Rational ClearQuest EmailPlus Package 2.1
Administrator's Guide
Issue 1.0

Table of Contents

1 ClearQuest EmailPlus Package

The Rational ClearQuest EmailPlus Package (EmailPlus) provides a more extensive solution to the problem of email notifications.

This guide describes version 2.0 (and possibly later versions) of the EmailPlus Package.

The ClearQuest Email Rules package provides basic facilities to communicate the changes in records to users. EmailPlus provides several additional features for an enhanced email experience for users, some of which are listed below:

- Customizing the content of the message subject and body with boilerplate text
- Including dynamic content in emails in addition to field old or new values
- Including rich content (HTML)
- Providing access to SMTP restricted servers
- Attaching files to emails
- Allowing enhanced conditions to construct the rules
- Allowing the self-subscription of users to rule or record notifications
- Enabling enterprise-wide configuration of email notifications, which can be enabled or disabled by users
- Providing performance improvements through queries
- Extending the rules and data displayed through customer-provided extensions

2 Package Design

The EmailPlus package is implemented in PERL as a ClearQuest package. The package can be added to any ClearQuest schema. Although it is implemented entirely in PERL, it can be used to enable VBScript based schemas because packages have their own scripting language defined. There is, however, a restriction in VBScript based schemas with respect to stateless record types (For details, see Section 6.3).

Possibilities of name clashes are minimized by prefixing routine names with "EMP_" and by prefixing Session Name values with "EMP::". Avoiding the use of names like these will ensure that the package can coexist with other packages in your schema.

2.1 Record Types

The EmailPlus package introduces seven new stateless record types into the schema to which it is applied. These record types are listed below:

- EmailPlusConfig
  Contains the main controls for configuring the package.
- EmailPlusTemplate
  Defines the templates for the content of an email.
- EmailPlusRule
  Defines the conditions under which an email should be generated.
- EmailPlusSiteConfig
  Defines the information about the SMTP and web servers at each site in a

ClearQuest MultiSite installation or the single site in a non-replicated environment.

- EmailPlusAction
  Is used by an internal caching mechanism to minimize the impact on performance of rule evaluation. There is no direct interaction with this record type.
- EmailPlusRuleSubscription
  Permits individual users to subscribe to notifications generated by individual rules.
- EmailPlusPostOffice
  Allows requests for sending emails to be written to the ClearQuest database which the relay service then processes.

The relationship between these record types is shown in Figure 1.

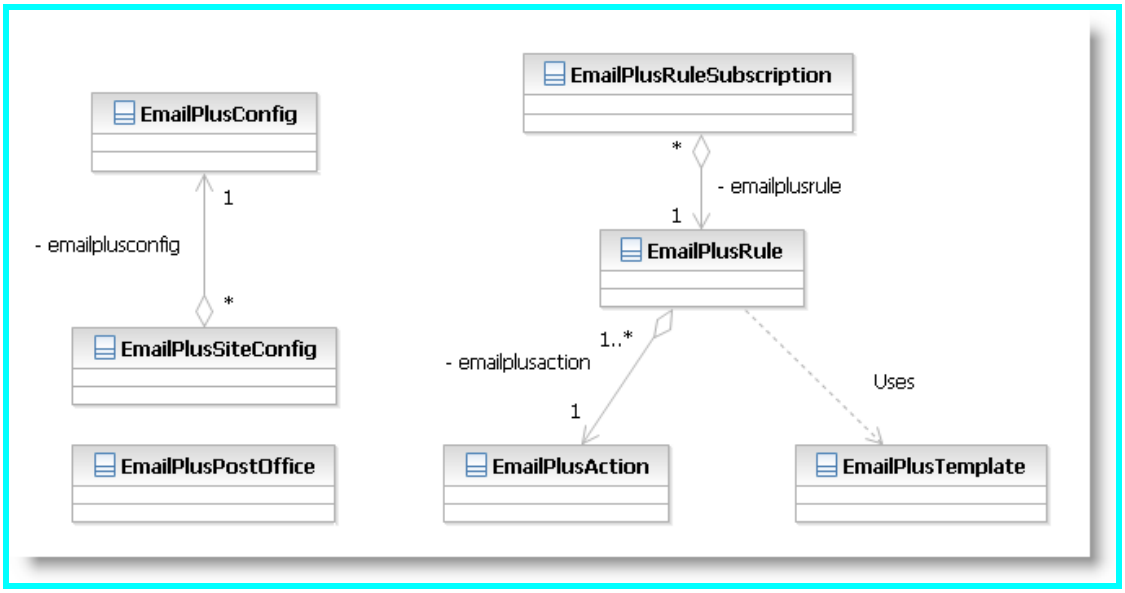


Figure 1. EmailPlus Record Relationships

2.1.1 Replicated and non-replicated environments

The EmailPlus package is designed to work in both replicated and non-replicated ClearQuest environments. In this guide, 'Site' refers to either the ClearQuest MultiSite replica name or, in a non-replicated environment, [local], that is the only site.

There is another special value of 'Site', that of <default>**.** This value applies in the absence of a specific site definition.

2.2 EmailPlusConfig Record Type

There must be exactly one EmailPlusConfig record. Its purpose is to control the operation of the package and to record some configuration parameters. The generation of email notifications by the package is controlled globally by the EmailPlus Active checkbox ① (see Figure 2), which, when not selected, stops EmailPlus from generating any emails. This is useful, for example, during maintenance activity on the database during which unwanted emails might be generated.

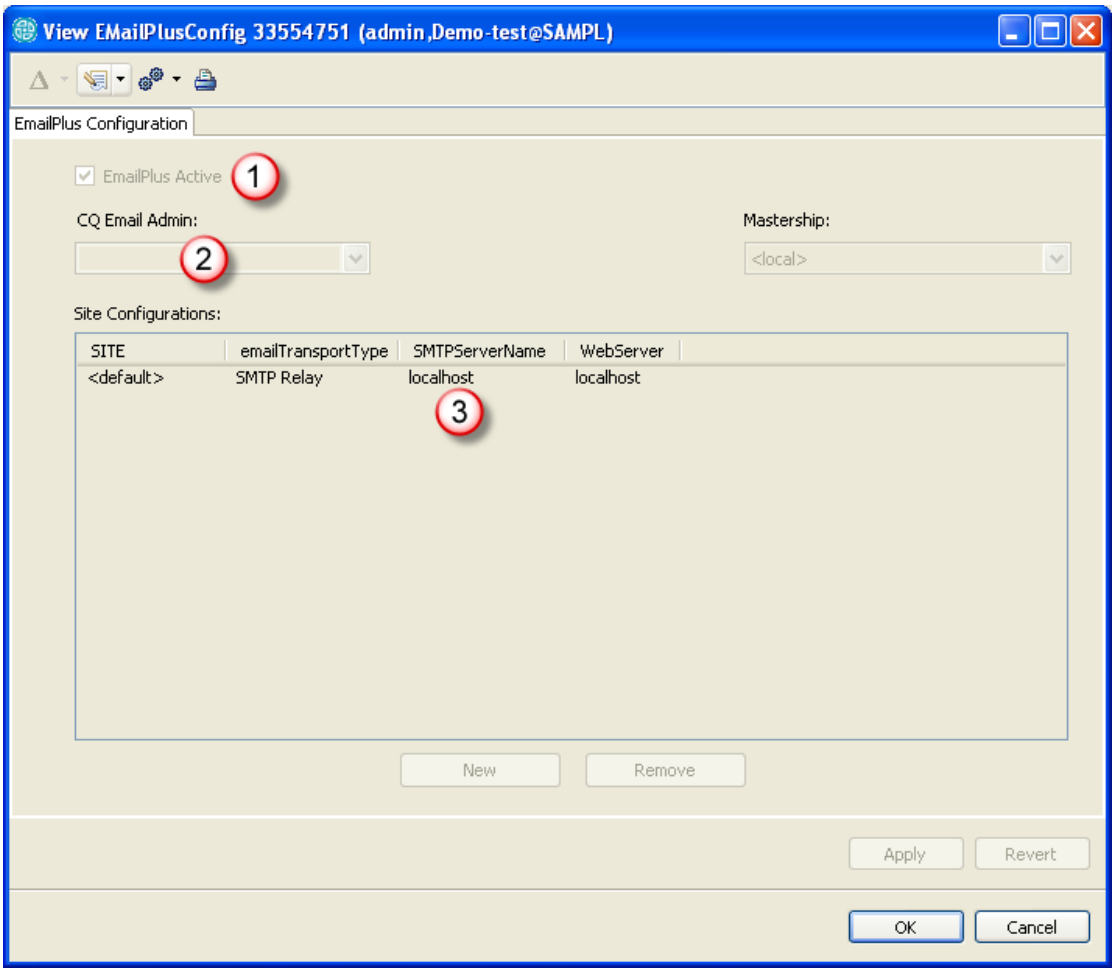


Figure 2. EmailPlus Configuration Record

If there are problems in sending emails, EmailPlus generates a notification of the fault and sends it to the ClearQuest user identified as the ***CQ Email Admin*** (2), provided this user has an email address defined in the ClearQuest user database. If this field is left blank, no notifications will be sent about email sending failures. The level of detail included in any notification depends upon the transport mechanism used and what it reports, if anything.

The site configurations field is a list of references to EmailPlusSiteConfig records that define the information for each site in the ClearQuest installation.

Note: When a site configuration record is removed, it is also deleted. Orphaned EmailPlusSiteConfig records are not allowed.

In EmailPlus 2.1, a new utility has been introduced on the EmailPlusConfig record to assist administrators who are migrating from previous versions of the EmailPlus package. The utility is called "RebuildEMPActions" and it will automatically rebuild the EmailPlusAction records utilizing any new algorithms used internally in the EmailPlus package to build EmailPlusRule expressions. If you are upgrading from an earlier release of the EmailPlus package, you should run this utility once after upgrading, otherwise EmailRules may not work as you expect them to. While viewing the EmailPlusConfig record, the "RebuildEMPActions" utility is available from the Utilities menu in ClearQuest web and the ClearQuest Eclipse client and is available from the Actions menu in the ClearQuest Windows client.

2.3 EmailPlusSiteConfig Record Type

The EmailPlusSiteConfig records define the information about the SMTP and ClearQuest web servers available at a given site. There must be at least one EmailPlusSiteConfig record. In a non-MultiSite environment that is all that is needed. In a MultiSite environment, there can be up to one record per MultiSite replica. This record type has two tabs that are described in the following sections.

2.3.1 EmailPlusSiteConfig – SMTP Tab

This tab is for specifying the SMTP mail server details (see Figure 3). The box identified by ① is for selecting the name of the replica to which the configuration applies. If this is a non-replicated ClearQuest installation, select <local>. You can also select the <default> site, in which case the <default> definitions will apply if a specific record for any given site does not exist.

The area identified by ② is for the credentials for the SMTP server that is accessible to this site.

The SMTP Reply-to address, if set, is the address used by email clients to respond to generated emails. Each site can have a different reply to address.

Note: Although a "reply-to" address can be specified, it is supported only by the transport types not based on the ClearQuest mail object. Custom transport types might support this feature.

The field identified by ③ allows you to select the transport type to be used at that site. Features offered by EmailPlus (such as the support for rich text in emails) vary depending upon the transport type selected. See section 2.3.1.1 for a description of the transport types supported by EmailPlus.

The field identified by ⑤ is used only in PostOffice Relay mode. It identifies the site name of the MultiSite replica where the Post office deliveries are undertaken for this site. This is necessary because the records can only be deleted on the site where they are mastered. Having a remote site delivery delays emails by up to one replication cycle period. If delivery time is of the essence, all Post Office deliveries should be performed locally. This means having a relay service running at each replica you need. In non-MultiSite environments, this should be set to <local>.

Finally, the fields identified by ④ are not used by EmailPlus directly, but are made available to any custom transport type that is defined.

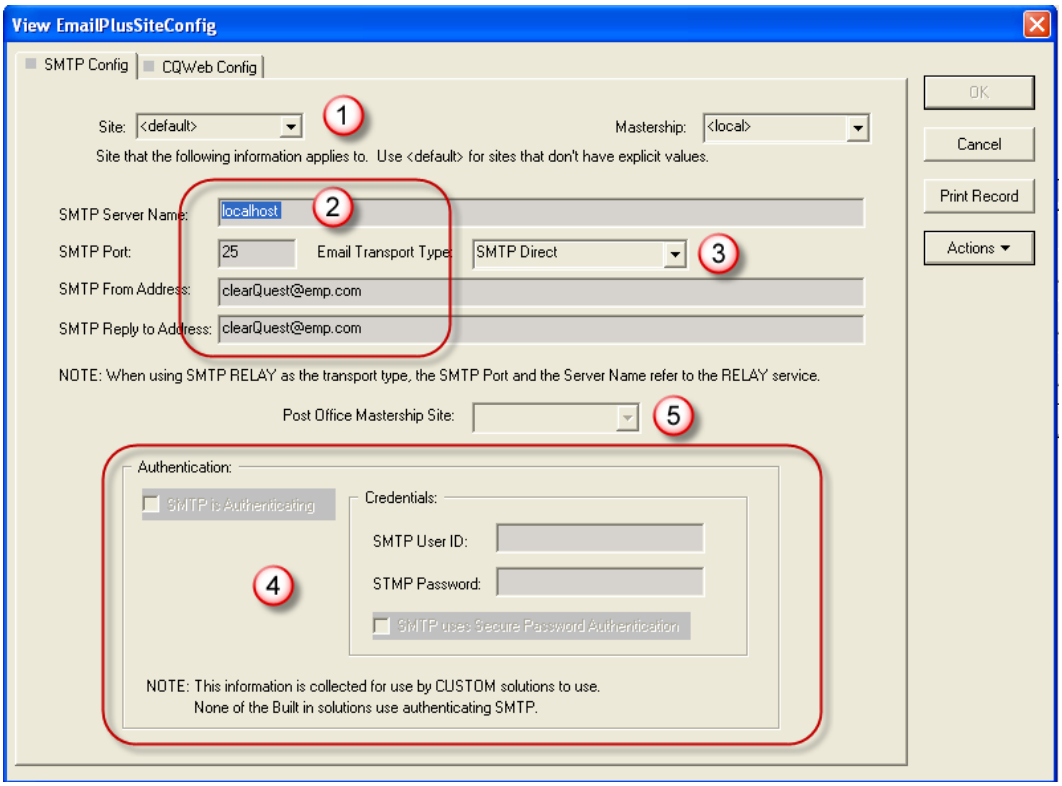


Figure 3. EmailPlus Site Configuration Record – SMTP Config Tab

2.3.1.1 Email Transport Types

The EmailPlus package offers a number of different transport types to suit different needs within the enterprise. You can select different transport types for different sites in a ClearQuest MultiSite environment. The possible email transport types are shown in Figure 4 and explained in the following paragraphs.

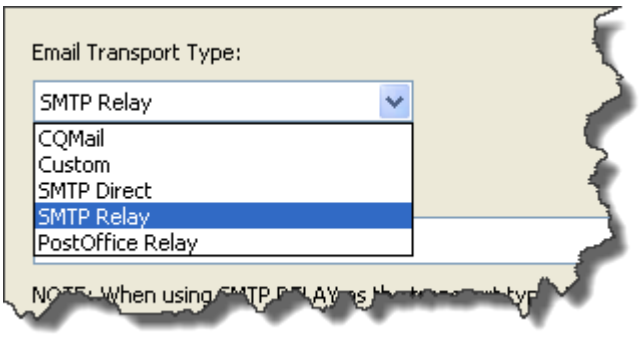


Figure 4. Email Transport Types

CQMail
This transport type uses the standard ClearQuest mail object to transport emails. If this transport type is selected, the server details are taken from the client configuration as normal and not taken from this EmailPlus record. The CQMail transport type has all the limitations of the standard Email Rules package regarding HTML Content, Mail headers and attachments. However, it is the only way if you want to use a MAPI mail transport.

Custom

This transport type gives you all the benefits of the rich content and rules of the EmailPlus package but leaves the actual relay of messages to a custom solution provided by you.

```
sub EMP_SendMail_CUSTOM {
    my $fromAddr            = shift; # String
    my $toList_ref          = shift; # Reference to an array
    my $ccList_ref          = shift; # Reference to an array
    my $bccList_ref         = shift; # Reference to an array
    my $subject             = shift; # String
    my $body                = shift; # String
    my $isHTMLContent       = shift; # Boolean flag
    my $ccActioner          = shift; # Boolean flag
    my $attachmentList_ref  = shift; # Reference to an array
    my $mailHeaders_ref     = shift; # Reference to an array

    # This is a customisable template to deliver mail in whatever way
    # you see fit.  It is only called for sites configured to use the
    # 'CUSTOM' email transport type.  Its up to you to get your mail there

    # You can retrieve information about the local SMTP server by calling the function
    #       EMP_LookupSMTPServerAttribute(AttName)
    # with one of these strings as AttName to request that item of information
    #       "SITE", "SMTPServerName", "SMTPPort","isSMTPAuthenticating", "SMTPUserID"
    #       "SMTPPassword", "isSMTPSecurePw" "SMTPReturnAddress", "SMTPFromAddress",
    #       "emailTransportType"
    EMP_DebugOut("START - Deliver mail using custom format");
    # EMP_WebSafeMsgBox("REMEMBER to customise the EMP_SendMail_CUSTOM routine to your specif
    EMP_DebugOut("REMEMBER to customise the EMP_SendMail_CUSTOM routine to your specification
    EMP_DebugOut("END");
```

Figure 5. Custom Email Transport Routine

In ClearQuest Designer, a function in the 'EMP_Customisable' PERL Global Script called **EMP_SendMail_CUSTOM** can be edited to provide for the transport functionality you want. If you do not customize this function, the custom transport only displays a warning message when running EmailPlus in debug mode (see Appendix G – Debugging EmailPlus).

