

# IBM Rational Software Development Conference 2006



Software.  
**IN CONCERT.**

IBM Rational Software  
Development Conference 2006

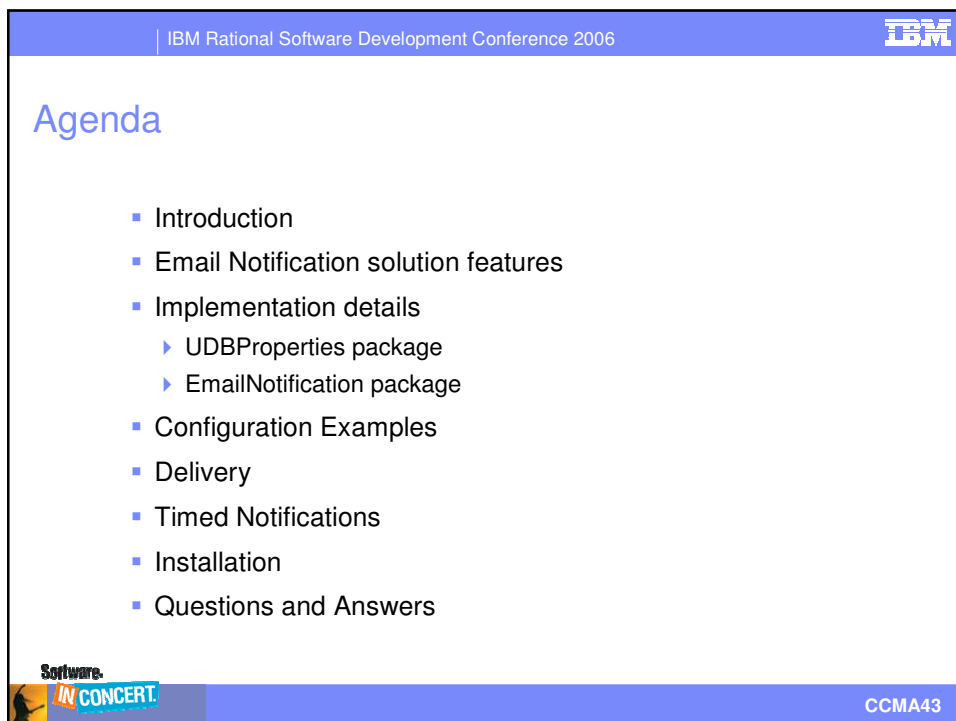
IBM

Extending IBM(R) Rational(R) ClearQuest(R)  
Email Notifications

*Pavel Dubovitskiy*  
*pdubovitsky@yahoo.com*

Rational. software

CCMA43



IBM Rational Software Development Conference 2006

IBM

## Agenda

- Introduction
- Email Notification solution features
- Implementation details
  - ▶ UDBProperties package
  - ▶ EmailNotification package
- Configuration Examples
- Delivery
- Timed Notifications
- Installation
- Questions and Answers

Software.  
**IN CONCERT.**

CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Email Notification Package Features

- Simple installation and support
- No additional software or tools need to be installed on clients
- Centralized configuration and management
- Flexible message and notification condition definitions
- Reliable delivery
- Ability to use password authentication with SMTP server
- IBM Rational ClearQuest Multisite considerations
- Ability to create notification rules for state-based and stateless record types
- Timed notifications



CCMA43

IBM Rational Software Development Conference 2006



## Agenda

- Introduction
- Email Notification solution features
- **Implementation details**
  - ▶ UDBProperties package
  - ▶ EmailNotification package
- Configuration Examples
- Delivery
- Timed Notifications
- Installation
- Questions and Answers



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Implementation Details

- Solution has to be a part of ClearQuest schema
- IBM Rational package utility (packageutil) was selected
  - ▶ Simplifies installation of the package and updates
  - ▶ Package updates do not interfere with schema development lifecycle
  - ▶ Allows Perl to be used for implementation and used with Perl and VBScript based schemas



CCMA43

IBM Rational Software Development Conference 2006



## Email Notification Package

- Configuration items are stored in ClearQuest database
  - ▶ UDBProperties package is used
  - ▶ No configuration is required on the client
- Email\_Notification\_Rule record type is used to configure email notifications
  - ▶ Ability to use parameterized and expression fields in Email\_Notification\_Rules:
    - record fields,
    - session-wide variables and persistent properties
    - SQL queries
    - package built-in, user-defined and standard perl functions



