. However when an organization has more and more users and more and larger collections of source code and documents, the need for an understandable organization of project-related material and for differential access to that material increases to the point where it is easier to create multiple CEE projects than to try to organize all activities within a single project.

You may encounter two typical situations when you consider how to organize multiple projects in CEE:

- When attempting to separate material and users into separate projects, you may find that the projects are (for the most part) unrelated to and independent of each other. In this case it is also likely that the people who need access to a particular project are (for the most part) different than the people who need access to any other project. (In other words, each project may have a set of users mostly unique to it and not shared with any other project.)
- On the other hand, you may find that the projects are (for the most part) related to each other, and can be thought of parts within a larger whole, i.e., as individual efforts within an overall initiative encompassing all of them. In this case you may also find that there is a core group of people who need some basic level of access

to all projects, with some of those people needing a higher level of access to only a few of those projects.

Where projects are not (for the most part) related to each other and do not necessarily have the same users, we can treat the projects independently, creating each one as a top-level project. Where projects are parts of a larger whole and may share a core group of users, we can organize the projects as subprojects of one overall project. We discuss both these approaches in the following sections.

Multiple Independent Projects

In this section we consider the case where projects are not related to each other and each project's users are (mostly) unique to it and are not shared with any other project (at least, not shared to any great degree).

For example, suppose that we have three disjoint groups of developers and other users within an organization: The three groups are working on development projects that are completely independent of each other, so that the groups do not share code or other data or otherwise interact with each other. However the groups wish to share the services of a single CEE system provided as a central service for use by anyone in the organization.

The most straightforward approach to meeting the groups' needs is simply to create three separate CEE projects, and assign people to each project as appropriate:

- Create the first project as a top-level project and as a private project.
- Repeat the process to create the second and third projects.
- Create a CEE user-id for each person in the three groups to be using the projects.
- For each person in the first group, add their user-id to the first project and grant an appropriate role to the user-id: Developer, Observer, or whatever other role is appropriate given the part that person will play in the project.
- For each person in the second group, add their user-id to the second project and grant an appropriate role to the user-id as previously discussed.
- Repeat the process for everyone in the third group, adding each person's user-id to the third project and granting to them appropriate roles.

Even though everyone in this example has a user-id on the same CEE system, any given person has access only to the project to which that person's user-id has been added, and within that project the person has permissions only for those actions allowed by the person's assigned role. For example, a given person might have a user-id that was granted the Developer role in the first project, allowing the person to add and modify data in the project's version control repository; however the person would not have access to any data associated with the second or third projects. (In fact, since in our example the projects are private and not public projects, the person in question would not even see the other projects listed under the "Projects" page.)

We could continue to add additional projects and users as desired, so that a given CEE system might contain dozens or even hundreds of independent projects, each with its own group of users.

User Access to Multiple Projects

In the above example all the projects were assumed to be independent and to have their own exclusive user communities, so that a person working on one project would not have access to any other project on the same CEE system. However in some cases we may wish to have a given person participate in multiple projects.

This is quite simple to do: Simply add the person's user-id to each of the other projects, exactly as was previously described, and grant to the user-id an appropriate role within each project. Note that a given user-id may be granted different roles in different projects. For example, a given person may have the CEE Developer role in one project, but have only the Observer role in another project.

A Project with Subprojects

In the previous section we discussed configuring access where projects are not related and each project's users are mostly unique to it and not shared much with any other project. In this section we discuss the situation when we have multiple projects that form part of a larger initiative and most of the users may be in a core group that needs access to all projects.

Where projects are related to each other and/or share a large core group of users, you can create a top-level CEE project and then create other projects as *subprojects* of the top-level project, with one set of activities per subproject. You can then grant roles in the top-level project to the core group of users, and depending on the role configuration, they will then also have those roles in each of the subprojects. (As with a single project, you can grant different roles to different users.) Users assigned to roles which do NOT have 'Block recursion into private projects' checked will be able to access sub-projects under the top level project. They will inherit the permissions associated with the role(s) they are playing in the top level project. Users playing roles with the 'Block recursion into private projects' checked will not automatically gain access to the sub-projects.

Previously in this document we discussed a situation where twenty people needed access to a project, with five people in the Developer Role and the other people in the Observer role. Continuing with that example, suppose that this group's company has an initiative in which it works with ten other companies acting as partners on individual projects within the initiative, one project per company. In this case the core group of 20 people will need access to the initiative as a whole and to these ten projects within the initiative, with each individual project also being accessed by one or more representatives of the company acting as a partner for that project.

You can create a suitable configuration for this in CEE as follows:

- Create an initial project as a top-level private project as described previously. This project will serve as a parent project for the other projects, and can be used to manage material and activities related to the initiative as a whole.
- Create a CEE user-id for each of the twenty people, as described previously.
- Add each user-id to the top-level project as described previously, granting to each user either the Developer role or the Observer role as appropriate.
- Create a project for the joint work with the first partner company, as a private subproject of the top-level project. (In other words, select the "initiative" project for the "Parent Project" field, and leave unchecked the "Public Project" checkbox.)
- Create CEE user-ids for the representatives of the first partner company, as described previously.
- Add each of the partner company's user-ids to the subproject, granting to them an appropriate role (e.g., Observer or Developer) in the subproject. (Be careful to ensure that you have the subproject properly selected first before adding user-ids.)
- Repeat the previous three steps for each of the remaining partner companies.

In this configuration each partner company's representatives can access only the particular (sub)project in which they have been granted roles; they cannot access other subprojects, nor can they access the top-level project. (In fact, since both the top-level project and its subprojects are

private, the partner company's representatives will not even see the top-level project or the other subprojects listed on their "My start page" or "Projects" pages.)

The members of the 20-person group can access the top-level parent project for the initiative as a whole, and any of the materials and activities associated with that project. Based on the roles they are playing, they can also access each of the ten subprojects, and will have the same role within those subprojects as they were granted in the top-level project – in other words, roles in the subprojects are inherited from roles in the parent project.⁴ (In CEE such an inherited role is a special case of what are referred to as *derived roles*, as opposed to *direct roles* that are explicitly granted to user-ids within a project. We will consider other kinds of derived roles later in this document.)

Using Inherited Roles

Note that when a user's role in a project is inherited from a parent project the CEE interface appears differently than when the user's role is directly granted within the project itself. (This is true for derived roles in general, as discussed in subsequent sections.) In the example, although members of the core group have (inherited) roles in the ten subprojects, the subprojects do not appear on the users' "My start page", and thus cannot be accessed from that page. To access one of the subprojects, a core group member could go to the top-level parent project (from either their "My start page" or the "Projects" page) and then access the desired subproject using the appropriate link under the "Subproject" section of the parent project's home page.