There is also a function in the PERL Global Script 'EMP_Customisable' called **EMP_SiteConfig_CUSTOM_SetFormFieldRequiredness** with which you can customize the field requiredness of the fields on the EmailPlusSiteConfig record for the custom email transport type. By default, all of the fields are optional, but you could modify this to make certain fields mandatory or read-only.

```
sub EMP_SiteConfig_CUSTOM_SetFormFieldRequiredness {

    # Modify the values of the field requiredness for each fields on the EmailPlusSiteConfig record form

    $entity->SetFieldRequirednessForCurrentAction("SMTPServerName", $CQPerlExt::CQ_OPTIONAL);
    $entity->SetFieldRequirednessForCurrentAction("SMTPPort", $CQPerlExt::CQ_OPTIONAL);
    $entity->SetFieldRequirednessForCurrentAction("SMTPFromAddress", $CQPerlExt::CQ_OPTIONAL);
    $entity->SetFieldRequirednessForCurrentAction("SMTPReturnAddress", $CQPerlExt::CQ_OPTIONAL);
    $entity->SetFieldRequirednessForCurrentAction("isSMTPAuthenticating", $CQPerlExt::CQ_OPTIONAL);
    $entity->SetFieldRequirednessForCurrentAction("PostOfficeMastershipSite", $CQPerlExt::CQ_READONLY);
}
```

Figure 6. Custom SetFormFieldRequiredness

SMTP Direct

This transport type makes use of a direct connection to the SMTP server that allows for a more robust email transport, but only through SMTP. This transport type allows the package to send emails that have HTML body parts that might also contain embedded images and additionally allows the use of attachments and SMTP mail headers.

SMTP Relay
This is the same as SMTP Direct except that the message components are sent as an XML data stream to a relay component which receives the XML data and converts it to an email delivered to your SMTP server. This allows the package to operate in environments where the SMTP server accepts requests only from named IP addresses. For more information about the relay service, see section 7.

PostOffice Relay
This transport type records the details of emails required as records in the ClearQuest database. Each email required creates one record of type EmailPlusPostOffice. These records contain everything necessary for the relay service component to retrieve and generate the emails. For more information about the relay service, see section 7.

2.3.2 EmailPlusSiteConfig - CQ Web Tab

The CQWeb Config tab is for defining information about the ClearQuest web server, if any, at the site. These credentials are used in constructing click-through links in the email message body to take you to a specific record, chart, query, or report.

The server name and port number are defined in fields identified by ① in Figure 7. If the server name is not specified, it is assumed there is no web server at this site. The port number can be left blank if you are using the default port number.

You can pick the protocol ② you use to communicate with your web server, either http:// or https://, but to be able to use the latter, your ClearQuest web server must be configured to use the https:// protocol.

You can specify a ClearQuest login ID and password ③ so that the URLs generated take the user straight to the appropriate item. Using this facility, however, is a potential security risk because of the following two reasons:

- The user ID and password appear as plain text in each URL
- The user will be able to perform whatever actions that ClearQuest ID is permitted to

If you leave these fields blank, users who click the link need to provide their own ClearQuest credentials if they do not already have an active ClearQuest web session.

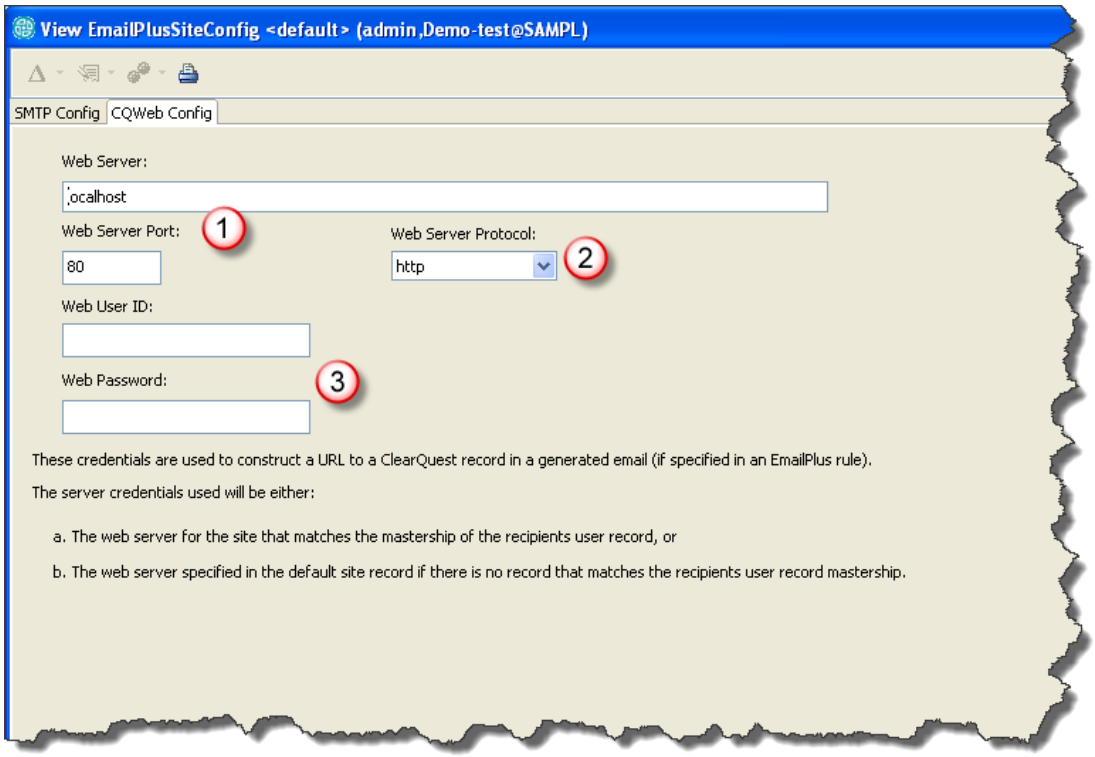


Figure 7. EmailPlus Site Configuration Record - CQWeb Config Tab

2.4 EmailPlusRule Record Type

The EmailPlusRule record type is used for defining the following settings:

- The rule conditions under which an email notification is generated
- Whether or not self-subscription to the rule is permitted
- The email addresses to whom notifications are sent
- The EmailPlus template used for generating the notification content

The EmailPlusRule record type is described in detail in section 4.

Note: In ClearQuest MultiSite environments, all rules relating to a single record type and its associated EmailPlusAction record are mastered at the same site. This is required because each time an EmailPlusRule is created or edited, the EmailPlusAction record is updated, and this can only be achieved if they are all mastered on the same site.

2.5 EmailPlusTemplate Record Type

The EmailPlusTemplate record type is used for defining the content and layout of an email. A template can be used by multiple rules for the same record type if required. The EmailPlusTemplate record type comprises five tabs which are described in the following subsections.

2.5.1 EmailPlusTemplate – Template Definition Tab

The template definition tab specifies basic information about the template such as its name, kind and a description of what it is used for.

To create a template, you must first select the record type that the template is to be defined for (see ① in Figure 8). If a record is not selected, it is not be possible to specify some information in the template record.

The Template Name ② is an arbitrary title for the template that identifies its purpose but it must be unique in the population of Templates for a given record type.

The Description ④ allows a fuller description of what the template is used for.

The Template Kind Field ③ allows you to define the type of template to be created. There are three kinds to choose from:

- **Special Interest:**
  This type of template is used by the EmailPlus package for generating notifications to individuals who have subscribed to notifications about changes to a specific records. There can be only one of this kind of template for each record type. Users receive notifications of any changes to the records they subscribe to. Special Interest templates are never explicitly associated with an EmailPlusRule. The rule is implicit in the EmailPlus package.

  Note: Users can also subscribe to an ***EmailPlus Rule*** in which case they are notified whenever the rule fires. See section 4.6.

- **Ownership:**
  This type of template is used for notifying a change of ownership (see section 6.4), where any change to the nominated ownership fields triggers a notification to both the old owner and the new owner by an email generated from this template. Like the Special Interest template type, there can be only one of these for each record type. Ownership templates are never explicitly associated with an EmailPlusRule. The rule is implicit in the package.

- **Rule:**
  The third template type is the Rule template. Templates of this type must be associated with one or more rules. When a rule is triggered, an email is generated using the template associated with it.

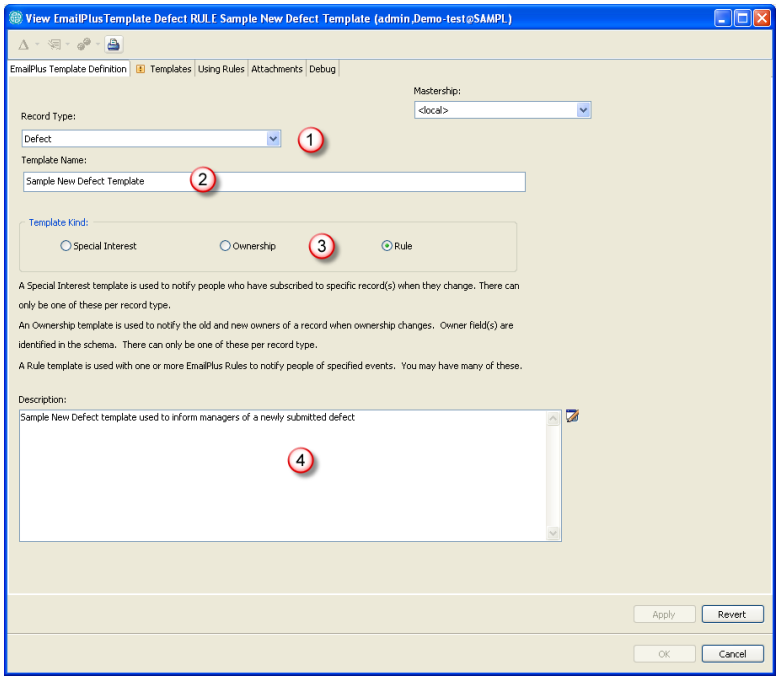


Figure 8. EmailPlus Template Record - EmailPlus Template Definition Tab

2.5.2 EmailPlusTemplate – Templates Tab

This tab (see Figure 9) contains the definitions to be used for the mail subject (header) 1 and mail body 3 of the message. These template fields contain plain text and special substitution tags that expand to the values they represent when the template is applied. Optionally, you can also specify that the body part be treated as HTML 2. If you do so, the body can contain all valid HTML tags that can be placed between the <BODY> and </BODY> tags. This facility is dependent on the email transport type selected and only works with the SMTP Direct, SMTP Relay, and Post-office Relay options, or the custom option if the custom transport so permits.

For template bodies that are specified as HTML, when you need to force a new line in the rendered HTML, place a
 tag into the HTML source as shown in Figure 9.

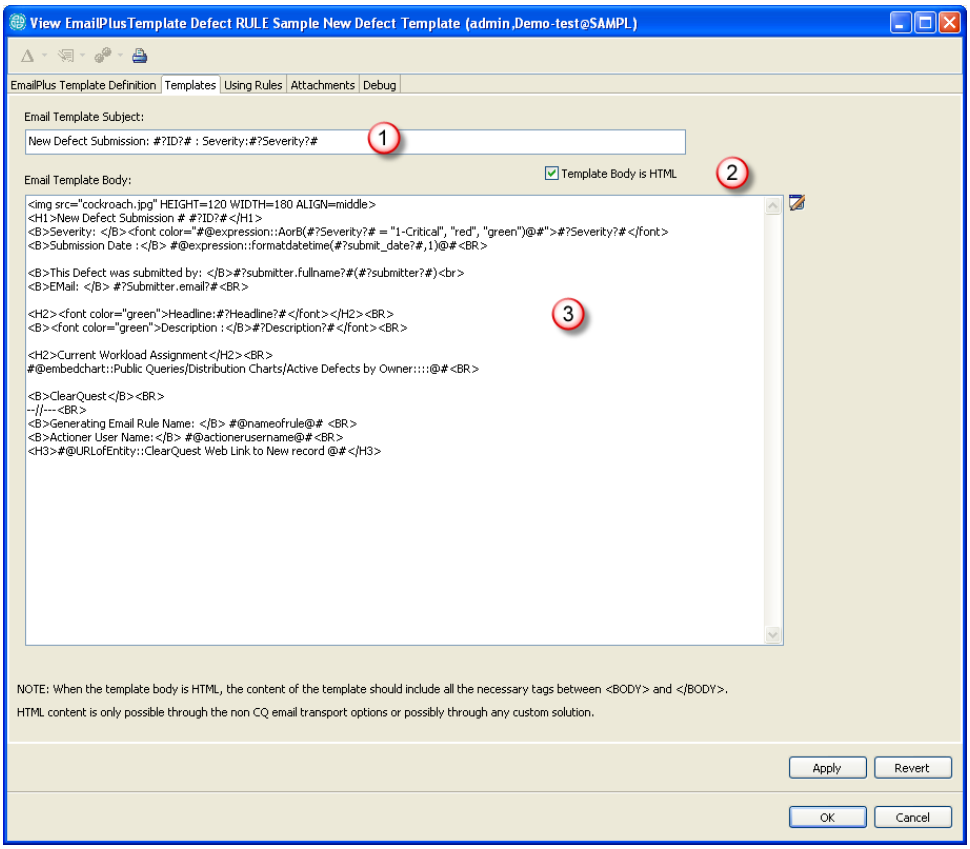


Figure 9. EmailPlus Template Record - EmailPlus Templates Tab

The list of template substitution markers is extensive and documented in section 3.1 below.

2.5.3 EmailPlusTemplate – Using Rules Tab

The Using Rules tab (see Figure 10) lists all the rules that are currently associated with the template. This helps you see, at a glance, what rules are affected if you change anything in the template and, thus, helps you decide whether to create a new template or to modify the existing one.

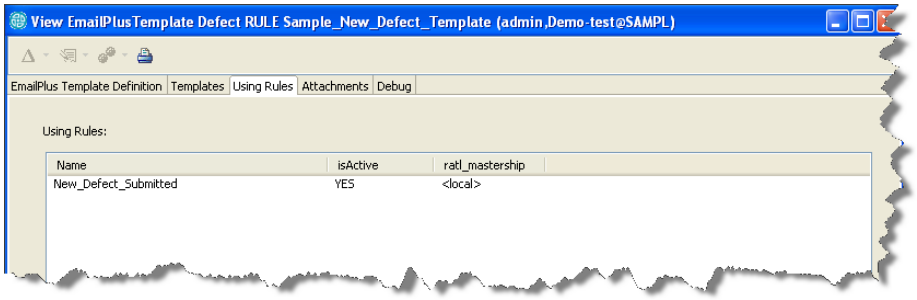


Figure 10. EmailPlus Template Record - Using Rules Tab

2.5.4 EmailPlusTemplate – Attachments Tab

EmailPlus does not use the attachments package and, therefore, does not use changes in the storage model that newer versions of the attachments package might provide. Template attachments are always stored in the ClearQuest database.

The Attachments tab (see Figure 11) enables you to do two things:

- Attach graphics or other files that are to be unconditionally attached to any generated emails created using this template. These attachments are used for embedded graphics in the template. So, for example, if you want a picture of a bug in your defect emails, you would attach the graphic for it here and then refer to it in an HTML tag. For details on how to do this, see section 3.2.
- Specify any mail headers. This enables you to do things like specify a priority for generated emails. Mail headers consist of two components separated by "::". The two parts are Name and value. So in the example, 'x-Priority' is the mail header name and '1-high' is the value of the mail header. Mail headers are supported only by the SMTP Direct, SMTP Relay, and Post-Office relay mail transports provided that the ClearQuest mail component is not used. The custom transport can also support headers but the support is implementation dependent.

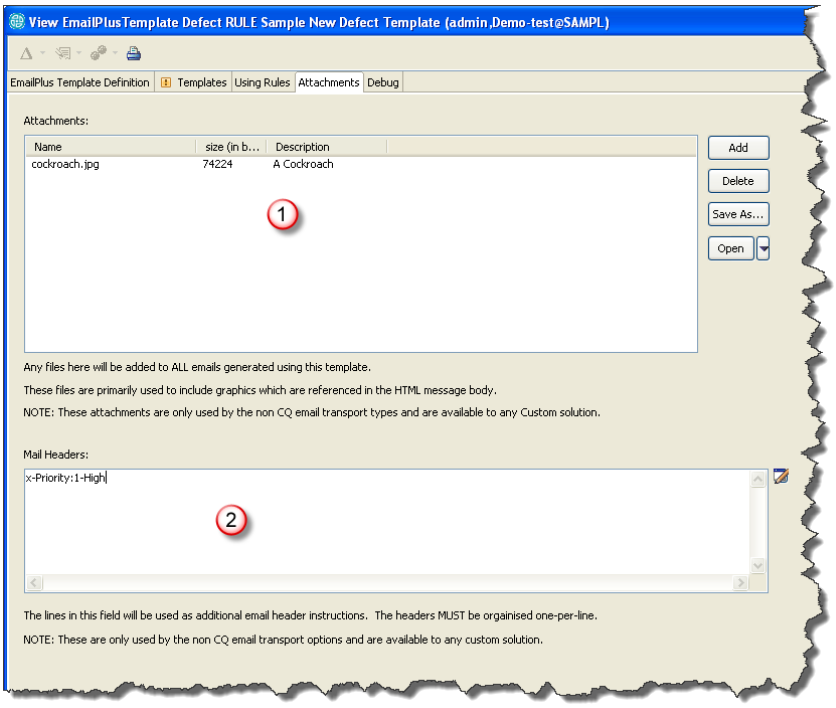


Figure 11. EmailPlus Template Record - Attachments Tab

For more information on Email Headers, see RFC 2822 - Internet Message Format.