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Email Notification Package

- en\_email\_message record type implements a message queue
  - ▶ Reliable delivery and error recovery
  - ▶ Debugging
    - Ability to verify that a message was sent
    - Saved error message if it was not sent
  - ▶ Audit trail
- SMTP delivery
  - ▶ ESMTP protocol realization is a part of CQPerl distribution
  - ▶ AUTH LOGIN extension is implemented in the package
  - ▶ Other delivery mechanisms can be added by customer



CCMA43

IBM Rational Software Development Conference 2006



## Agenda

- Introduction
- Email Notification solution features
- Implementation details
  - ▶ **UDBProperties package**
  - ▶ EmailNotification package
- Configuration Examples
- Delivery
- Timed Notifications
- Installation
- Questions and Answers



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Centralized Configuration

- Email Notification settings are ClearQuest user database properties
  - ▶ Functionality is grouped into UDBProperties package that is installed automatically when you install EmailNotification package
  - ▶ The settings are stored in dbs\_property table that is a part of the package
  - ▶ Settings are cached in session-wide variables to improve performance
  - ▶ Refreshed every 10 minutes



CCMA43

IBM Rational Software Development Conference 2006



## Centralized Configuration

- The following public methods provided to customers to access database properties:
  - ▶ GetProperty - to access persistent and session-wide properties
  - ▶ SetProperty - to set session-wide property
  - ▶ IsProperty - to check whether the specified property exists
  - ▶ DebugPrint(debug\_level, "message" [, "message1 ..."])  
to print debug message with OutputDebugString API call if current debug level set in "Debug" property is equal or higher than debug\_level



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Centralized Configuration - Security

- Visibility is restricted using security context feature
  - ▶ Based on `udb_visibility` - another record type that is included into `UDBProperties` package
  - ▶ Initially, no security context record is attached to a property. Super User or Security administrators have rights to view and modify the records
  - ▶ You can define your own visibility groups and assign them to properties
  - ▶ If '**Property\_Default\_Security\_Group**' property is specified – new property will be initialized with this visibility group on submit action



CCMA43

IBM Rational Software Development Conference 2006



## Centralized Configuration - Security

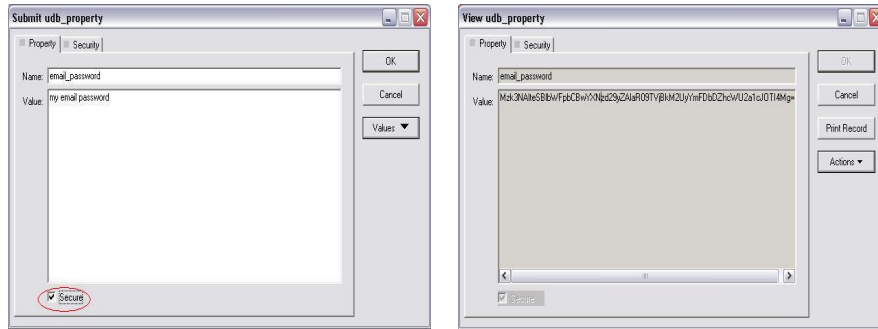
- Data encryption is used for 'Secure' properties
  - ▶ Simple encryption mechanism was provided by Reto Hersiczky and implemented in Perl
  - ▶ The algorithm ID is stored along with encrypted value
  - ▶ Stronger algorithms can be included in the package later.
  - ▶ Customers can modify the package and add their own algorithms
  - ▶ `GetProperty` methods returns decrypted value in hook
  - ▶ ClearQuest Export and Import utility operate with encrypted values
    - Export: the field encrypted value is stored in exported text file
    - Import: the field encrypted value is imported from the text file



CCMA43

# IBM Rational Software Development Conference 2006

## Centralized Configuration - Security



## Email Notification Delivery Properties

Name	Requiredness	Function
email_host		SMTP server domain name or address
email_login	optional	Email account login name that is used for SMTP server authentication. Turns on AUTH LOGIN authentication mechanism with SMTP server
email_password	optional	Email account password (it is better to be a 'secure' property)
email_address	optional	'From' email address for notifications. Current User is used when the property does not exist
email_reply_to	optional	Reply-To address, if different from 'From' address
email_enabled		Turns on email notification for the database when exists and is set to non-zero value