Also note in our example that when a core group member accesses the home page for a subproject, the subproject's home page will display a "Join this Project" link, even though the member already has a role in the subproject inherited from the parent project. This behavior occurs because a user is considered to have "joined" a project only if the user has been directly granted a role in the project and is on that project's membership list. However in this case there is no need to join the project explicitly, unless the user would like to request a different role in the project; the user can simply ignore the "Join this Project" link and proceed to access project functions as the user would normally do.

Finally, note that the subproject's home page does not show a user's role in the project unless the user has actually joined the project. In order to see what inherited role they might have in a project, users can look at their own profile (using the "Edit profile" link in the top navigation bar) on the "Edit user" page, follow the "Direct and derived roles" link (in the "Detailed role info" section of the page) to go to the "User roles" page, and then invoke the "Show derived roles" operation. The resulting page displays the user's derived roles for all projects on the CEE system (in the "Derived roles" section of the page).

Further Customizing Access in Subprojects

Continuing with our example, suppose that a given member of the core group has the Observer role in a particular subproject, having inherited that role from the parent top-level project; however suppose that instead the user in question actually needs to have greater access to the project, e.g., the access associated with the Developer role.

This can be easily done: We can simply add the person's user-id directly to the subproject in question, and grant to the person the Developer role in that subproject. The person will then have two roles in the project: the Developer role (directly granted) and the Observer role (inherited from the parent project). The user's permissions in the project are the union of the permissions from the two roles; in other words, the person can do anything allowed by the Observer role, as well as any-

⁴ Note that this is not necessarily true in all cases: as discussed later in this document, a particular role can be configured so that it is not inherited in a private subproject (which happens to be the type of the projects in this example). However the standard CEE Developer and Observer roles used in the example are inherited in all subprojects, private or public.

thing allowed by the Developer role. (Since the permissions associated with the Developer role are a superset of the permissions for the Observer role, in effect it is as if the user had only the Developer role.)

Note that in this case the user is considered to have joined the subproject (since they were directly granted a role in the subproject) and thus the user will now see the subproject listed on their “My start page”. (However they will still not see subprojects listed in which they have only an inherited role.)

User Groups and Project Groups

In the previous section we discussed two general ways to organize users into multiple projects and control access to those projects:

- For situations where the projects are part of a larger initiative and most of the users may need access to all the projects, you can create the projects as subprojects under a top-level parent project, and add the core group of users as members to the top-level project. The users can then access the subprojects using roles inherited from the parent project. In this scenario you may also have some users who should have access to only one or a few of the subprojects; they can be treated as exceptions, and explicitly granted roles in one or more of the subprojects as needed.
- For situations where the projects are independent and each has its own separate user base, you can create the projects as separate top-level projects, and add each person’s user-id to the particular project in which they will be participating. In this scenario you may also have some users who should have access to multiple projects; they can be treated as exceptions, and added to multiple projects as needed.

However when creating real CEE configurations you often need to create more complicated schemes for access control. In this section we discuss two CEE features allowing more extensive customization of access control: user groups and project groups.

User Groups

Previously in our examples we have granted access to projects on a user-by-user basis; in other words, to give to a person access to a project we either explicitly granted the person’s user-id a role in the project itself, or (in the case of subprojects) explicitly granted their user-id a role in the project’s parent project, with that role then being inherited in the (sub) project.

If your CEE system has a large number of users then granting access user by user can be time-consuming at times, especially if the users must have access to multiple independent top-level projects, where the projects have no parent project from which roles can be inherited. To help address this problem CEE allows you to create *user groups* consisting of multiple user-ids, and then to grant access by user group instead of by user-id.

User groups are useful when there are a number of users who need to participate in multiple projects, and the users can be classified into more than one distinct group, with each group needing a particular level of access. For example, suppose that we have a central 10-person “security team” consisting of developers who develop security-related code for several projects and who are responsible for responding to reports of security-related vulnerabilities in each project’s software.

We could add each of the ten people individually to each of the projects they are to work in, as described above, but this would require a lot of CEE administrative operations. In addition, we

might want to distinguish the security team members in some way from the normal project developers.

We can address both these issues by creating a CEE user group containing a set of user-ids that can be treated as a unit when it comes to granting roles. In our example we could create a user group "SecurityTeam" consisting of the ten user-ids for the people on the security team. We could then add the SecurityTeam user group to each of the several projects in which the team is to participate, assigning to the SecurityTeam user group the Developer role in each project (or some other appropriate role).

You can do this in CEE as follows:

- Create CEE user-ids for all the people in the security team group, as previously described.
- Create a new CEE user group; specify the name of the new user group as "SecurityTeam", and add a brief description of the group.
- Add the user-ids to the user group

Add new user group

* = Required fields

Group name	<input type="text" value="SecurityTeam"/>
	Up to 32 characters in length; valid characters are a-z, A-Z, 0-9 and underscores. Spaces are not permitted, and an underscore may not be the first character. Example: eng_group
Short description	<input type="text" value="Developers who develop security-r"/>
Initial users	<input type="text" value="dan
joe
mary
frank
jose"/>
	Enter users to be included either by their username, email addresses, or full names; one per line or separated by commas.

- For each of the projects in which the security team is to have access, add the user group (*not* the individual user-ids) to the project, and grant to the user group the Developer role.

If the user group SecurityTeam has been granted the Developer role in a given project, then all of the user-ids that are part of that user group have the Developer role within that project, and can perform any action associated with that role.

Note that when a user has a role in a project solely by virtue of being in a user group granted that role, that role is considered to be a derived role as discussed above, not a direct role. Thus within that project the CEE interface will appear differently to the user than it does when the user's role is directly granted within the project itself, as discussed earlier. So, for example, suppose a given person's user-id is in the SecurityTeam user group, that the SecurityTeam user group has been granted the Developer role within a given project, and that the person's user-id itself has not been granted any direct role in that project. In that case the CEE interface will appear as follows to the user:

- The project in question will not be listed on the user's "My start page" if the project is private. However the user can access the project by going to the "Projects" page, by directly entering in the project's URL into the browser's location field, or by other means discussed below.
- The project's home page displayed to the user will have a "Join this Project" link, even though the member already has a role in the project as a result of being in a

- user group granted a role in the project. However the user can simply ignore the “Join this Project” link and proceed to access project resources, as they would normally do.
- The project’s home page will not display the derived role that the user has in the project. However the user can determine their role in the project using the “Show derived roles” function of CEE, as described earlier.