2.6 EmailPlusAction Record Type

The EmailPlusAction records are created automatically by the EmailPlus package whenever the first EmailPlus Rule is created for a given record type. The purpose of the EmailPlusAction records is to cache all of the rules so that when the rules have to be evaluated, it minimizes the performance impact. The EmailPlusAction record type

is an internal-only record type and has no user serviceable parts. If you think that the cache could be out of date, modify any rule for the given record type and the cache record will be rebuilt.

Note: Modifying the rule results only in rebuilding the cache record, and not in updating the in-memory caches on any other active clients. To update these caches, users must log out and then log back in.

2.7 EmailPlusRuleSubscription Record Type

The EmailPlusRuleSubscription record type exists so that users can subscribe to register receiving notifications generated by the associated EmailPlusRule. The subscription record is a separate entity so that in a ClearQuest MultiSite environment there can be an editable subscription record at each replica you wants the facility to be available from. If no EmailPlusRuleSubscription records are created for a rule, then no subscription is possible for that rule.

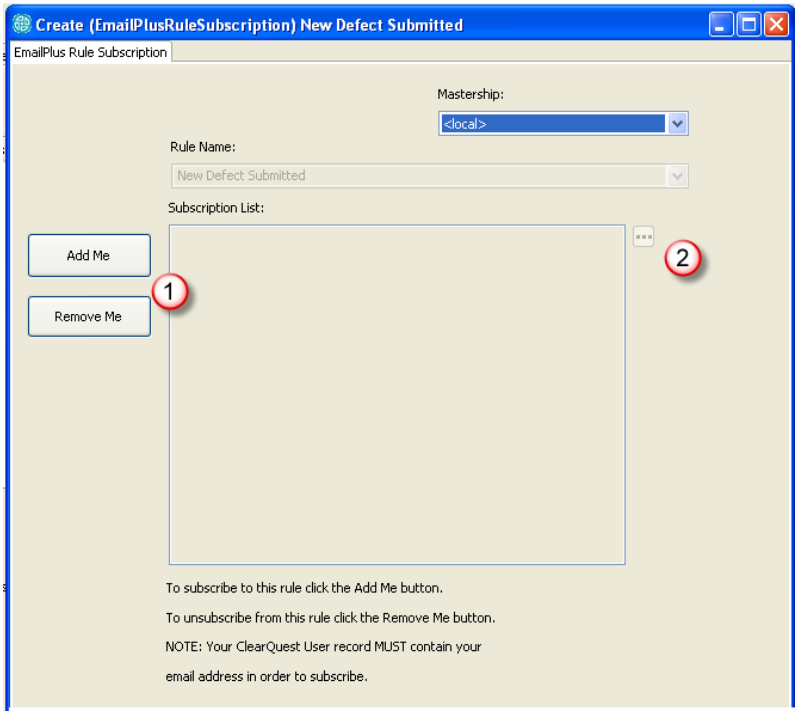


Figure 12. EmailPlus Configuration Record

Note: EmailPlus has to query the database to find the subscription records and their members but it only does this for rules that are evaluated as true. These are not cached so new subscriptions take effect almost immediately although, for MultiSite configurations, there might be a delay till the remote records are replicated.

Users subscribe to a rule by locating the subscription record for the rule they wish to subscribe to in their local replica. Users can add themselves to the subscription list by clicking the **Add Me** button (see 1 of Figure 12) or remove themselves by clicking **Remove Me**. The EmailPlus administrator can also restrict who can subscribe to a rule. See section 4.6 below.

2.8 EmailPlusPostOffice Record Type

When the PostOffice Relay transport type has been selected, the EmailPlusPostOffice records are created automatically when there is a need to send an email. The EmailPlusPostOffice records are processed and deleted by an instance of the relay service that is described in section 7.

The EmailPlusPostOffice record type is an internal only record type and has no user serviceable parts. If a specific record is causing problems, the record can be deleted by running a script after logging in with a login ID that is a member of the EmailPlusAdmins group.

If you are not using the PostOffice Relay transport type, then there should be no records of this type.

3 Working with EmailPlus Templates

Email templates consist of boilerplate text that has special substitution tags embedded in it. These tags are replaced by the value that they represent when the template is applied. The boilerplate can contain plain text or html.

Three types of tag are supported by EmailPlus:

- Current Value Field Tags
- Old Value Field Tags and
- Meta Tags

All tags are enclosed by special character sequences that define the tag type as shown in Table 1.

Opening Sequence	Closing Sequence	Tag Type	Example
#?	?#	Current field value. Any field defined in the ClearQuest record.	#?state?#
#%	%#	Old Field Value. Any field defined in the ClearQuest record	#%state%#
#@	@#	Meta Tag Name (see section 3.1 below)	#@URLofEntity@#

Table 1. EmailPlus Tag Definitions

EmailPlus is not limited to data contained in the record being modified. Information from anywhere in the ClearQuest environment can be used.

3.1 Template Meta Tags

The EmailPlus package contains several meta tags. This meta tagging facility is also extensible by the ClearQuest administrator. Custom meta tags can be added by modifying the schema (See Appendix A – Template Meta Tag Substitution Extension Point for details).

Table 2 describes the built-in meta tags supported by Email Plus. Some of these tags have parameters. Some parameters are optional or have default values. Optional parameters are enclosed in {} characters which are not part of the parameter syntax. Meta tags should be defined on a single line although in Table 2 the Meta Tag Name column might display them over multiple lines.

Meta tag name	**Meta tag purpose**
URLofEntity {::<linktext>}	Replaces the tag with a URL generated to take the user directly to the record that caused the email rule to be triggered. The web server credentials are those defined in the EmailPlusSiteConfig record for the site where the record is mastered. If no site record exists for the replica, <default> is used. If no web servers are defined then no URL can be generated. <linktext> is an optional string containing the text users see in HTML body parts, instead of the URL itself. **Example:** #@URLofEntity::Click Here To Visit Record@# Would insert the URL into the email and display it as: Click Here To Visit Record
URLofQuery::<queryname> {::<linktext>}	Replaces the tag with a URL of the named query. <queryname> is the name of a ClearQuest query. This should be a public query unless it can be ensured that every user has the same query in their personal queries folder. <linktext> is an optional string containing the text users see in HTML body parts instead of the URL itself. **Example:** #@URLofQuery::Public Queries/All Defects::Click Here To Run Query@# Would insert the URL into the email and display it as: Click Here to Run Query
URLofChart::<chartname> {::linktext>}	Replaces the tag with a URL of the named chart. <chartname> is the name of a ClearQuest chart. This should be a public chart unless it can be ensured that every user has the same query in their personal queries folder. <linktext> is an optional string containing the text that the users see in HTML body parts instead of

Meta tag name	Meta tag purpose
	the URL itself. **Example:** #@URLofChart::Public Queries/Defect Distribution::Click Here To Display Chart@# Would insert the URL into the email and display it as: Click Here To Display Chart
URLofReport::<reportname> {::<linktext>}	Replaces the tag with a URL of the named report. <reportname> is the name of a ClearQuest report. This should be a public report unless it can be ensured that every user has the same query in their personal queries folder. <linktext> is an optional string containing the text users see in HTML body parts instead of the URL itself. **Example:** #@URLofReport::Public Queries/Active Defects::Click Here To Display Report@# Would insert the URL into the email and display it as: Click Here To Display Report
ActionerUserName	Replaces the tag with the ClearQuest username of the person who performed the action that caused the rule to be triggered. **Example:** #@ActionerUserName@# If the username "owenja" had caused the EmailPlus rule to be triggered, this username would be inserted into the email.
NameofUserInField::<field>	Replaces the tag with the full name of the user whose ClearQuest login ID is held in the <field> field. **Example:** #@NameofUserInField::owner@# If the ClearQuest login ID in the owner field is "owenja" and the full name is "John A. Owen" , the full name would be inserted into the email.
NameOfRule	Replaces the tag with the name of the rule that triggered and expanded the template.

Meta tag name	Meta tag purpose
	Example: #@NameofRule@# If the rule that was triggered was "RequestHasChangedState", this text would be inserted into the email.
Expression::<expression>	The operand <expression> can be any valid PERL expression. The expression can include any PERL built-in function or any function defined in the global scripts section of the schema to which the package is applied. The value returned by the evaluation of the expression replaces the tag. Expressions can include embedded field expressions which are described in section 3.1.1. **Example:** #@Expression:: use Date::Format;use Date::Parse; my $template = "%d-%m-%Y %H:%M:%S"; time2str($template, str2time(#?SubmitDate?#, "UTC"));@# This expression takes the current value of the field SubmitDate and formats it using PERL. So if SubmitDate contains the date/time 23 February 2009 at 5:40am this would be formatted by the expression and the following text would be inserted into the email: 23-02-2009 05:40:00
AttachmentByName::<attFieldName> {::<Pattern>[1]}	Inserts attachments into the email from the attachments field called <attFieldName>. Optionally, a regular expression[2] can be provided for selecting which attachments are added. If no pattern is given all will match. The filenames (not the paths) are used in the pattern match. **Example:** #@AttachmentByName::Attachments::^Spec.*@# Matches all attachments whose file name starts

[1] Note: If the target schema uses a version of the ClearQuest attachments package that uses an external storage model, this feature will not work. However, it can be possible to include a link to the external storage site. For more information on where the link is stored, see Attachments package documentation.

[2] A full list of regular expression meta tags can be found in

Appendix B – Regular Expressions Meta Characters.

Meta tag name	Meta tag purpose
	with "Spec" and attaches these files to the email.
AttachmentByDescription::<attFieldName>{::<Pattern>[1]}	This is the same as for AttachmentByName except the pattern match is against the attachment description instead of the filename.
EmbedChart::Public Queries/Chart {{::<<height>>}{::<<Width>>}}	Inserts a ClearQuest chart into the email. The first parameter is the fully qualified name of the chart. **Note:** To ensure visibility for all users, this chart should be a public chart. You can optionally specify the size of the chart in pixels. If height or width are omitted, the default size of 500 x 800 pixels is used. **Example:** #@ EmbedChart::Public Queries/Chart::300:400@# Embeds the chart "Public Queries/Chart" with a size of 300 pixels high by 400 pixels wide into the email.
Other{::<your parameters>}	This is the extensibility facility. You can define your own meta tags in the schema by editing the EMP_SubstituteCustomMetaTag function in the EMP_Customisable global script. See Appendix A – Template Meta Tag Substitution Extension Point for further details.
RestUriOfEntity {::<linktext>}	Replaces the tag with a REST URI generated to take the user directly to the record that caused the email rule to fire. The web server credentials are those defined in the EmailPlusSiteConfig record for the site where the record is mastered. If no site record exists for the replica, <default> is used. If no web servers are defined then no URL can be generated. <linktext> is an optional string containing the text users see in HTML body parts, instead of the URL itself. **Example:** #@ RestUriOfEntity::Click Here To Visit Record@# Would insert the Rest URI into the email and display it as: Click Here To Visit Record ClearQuest Web 7.1 and later support REST URIs. The REST URIs are generated in previous

Meta tag name	Meta tag purpose
	versions but do not work with earlier versions of ClearQuest Web.
RestUri::CHART::<Chart URL> {::<linktext>}	Replaces the tag with a REST URI of the named chart. <Chart URL> is the name of a ClearQuest chart. This should be a public chart unless it can be ensured that every user has the same query in their personal queries folder. <linktext> is an optional string containing the text users see in HTML body parts instead of the URL itself. **Example:** #@RestUri::CHART::Public Queries/Defect Distribution::Click Here To Display Chart@# Would insert the REST URI into the email and display it as: Click Here To Display Chart ClearQuest Web 7.1 and later support REST URIs. The REST URIs are generated in previous versions but do not work with earlier versions of ClearQuest Web.
RestUri::REPORT::<Report URL> {::<linktext>}	Replaces the tag with a REST URI of the named report. <Report URL> is the name of a ClearQuest report. This should be a public report unless it can be ensured that every user has the same query in their personal queries folder. <linktext> is an optional string containing the text users see in HTML body parts instead of the URL itself. **Example:** #@RestUri::REPORT::Public Queries/Active Defects::Click Here To Display Report@# Would insert the REST URI into the email and display it as: Click Here To Display Report ClearQuest Web 7.1 and later support REST URIs. The REST URIs will be generated in previous versions, but will not work with earlier versions of ClearQuest Web.
RestUri::QUERY::<Query URL>	Replaces the tag with a REST URI of the named

Meta tag name	Meta tag purpose
{::<linktext>}	query. <Query URL> is the name of a ClearQuest query. This should be a public query unless it can be ensured that every user has the same query in their personal queries folder. <linktext> is an optional string containing the text users see in HTML body parts instead of the URL itself. **Example:** #@RestUri::QUERY::Public Queries/All Defects::Click Here To Run Query@# Would insert the REST URI into the email and display it as: Click Here to Run Query ClearQuest Web 7.1 and later support REST URIs. The REST URIs are generated in previous versions but do not work with earlier versions of ClearQuest Web.

Table 2. EmailPlus Meta Tags

3.1.1 Working with Embedded Field Expansions in meta tags

It is possible to embed current and old field value tags within the expression tag itself. This is fine if you are certain that the field value does not contain single or double quotes. However, if a field value contains an odd number of single or double quotes it may cause an expression error and the EmailPlusTemplate will not validate correctly. This is a known defect in the EmailPlus package.

A better approach is to use the EmailPlus functions described in section 4.7 of this EmailPlus Administration Guide to evaluate field values. In conjunction with this approach, if you perform field value comparisons use the PERL function quotemeta() to escape any non-alphanumeric characters to ensure that your expression evaluates correctly.

For example, the following expression would evaluate correctly if the field value does not contain single or double quotes :

```
#@expression::if (#?Description?# ne #%Description%#) { "Description field has changed\n".#%Description%#; }@#
```

However, if you are unsure whether the field value will contain single or double quote, then the following example is a better expression to use that would evaluate correctly:

```
#@expression::if (quotemeta(Gfv("Description")) ne quotemeta(Gfov("Description"))) { "Description field has changed\n".Gfov("Description"); }@#
```

As another example, assume that you wanted to display the submit date from your defect record, but you did not want to use the default date format but, instead, wanted to display it in a more readable form.

So here is a simple example expression using a meta tag:

```
#@expression::use Date::Format;use Date::Parse; ctime( str2time( #?SubmitDate?#
));@#
```

This is expanded to

```
#@expression::use Date::Format;use Date::Parse; ctime( str2time( "2010-07-12
10:45:18" ));@#
```

This is a syntactically valid expression; it returns a date object. When the above expression is evaluated, it returns the following:

```
Mon Jul 12 10:45:18 2010
```

This is what is substituted into the output message.

Error! Reference source not found. lists the kinds of data types returned when an old or current field value is expanded within a meta tag.

Field Type	Expression Retuned
INT	Integer For example, 54
DATE_TIME	24 hour date string "YYYY-MM-DD hh:mm:ss" For example "2009-10-20 15:23:04"
SHORT_STRING, MULTILINE_STRING, ID, STATE, RECORDTYPE, DBID, REFERENCE & any other types not specified	Quoted string value For example "some text" Quotes embedded in the original field value will be properly dealt with to yield a syntactically correct string value
REFERENCE_LIST	A reference to an array of quoted values For example: ["value 1", "value 2"]

Table 3. EmailPlus data types returned in meta tags

3.2 Embedding Images in an HTML Body Template

To embed an image in the body of an HTML email message, insert a tag like this one in the body template:

```
<img src="image.gif">
```

Store the image.gif image file in the attachments field of the template (see section 2.5.4). It is important that the image name you specify in the <img> tag matches the filename of an attachment to the template. Attachments with one of the following file extensions can be used as embedded graphics: .jpg, .jpeg, .png, .gif

All files found in the attachments field are added to the body, regardless of whether they are used in an <img> tag. The treatment of additional files depend on the email client. Some clients could treat them as simple attachments while others could display them at the end (or the beginning) of the email.

Do not use a pathname with the <img> tag.

3.3 EmailPlus Expressions and Custom MetaTags and HTML Formatting

A major change has been made in EmailPlus 2.1 to remove inconsistent behaviour in EmailPlus 2.0. This change means that the the output generated by any Expressions or Custom MetaTags in your EmailPlusTemplates may change if you are upgrading from EmailPlus 2.0. When preparing to upgrade from EmailPlus 2.0 to 2.1 it is recommended that you verify the output generated by your EmailPlusTemplates in a test environment prior to upgrading any production databases.

In EmailPlus 2.0 there was an inconsistency with the output of MetaTags and Expressions and Custom MetaTags in HTML formatted emails. Field metatags such as #?fieldname?# and #%fieldname%# were manipulated so that they were formatted in the same way that they were entered in ClearQuest - for example multiline strings would retain the line breaks. However, the results of an expression #@EXPRESSION::<perl expression>@# and the results of any Custom Metatags you may have defined were not manipulated in the same way. As a result formatting such as newlines was not retained.

In order to fix this inconsistency EmailPlus 2.1 now manipulates all output from MetaTags, Expressions and Custom MetaTags so that formatting is retained in HTML emails. This may mean that the output of any HTML EmailPlusTemplates in EmailPlus 2.0 will differ to the new output generated in EmailPlus 2.1. This change will not alter the behaviour when using plain text email templates.