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Agenda

- Introduction
- Email Notification solution features
- Implementation details
  - UDBProperties package
  - **EmailNotification package**
- Configuration Examples
- Delivery
- Timed Notifications
- Installation
- Questions and Answers



CCMA43

IBM Rational Software Development Conference 2006



## Email Notification Rule Security

- Security Based on Security Context feature
  - Based on `udb_visibility` - a record type that is included into UDBProperties package
  - Initially, no security context record is attached to a notification rule. Super User or Security administrators have rights to view and modify the records
  - You can define your own visibility groups and assign them to notification rules
  - If '**Notification\_Default\_Security\_Group**' property is specified – new notification rule will be initialized with this visibility group on submit action



CCMA43



# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Email\_Notification\_Rule static fields

Field Name	Requiredness	Function
Name	Mandatory	The email rule unique name
Entity_Def	Mandatory	Record type on which to base the email rule (Defect, Enhancement Request, etc)
Field_Change	Optional	Fields to check for change Fields changed in action of field initialization hooks are not counted
Actions	Optional	Action controls for the email rule
Active	Optional	You can temporarily disable the rule by clearing 'Active' checkbox



CCMA43

IBM Rational Software Development Conference 2006



## Email Notification Rule Static Fields

- If you change the names of fields, record types or actions in a schema that has existing email rules, you must update email rules to reflect the change
- 'AND' operator is used for **Field\_Change** and **Actions** condition
  - ▶ It is not critical, because any desirable condition can be implemented using parameterized **Condition** field



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Email Notification Rule Static Fields

- Action selections are not limited to existing actions for the record type.
  - ▶ An unique value can be set to the **PseudoAction** session-wide variable before notification hook execution and unset it after.
  - ▶ The variable can be used instead of action name for **Actions** control evaluation
    - To use **Actions** control with RECORD\_SCRIPT\_ALIAS actions (when \$entity->GetActionName() API call fails)
    - To use **Actions** control with timed notification script



CCMA43

IBM Rational Software Development Conference 2006



## Email\_Notification\_Rule Parameterized Fields

Field Name	Type	Function
Condition	Expression	Logical condition. Notification is sent when evaluation result is 'True'
To CC BCC	Text	Destination addresses, one address per line, empty lines are ignored
Priority	Expression	< 0     Low == 0    Normal > 0     High
Header (Header Add-in)	Text	Additional header information charset, content-type, x-confirm-reading-to, etc in RFCs compliant format
Subject	Text	Message subject
Body	Text	Message body



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Parameterized Fields Syntax

- Fields Evaluation
  - ▶ In **Text** mode the following items are expanded:
    - Variables
    - SQL statements
    - Functions
    - Characters escaped with '\'
  - ▶ In **Expression** mode, the following items are expanded in addition to the Text mode:
    - Numeric constants, single and double quoted strings
    - Operators, including regular expression operation



CCMA43

IBM Rational Software Development Conference 2006



## Variables

- Syntax  
 **$\$$ variable\_name** or  **$\$\{$ variable\_name $\}$**
- ▶ Variable name has to start with alphabetic character, can include alpha-numeric characters, underscores and dots '.'
- Variable name is resolved in the following order:
  1. Database property variable\_name.
  2. Predefined variable name.
  3. ClearQuest record field name on which the rule is based
- Notes:
  - ▶ property names and predefined variable names are case sensitive
  - ▶ ClearQuest record field names are case insensitive
  - ▶ Date\_Time variables are represented in 'yyyy-mm-dd hh:mm:ss' format



CCMA43

# IBM Rational Software Development Conference 2006

## Predefined Variables

Action	Current action name Returns a value of <b>PseudoAction</b> session-wide variable or result of \$entity->GetActionName() API call if <b>PseudoAction</b> variable is not set
EntityDefName	Current entity definition name For example: Defect, EnhancementRequest, etc
SiteName	Site name in multisited environment

## Variables

- Examples
    - ▶ Database property property\_name  
*`\${property\_name}`*
    - ▶ Record field value  
*`\${owner.email}`*
    - ▶ Record field value and predefined variable  
*Subject: `\${EntityDefName} \${ID} has been \${state}`*
- Result  
Subject: Defect SAMPL00000041 has been Closed