As we discussed in connection with inheriting a role in a subproject, a user might have a derived role in a project and also have a direct role as well.

Returning to our example above, consider a member of the security team who has the (derived) Developer role in a given project as a result of that person’s membership in the SecurityTeam user group. Suppose that this person is then also placed in charge of the project activities, and is granted the Project Owner role in the project; this person then will have a direct role in the project (Project Owner) and also a derived role (Developer).

As we discussed previously, when a person has both a direct role and a derived role in a project, the user’s permissions in the project are the union of the permissions from the two roles; in other words, in our example the user can do anything allowed by the Developer role, as well as anything allowed by the Project Owner role. (As it happens, the permissions associated with the Project Owner role are a superset of those for the Developer role. However this may not be true in general when a user has two different roles; in fact, the two roles might grant completely different sets of permissions.)

Multiple User Groups

Although in our example thus far we have mentioned only a single user group, in fact a given user can belong to multiple user groups. For example, suppose that we have a separate group of Quality Assurance managers who oversee QA-related activities for multiple CEE projects associated with several development teams. We could create a user group QAManagers containing the user-ids of all the QA managers, and then add the QAManagers user group as a member to all the projects overseen by the QA managers, granting the user group the Observer role in those projects.

It may be that a given QA manager is also a member of the security team; in that case the QA manager would be a member of both the SecurityTeam user group and the QAManagers user group. It might happen that in a given project the SecurityTeam user group is granted the Developer role and the QAManagers user group is granted the Observer role. In this case our example QA manager would have two derived roles in the project: the Developer role (as a consequence of being a member of the SecurityTeam user group) and the Observer role (as a consequence of being a member of the QAManagers user group). As previously discussed, the QA manager’s permissions in the project would be the union of the permissions from the two roles.

In general there can be many user groups defined in a CEE system, and any user could be a member of multiple user groups. However note that user groups cannot contain other user groups as members, but only users (i.e., user-ids). Thus in our example suppose that we had a third user group, Managers, consisting of all managers, including development managers, QA managers, etc. Because of the restriction that user groups can have only users as members, we must first add all the QA managers’ user-ids to the QAManagers user group, and then separately add those same user-ids to the Managers user group; we cannot simply add the QAManagers user group as a member of the Managers user group.

Project Groups

We previously discussed organizing projects as subprojects under a parent project, with users granted roles in the parent project then inheriting those roles in the subprojects. Although a parent project may have multiple subprojects, a given project can have only one parent project.

While this arrangement works well for many situations, some real-life situations are too complex to model using a standard parent project-subproject scheme.

In the previous section we discussed an example where we created two user groups: a SecurityTeam group and a QAManagers group, each containing a specific group of users (i.e., user-ids). In the example we explicitly granted to the SecurityTeam user group the Developer role in each and every project in which the security team members were to participate, and explicitly granted to the QAManagers user group the Observer role in each and every project to be overseen by QA managers.

It would be nice if we could avoid having to add the user groups explicitly to each and every project in which their members are to participate, in order to reduce the amount of CEE administrative work that needs to be performed. So, for example, we could take all the projects in which the security team is to participate, and make those subprojects of a parent project; we could then explicitly grant to the SecurityTeam user group the Developer role in the parent project. The security team members would then have the (derived) Developer role in the parent project (because their user-ids are in the SecurityTeam user group), and they would inherit the Developer role in all of the subprojects of the parent project. (In general any role which a user has in a project, no matter how obtained, will be inherited as a derived role in a subproject of that project.⁵)

Similarly, we could take all the projects that the QA managers are to oversee, and make those subprojects of a parent project; we could then explicitly grant to the QAManagers user group the Observer role in the parent project. The QA managers would then have the (derived) Observer role in the parent project, and would inherit that role in the subprojects.

Ideally we'd like to combine these two schemes. For example, we could have a single parent project in which we'd grant the Developer role to the SecurityTeam user group and grant the Observer role to the QAManagers user group. However this assumes that all the subprojects have both the security team and the QA managers participating in them, since both the security team members and the QA managers will have derived roles (of Developer and Observer respectively) in all the subprojects.

On the other hand, we could have two separate parent projects: In one parent project we would grant the Developer role to the SecurityTeam user group, and in the other parent project we would grant the Observer role to the QAManagers user group. However this assumes that no project has both the security team and the QA managers participating in it, since a project can have only one parent project: If we create the project as a subproject of the parent project in which the SecurityTeam user group has been granted the Developer role, we cannot also create the project as a subproject of the (separate) parent project in which the QAManagers user group has been granted the Observer role.

Thus we have a problem: What if some projects have only the security team participating in them, while other projects have only the QA managers participating in them, and some projects have participation by both the security team and the QA managers? How can we configure the CEE system to address this situation?

The *project group* feature of CEE provides an answer to our problem. Project groups are a generalization of the parent project/subproject feature we've already discussed: Like a parent project, a project group is a project that can contain other projects as members. As with a parent project and its subprojects, any direct or derived role that a user has in a project group is inherited in the projects that are members of the project group. However there is an important difference be-

⁵ There is a minor exception to this rule, as mentioned in a previous footnote and discussed later in this document; however the Developer and Observer roles follow this rule.

tween project groups and parent projects: A given project may have only one parent project, but it can be a member of multiple project groups.

Project groups

Name	Summary	Subgroups	Projects
cdn-core-projects	All the main SourceCast components	0	21
collabnet-engineering	Group for Engineering Projects	0	40
collabnet-group	Your portal to the other CollabNet projects	0	134
collabnet-sales	All projects serving 'sales' and 'pre-sales' teams	0	21
sourcecastnetwork	components in the sourcecast suite open to sourcecat gated commu	0	15

We can use the project group feature in our example as follows: We create one project group (call it “SecurityTeamProjects”) containing all the projects in which the security team is to participate, and in that project group we grant to the SecurityTeam user group the Developer role. We create a second project group (call it “QAManagerProjects”) containing all the projects that the QA managers are to oversee, and in that project group we grant to the QAManagers user group the Observer role. Any given project may be a member of the SecurityTeamProjects project group, a member of the QAManagerProjects project group, or a member of both project groups. If a project is a member of both project groups then members of the security team will have the (derived) role of Developer in that project, while QA managers will have the (derived) role of Observer in the project.

In CEE you can do this as follows:

- Create the SecurityTeamProjects project group as a new top-level project group. In the “Initial projects” field specify the names of the projects that are to be members of the SecurityTeamProjects project group, and then invoke the “Create group” operation.
- Add the SecurityTeam user group to the just-created project group and grant to it the Developer role.
- Repeat the steps above to create the QAManagerProjects project group, specifying all the projects that are to be members of that project group, and then add the QAManagers user group to the project group, granting to it the Observer role.