Two new functions have been included in EmailPlus 2.1 to provide you with more control over the output from Expressions and Custom MetaTags. These functions can be called in an EmailPlusTemplate Expression or included in your custom schema code for Custom MetaTags to switch off the default manipulation of text output in HTML emails and also to enable the ability to escape html content so HTML restricted characters are converted to HTML representations.

3.3.1 EMP_Verbatim([$string])

EMP_Verbatim will switch off the automatic manipulation of the output from Expressions or Custom MetaTags. When called in an Expression or called in the schema code for a Custom Metatag the resultant output will not be formatted for HTML emails - this output will in fact be the same as the output generated from EmailPlus 2.0 expressions. If you want to revert the behaviour of Expressions and/or Custom Metatags to EmailPlus 2.0 then you will need to modify your templates to include a call to the EMP_Verbatim() function in Expressions or alter your schema code to call EMP_Verbatim in any Custom MetaTags code you have defined.

EMP_Verbatim can be passed a string as a parameter and it will return a string which has not been formatted for HTML email.

Examples:

A simple solution for those upgrading from EmailPlus 2.0 to retain EmailPlus 2.0 functionality in EmailPlus 2.1 EmailPlusTemplates is to add a call to EMP_Verbatim() in all of your Expressions and Custom MetaTags. This example shows how you would modify an Expression in an EmailPlus 2.1 EmailPlusTemplate record so that the resultant output is the same as it was in EmailPlus 2.0 :

EmailPlus 2.0 Expression:

```
#@EXPRESSION::if ( quotemeta(Gfv("Description")) ne quotemeta(Gfov("Description") ) {
"The Description field has changed: ".Gfv("Description") ; }@#
```

EmailPlus 2.1 Expression:

```
#@EXPRESSION::EMP_Verbatim(); if ( quotemeta(Gfv("Description")) ne
quotemeta(Gfov("Description") ) { "The Description field has changed:
".Gfv("Description") ; }@#
```

If you want to override the default behaviour of auto formatting field value MetaTags (this functionality was introduced in EmailPlus 2.0 and is retained in EmailPlus 2.1) you can use EMP_Verbatim in an expression as follows :

```
#@EXPRESSION::EMP_Verbatim(Gfv("Description"));@#
```

It should be noted that the EMP_Verbatim can not be used in conjunction with the field MetaTags because of the way EmailPlusTemplates are parsed. i.e. #@EXPRESSION::EMP_Verbatim(#?Description?#);@# will not work. You must use the Gfv() function and the Gfov() function as documented in section 4.7.

3.3.2 EMP_EscapeHTML($string)

EMP_EscapeHTML takes a string as a parameter and it converts HTML reserved characters such as <, >, &, " and newlines into HTML representations <, >, &, " and
.

In EmailPlus 2.1 this function is automatically called on the results of all field MetaTags, Expressions and Custom MetaTags and this automatic behaviour can be overridden using EMP_Verbatim. However, EMP_EscapeHTML can also be used in Expressions and Custom MetaTags in conjunction with EMP_Verbatim to allow control over specific sections of resultant output to be manipulated so it is displayed correctly in HTML format email notifications. Using these two new functions will allow the ability to create Expressions or Custom MetaTags that contain text that should be interpreted as HTML (such as formatting) and text that should not be interpreted as HTML (field values).

Example:

The following example is a good illustration of the resultant output from an expression contains HTML code which needs to be interpreted by the email client and field values which should not be interpreted as HTML. It will generate a table of the new and old values of all fields which have been modified in a record except the history field. EMP_Verbatim is called in the expression to disable the automatic HTML formatting of the output and EMP_EscapeHTML is called to only modify the field values that are presented in the table.

```
#@EXPRESSION::
EMP_Verbatim();
$table="";
$fieldsInfos = $entity->GetFieldsUpdatedThisEntireAction();
if ($fieldsInfos->Count() > 0) {
    $table .= "The following fields were modified : <BR>";
    $table .= "<TABLE>";
    $table .= "<TR><TH> Field </TH><TH> New Value </TH><TH> Old Value </TH></TR>";

    for (my $i=0; $i < $fieldsInfos->Count(); $i++) {
        $fieldInfo = $fieldsInfos->Item($i);
        $fieldName = $fieldInfo->GetName();
        next if (lc($fieldName) eq "history");

        $fieldValue = $entity->GetFieldValue($fieldName)->GetValue();
        $oldFieldValue = $entity->GetFieldOriginalValue($fieldName)->GetValue();

        $table .=
"<TR><TD>".EMP_EscapeHTML($fieldName).":</TD><TD>".EMP_EscapeHTML($fieldValue)."</TD><
TD>".EMP_EscapeHTML($oldFieldValue)."</TD></TR>";
    }
    $table .= "</TABLE>";
}
$table;
@#
```

As this is a reasonably complex Expression which may be used in several EmailPlusTemplates you might consider adding this functionality as a CustomMetaTag. Below is a worked example of the segment of code which would need to be inserted into the EMP_SubstituteCustomMetaTag function in the EMP_Customisable global script to create a new Custom MetaTag called #@FIELDSCHANGEDHTMLTABLE@#.

```
    elsif ($customTag eq "fieldschangedhtmltable") {
        if ($isTest) {
            $errorTags = "";
        }
        else {
            EMP_Verbatim();
            $table="";
            $fieldsInfos = $entity->GetFieldsUpdatedThisEntireAction();
            if ($fieldsInfos->Count() > 0) {
                $table .= "The following fields were modified : <BR>";
                $table .= "<TABLE>";
```

```
                $table .= "<TR><TH> Field </TH><TH> New Value </TH><TH> Old Value
</TH></TR>";


                for (my $i=0; $i < $fieldsInfos->Count(); $i++) {
                    $fieldInfo = $fieldsInfos->Item($i);
                    $fieldName = $fieldInfo->GetName();
                    next if (lc($fieldName) eq "history");


                    $fieldValue = $entity->GetFieldValue($fieldName)->GetValue();
                    $oldFieldValue = $entity->GetFieldOriginalValue($fieldName)-
>GetValue();


                    $table .=
"<TR><TD>".EMP_EscapeHTML($fieldName).":</TD><TD>".EMP_EscapeHTML($fieldValue)."</TD><
TD>".EMP_EscapeHTML($oldFieldValue)."</TD></TR>";
                }


                $table .= "</TABLE>";
            }


        $fieldValue = $table;
        }
    }
```

4 Working with EmailPlus Rules

The EmailPlusRule record type provides the EmailPlus package with a rich environment for defining conditions under which email notifications are generated and the users who receive those emails. A correctly defined EmailPlus rule includes the following conditions:

- An initial rule definition, comprising the following details:
    - A name, to uniquely identify the rule by
    - The record type to which the rule applies
    - The name of the EmailPlus template to be used to construct the email header and body
- The definition of the rule conditions that must be met for the rule to trigger
- Details of the users or subscribers who receive the email when the rule is triggered

The rule conditions are constructed from one or more rule phrases. These phrases are logical expressions that evaluate to TRUE or FALSE when a change is made to the record type to which the rule applies. Example phrases are:

Project IS IN "Project A", "Project B", "Project C"

Priority > 1

Action performed on record is "Complete"

These phrases are defined on the EmailPlus Rule, Actions / States and Advance Rule tabs. Rule phrases are linked to each other using an AND/OR logical operator to create the overall rule conditions. Building upon the example above, a rule condition might be:

Project IS IN "Project A", "Project B", "Project C"
AND Priority > 1
OR Action performed on record is "Complete"

In this example, the rule is triggered only if the project field contains one of the values "Project A", "Project B" or "Project C" and if the priority is either greater than 1 or the complete action has been executed on the record. In this case, the rule conditions have evaluated to TRUE. If one of the above conditions had not been met (say, because the project was called "Install new GUI"), the rule conditions would have evaluated to FALSE.

The rule will be triggered only if the rule conditions evaluate to TRUE.

4.1 Initial Rule Definition

When a new rule is created, three things must be established:

- First, the type of record that the rule applies to must be selected from the dropdown list (see 1 in Figure 13. EmailPlus Rule Record) so that EmailPlus can determine the characteristics of the record associated with the rule.
- Then, the template associated with the rule must be specified 2, for defining the structure and content of the notification message.
- Finally, the rule must be given a name 3, which can be anything that adequately expresses the purpose of the rule.

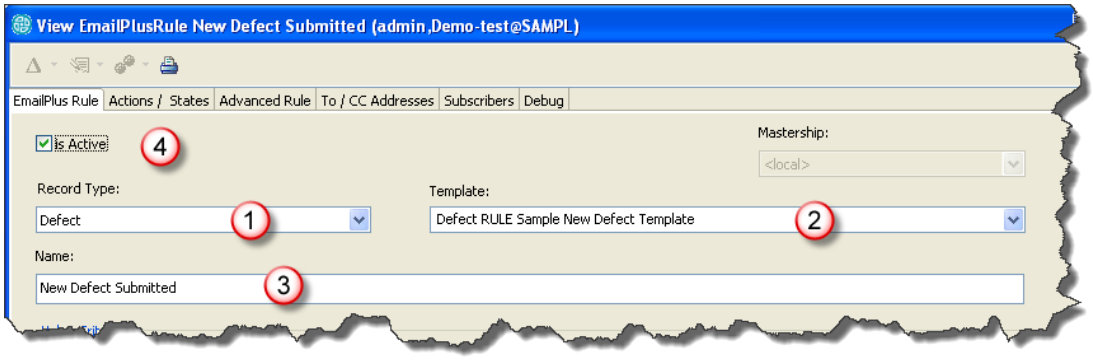


Figure 13. EmailPlus Rule Record Initial Definition

Note: Individual EmailPlus rules can be turned on and off using the "is active" check box 4.

After these basic pieces of information have been provided, the next step is to provide the definition of the rule itself using one or more rule phrases. The rule conditions can use different components, which are described in the following sections.

4.2 Defining a Rule Phrase Using Value Criteria

The value criteria or simple expression fields on the EmailPlus Rule tab allow you to create simple conditions around the values in fields. Up to five groups of value criteria can be provided. Unlike the default Email Rules package, EmailPlus value criteria does not use queries and, therefore, value criteria provide an efficient way to perform comparison operations.

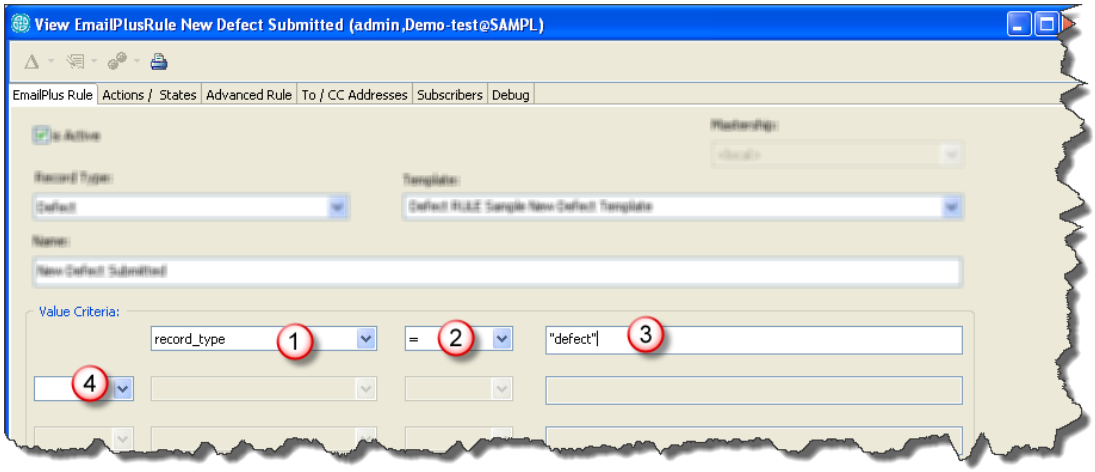


Figure 14. EmailPlus Rule Record - Specifying Value Criteria

The value criteria are made up of three parts:

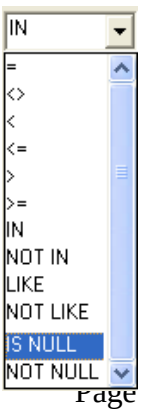
< Field Name Selection > < Comparison Operator > < Comparison Value >

Together, these parts equate to a logical expression which evaluates to TRUE or FALSE. Examples of value criteria are:

ProjectName	IS LIKE	"Proj"
Priority	>=	1
Owner	NOT NULL	
ProjectName	IN	"Project A","Project B","Project C"

These three parts are described in further detail below.

- <Field Name Selection> (1)
  This is a simple list from which a field name associated with the record type can be selected.
- <Comparison Operator> (2)
  This is a list, with the supported comparisons. Available operators are the normal logical operations such as:



=	(equals)
<>	(not equals)
<	(less than)
<=	(less than or equal)

> (greater than)
>= (greater than or equal)

>=

In addition, Table 4 lists further comparison operators. These operators are similar to comparison operators provided in SQL.

Comparison Operator	Description
IN	The value criteria will return TRUE if the value of the field provided in the <Field Name Selection> is one of the values provided in the <Comparison Value>. **Example:** Field Name Selection = *Project* Comparison Operator = *IN* Comparison Value(s) = *"Project A", "Project B", Project C"* When the EmailPlus rule is evaluated if the value of the field Project is "Project A" then the value criteria will return TRUE. However, if the value of the field Project is "Project Z" (which is not in the list of comparison values) then the value criteria will return FALSE.
NOT IN	This is similar to IN, except that the value criteria returns TRUE if the value of the field provided in <Field Name Selection> is not one of the values provided in the list of <Comparison Value>.
LIKE	The value criteria will return TRUE if the value of the field provided in the <Field Name Selection> contains the text provide in <Comparison Value>, bearing in mind that string comparisons are case sensitive. **Example:** Field Name Selection = *Project* Comparison Operator = *LIKE* Comparison Value = *"Proj"* When the EmailPlus rule is evaluated if the value of the field Project is "Project A" then the value criteria will return TRUE since "Project A" contains the text "Proj". However, if the value of Project is "New System Icon" then the value criteria will return FALSE since this project does not contain the text "Proj".
NOT LIKE	This is similar to LIKE, except that the value criteria returns TRUE if the value of the field provided in <Field Name Selection> does not contain the value provided in <Comparison Value>.

Comparison Operator	Description
IS NULL	The value criteria will return TRUE if the value of the field provided in the <Field Name Selection> is empty or has no value. **Note:** This comparison operator does not require the <Comparison Value> to be set. **Example:** Field Name Selection = *Project* Comparison Operator = *IS NULL* Comparison Values are not used When the EmailPlus rule is evaluated if the value of the field project is "" (empty) then the value criteria will return TRUE. However, if the value is not-empty (for example, "Project A") then the value criteria will return FALSE.
NOT NULL	This is similar to IS NULL, except that the value criteria returns TRUE if field provided in <Field Name Selection> is not empty.

Table 4. EmailPlus Rule Record - Additional Logical Operators Supported

- <Comparison Value(s)> 

This provides the value or list of values (when using the IN/NOT IN comparison operators) against which the value in the field is compared using the comparison operator.

Comparison values do not need to be provided when using the comparison operators IS NULL and NOT NULL.

String values must be enclosed in quotation marks, for example "Project A".

Lists of values must be represented as a comma-separated list, where strings are enclosed in quotes, this is especially important for values that contain spaces. For example "Project A", "Project B", "Project C" for a list of strings or 1,2,3 for a list of numbers.

Comparison values can also be specified as a PERL expression which returns a value which is of the same type as the field in the value criteria and which is also valid for the comparison operand used. The expression could call a routine defined in your schema, or a ClearQuest API call. So, for example, to only send an email if the current user was the submitter, you could use the ClearQuest GetUserLoginName() API call to return the current user as shown in Figure 15. EmailPlus Rule Record - Example value criteria using a ClearQuest API call.

Figure 15. EmailPlus Rule Record - Example value criteria using a ClearQuest API call

You can also use this mechanism to add your own dynamic comparison values to the EmailPlus value criteria. For example, you could add your own function to the schema which returns date comparison values like YESTERDAY, LAST_WEEK, etc.

- When multiple value criteria conditions are in use, you can combine them by selecting the required AND/OR logical operator (see 4 in Figure 14).

An advanced expression facility, with which you can formulate more complex expressions, is explained further in section 4.7.

4.3 Defining a Rule Phrase Using Field Changes

The field changes section of the EmailPlus Rule tab allows the addition of a phrase to the rule condition, which looks for fields which have changed rather than the values they contain. This rule phrase comprises three parts:

< Phrase Operand > < Fields Operand > < Fields Changed >

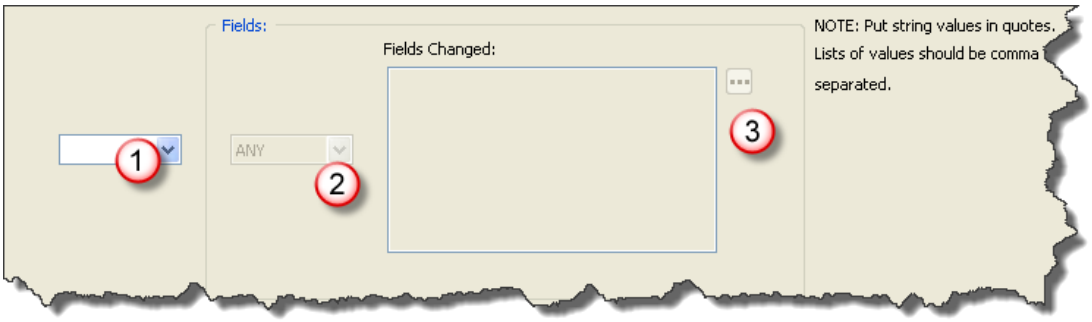


Figure 16. EmailPlus Rule Record - Specifying Field Changes

< Phrase Operand >

If you want to look for changed fields, you must first select the phrase operand (AND or OR) from the list (see 1 in Figure 16). Doing so combines the fields changed phrase with any value criteria phrases defined previously.