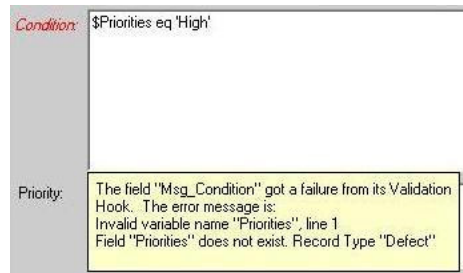
# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Variables

- Avoid Errors
  - ▶ Variable name is validated when you create a notification rule



- ▶ Non-zero value of **EN\_Disable\_Variable\_Name\_Validation** property suppress the validation



CCMA43

IBM Rational Software Development Conference 2006



## SQL Statements

- Syntax
  - SQL(<statement>)**
    - ▶ No space is allowed between SQL keyword and open bracket
    - ▶ Statement can be single-quoted non-expandable string, double-quoted string with variables, functions, SQL queries, etc
    - ▶ Statement is not limited to a single line
    - ▶ Only one column is retrieved from result set
    - ▶ May have more than one row in the result (Multiline string)
    - ▶ Statement is executed at run-time using BuildSQLQuery API call and has all limitations related to that call



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## SQL Statements

- Examples

- ▶ A hard way to get \$owner.email value for Defect record type

```
SQL( "select T1.email  
      from users T1  
      where T1.login_name = '$owner' ")
```

- ▶ Get email addresses of all users that are listed in Defect's history

```
SQL( "select distinct T2.email  
      from defect T1, users T2, history T3  
      where T1.dbid = T3.entity_dbid  
            and T2.login_name = T3.user_name  
            and (T1.dbid=$dbid) and (T1.dbid <> 0) ")
```



CCMA43

IBM Rational Software Development Conference 2006



## Functions

- Syntax

**function\_name( [ parameter [, parameter1 ... ] ] )**

- ▶ Function name has to start with alphabetic character, can include alpha-numeric characters and underscores
- ▶ No space is allowed between function name and open bracket
- ▶ Optional function parameters is a list of comma separated expressions, which can be numeric constants, single and double quoted strings, variables, SQL statements, functions or other expressions that must be grouped in round brackets ().



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Functions

- Function name is resolved in the following order
  - ▶ User-defined functions
  - ▶ Predefined functions
  - ▶ The function name is expected to be a Perl function, including functions defined in a schema global scripts section and perl standard functions
  
- Note
  - ▶ There is no reliable way to validate Perl function names when you create a notification rule, but the package provides you a way to debug notification rules including syntax check and run-time error messages



CCMA43

IBM Rational Software Development Conference 2006



## User-defined functions

- Function body is a CQPerl code
- ClearQuest CQPerl API is available
- \$entity and \$session local variables are available within a function scope
- Function name is a database property name
- Function body is the database property value
- Function parameters are passed as @\_ array to the function
- Return operator should be used at the end
- Scalar (string, number, etc) is expected as a return value



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## User-defined functions

- The following code is equivalent to the user function definition at the next slide

```
sub RT_GetFieldOriginalValue {
  my ($name) = @_;
  my $value = '';
  eval {
    $value=$entity->GetFieldOriginalValue($name)->GetValue();
  };
  $@ = '';
  return $value;
}
```