Note that a given project may be a subproject of a parent project, and also at the same time be a member of one or more project groups. You can create a parent project to organize multiple sub-projects with similar access control requirements, and then use project groups in addition to take care of special cases where some groups of people need access to some (but not all) of the sub-projects.

Previously in this document we discussed the example of a company that had a group of twenty people working together with representatives of ten partner companies. In the example we assumed that each of the ten partner companies worked on one and only one project, a project different than those in which other partner companies participated; we created each of the ten projects as subprojects of a top-level parent project, and we granted to each of the representatives of the partner companies a role in the particular subproject associated with their company.

However suppose that each partner company might actually participate in several projects with the original company, and that in some cases two or more partner companies might work with the original company on a single project. How can we create an appropriate access control scheme in CEE?

We can start with the scheme discussed in the previous example: We create a top-level parent project, and grant roles in that parent project to the twenty people in the original company. (We could also create two user groups for the twenty people in the original company, one user group for the five people with observer status and a second user group for the fifteen developers; however this is not necessary for the purposes of this example.) We then create all the partner projects as subprojects of the initial parent project. In this example we might have dozens or even hundreds of such projects in total. As in the previous example, the twenty people in the original company will have (derived) roles in all the subprojects, either the Observer role or the Developer role depending on which role they were granted in the parent project.

As mentioned above, each partner company might participate in several projects, and several partners might participate in a single project. We therefore create a project group for each project; for this example we'll refer to these project groups as "partner1-projects", "partner2-projects", and so on, through "partner10-projects". (Recall that there are ten partner companies in the example.)

We then add all of the first partner's representatives as members in the partner1-projects project group, granting to them the Observer role. If we'd like we can create a user group Partner1Employees consisting of the user-ids for all the representatives of the first partner, and grant the user group Partner1Employees the Observer role in the partner1-projects project group. However this is not strictly speaking necessary: We could instead just add to the partner1-projects project group all the individual user-ids for the representatives of the first partner.

We repeat this process for the second partner, adding all of that partner's representatives as members of the partner2-project group and granting to them the Observer role in that project group. (Again, we could create a user group for the second partner's employees, or just add individual user-ids to the project group.) We repeat this process for the third partner, and continue until we have added all of the partner representatives to one of the ten partner project groups.

We then consider each individual project in which partners are to participate (i.e., the subprojects of the parent project created earlier). If a particular partner is to participate in a project, we add that project to the project group associated with that partner. For example, if the first partner is to participate in a project "abc" then we add "abc" to the "partner1-projects" project group. All the representatives of the first partner will then have a derived role of Observer in the project "abc", a role inherited from the role they were granted in the "partner1-projects" project group.

If two or more partners are to participate in a given project then we make that project a member of multiple project groups. So, for example, if both the second partner and the fifth partner are to participate in a project "xyz" then we add "xyz" to the "partner2-projects" project group, and then also add "xyz" to the "partner5-projects" project group. (Recall that projects can belong to multiple project groups, although they can have only one parent project.) The representatives of the second partner will then have a derived role of Observer in the project "xyz", a role inherited from the role they were granted in the "partner2-projects" project group; the representatives of the fifth partner will also have a derived role of Observer in the project "xyz", a role inherited from the "partner5-projects" project group

Public Projects

Thus far we've discussed access control on private CEE projects, i.e., projects for which the "Public project" checkbox was *not* checked when creating the project. As we discussed above, private projects are not generally visible to users of the CEE system on which the projects are located. The exact rules are somewhat complicated, but in general the existence of a private project can be discovered through the standard CEE interface only by those users who are in the actual membership list for the project, or by those users who have a derived role in the project. (For users who are on the membership list for a top-level private project, the project would be listed on the users' "My

start page”, as well as on the “Projects” page. For users who have only a derived role in the project, the project would be listed only on the “Projects” page, and not on the “My start page”).

In some cases we may wish to make the existence of the project known to every CEE user on the system, and allow users to view basic information about the project and then apply for membership in the project if desired. For example, we might want to create a project “utilities” to develop and distribute utility software that might be useful to any user on the CEE system.

We can create such a project as a top-level public project. (In other words, when creating the project select the “New top-level” item for the “Parent project” field, and check the “Public project” checkbox.)

The “utilities” project will then be visible to any CEE user through a listing on the “Projects” page. By following the project’s link any user may also view the project’s home page and view the project’s file sharing area, source code, issues, and other information. (We discuss below in more detail how access control works with public projects.) If they wish, users can make a request to join the project and be granted a specific role (e.g., Observer or Developer); their request is sent to the project owner(s), who can then approve or deny the request.

The Registered User Role

We noted above that any user of the CEE system can access the home page of a public project,⁶ and can also access various content areas within the project. In fact, any user accessing the project has most of the access that a user with the Observer role would have; the major exceptions are that a user must have (at least) Observer status to be able to subscribe to project discussions or submit issues to the project’s issue tracking system.

Such default access to public projects is made possible through a special Registered User role. Each new user-id created in the CEE system is immediately granted the Registered User role, prior to the user being added as a member of any projects, project groups, or user groups. A user will then have the Registered User role in any⁷ public project, even if their user-id has no other direct or derived role in that project.

In essence it is as if there were a project group consisting of all public projects in a CEE system, with all user-ids granted the Registered User role in that project group. Any user therefore has the ability to view public projects (in that unnamed project group), by virtue of having inherited the Registered User role within that project.

Private Projects vs. Public Projects

What about private projects? The Registered User role is configured so that it will *not* be inherited in private projects. (In CEE terms, the Registered User role has the “Block recursion into private projects” checkbox checked.) Thus any given user will *not* automatically have access to any private project, whether top-level private projects and private subprojects of public projects. As previously discussed, in order to access a private project a user must be granted at least the Observer role (or an equivalent role) in the project, or must otherwise have a derived role of least Observer (or equivalent) in the project.

In fact, this is probably the most important difference between the Observer role and the Registered User role (which otherwise have similar permissions): the Observer role is inherited in private subprojects, and the Registered User role is not. Thus if a user has the Observer role in a given project (or project group) then that user will have (at least) that same role in any subproject (or mem-

⁶ This is not quite correct; as noted elsewhere in the document, users cannot access a public subproject of a private project unless they have access to the private project.