Note: Even if there are no value criteria, the AND/OR operand still needs to be selected to make the input fields associated with this rule phrase editable.

< Fields Operand >

You then have to decide what combination of field changes you are looking for by selecting the required fields operand from the list (see 2 in Figure 16):

ALL

All the fields listed in < Fields Changed > must have changed for this phrase to evaluate to TRUE.

ANY

At least one of the fields listed in < Fields Changed > must have changed for this phrase to evaluate TRUE.

NONE

None of the fields listed in < Fields Changed > must have changed for this phrase to evaluate to TRUE.

< Fields Changed >

Finally, you must select the fields on which to base this rule phrase. To do this, click the multi-select box icon (see 3 in Figure 16). You will see a window similar to that shown in Figure 17 but with all the fields in your selected record type displayed instead.

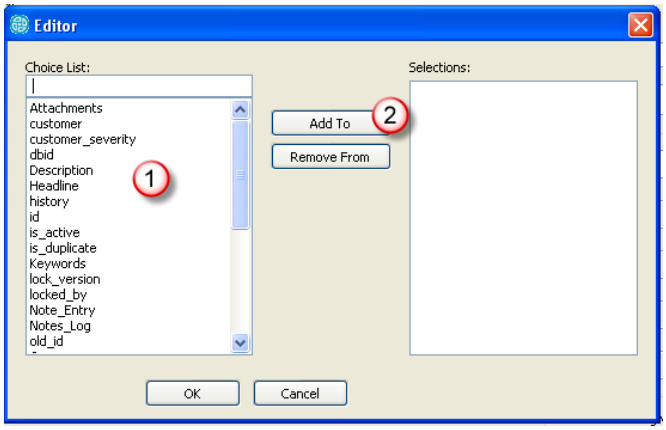


Figure 17. Field selection choice list

Select the field that you want 1 and then click the Add To button 2 to add it to the selected list of fields. Repeat this step for all the other fields you want to use, and click **OK**. The chosen fields are displayed in the Fields Changed pane (see 3 in Figure 16).

4.4 Defining Rule Phrases Using Actions and States

The EmailPlus Rule – Actions / States tab allows you to check for specific changes in state or for particular actions being performed. Figure 18 shows an example of this tab.

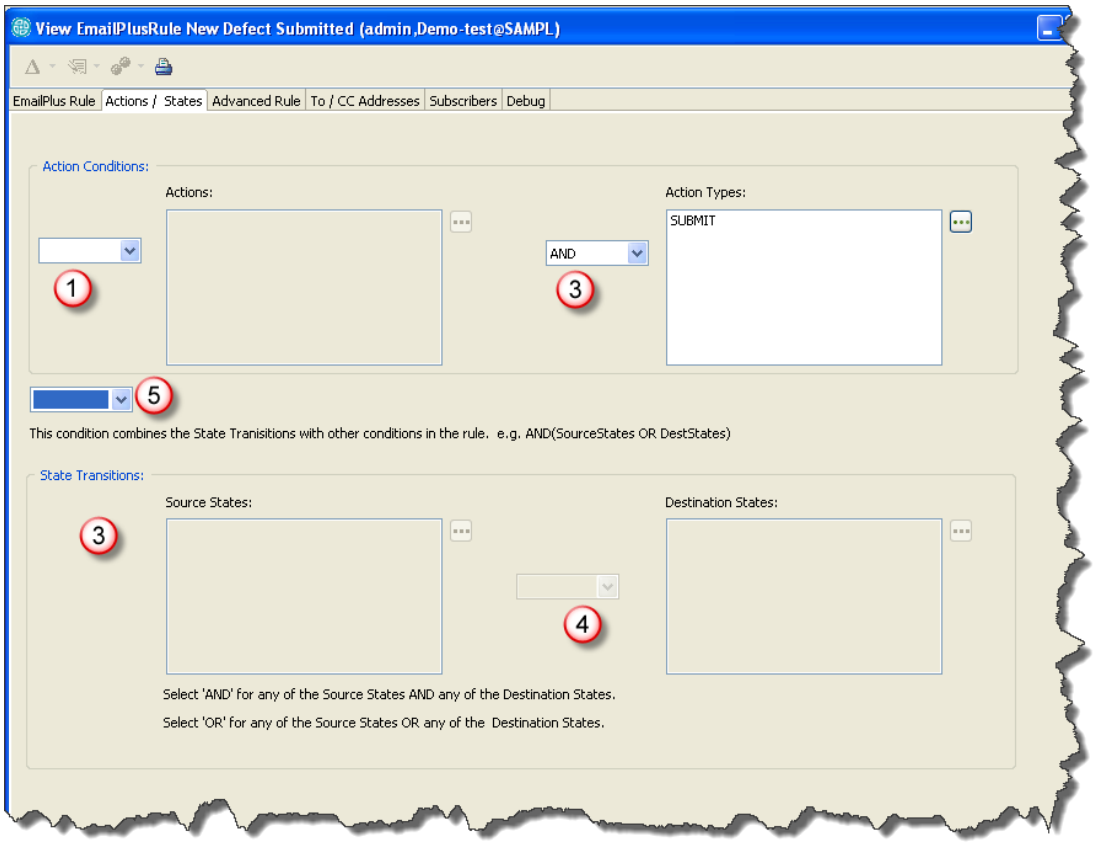


Figure 18. EmailPlus Rule Record - Actions / States Tab

4.4.1 Action Conditions

Two kinds of action conditions can be specified:

Actions: (1)

If the record was updated using one of the action names (for example: Submit, Close, Open, Assign) specified in the list, the rule phrase evaluates to TRUE.

Action Types: (2)

If the record was updated using one of the action types (for example: CHANGE_STATE, MODIFY) specified in the list, the rule phrase evaluates to TRUE.

In each of the cases, the phrase operand (AND/OR) must be selected first to make the rule phrase editable. You can then choose the list of actions and action types associated with the rule phrase by clicking the multi-select icon and picking from the list of the possible choices.

4.4.2 State Transitions

The State Transitions section of the Actions / States tab allows the option of choosing to add a rule phrase based upon the source or destination states associated with the record being updated. To make the state transitions section editable, the required AND/OR phrase operand (5) must be selected.

The state transitions rule phrase has the following format:

(<Source States> <AND/OR operand> <Destination States>)

The phrase is enclosed in brackets ( ) to indicate that the whole phase is evaluated as TRUE or FALSE. The rule can include either source states or destination states or both.

<Source States> 3

Specify any source states which are needed to trigger the rule. If no source states are required, the list can be left blank.

<AND/OR Operand> 4

If the rule needs to include destination states, ensure the required AND/OR operand is selected.

AND

For the complete state transitions rule phrase to evaluate to TRUE, both the source state for the record and the resultant destination state must appear in the respective lists of source and destination states.

OR

For the complete state transitions rule phrase to evaluate to TRUE, either the source state or the resultant destination state must appear in their respective lists.

<Destination States> 4

Specify any destination states that are needed to trigger the rule. If no destination states are required, the list can be left blank.

4.5 To/CC Addresses Tab

The To / CC Addresses tab allows you to specify who should receive email notifications when the rule is evaluated and found to be TRUE.

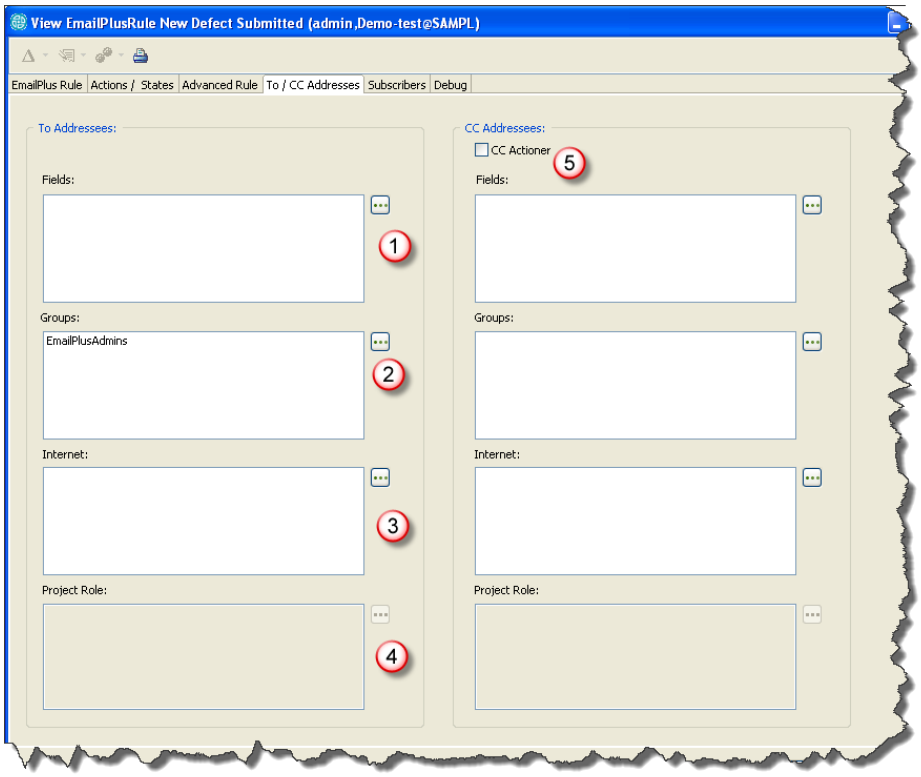


Figure 19. EmailPlus Rule Record - To / CC Addresses Tab

You can notify users by using any of the options included with the out-of-the-box ClearQuest Email package.

Fields:

The multiselect icon next to the Fields pane (see 1 in Figure 19) allows you to select fields from the ClearQuest record that are of type *Reference to Users*.

EmailPlus also searches referenced records for fields of type *Reference to Users*. So, for example, if you had a reference to a project record and the project record contained a field called ProjectMembers which was of type *Reference to Users*, then this field would also appear as an option Project.ProjectMembers. This facility is configurable. See Appendix D - Configuring Addressing Options.

When the EmailPlus rule evaluates to TRUE, the ClearQuest user identified in the selected fields receives a copy of the email notification as a primary recipient.

A similar action for the CC Addresses column sends a notification to the user as a CC addressee.

Groups:

The multiselect icon next to the Group pane 2 allows you to select one or more ClearQuest groups.

When the EmailPlus rule evaluates to TRUE, everyone in each of the selected groups receives an email notification as a primary recipient.

A similar action for the CC Addressees column results in users being notified as a CC addressee.

Internet:

The multiselect icon next to the Internet pane (3) allows you to enter Internet style addressees. You can use these for recipients who are not ClearQuest users or who do not have a ClearQuest login ID. A similar action for the CC Addressees Column results in users being notified as a CC addressee.

Project Role:

The multiselect icon next to the Project Role pane (4) allows you to select the project roles whose members will be notified when the EmailPlus rule is triggered.

This facility is available only if the companion ClearQuest package ProjectRoles is also installed in your schema. Otherwise, these fields will remain read-only. The project name is taken from the record just edited and this is used to find the role membership for the required project.

EmailPlus does not provide a way to add ClearQuest User IDs to the addressees because this facility is provided by the self subscription mechanism described in section 4.6. Subscriptions can be defined either on the EmailPlus rule record or the record types to which the EmailPlus package has been applied. This reduces administrative load since this is a self-service operation.

You can also generate a CC Copy for the person who performed the action on the record that caused the notification to be generated. To do so, select the CC Actioner checkbox (5).

Note: When the same address appears in both the To and CC lists, the address is removed from the CC list so that any given person is notified only once.

4.6 Subscription

The Subscribers tab gives you control over the subscription facility. For individuals to be able to subscribe to a rule, you as the administrator must create a subscription record for each site or replica that needs the facility. If no subscription records exist, the subscription facility won't be available for that rule.

First, you must create the subscription record. This is done by clicking the **New** button (see (1) in Figure 20. EmailPlus Rule Record - Subscribers Tab).

You can also restrict who is allowed to subscribe to a rule by specifying that prospective subscribers must belong to a list of specified ClearQuest user groups (2). This is done so that users, who are not normally able to see a record because of ClearQuest security contexts, cannot subscribe to a rule by circumvent the security settings.

Note: This mechanism only prevents a person subscribing; it does not automatically remove them if the group memberships subsequently change. Removing a person from a group remains a manual operation.

If no subscription groups are specified, anyone can subscribe.

If you have created a *Special Interest* EmailPlus template for an enabled record type, users are able to subscribe to change notifications of individual records. They do this by selecting the records and adding themselves to the records subscription list. To be able to subscribe oneself, the user must be able to perform some action on the record. If you have not created a Special Interest template, this type of subscription does not work.

Note: If you remove a Subscriber list entry, the referenced subscription record is also deleted if possible. In ClearQuest MultiSite environments, it is not possible to delete subscription lists that are mastered elsewhere. In such cases, removed subscription records must be manually deleted before it is possible to create a new subscription record for that remote site. After a rule has been removed, you cannot reattach a subscription list record to the rule.

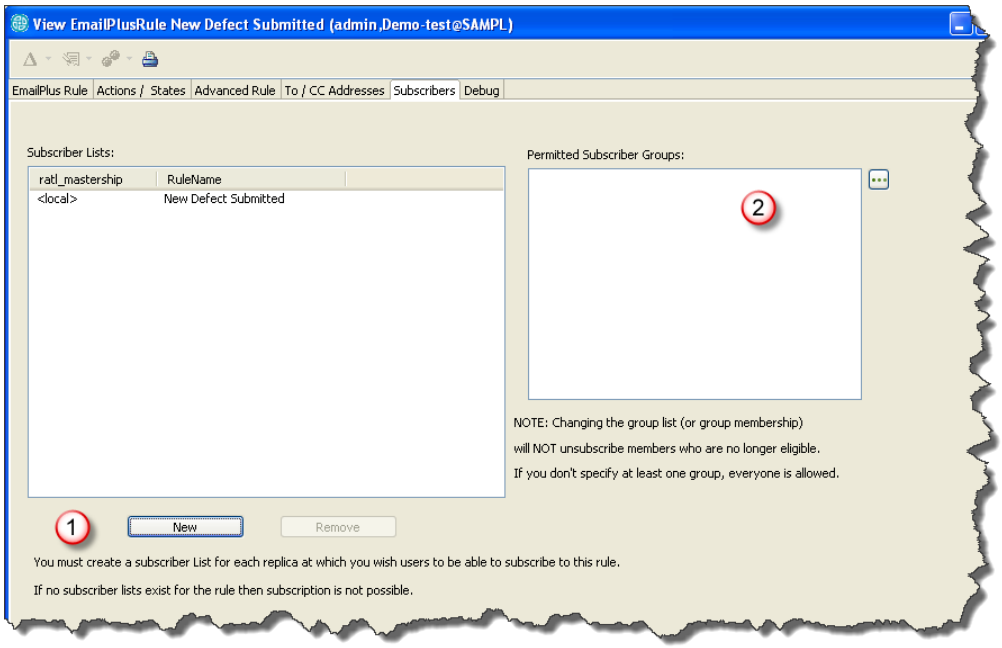


Figure 20. EmailPlus Rule Record - Subscribers Tab

4.7 Defining a Rule Phrase Using an Advanced Rule

The Advance Rule tab, as shown in Figure 21, provides the final method for defining an EmailPlus rule phase.

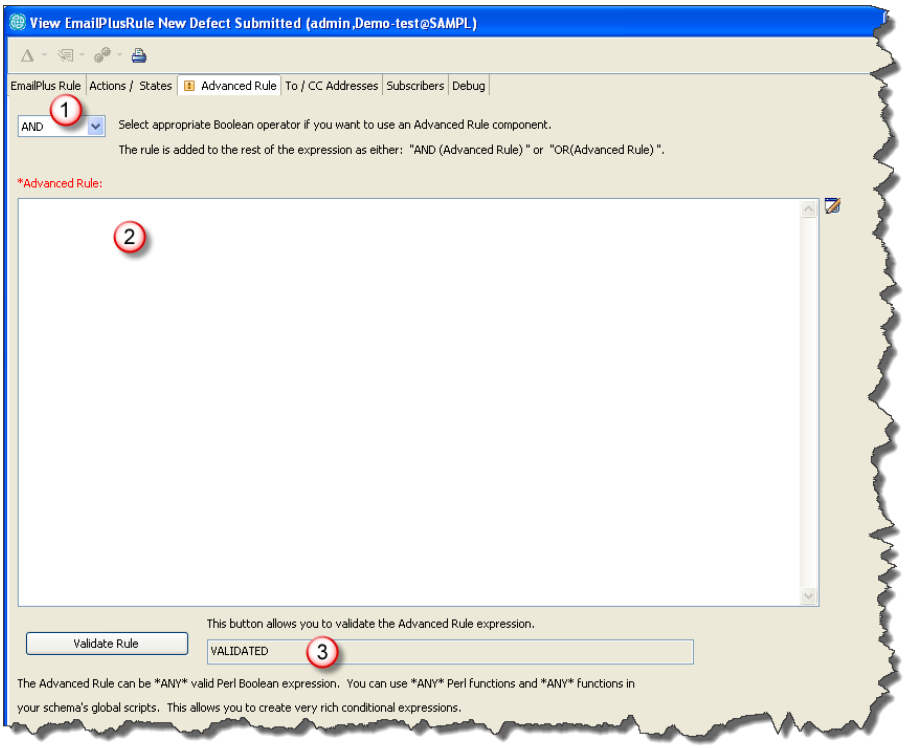


Figure 21. EmailPlus Rule Record - Advance Rule Tab

The advanced rule allows you to combine any valid PERL expression with the rest of the rule conditions. In fact, all of the other parts of the rule could be expressed as a PERL expression. They are there more for your convenience. This means you can construct any expression you like and that expression can do the following actions:

- Call any built-in PERL function
- Call any function defined in the global scripts section of your schema
- Call any function from the PERL ClearQuest API itself, or any other object available to PERL

Additionally, the expression created using the advanced rule is not limited to working with just the record being edited.