⁷ Or almost any – see the discussion below.

ber of the project group), whether public or private. However if a user has only the Registered User role in a given project (or project group) then that user will have that role only in subprojects (or members of the project group) that are public.

Incidentally, note that registered users will not automatically have access to a public subproject of a private project (or a public project in a private project group). That's because the Registered User role is not inherited into the private project (or project group), as discussed above, and thus is not available to be inherited into any subprojects (or members of the project group), even if they were explicitly marked as public.

"Semi-Public" Projects

Thus under ordinary circumstances it does not make sense to create public subprojects of a private project (or add public projects to a private project group). One exception is if you have the following environment and requirement:

- Within a larger CEE system you have a group of projects that are in general restricted to a specific group of people (call them "authorized users"), but not every person in the group has access to all the projects.
- You wish to make certain sets of documents and/or source code generally available to everyone in that specific group of people, while protecting the information or code from everyone else.

You can address this requirement by creating what we'll call "semi-public" projects – essentially imitating what the standard CEE system does with public projects and the Registered User role, but within a more private context. The specific steps to do this are as follows:

- Create a private project (or project group) to be associated with the "authorized users", i.e., the specific group of people with which we are concerned.
- Create a special role (for this example we'll call it the "Authorized User" role) to be associated with the previously created private project (or project group). (In other words, make sure that the "Role visibility" field for the Authorized User role is set to the project or project group just created.) Give the role the same set of permissions as the Registered User role, and set the role to not be inherited in private subprojects. (In other words, make sure that the "Block recursion into private subprojects" checkbox is checked.)
- Grant the Authorized User role to all users in the authorized group of people. (You can do this either by granting the role individually to each and every user in the authorized group, or by granting the role to a special user group and then adding all the people to that user group.)

You can then create subprojects of the original project (or new projects in the project group), marking the new (sub)projects as either private or public. If the new (sub)project is to be a closed project that is not open (or even made known) to everyone in the authorized group of people, then you should set the (sub)project to be private. On the other hand, if anyone in the authorized group is free to join the project then you should set the (sub)project to be public. This allows anyone having the Authorized User role to inherit that role in the public (sub)project, and to be able to view project documents and request project membership.

However other CEE users not in the Authorized User role cannot access or view the (sub)project at all, even though it is ostensibly "public", since such users do not have a role (whether direct or derived) in the (sub)project's private parent project (or a project group of which it is a member). We therefore use the term "semi-public" to describe the (sub)project.

Public CEE Sites

Above we discussed how to create public CEE projects that are visible to every user of a CEE system. However thus far we've assumed that the only way to access CEE is to login under a given user-id, so that in order to see even the public projects a person must be a registered CEE user.

In some cases we may want to make CEE project information publicly accessible to anyone, whether they are a registered CEE user or not. For example, we may be using CEE to support a public open source development project, and we may wish to allow any Internet user to access basic project information. Alternatively, we may be using CEE on an enterprise intranet to support a "development portal" open to all developers and others within the organization, and would like employees to be able to try out the system without having to obtain a user-id first.

In order to support such use, a CEE system can be configured to make user registration optional. When a person connects to such a CEE system, they can skip the login step and proceed directly to access information relating to public projects on the system.

Even though such a person does not login to CEE, they are still treated as if they have a valid CEE role, namely the Anonymous Guest role. Unlike any other CEE role previously described, it is not necessary to login to a CEE site in order to be associated with the Anonymous Guest role; instead the Anonymous Guest role is assigned by default to any person who is accessing a public CEE site and who has not explicitly logged in to the site.

The actions permitted to the Anonymous Guest role on public CEE sites are similar to those permitted to the Registered User role: Guest users may view documents and announcements in public projects, look at discussion archives, view and check out source code in the version control repository, and query the issue tracking system for issues. (When checking out source code, guest users should login using the user-id "guest" and a null password.)

However unlike registered users, guest users may not request roles in projects, submit issues to the issue tracking system, or do other things that require a person to have a valid CEE user-id. People wishing to have greater access to public CEE sites should register with the site to select a user-id and be assigned a password (or other authentication credential). They may then login to the site and have the access privileges associated with a registered user, as previously described.

Like the Registered User role, the Anonymous Guest role provides access only to top-level public projects and their public subprojects; the Anonymous Guest role is not inherited in private subprojects, so guest users cannot access private projects (if there happen to be any on a public CEE site) or "semi-public" subprojects of private projects (i.e., subprojects whose parent projects are private but which are themselves marked as public). However once a person registers as a user their user-id can then be granted access to private and semi-public projects, as previously described.

Project Categories

As more and more projects are created on a CEE site, it can become difficult for users to keep track of all the projects that are available. You can make it easier for users to find projects (whether public or private) by organizing them into project categories. A *category* is similar to a project group, but used for site organization rather than access control; a given category can contain multiple projects, and a given project can be a member of multiple categories. A category can contain either public projects or private projects or both, and can itself be either public or private, as described below.

Categories



Name	Summary	Subcategories	Projects
departments	Internal Department Projects	1	95
engagements	Customer-Specific Projects	0	153
products	Internal/External Product Projects	0	92

Categorizing Public Projects

Suppose that our CEE site contains several dozen public projects, open to anyone who has access to the site. Because there are so many projects we'd like to organize them in some way, so that users can get a better idea of which projects might be of interest to them.

One way to organize projects is by the type of software applications they represent. Thus, for example, your organization may develop logistics applications for internal use, and have several public development projects related to such applications. (The projects in this example are public in the sense that any employee with access to CEE may view them; the projects would not necessarily be accessible by anyone outside the organization, unless there were good reasons to allow such access.)

Since the projects are public, any registered user of the CEE system can see the projects listed by going to the "Projects" page and selecting the "Projects" tab; however if there are many projects then the logistics software projects may be jumbled together with unrelated projects, and thus may be difficult to find.

To address this problem, we can create a new category, call it "logistics", and add to the category all the projects related to logistics. Registered users can then go to the "Projects" page, select the "Categories" tab, and see "logistics" listed as a category. If they follow the "logistics" link they will go to a page wherein are listed all the projects belonging to the "logistics" category. From there they can decide which projects are of interest, and then access the projects by following their links. For more information on creating categories, see the CEE online help document.

Once you have created the category, it should be displayed on the "Projects" page under the "Categories" tab. If you later wish to add new projects to the category, you can do so in one of two ways:

- You can edit the category and add one or more projects to it.
- You can edit the project and change the "Project categories" field to select the category to which you wish to add the project.

You can also add a project to a category when you create the project originally, as described above.

Multiple Categories and Subcategories

In the example above we decided to organize projects based on the type of software they were developing, and created a "logistics" category to collect together projects related to logistics applications. You may also wish to organize projects based on other criteria; for example, you may wish to allow users to see all the projects sponsored by a particular part of the organization. You could then create categories such as "manufacturing", "engineering", "research", etc., one for each organization, and then assign projects to each category as described above.

Note that it is possible for a given project to be included in two different types of category; for example, a logistics application sponsored by the manufacturing division could be included in both the "logistics" category and the "manufacturing" category. Similarly, a project jointly sponsored by two different divisions could be included in the categories associated with each division.

A project can be added to multiple categories either by editing each category individually to add the project, or simply by editing the project itself and changing the "Project category" field to have multiple values. (For example, if you are using Microsoft Internet Explorer for Windows, you can select multiple categories in the "Project category" field by holding down the "control" key while clicking on each of the categories you wish to select.)

If you do use multiple sets of categories, and you specify all the categories as "new top-level" categories (using the "Parent category" field as described above), then all the categories will be listed together on the categories part of the "Projects" page. If you intend to use multiple sets of categories to implement "multi-axis" browsing for projects, then you may wish to use subcategories to create multiple hierarchies of categories.

For example, suppose you wish to have three sets of criteria by which you organize projects: the problem area for which the application is intended, the sponsoring division, and the development and run-time infrastructure on which the application is dependent. You can create category hierarchies to support this project organization as follows:

- Create three new top-level categories, say "application-domain", "application-sponsor", and "application-environment", as described above.
- Create the categories for application domain, e.g., "logistics", "financial", etc. When creating each category, specify the "Parent category" field to be "application-domain".
- Create the (sub)categories for application sponsor, e.g., "manufacturing", "engineering", etc. When creating each category, specify the "Parent category" field to be "application-sponsor".
- Create the (sub)categories for application environment, e.g., "apache-php", "ibm-websphere", "visual-basic", etc. When creating each category, specify the "Parent category" field to be "application-environment".

You may then add projects to one or more of the subcategories you have created. Note that when adding a project to a subcategory, you must edit the subcategory itself; it is not possible to add the project by editing the project itself, because subcategories do not show up as choices in the "Project categories" field on the "Edit project" page.

When users then go to the category listing (using the "Categories" tab on the "Projects" page), they will see the top-level categories listed. Subcategories can be seen by checking the "View subcategories" box. If they select a particular top-level category, say "application-type", they will be shown the home page for that category, which will list the various subcategories within the category. Users can then select a particular subcategory to see the projects belonging to that category.

Note that in this scheme a particular project could be added to a top-level category instead of (or in addition to) a subcategory, if there were some reason you wanted to do that.

Categorizing Private Projects

Thus far we have assumed that we will be categorizing only public projects, and that the category lists will be visible to any registered user of the CEE system. However you can also categorize private projects, and even have categories that themselves are private.

Continuing with our example above, suppose that the research division sponsors certain private projects that are open only to employees of that division, and that are not publicly known outside that division. If you wish, you can include such private projects in the “research” (sub)category containing the projects sponsored by the research division.

If an employee of the research division has at least the Observer role (or an equivalent role) in the private projects sponsored by that division, then that employee would see those private projects listed under the “research” category, along with all the public projects. However if the “research” catalog listing is viewed by an employee of another division, one who does not have any access to the research division private projects, then the list displayed to that employee would not include any of the private projects, only public projects.

Creating Private Categories

In addition to including private projects in a public category, you can also create a private category, so that even the category itself is hidden except to those specifically allowed access to it.

For example, suppose that in addition to the various corporate divisions mentioned above (“research”, “engineering”, etc.), the corporation in our example is working with a third party (“Acme Widgets”) to fund and manage various confidential joint development projects. For convenience we would like to include a category listing such Acme Widgets projects as one of the subcategories under the “sponsor” category, so that corporate employees working with Acme Widgets may have an easy way to find all these confidential projects. However at the same time we do not want other employees to know about the projects, or even to know that our example corporation is working with Acme Widgets.

We can do this by creating an “acme-widgets” category under the “sponsors” category, similar to the (sub)categories “research”, “engineering”, etc., that we created for projects sponsored by those divisions. However instead of making the “acme-widgets” (sub)category a public category, you should create it as a private category; in other words, you create the category as described above, except that you should leave the “Public category” checkbox unchecked.

In order to view a private category (or to see it listed as one of the available categories), a user must have at least the Observer role (or equivalent role) in the category itself. You can grant users roles in the category in the same manner as you would grant roles in a project or project group.

Note that by granting a user a role in the category you simply make it possible for the user to view the category itself (i.e., its existence). If the category includes private projects (which is the case in our example) then a user viewing the category would also have to have a role in those projects in order to see them listed in the category, as described above.

In our example we would also grant certain users roles within the various confidential projects relating to Acme Widgets, as well as granting them a role within the “acme-widgets” category itself. Such users would then be able to see the “acme-widgets” (sub)category listed under the “sponsors” category, as well as be able to see all the Acme Widgets projects listed under the “acme-widgets” category.

Note that roles can be inherited within categories and subcategories in a similar manner to regular projects and subprojects. Thus, for example, if a user is granted the Observer role in the “acme-widgets” private category described above, and you create new private subcategories within the “acme-widgets” category, the user in question will have the (derived) role of Observer within that category as well. However note that not all roles are inherited from a public category into a private subcategory; as with projects, the Registered User role is not inherited from a public parent cate-

gory into private subcategories. In our example this allows a registered user to view the “sponsors” public category, but prevents them from viewing the “acme-widgets” private subcategory.

Also note that a role granted in a category is not inherited in projects that are included in that category. This situation is different from that of project groups, where a role granted in a project group is inherited in the projects that are members of that group. This difference reflects the different purpose of project groups versus categories: project groups are specifically designed as an administrative mechanism to support more flexible access control schemes, while categories are designed simply to organize projects for easier browsing by users (who presumably have been separately granted access to the projects).