To use the Advanced Rule feature, select the required AND/OR phase operand (1) to combine it with the rest of the rule and then type your expression in the space provided (2).

You can check whether the expression will work by clicking on the Validate Rule button (3). If the rule is valid, the string **VALIDATED** is displayed next to the button, else the PERL expression evaluation error is displayed.

The global scripts of the EmailPlus package contain a number of useful functions. These are described in Table 5.

Function	Description
OneOf($list, $item)	This routine takes a list and searches it for the presence of the given item. It returns TRUE if the item is found, else FALSE. $list - A reference to an array of string values $item - The value to search the list for
Gfv($fieldName)	Simple encapsulation of GetFieldValue which is equivalent to: GetFieldValue($fieldName)->GetValue() $fieldName – The name of the field
Gfov($fieldName)	This is similar to Gfv, except that it is the simple encapsulation of GetField***Original***Value, which returns the original value of a field before any changes were made.
Gfvs($fieldName)	This function is the simple encapsulation of ClearQuest API call GetFieldValueStatus and is equivalent to: GetFieldValue($fieldName)->GetValueStatus()
FChg($list)	Examines the list of given fieldnames and returns a string to indicate what fields have changed: "ANY" means one or more of the fields in the list have changed "ALL" means all the fields in the list have changed "NONE" means none of the fields have changed $list - A reference to an array of field names to check for change.
StoDT($dateString)	Converts a date string value from a ClearQuest DATE_TIME field to a UTC date/time in seconds. The return value is the number of non-leap seconds since the epoch. On most systems the epoch is 00:00:00 UTC, January 1, 1970. $dateString - A date string of the format "YYYY-MM-DD hh:mm:ss".

Function	Description
DTtoS($timestamp)	Converts a UTC Date/Time in seconds into a date string of format YYYY-MM-DD hh:mm:ss. $timestamp – A UTC date/time in seconds.
EMP_ReFormatDateString($dateString, $template, $timezone)	This converts a ClearQuest DATE_TIME field to a formatted string of your choice, by defining a template and a time zone. For details, see Appendix H – Reformatting DATE_TIME fields for use in EmailPlus notifications.

Table 5. EmailPlus Rule Functions

5 Administration of EmailPlus

All administration activities for the package are controlled by membership of the ClearQuest user group called EmailPlusAdmins. This group should be created and then subscribed to all databases that use a schema to which the EmailPlus package has been applied. This separate group is used so that administration of EmailPlus capability can be delegated to specific individuals. If you already have a ClearQuest administrators group and want the same group of people to administer EmailPlus, make your existing administrators group a member of the EmailPlusAdmins group.

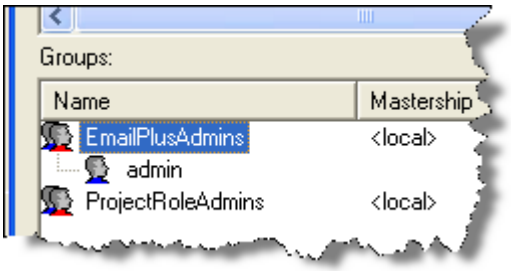


Figure 22. EmailPlusAdmins ClearQuest Group

6 Applying the EmailPlus Package

Because the package is not a default offering for a ClearQuest installation, you must manually configure it to make the package available to ClearQuest. These steps need to be done only on the machine you use to add the package to your schema. After the package is applied, it is copied into your schema and available to all other clients. Any vendor components used by the package must be installed separately.

Note: If you ever export your schema with the intent of creating a new schema from it in a new schema repository, you must also provide a copy of the package install files for the recipient to be able to import your schema.

6.1 Installing the Package

The package is supplied as an archive containing all the necessary files. To install the package, you must first read the EmailPlus Release Notes and Installation Guide, then

extract the contents of the archive and save the files in the correct locations. Then, proceed as follows:

1. Restart the designer if it was already running when you installed the package files.
2. Use ClearQuest Designer to register the new package by starting the package wizard (see Figure 23. ClearQuest Designer Start Package Wizard).

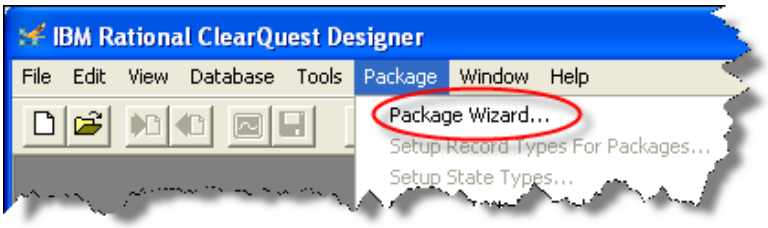


Figure 23. ClearQuest Designer Start Package Wizard

3. In the package wizard, if the EmailPlus package is not listed, select More Packages (see Figure 24).

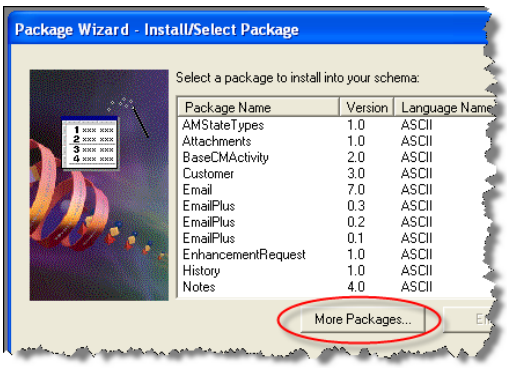


Figure 24. Locate Unregistered Packages

4. Find the newly installed package, select it, and click OK (see Figure 25).

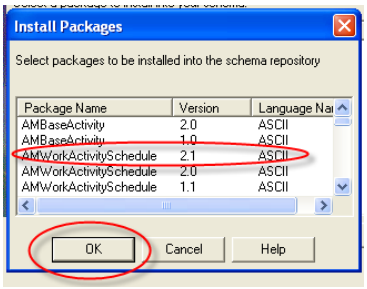


Figure 25. Selecting the package

After the package is installed and registered, it can be applied to the record types in the schema.

6.2 Applying the Package to your schema

To apply the package to your schema, perform the following steps:

NOTE: In a ClearQuest MultiSite environment, the package should be registered only at the working master site.

1. Start the ClearQuest designer and then start the package wizard (see Figure 26).

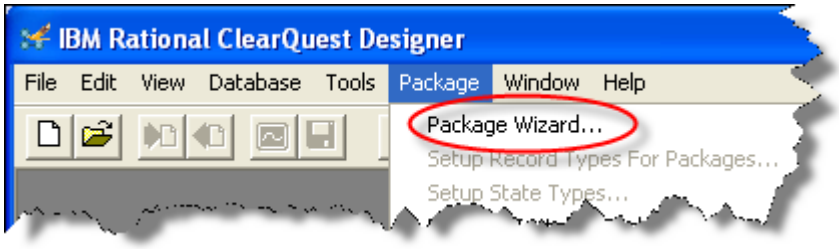


Figure 26. Starting The Package Wizard

2. Select the EmailPlus package to be applied and click Next (see Figure 27).

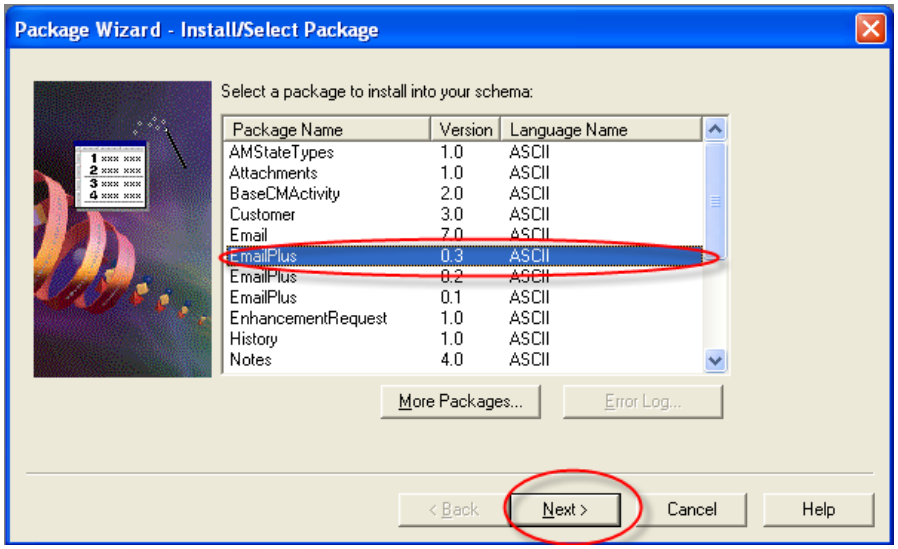


Figure 27. Package Wizard - Selecting The EmailPlus Package

3. Select the schema version you want to apply the EmailPlus package to and click **Next** (see Figure 28).

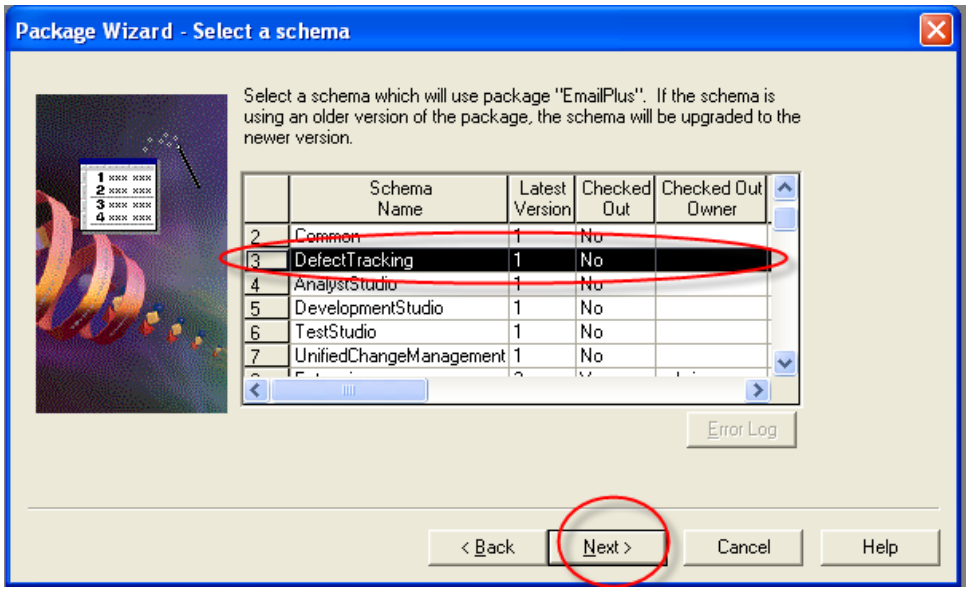


Figure 28. Package Wizard - Selecting the schema which will use EmailPlus

4. Enable all the record types that you want the package to work with (see Figure 29). To complete the process, click Finish.

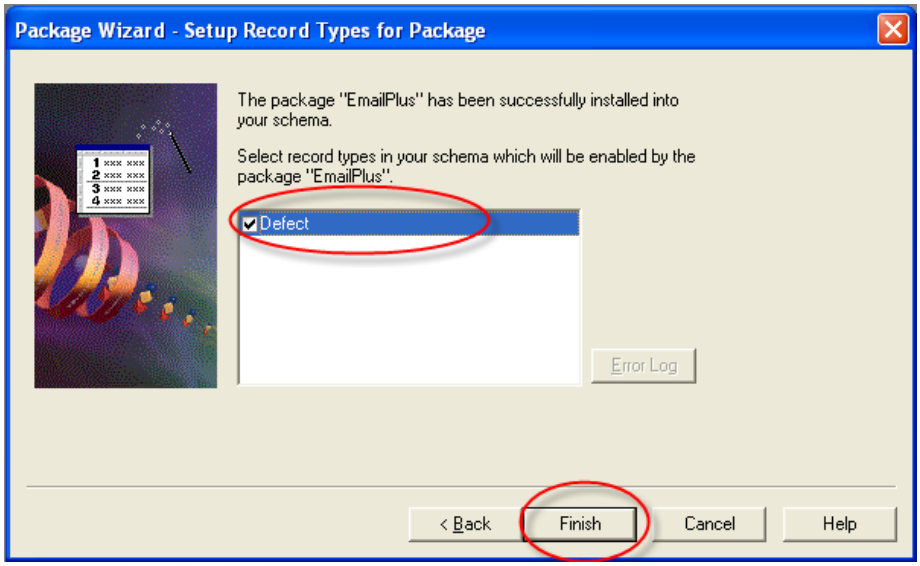


Figure 29. Package Wizard – Enabling record types with EmailPlus

6.3 Enabling a Stateless Record Type

In EmailPlus 2.1 it is now possible to apply the package to stateless records either through the ClearQuest Designer or from the command line.

Using the ClearQuest Designer the process is the same as that used when applying the package to a stateful record. From the ClearQuest Designer click Packages > Setup Record Types for Packages. For further information about applying packages in ClearQuest see the ClearQuest Administrator's Guide.

From the command line it is also possible to apply the package to stateless record types using the command "packageutil enablerecordtype" :

```
packageutil enablerecordtype -dbset <dbset_name> <clearquest_login>
<clearquest_password> <schema_name> <package_name> <record_type_name_to_enable>
```

Refer to the command documentation by typing "packageutil enablerecordtype –help" for the full list of options.

6.4 Configuring an Enabled Record Type

Record types enabled by the package can be customized.

1. If you do not want users to have the ability to subscribe to notifications associated with an individual record, either hide or restrict access to the Subscriber List tab. If you leave the tab visible and do not create a Special Interest template, user subscription is refused.
2. If you want to take advantage of the change of owner notifications the package provides, locate the following line of code in the EmailPlusNotify Notification hook:

   ```
   EMP_ExecuteNotification(undef, "SubscriberList");
   ```

   Change the undef part to include a reference to an array of all the fields you regard as ownership fields. For example,

   ```
   EMP_ExecuteNotification( [ "Owner", "Submitter" ], "SubscriberList");
   ```

3. Provided an ownership type EmailPlusTemplate record exists for the enabled record type, the package generates notifications whenever any of the listed fields change. The notifications are sent to the old and new owners.

   Note: An email is generated even if one of the designated owner fields was blank originally or becomes blank later. Any change to any of the designated owner fields triggers the email generation if an ownership template exists for the enabled record type.

6.5 Recommended Testing Strategy

Because ClearQuest packages cannot be removed from your schema after they have been applied (except by deleting the schema version they were applied to and all later versions), you must ensure the package does what you want and will work in your environment before you apply it to your live schema.

Therefore, evaluate the package before applying it to your live schema.

Take a full backup of all your ClearQuest databases before you apply the package.

6.5.1 Copy Live Environment

This strategy involves making an entirely independent and complete copy of your live environment databases so that the effects of the package can be judged. To do this, make copies of your schema repository and user databases using the command line utility 'installutil'. The process for doing this is documented in Tech Note #1118690.

Note: If you are in a MultiSite environment, you should ensure that there is no possibility of shipping updates packets from your copy to the live environments.

Copies of ClearQuest databases are must never be used until you have updated the database credentials in the copy of the schema repository. Until these are updated, the

copy of the schema repository points to your live databases and they are at risk if they can be accessed.

If this arrangement is not satisfactory, seek advice or use the Copy Schema strategy outlined below.

6.5.2 Copy Schema

This strategy needs the following steps:

1. Using the ClearQuest Designer, create a new schema based upon the most recent version of your live schema.
2. Apply the package to the newly created copy of your schema.
3. Create a new database and base it on the newly created copy of your schema.
4. Test the amended schema in this new database and verify that it works in your environment.
5. If you encounter no difficulties, you can apply the package to your live schema.

7 Mail Relay

Access to Simple Mail Transfer Protocol (SMTP) servers is strictly controlled in some organizations. This often means that the ClearQuest clients cannot directly access the SMTP server for email notifications to be sent without the IP address of the client machine being registered with the SMTP server or a firewall surrounding it.

In these kinds of environments, it could be difficult to register every ClearQuest client machine because the whole purpose of this restriction is to reduce the risk of the SMTP server being used to send spam mails. In such a scenario, you can consider the following options:

- Implement your own SMTP server for ClearQuest emails.
- Implement some kind of private relay service to forward the email request to a server that is registered for SMTP server access. This cannot simply be an SMTP relay because otherwise the problem of spam emails continues to exist.

The EmailPlus package implements the second of these options in two different ways. These are implemented using two different transport types: **SMTP Relay** and **PostOffice Relay**.

7.1 SMTP Relay

This transport type is implemented using a Windows application. The application must be started manually for it to receive email relay requests.

When this transport method is used, the EmailPlusSiteConfig fields for the SMTP server are used in a slightly different way. The SMTPServerName field becomes the name of the server where the relay service is installed. Similarly, the SMTPPortNumber field is the port number the mail relay service listens on. It is important that the mail relay service is configured to use the same port as defined in your EmailPlusSiteConfig records.

Figure 30 presents an overview of the communications path taken to deliver mail.

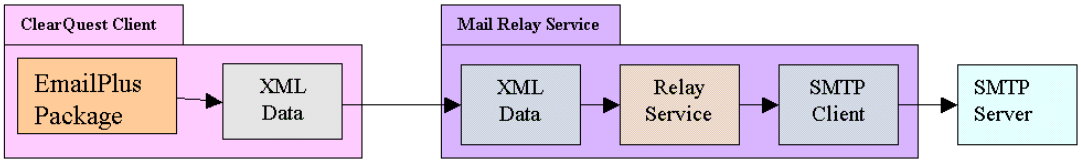


Figure 30. SMTP Relay Architecture

When the ClearQuest client executes the hook code in the EmailPlus package, it calls upon the installed component to establish a link with the Mail Relay service that receives the email encoded in an XML stream. This is then decoded by the relay service and email sent to the SMTP server using either the SMTP client provided by the ClearQuest mail transport or the one provided in the installed component.

The default port number used is 36001. Whatever port you select, that is the only one required for incoming requests should the relay service be behind a firewall. Access to the SMTP server port configured is required for outgoing requests.

7.2 PostOffice Relay

This transport type achieves the same results as the SMTP Relay transport type but by using a different method.

The reasons for using this transport type are as follows:

- Increases resilience of generated emails. Because they are registered in the ClearQuest database, they cannot be lost even if the relay service goes offline for a while. When it gets restarted, the service picks up from where it left off and delivers all the outstanding emails.
- Eliminates the need for an installable component from client desktops. You still have the choice of using the installable component or CQMail transports in the relay component itself. If you do not want to use the HTML body part, and embedded graphics and attachments, you can entirely eliminate the need for the installable component.

Figure 31 presents an overview of the communications path taken to deliver mail.

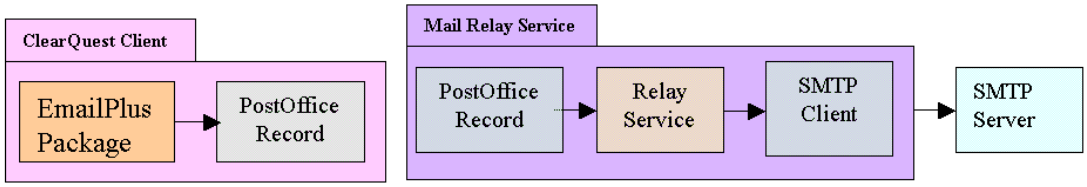


Figure 31. PostOffice Relay Architecture

7.3 EMP Relay Application

The EMP Relay application is a Windows-only application that is responsible for receiving XML-encoded mail notifications from your EmailPlus-enabled ClearQuest database and then translating those requests into actual emails. This application is intended to be used in environments where access to standard SMTP servers is

restricted. The EMP Relay application is intended to be used with the following EmailPlus mail transport modes:

- **Network Relay Mode**
  The service listens for incoming network connections from the EmailPlus enabled ClearQuest database. It accepts the XML encoded information, then decodes it, and establishes a connection to the SMTP server to send the mail.
- **PostOffice Relay Mode**
  The service periodically polls your EmailPlus-enabled ClearQuest database for EmailPlusPostOffice records. Each EmailPlusPostOffice record contains an XML encoded mail notification. The data contained in this record is decoded and then forwarded through email to the SMTP server.

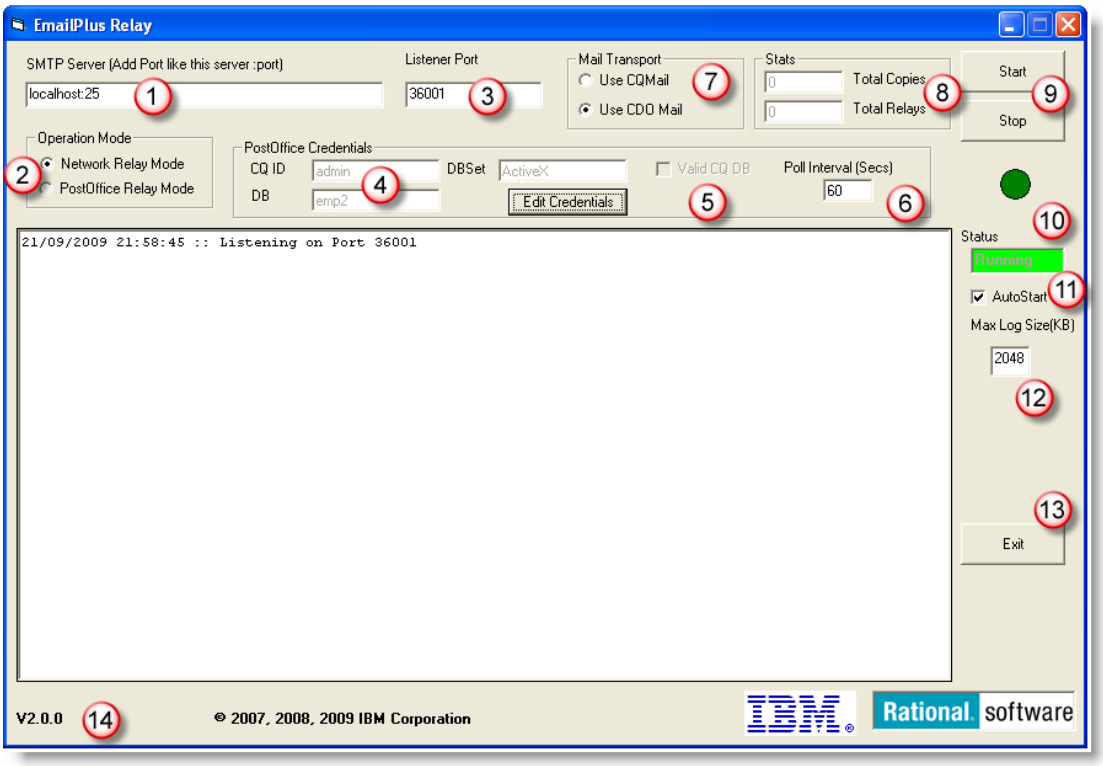


Figure 32. EmailPlus Relay Interface

Table 6 describes the various parts of the EmailPlus Relay Interface shown in Figure 32.

Field	Definition
1	Specifies the name of the SMTP server and port to use. This is used only by the CDO[3] Mail transport option. If the CQMail transport option is used, then the ClearQuest client settings are used.

[3] CDO – Collaboration Data Objects – A Microsoft® Windows® component providing SMTP Client services.

Field	Definition
2.	This area selects the operating mode of the tool. • When in Network Relay mode, the service listens for incoming forwarding requests on the port specified 3. • When in PostOffice Relay relay mode, the service checks the designated ClearQuest database for PostOffice requests.
3.	This is the port the service listens to for incoming forwarding requests. This is only used in Network Relay mode. This must match the port given in the EmailPlusSiteConfig record.
4	This area identifies the ClearQuest database containing EmailPlusPostOffice records to process. The database must be based on a schema to which the EmailPlus package has been applied. This is only used in the PostOffice Relay mode. In order to edit the credentials click on the button. **Note**: The CQ Login ID provided must be a member of the EmailPlusAdmins group for the service to function correctly.
5	This indicates if the ClearQuest database credentials supplied are for a database that the service can run with. Only applicable in PostOffice Relay mode.
6	Specifies the database poll interval in seconds. This is only used in PostOffice Relay mode and determines the period between queries looking for post office records to process.
7	Selects the mail forwarding transport mode. In CQMail Mode, the emails are sent through the ClearQuest mail component that must have been correctly configured using the ClearQuest Client. In CDO Mail transport mode, the mail is forwarded to the SMTP server identified in 1. This transport mode affords the greatest level of functionality.
8	Provides statistics surrounding the amount of mail sent. The statistics are reset only when the application is closed. • **Total Copies** records the total number of mail recipients of all emails sent. • **Total Relays** records the number of mail requests.
9	The Start and Stop buttons start or stop all forwarding. Pressing the Start button causes an immediate database poll when in PostOffice Relay mode.

Field	Definition
10	These are indicators of the current status of the service. • The green circle flashes once a second to indicate that the service is alive. • The status box indicates if the service is forwarding mail or not. ○ When the status box contains the word running the service is actively forwarding emails. ○ When the status box is highlighted red it is not forwarding emails.
11	The autostart box indicates whether the service should enter the running mode immediately without any intervention using the settings that were in place when it was last run.
12	This is the size in KB of the space allocated to the log. The minimum size is 16 KB but can be increased. Logs cannot be saved although you can select the content, copy it to a text editor, and save the file.
13	The Exit button causes the application to terminate. Current settings are saved at this point. The application might not terminate immediately if it is actively processing requests.
14	This is the version number of the application and must be includes in all problem reports.

Table 6. EmailPlus Relay Interface Window Definitions

8 Performance

The EmailPlus package is implemented making extensive use of caching techniques to avoid continually retrieving information from the ClearQuest database and thus minimize the impact of evaluation and execution of the email rules on the ClearQuest environment.

However, because the email rules provide significant flexibility, care should be taken when constructing the rules to ensure that any *Advanced Rules* minimize their own impact on performance.

EmailPlus caches all rules for any given record type in memory the first time they are examined in any ClearQuest session. No matter how many rules there are for a single record type, only 1 record is fetched from the database.

The Simple EmailPlus rule phrases are not evaluated using ClearQuest queries. Evaluation of these phrases is undertaken entirely in memory.

If you make use of the PostOffice transport mode, you should consider the following points:

- There is an additional load on the database. Each email generated results in a record and its history being created in the ClearQuest database. There is a corresponding cost when the relay service processes the record and subsequently deletes it. Additionally, every time the relay service polls the database for EmailPlusPostOffice records, a ClearQuest query is run against

the database.
You must find a balance between the number of queries and the number of records processed at one time to even out load. The additional load presented to the database is dependent on the number, kind, and frequency of email rules generating messages. This balance needs to be achieved by a process of trial and error. Ideally, the relay service should not be processing large numbers of records in a batch, nor should it be running excessively frequently.

- If Post Office records are not processed by a local instance of the relay service, emails are delayed by approximately one MultiSite replication cycle period.

9 Email Attachments

EmailPlus includes the ability to send ClearQuest record attachments in emails when using one of the following email transport types:

SMTP Direct

SMTP Relay

PostOffice

The EmailPlus Template records also allow you to include attachments, such as images, when generating HTML emails with inline images. The record and template attachments are cached to the ClearQuest client when an EmailPlus notification is sent. Because of this feature, consider the following things when enabling attachments in EmailPlus.

Security:

Due to the confidential nature of some record attachments, you must take precautions to ensure that record attachments are not sent to people who do not have authority to view them.

Performance:

When attaching record attachments to EmailPlus notifications, you must consider the performance of record modification due to the fact that the ClearQuest client downloads the record attachments to your local machine or ClearQuest Web server when constructing the EmailPlus notification. Attaching large record attachments to EmailPlus notifications impacts the performance of downloading the attachments to the ClearQuest client depending on the network speeds and proximity to the ClearQuest server. You should consider how your users access ClearQuest in your environment.

9.1 Default ClearQuest Client Temporary Cache Directory

When attachments are included in an email, the files are downloaded to a temporary directory on the ClearQuest Client local machine. The tables below illustrate the order of the defined default locations for these temporary directories.

Windows location	**Example**
1. The value of the %TEMP% environment variable	C:\Documents and Settings\username\Local Settings\Temp
2. The value of the %TMP% environment variable	C:\Windows\Temp

Table 7. Windows Temporary Cache Directory Location

UNIX location	**Example**
1. The default tmp directory	e.g. /tmp/ClearQuest/EmailPlus/Cache

Table 8. UNIX Temporary Cache Directory Location

9.2 Define your own ClearQuest Client Temporary Cache Directory

The location of the temporary cache directory can be customized on Windows, Linux, and UNIX platforms by defining the system environment variable EMP_TEMPDIR. This environment variable should be set to an absolute file path of your choice. Your ClearQuest client must have read/write access to this directory and have the ability to create files within the directory.

On Linux and UNIX systems which multiple users access, you might need to define one EMP_TEMPDIR and create the directory with write permission for all users of that system. Alternatively, it might be necessary to define EMP_TEMPDIR environment variables and directories for each individual user if you are concerned about file system security issues.

Note: It is also recommended that you use the EMP_TEMPDIR system environment variable on ClearQuest Web servers if you plan to send attachments in EmailPlus notifications.

9.3 ClearQuest Client Temporary Cache Directory Housekeeping

Many of the record attachments are deleted after they have been included in an EmailPlus notification. However, to improve performance, template attachments are cached for the duration of a ClearQuest session and might not get deleted. It is good practice to periodically remove any of the files from the temporary directory so that filesystem capacity issues are prevented. This housekeeping could be done using a cronjob on Linux and Unix systems or a scheduled task on Windows systems.

Be careful when deleting these files because some of the files may still be needed by open ClearQuest sessions. You might consider deleting cached attachment files only if they are more than 24 hours old.

10 Appendix A – Template Meta Tag Substitution Extension Point

```
sub EMP_SubstituteCustomMetaTag {
    my $metaTagParts_ref  = shift; # a reference to an array
    my $entityDefObj      = shift;
    my $isHTMLContent     = shift;
    my $isTest            = shift;

    # This routine provides a custom extension to the Meta Tag substitution processing.
    # In the event that none of the built in meta tags is sufficient for the CQ Admins needs
    # this extension allows admins to add their own meta tags.  You should explore the
    # EXPRESSION meta tag before hand thought since it its envisaged that most needs will
    # be met using that facility.  Bear in mind that you can call functions defined in the
    # schema as well.  It is envisaged that this customisation point would be used where it
    # is desired to be able to test the function and / or switch between HTML output and plain text
    #
    # Parameters:
    # metaTagParts_ref - This is a reference to an array of the decoded parts of the metatag as read from the
    #                    template provided.  The 0 element is the name of the metatag whilst any other
    #                    elements that may be present hold parameters decoded from the original input.
    #                    The original input is split on "::" Boundaries.  The complete original tag can
    #                    be reconstituted with this statement if desired
    #                    $origMetaTag = join("::", @{$metaTagParts_ref});
    #  entityDefObj    - This is the ClearQuest Entity Def object for the record type to which the template pertains
    #  isHTMLContent   - This is a boolean flag that when set to 1 means you can render the result as properly formed
    #                    HTML if desired.  Otherwise whatever you return will be treated as plain text.
    #  isTest          - This is a boolean flag and when set to 1 indicates that you should return any Error that the expanding the Tag may
    #                    generate AND NOT the value.  Returning a null string in isTest mode means there were no errors
    #                    This feature is used to validate the usage of a tag.

    EMP_DebugOut("START - Add custom meta tags");
    my $errorTags = "";     # Put any error text in this variable, this will be displayed in the Debug window on the EmailPlusTemplate
    my $fieldValue = "";    # Put the result of the MetaTag expansion in this variable, this value will be included in the EmailPlus notification

    # Process the custom tags

    my $customTag = lc($metaTagParts_ref->[0]);

    # Add your processing here
    # ========================
    # Note that the Meta Tag is converted to lower case so all of your literal values should also be in lower case
    # this handles the situation when the template rule writer uses mixed or upper case for the tag name

    if ($customTag eq "nameoftemplate") {
        # This is an example of a custom metatag that will return the unique name of the EmailPlusTemplate
        # In the EmailPlusTemplate you would use the case insensitive metatag #@NAMEOFTEMPLATE@#
```

```
                if ($session->HasValue("EMP::CONTEXT")) {
                my $context = $session->GetNameValue("EMP::CONTEXT");
                        my($ruleName,$templateName) = split(/\|\|/,$context);
                        $fieldValue = $templateName;
                        # The unique TemplateName consists of 3 parts : RecordType TemplateKind TemplateName
                }
                else {
                    # If the EMP::CONTEXT session variable is not defined then return an error message as the result
                    my $errmsg = 'TemplateName could not be established using the MetaTag: '.$metaTagParts_ref->[0];

            # Use the function EMP_MakeEmbeddedErrorMessage to format the error if the message body format is HTML
                        $fieldValue = EMP_MakeEmbeddedErrorMessage($errmsg,$isHTMLContent);
                }
    }
    # ================================================
    # This dummy is for test only and can be removed START->
    elsif ($customTag eq "dummy") {
        $fieldValue = "Customized test tag ".localtime();
    }
    # <- END This dummy is for test only and can be removed
    # ================================================
    else {
        # Catches Undefined tags
        $errorTags .= "ERROR: Undefined MetaTag: ".$metaTagParts_ref->[0]."\n";
    }

    # Return either value or error string dependent on mode
    my $result = "";    # The return value
    if ($isTest) {
        # Any errors will be displayed in the Debug window on the EmailPlusTemplate record
        $result = $errorTags;
    }
    else {
        # This is what will be inserted into the EmailPlus notification message
        $result = $fieldValue;
    }

    EMP_DebugOut("END - result \= \<$result\>");
    return $result;
}
```

11 Appendix B – Regular Expressions Meta Characters

Table 9 lists some of the standard escape characters that can be used in PERL Regular Expressions. These are all supported in the EmailPlusRule and EmailPlusTemplate records.