Conclusion

We have now completed our overview of the various CEE features to organize projects and control access to them, and have given a number of examples of how these features can be used. The following summarizes the key features of CEE in this area:

- Every CEE user is typically assigned an individual *user-id* (except for the special case of guest users accessing public CEE sites as noted below).
- CEE users can be granted *roles* in one or more CEE *projects*, with each such role providing a particular set of *permissions* with respect to a project’s file sharing area, version control repository, discussions, issue tracking system, etc. Different users can be granted different roles.
- CEE provides a number of pre-configured roles to correspond to particular standard sets of permissions; such standard roles include the *Observer* role, the *Content Developer* role, the *Developer* role, and the *Project Owner* role.
- *Resources* can be used to provide finer-grained control over access to a project’s version control repository, and are used, for example, to distinguish between the Developer role and the Content Developer role.
- When multiple projects are unrelated to each other and do not have users in common (for the most part), you can organize them as independent *top-level projects*.
- When multiple projects are part of a larger initiative and may share similar needs for access control, you can organize them as *subprojects* of a single *parent project*. With certain exceptions as noted below, a user’s role in a parent project is inherited in subprojects of that project.
- When identifiable sets of users participate in multiple projects, you can define *user groups* containing user-ids for each respective set of users, and then grant a role to a user group instead of individual user-ids. Users then have roles in the project based on the role granted to their user group. Users may belong to more than one user group.
- When a group of users needs access to an otherwise independent set of projects, you can create a *project group* of which the projects are members, and grant the users (or a user group containing the users) a role in the project group; the role will be inherited in projects that are members of the project group. A project may be a member of more than one project group, and may be a member of a project group in addition to being a subproject of a parent project.
- A project role that a user has inherited from a parent project (or a project group), or that a user has by virtue of being in a user group, is a *derived role*, as opposed to a *direct role* that is explicitly granted to the user’s user-id in the project in question. A user is not considered to have *joined* a project unless their user-id has been granted a direct role in that project.
- Projects can be either *private* or *public*. Any logged-in user can access top-level public projects and their public subprojects, using the *Registered User* role. However the Registered User role is not inherited in private subprojects of public projects.

- A CEE system can be a public site, allowing guest users access to the site without having to login, using the *Anonymous Guest* role. Like the Registered User role, the Anonymous Guest role is not inherited in private subprojects of public projects.
- You can use *categories* to group projects together to help users discover projects of interest to them. Like projects, categories may be public or private; private categories are made known only to users granted at least the Observer role (or equivalent role) in the category. Categories may contain public or private projects; a user viewing a category will not see private projects in a category unless the user has roles (direct or derived) in those projects.

Because of the many CEE features relating to access control and project organization, and the many ways in which those features can interact, we recommend that you take the time to plan out the ways in which you plan to organize your projects and users; you may also wish to experiment with test projects and test users to ensure that the permissions for users are configured as you intended.

Appendix A: Standard CEE Roles

This document has described briefly the various roles used in a standard CEE system. This appendix describes the standard CEE roles in more detail. The roles are divided into two general types, domain-wide roles and project roles, distinguished as follows:

- Domain-wide roles are not associated with any particular project or projects, but are granted for the entire CEE domain.
- Project roles are granted within one or more projects, project groups, or categories.

The standard CEE roles are as follows, in order of increasing privileges:

- Anonymous Guest
- Registered User
- Observer
- Content Developer
- Developer
- Project Owner
- Domain Admin

(There is also a separate Click-through Viewer role used for a single special purpose as described below.)

Note that a CEE administrator (i.e., someone granted the Domain Admin role) may edit one or more of the standard CEE roles to add new permissions to the role(s), or delete existing permissions from the role(s). However we recommend that you consider creating new roles instead of changing the existing ones. (Among other things, this produces less confusion for users reading standard CEE documentation and training materials.)

Such new roles can be created as domain-wide roles or project roles. Project roles can in turn be visible domain-wide (and hence can be granted in any project) or can be visible only within a particular project or project group (in which case the role can be granted only within that project or its subprojects, or within that project group and projects that are members of that project group).

Anonymous Guest

The Anonymous Guest role is a domain-wide role used for guest users on public CEE sites (i.e., sites that do not require users to register and login). Unlike the other roles described below, it is not necessary to login to a CEE site in order to be associated with the Anonymous Guest role; instead this role is assigned by default to any person who is accessing a public CEE site and who has not explicitly logged in to the site.

The Anonymous Guest role provides access only to top-level public projects and their public subprojects (or public project groups and public projects that are members of such groups); it is not inherited in private subprojects, so guest users cannot access private projects on a public CEE site, or “semi-public” subprojects of private projects (i.e., subprojects whose parent projects are private but which are themselves marked as public).

The actions permitted to the Anonymous Guest role on public CEE sites are similar to those permitted to the Registered User role: Guest users may view documents and announcements in (top-level) public projects, look at discussion archives, view and check out source code in the version

control repository, and query the issue tracking system for issues. (When checking out source code, guest users should login using the user-id “guest” and a null password.)

Registered User

The Registered User role is a domain-wide role granted to any registered user on a CEE site (i.e., a user who has been assigned a CEE user-id and, and who logs into the CEE system using that user-id).

The Registered User role provides access only to top-level public projects; it is not inherited in private subprojects, so users cannot access either private projects or “semi-public” subprojects of private projects solely by virtue of being registered users – registered users must have been granted some other role (e.g., the Observer role) to access private or semi-public projects.

Registered users may view documents and announcements in (top-level) public projects, look at public discussion archives, view and check out source code in the version control repository, and query the issue tracking system for issues. Unlike guest users, registered users may also suggest new projects (e.g., using the “Start project” function), submit new issues to the issue tracking system for projects to which they have access, and request to be granted new roles in existing projects to which they have access. Registered users also have a personalized “My start page”, and may view and edit their own CEE user profile.

Note that a request for a role in a project must be approved by a user who has the Project Owner role (or equivalent) for that project. A request to start a new project must be approved by a user who has the Domain Admin role (or equivalent) for the CEE system.

Click-through Viewer

The Click-through Viewer role is a domain-wide role used solely as part of the user registration process, in order to ensure that newly registered users agree to the site terms of use. (This agreement also known as the “click-through page”, hence the name of the role.)

The Click-through viewer role works as follows: The Click-through Viewer role is associated with one and only one permission, “Click-through Page – Seen”. When a new user is added to the system (either by an administrator or through user self-registration, if enabled) they are immediately granted the Registered User role upon assignment of the user’s CEE user-id; however they are not at that time granted the Click-through Viewer role.

When the new user then attempts to login to the CEE system, CEE checks to see if they have the permission “Click-through Page – Seen” associated with the Click-through Viewer role. Since the user does not have such permission (not having yet been granted the Click-through Viewer role), the CEE system redirects the user to a page on which is displayed the site’s terms of use agreement. If the user reads the page, selects the “Agree” option, and clicks the “submit” button, CEE then grants the Click-through Viewer role to that user. Once that is done, the user can then access the CEE system normally.

(If, on the other hand, the user selects the “Disagree” option and clicks the “Submit” button, they are redirected to the terms of use page once more, and cannot proceed further with CEE until they select the “Agree” option.)