Meta Character	**Description**
\	Marks the next character as either a special character or a literal. For example, "n" matches the character "n". "\n" matches a newline character. The sequence "\\" matches "\" and "\(" matches "(".
^	Matches the beginning of input.
$	Matches the end of input.
*	Matches the preceding character zero or more times. For example, "zo*" matches either "z" or "zoo".
+	Matches the preceding character one or more times. For example, "zo+" matches "zoo" but not "z".
?	Matches the preceding character zero or one time. For example, "a?ve?" matches the "ve" in "never".
.	Matches any single character except a newline character
(pattern)	Matches pattern and remembers the match. The matched substring can be retrieved from the resulting Matches collection, using Item [0]...[n]. To match parentheses characters ( ), use "\(" or "\)".
x|y	Matches either x or y. For example, "z|wood" matches "z" or "wood". "(z|w)oo" matches "zoo" or "wood".
	{n} n is a non-negative integer. Matches exactly n times. For example, "o{2}" does not match the "o" in "Bob," but matches the first two o's in "foooood".
{n,}	n is a non-negative integer. Matches at least n times. For example, "o{2,}" does not match the "o" in "Bob" and matches all the o's in "foooood." "o{1,}" is equivalent to "o+". "o{0,}" is equivalent to "o*".
{n,m}	m and n are non-negative integers. Matches at least n and at most m times. For example, "o{1,3}" matches the first three o's in "fooooood." "o{0,1}" is equivalent to "o?".

[xyz]	A character set. Matches any one of the enclosed characters. For example, "[abc]" matches the "a" in "plain".
[^xyz]	A negative character set. Matches any character not enclosed. For example, "[^abc]" matches the "p" in "plain".
[a-z]	A range of characters. Matches any character in the specified range. For example, "[a-z]" matches any lowercase alphabetic character in the range "a" through "z".
[^m-z]	A negative range characters. Matches any character not in the specified range. For example, "[m-z]" matches any character not in the range "m" through "z".
\A	Match only at beginning of a string
\b	Matches a word boundary, that is, the position between a word and a space. For example, "er\b" matches the "er" in "never" but not the "er" in "verb".
\B	Matches a nonword boundary. "ea*r\B" matches the "ear" in "never early".
\d	Matches a digit character.
\D	Matches a nondigit character.
\f	Matches a form-feed character.
\n	Matches a newline character.
\r	Matches a carriage return character.
\s	Matches any white space including space, tab, form-feed, etc.
\S	Matches any nonwhite space character.
\t	Matches a tab character.
\v	Matches a vertical tab character.
\w	Matches any word character including underscore. Equivalent to "[A-Za-z0-9_]".
\W	Matches any nonword character. Equivalent to "[^A-Za-z0-9_]".
\z	Match only at the end of a string.

\Z	Match only at the end of a string, or before newline at the end.

Table 9. PERL Regular Expression Meta Characters

For more information regarding PERL Regular Expressions, see the PERL documentation. Reference: http://perldoc.perl.org/perlre.html

12 Appendix C – Features-Mail Transport Matrix

This appendix documents what features are available with each of the different mail transport types. The custom variant is blank because what it supports depends on your own implementation.

	Mail Transport Types[4][5]					
Feature	**CQMAIL**	**SMTP Direct**	**SMTP Relay**[6]	**PostOffice Relay**		**Custom**
				with CQMail	**with CDO**	
Templates	**Y**	**Y**	**Y**	**Y**	**Y**	
Email Rules Simple	**Y**	**Y**	**Y**	**Y**	**Y**	
Email Rules Advanced	**Y**	**Y**	**Y**	**Y**	**Y**	
HTML Content	**N**	**Y**	**Y**	**N**	**Y**	
Mail Headers	**N**	**Y**	**Y**	**N**	**Y**	
Embedded Graphics	**N**	**Y**	**Y**	**N**	**Y**	
Attachments	**N**	**Y**	**Y**	**N**	**Y**	
Mail Relay	**Y/N**[7]	**Y**	**Y**	**Y**	**Y**	

[4] In a ClearQuest multisite environment you can have different transport types at different sites. However, this is harder to manage and you might need to replicate rules that use different templates depending on the site the rule executes.

[5] For all of the 'N' in this table, EmailPlus does not stop you from trying to use these features; however, the features will not work. HTML for example will come out as raw HTML, attachments and graphics will be ignored.

[6] Assuming the Network Relay mode is enabled in the EMP Relay application, otherwise the CQMail restrictions will apply.

[7] For mail relay, you can select CQMAIL as the delivery mechanism at the receiving end even though the transport type is set to SMTP Relay. If you do this, you will lose some of the features by using the CQMAIL delivery. It is recommended that you choose SMTP Direct for the delivery mechanism.

Subscriptions	**Y**	**Y**	**Y**	**Y**	**Y**	
MultiSite Support	**Y**	**Y**	**Y**	**Y**	**Y**	
UNIX Platform Support	**N**	**N**	**N**	**N**	**N**	**N**
Windows Platform Support	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**

Table 10. EmailPlus Features Supported by Mail Transport Types

13 Appendix D – Configuring Addressing Options

The EmailPlus package allows you to address emails by identifying fields that are a reference or reference list to users. It also searches referenced record types for such references, but a configuration item limits the depth of recursion. By default, EmailPlus searches the following records:

- The current record
- The child records of the current record
- The child records of the child records of the current record

If you want to change this depth, you must define a constant in the 'EMP_Customisable' PERL Global Script.

$MAX_FIELDPATH_RECURSE_DEPTH = 2;

Setting the value to 2 (as shown in the example) sets the depth of recursion to two, i.e. the child record and the child records of the child records of the current record.

Setting the value to 0 limits the search to the current record only.

Setting the value to 3 will search child records up to the third level, in addition to the current record.

There is another option to widen the kind of fields searched for:

$FIELDPATH_INCLUDE_STRINGS = 1;

This option makes EmailPlus include all string fields discovered in records at any depth of recursion. It is up to you that the selection includes only fields that contain lists of valid ClearQuest User IDs that are specified one to a line or are separated by commas. The results are otherwise undefined.

These constants should be defined in the 'EMP_Customisable' PERL Global Script with GLOBAL SCOPE, that is, not inside a function or subroutine. See the code extract below for an example.

```
# Start of Global Script EMP_Customisable
# The routines in the module are customisable.

$MAX_FIELDPATH_RECURSE_DEPTH = 2;
$FIELDPATH_INCLUDE_STRINGS = 1;
```

```
sub EMP_SubstituteCustomMetaTag {
  my $metaTagParts_ref = shift; # a reference to an array
  my $entityDefObj     = shift;
  my $isHTMLContent    = shift;
  my $isTest           = shift;
….
```

If these constants are not defined, the package adopts the following default behavior:

- Using only the User fields of type *Reference To Users* to determine email addresses
- Searching to a depth of recursion equal to 2 to determine the fields that can be used to determine the email addresses

14 Appendix E – Example EmailPlus Template Expressions

Included below are some examples of valid expressions that can be used in EmailPlusTemplate records to reformat information retrieved from ClearQuest or to enhance the content of the email notifications:

- This expression adds HTML code to change the color of the text depending on the value of the priority field:

```
#@EXPRESSION::if ( #?Priority?# eq "1 - High" ) { "<FONT COLOR=\"red\">"; } elsif ( #?Priority?# eq "2 - Medium" ) { "<FONT
COLOR=\"orange\">"; } elsif ( #?Priority?# eq "3 - Low" ) { "<FONT COLOR=\"yellow\">"; } @#Priority: #?Priority?# </FONT>
```

  For example, if the priority has a value of 1, the text is displayed in red, i.e. Priority: 1-High.

- This expression prints all of the record IDs for the related records on the same line separated by spaces. If there are no related records associated with the record, nothing is printed:

```
#@EXPRESSION::if ( scalar( @{#?RelatedRecords?#} ) ) { "Related Records: ".join(" ",@{#?RelatedRecords?#}); } else { ""; }@#
```

  For example, Related Records: mydb00001201 mydb00001543 mydb00002175

- This expression prints the current value of the "State" field and if, it has changed, also the old value of the "State" field:

```
State: #?State?# #@EXPRESSION::if (#?State?# ne #%State%#) { "(Old Value: ". #%State%# .")"; }@#
```

  For example, State: Activated (Old Value: Submitted)

- It should be noted that although field meta tags can be used in EmailPlus expressions, on some occasions expressions may not evaluate correctly – specifically if a field value contains an odd number of double or single quotes. To work around this issue you can use the built in EmailPlus functions that are documented in section 4.7 of this EmailPlus Administation Guide to resolve field values instead of using the field meta tags. In addition to this you should use the PERL function quotemeta to escape any non alphanumeric characters in a field value. This will allow you to perform effective field value comparisons in EmailPlus expressions. The example below compares the field value of the Description field with the original field value of the Description field. If they different then a message will be included in the EmailPlus nofication that states the Description field has changed. :

```
#@EXPRESSION::if ( quotemeta(Gfv("Description")) ne quotemeta(Gfov("Description") ) { "The Description field has changed:
".Gfv("Description") ; }@#
```

- If you want to add details about record attachments but do not want to add attachments, you can include this expression :

```
#@EXPRESSION::
  # Get a list of the attachment fields in this record type...
  my($AttachmentFields) = $entity->GetAttachmentFields();
```

```
  # Tell how many attachment fields there are and show their
  # names...
  $M = $M . "This entity contains " . $AttachmentFields->Count() .
      " attachment field(s)\n";
  for ($A = 0; $A < $AttachmentFields->Count(); $A++)
  {
    $M = $M . "    " . ($AttachmentFields->Item($A) )->GetFieldName() . "\n";
  }
  $M .= "\n";

  # Iterate over the attachment fields; for each one, list the
  # attachments it contains in the current record...
  for (my($AF) = 0; $AF < $AttachmentFields->Count(); $AF++) {
    my ($AttachmentField) = $AttachmentFields->Item($AF);

    $M = $M ."Attachment field '"
          . $AttachmentField->GetFieldName().
           "' contains:\n";

       # Iterate over the attachments in this field...
       my($Attachments) = $AttachmentField->GetAttachments();

    for (my($A) = 0; $A < $Attachments->Count(); $A++) {
      my($Attachment) = $Attachments->Item($A);

      # Report info about this attachment...
      $M = $M .
      "    Filename='" . $Attachment->GetFileName() . "'" .
      " FileSize=" . $Attachment->GetFileSize() .
      " Description='" . $Attachment->GetDescription() . "'" .
      "\n";
    }
    $M = $M ."Total attachments: ". $Attachments->Count() ."\n\n";
  }
  # Display the results...
  $M;
@#
```

Using this expression adds a report to the notification like this:

This entity contains 1 attachment field(s)

Attachments

Attachment field 'Attachments' contains:

Filename='word.doc' FileSize=24064 Description='A Sample Word Doument'

Filename='jpg.jpg' FileSize=4822 Description='A Sample JPG Picture File'

Filename='Text.txt' FileSize=27 Description='A Sample Text File'

Total attachments: 3

15 Appendix F – Example EmailPlusRule Advanced Rules

This section includes examples of some expressions that you can use in the EmailPlusRule Advanced Rules to define or enhance your rule conditions:

- This expression will return TRUE if the value of the SubmitDate field is within the last 24 hours.

  ```
  StoDT( Gfv( "SubmitDate" ) ) > ( time() - ( 24*60*60 ) )
  ```

  Function Gfv() retrieves the value of SubmitDate in UTC Date/Time format (for example 2009-08-16 09:17:54).

  StoDT converts the UTC Date/Time into an elapsed time in seconds (i.e. 1250414274).

  Time() returns the current elapsed time (for example 1250439474).

  This elapsed time is reduced by 24 hours represented in seconds (24*60*60).

  If the time now is 7 hours later than when the record was submitted:

  StoDT(("2009-08-16 09:17:54") > (1250439474 – 86400)) evaluates to

  (1250414274 > 1250353074) which evaluates to 1 (TRUE).

- This expression will return TRUE if the value of the Project.Name field contains the case insensitive word "test" :

  ```
  ( Gfv( "Project.Name" ) =~ /test/i )
  ```

  For example, if the name of the project associated with the modified record is "Project Omega" then:

  ("Project Omega" =~ /test/i) evaluates to 0 (FALSE).

  For example, if the name of the project associated with the modified record is "My Test Project" then:

  ("My Test Project" =~ /test/i) evaluates to 1 (TRUE).

16 Appendix G – Debugging EmailPlus

If you need to debug the EmailPlus operations, enable debugging as follows:

- Define a system environment variable called EMP_DEBUGOUT_LEVEL and set the value to one of the following integer values:

  1 - List all EmailPlus function calls and the parameters passed to the functions

  2 - List all EmailPlus function calls, the parameters passed to the functions and the return values

  3 - Full Debugging – This can generate a lot of output

- If you are debugging on Linux or UNIX systems, you must also set the value of the EMP_DEBUGOUT_FILE system environment variable to the name of a file that ClearQuest can access and write to. You must not include any spaces in the full path of the name of the debug file; example, EMP_DEBUGOUT_FILE=/tmp/EmailPlusDebug.log.

To view the debug output on Windows systems, run the Windows Debugger (dbwin32.exe). The debug output generated by EmailPlus is prefixed by:

```
#EMP_DEBUGOUT# <DATE> : <FUNCTION NAME>: <DEBUG MESSAGE>
```

For example, when EMP_DEBUGOUT_LEVEL is set to 2, the following output might be generated:

```
3712: #EMP_DEBUGOUT# 2009-10-23 11:45:21 :EMP_isEmailPlusActive: START
3712: #EMP_DEBUGOUT# 2009-10-23 11:45:21 :EMP_CacheConfig: START - Cache EmailPlusConfig
3712: #EMP_DEBUGOUT# 2009-10-23 11:45:21 :EMP_CacheConfig: END
3712: #EMP_DEBUGOUT# 2009-10-23 11:45:21 :EMP_isEmailPlusActive: EMP::ConfigCache::isEmailPlusActive = <YES>
3712: #EMP_DEBUGOUT# 2009-10-23 11:45:21 :EMP_isEmailPlusActive: END  - isEmailPlusActive=<1>
```

To view the debug output on Linux or UNIX systems, view the contents of the file that you defined in the EMP_DEBUGOUT_FILE system environment variable.

Enabling debugging for EmailPlus impacts the performance of ClearQuest. It should be enabled only when you need to debug something and then disabled when you complete debugging. Also, you must define or modify the system environment variable before starting the ClearQuest client. Any changes made to the system environment variable while the ClearQuest client is running is not recognized until the ClearQuest client is restarted.

17 Appendix H – Reformatting DATE_TIME fields for use in EmailPlus notifications

A new feature in EmailPlus 2.1 is the addition of a function that helps to reformat date strings. This function utilizes the Date::Format PERL module included with CQPerl to reformat a DATE_TIME string from ClearQuest (YYYY-MM-DD hh:mm:ss) to a format of your choice.

The function is called EMP_ReFormatDateString and it accepts three parameters:

$datestring – A date string in the format "YYYY-MM-DD hh:mm:ss"

$template – A template defined using the LC_TIME categories of the program's locale e.g. "%d-%m-%Y %H:%M:%S"

$timezone – An optional parameter to define the timezone. If no timezone is defined, it will default to the timezone of the host that sends the notification.

```
#@EXPRESSION::EMP_ReFormatDateString( Gfv("DueDate"), "%d-%m-%Y %H:%M:%S", "UTC" );@#
```

The following characters can be used in the templates:

```
%%      PERCENT
%a      day of the week abbr
%A      day of the week
%b      month abbr
%B      month
%c      MM/DD/YY HH:MM:SS
%C      ctime format: Sat Nov 19 21:05:57 1994
%d      numeric day of the month, with leading zeros (eg 01..31)
%e      like %d, but a leading zero is replaced by a space (eg  1..32)
%D      MM/DD/YY
%G      GPS week number (weeks since January 6, 1980)
%h      month abbr
%H      hour, 24 hour clock, leading 0's)
%I      hour, 12 hour clock, leading 0's)
%j      day of the year
%k      hour
```

```
%l      hour, 12 hour clock
%L      month number, starting with 1
%m      month number, starting with 01
%M      minute, leading 0's
%n      NEWLINE
%o      ornate day of month -- "1st", "2nd", "25th", etc.
%p      AM or PM
%P      am or pm (Note that the case returned by %p and %P is reversed.)
%q      Quarter number, starting with 1
%r      time format: 09:05:57 PM
%R      time format: 21:05
%s      seconds since the Epoch, UCT
%S      seconds, leading 0's
%t      TAB
%T      time format: 21:05:57
%U      week number, Sunday as first day of week
%w      day of the week, numerically, Sunday == 0
%W      week number, Monday as first day of week
%x      date format: 11/19/94
%X      time format: 21:05:57
%y      year (2 digits)
%Y      year (4 digits)
%Z      timezone in ascii. eg: PST
%z      timezone in format -/+0000
```

The expression below generates the date in the ctime format :

```
#@EXPRESSION::EMP_ReFormatDateString( Gfv("DueDate"), "%C", "UTC" );@#
```

An example of the date formatting for the above expression would be: Thu Oct 15 08:16:19 2010

If the value of the date string passed to this function is empty, an empty string is returned.

References: http://search.cpan.org/~gbarr/TimeDate-1.19/lib/Date/Parse.pm

http://search.cpan.org/~gbarr/TimeDate-1.19/lib/Date/Format.pm

18 Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software
IBM Corporation
5 Technology Park Drive
Westford, MA 01886
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

18.1 Trademark acknowledgments

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both. UNIX is a registered trademark of The Open Group in the United States and other countries. Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at www.ibm.com/legal/copytrade.shtml.