Observer

The Observer role is a project role used for users who are granted “view-only” membership in projects, e.g., the user is to be prevented from changing content in the project’s version control repository.

When a user is granted the Observer role in a particular project, the user will have that (derived) role in all subprojects of that project. Similarly, if a user is granted the Observer role in a project group, the user will have that (derived) role in all projects that are members of that project group.

The permissions granted as part of the Observer role are similar to those associated with the Registered User role: a user with the Observer role in a project can view documents and announcements for the project, look at public discussion archives, view and check out source code in the version control repository, query the issue tracking system for issues, and submit new issues. (Of course, the Registered User role is effective only for public projects, while the Observer role can be granted to a user for any type of project, public or private.) Users with the Observer role can also suggest new projects and request new roles in existing projects.

However a user with the Observer role can perform some additional actions that are not allowed to all registered users: A user with the Observer role can also suggest new project documents and announcements, and can subscribe to and view the archives of discussions associated with the project.

Note that although users with the Observer role may submit new issues to the issue tracking system, they may not add comments to an issue that has already been submitted (whether by themselves or by others). This is because the Observer role does not include the "Project Issue Tracking – Change" permission, which in the standard CEE configuration requires having at least the Developer role.

If you wish to allow users to add comments to issues, but do not wish to grant them the Developer role, then you can edit the standard Observer role to add the "Project Issue Tracking – Change" permission, or you can create a new role based on the Observer role but including that additional permission, and then use that role instead of the Observer role. However note that the "Project Issue Tracking – Change" permission also allows users to change issue milestones, reassign issues, mark issues as resolved, and otherwise make arbitrary modifications to issue data.

Content Developer

The Content Developer role is a project role used for users who are granted permission to modify the web pages associated with the project, but who are prevented from modifying source code and other content in the project's version control repository.

When a user is granted the Content Developer role in a particular project, then the user will have that (derived) role in all subprojects of that project. Similarly, if a user is granted the Content Developer role in a project group, the user will have that (derived) role in all projects that are members of that project group.

The permissions granted as part of the Content Developer role include all the permissions associated with the Observer role⁸. Users having the Content Developer role in a given project can also add new documents to the project file sharing area, and edit existing documents. They can also add and modify content within the "www" directory in the project's version control repository, i.e., they can do adds, commits, and similar operations against the repository, but only for files and subdirectories within the "www" directory.

This allows users with the Content Developer role to modify the optional project home page ("www/index.html") and to add other project pages. (Note that in order for the project web page

⁸ In some versions of CEE the Content Developer role does not have permission to suggest new project announcements, even though this capability is allowed to both the Observer role and the Developer role. You can correct this oversight by editing the Content Developer role to add the permission "Project News - Suggest".

to be used as the default start page for the project, a user with the Project Owner role for the project must edit the project and make sure that the “Use project index.html” checkbox is checked.) However users with the Content Developer role are prohibited from modifying version control content outside the “www” directory; if they attempt to do a commit or similar operation in other directories in the repository, they will get an error message.⁹

Note that although users with the Content Developer role may submit new issues to the issue tracking system, they may not add comments to an issue that has already been submitted (whether by themselves or by others). This is because the Content Developer role does not include the “Project Issue Tracking – Change” permission, which in the standard CEE configuration requires having at least the Developer role.

If you wish to allow users to add comments to issues, but do not wish to grant them the Developer role, then you can edit the standard Content Developer role to add the “Project Issue Tracking – Change” permission, or you can create a new role based on the Content Developer role but including that additional permission, and then use that role instead of the Content Developer role. However note that the “Project Issue Tracking – Change” permission also allows users to change issue milestones, reassign issues, mark issues as resolved, and otherwise make arbitrary modifications to issue data.

Developer

The Developer role is a project role used for users who are granted permission to modify source code, project web pages, and any other content in the project’s CVS repository.

When a user is granted the Developer role in a particular project, then the user will have that (derived) role in all subprojects of that project. Similarly, if a user is granted the Developer role in a project group, the user will have that (derived) role in all projects that are members of that project group.

Users granted the Developer role have all the permissions associated with the Content Developer role; in addition users having the Developer role in a given project can make changes to any file or directory in the version control repository (not just in the “www” directory), and can also make changes to issues already submitted to the issue tracking system.

Project Owner

The Project Owner role is a project role used for users who are granted permission to administer projects.

When a user is granted the Project Owner role for a particular project, then the user will also have that (derived) role in any subprojects created in that project. Similarly, if a user is granted the Project Owner role in a project group, the user will have that (derived) role in all projects that are members of that project group.

A user with the Project Owner role in a project has all the permissions associated with the Developer role, and in addition can perform the following actions within the project:

⁹ In some versions of CEE users with the Content Developer role will get an error message even when doing a commit within the “www” directory, due to a CEE error in interpreting the resource associated with the Content Developer permissions. You can correct this by editing the Content Developer role and changing the resource associated with the version control-related permissions from “/www/*” to “www/*”. To do this, first add a new resource with resource pattern “www/*”. Then edit the Content Developer role to delete all the VersionControl permissions referencing “/www/*”, and add all those permissions back again, this time referencing the resource “www/*”.

- Edit the information about the project
- Add, delete, and modify project announcements, and approve project announcements suggested by other users
- Add, delete, and modify project discussions, and subscribe or unsubscribe users to and from discussions
- Configure the project's issue tracking system (e.g., to define new components and/or subcomponents)
- Administer the project's version control repository
- Add and delete users from the project's membership list, and change the roles granted users in the project
- View audit log entries associated with the project

Recall that any registered user may suggest that a new project be created; the request must then be approved by a user with the Domain Admin role (or equivalent). The user suggesting the project is granted the Project Owner role for that project, and may then invite or add additional users to the project as desired.

Domain Admin

The Domain Admin role is a domain-wide role used for users granted permission to administer the overall CEE domain and all projects within it.

A user with the Domain Admin role has the permissions of the Project Owner role for any project created within the CEE system. In addition a user with the Domain Admin role may perform the following actions:

- Add, delete, and edit user-ids, and change users' passwords
- Add, delete, and edit user groups, including adding and deleting users to and from user groups
- Add, delete, and edit domain-wide and project roles
- Create new projects without needing approval, and approve projects suggested by others
- Lock projects from being used
- Create new project groups and categories, and modify and delete existing project groups and categories
- Display audit log entries associated with all projects
- Display users currently actively using the CEE system

In essence a user with the Domain Admin role can perform all functions that can be performed using the standard CEE user interface. Thus you should grant the Domain Admin role only to a few trusted people.