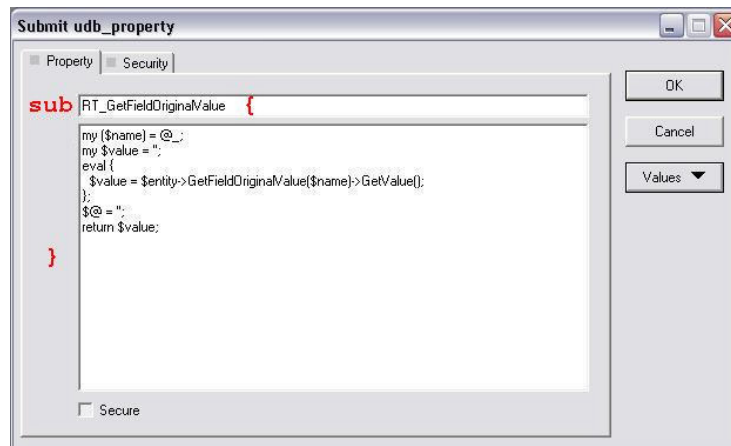


CCMA43

IBM Rational Software Development Conference 2006



## User-defined Functions



Equivalent representation of that function in udb\_property record



CCMA43



# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Built-in Functions

- Conditional functions
  - ▶ **IF( ( <expression> ) , <val1>, <val2> )**
    - **<expression>** Boolean expression
    - **<val1>** returned if <expression> is true
    - **<val2>** returned if <expression> is false
  - ▶ Example  
*Defect \$ID has been IF(( \$Action ne 'Modify' ), lc(\$state), 'updated')*
  - ▶ Result  
Subject: Defect SAMPL00000044 has been assigned  
Subject: Defect SAMPL00000044 has been updated



CCMA43

IBM Rational Software Development Conference 2006



## Built-in Functions

- Date/Time functions
  - ▶ **GetDate( [ '<offset><modifier>' ] )**
  - ▶ **GetDateTime( [ '<offset><modifier>' ] )**
  - ▶ **DateTimeToDate( <date\_time> )**
  - ▶ **DateTimeDiff(<date\_time1>, <date\_time2>, '<modifier>')**
    - Returns <date\_time1> - <date\_time2> difference
  - ▶ **DateTimePlusOffset( <date\_time> , '<offset><modifier>' )**
  - ▶ **DateTimeFromTimestamp(<timestamp>)**
    - **<date\_time>** date/time in 'yyyy-mm-dd [hh:mm:ss]' format
    - **<offset>** numeric offset
    - **<timestamp>** number of seconds since 01/01/1970  
return value of time() function



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Built-in functions

- Date/Time functions
  - ▶ <modifier> values
    - none/undef       seconds
    - s                   seconds
    - m                   minutes
    - h                   hours
    - d                   days
  - Examples
    - ▶ *GetDateTime()*                   now
    - ▶ *GetDateTime( '+10m'* )       10 minutes from now
    - ▶ *GetDate( '-1d'* )               yesterday
    - ▶ *DateTimeDiff(\$d1, \$d2, 'h'* )   difference (\$d1-\$d2) in hours



CCMA43

IBM Rational Software Development Conference 2006



## Strings

- Double quoted strings
  - ▶ The following items are expanded
    - Variables
    - SQL statements
    - Functions
  - ▶ Escape characters
    - \n new line
    - \r carriage return
    - \t tabulation
    - \f form feed
    - \b backspace
    - \a bell
  - ▶ Backslash '\ ' at the end of the line escapes new line symbol
  - ▶ Backslash '\ ' in front of any other characters escapes the character



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Strings

- Single quoted strings
  - ▶ Single quoted strings are not expanded
  - ▶ Escape characters
    - \\ backslash
    - \ ' single quote



CCMA43

IBM Rational Software Development Conference 2006



## Operators and Precedence

Operator	Comments
! ~ + -	Unary operators
=~ ~!	Binding operators. Regular expression at the right side can include modifiers (/.../[misx])
* / % x	Multiplicative operators
+ - .	Additive operators
gt ge lt le > >= < <=	Relational operators
eq ne cmp != == <=>	Equality operators
&&	C-style logical AND
	C-style logical OR



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Operators

- Note
  - ▶ There has to be a space between division (/) operator and operand
- Examples
  - ▶ Check if **Due\_Date** is not null and less then one day away:  
`$Due_Date ne '' &&  
DateTimeDiff( $Due_Date, GetDateTime('+1d') ) < 0`
  - ▶ Check for a word in description field  
`$Description =~ /critical/im`



CCMA43

IBM Rational Software Development Conference 2006



## Agenda

- Introduction
- Email Notification solution features
- Implementation details
  - ▶ UDBProperties package
  - ▶ EmailNotification package
- **Configuration Examples**
- Delivery
- Timed Notifications
- Installation
- Questions and Answers



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Address Fields

- We can use text, variables, SQL and functions that return email addresses in one-address-per-line format.

- ▶ For example:

```
reader@email.net
$owner.email
SQL( "select distinct T2.email
      from   defect T1, users T2, history T3
      where  T1.dbid = T3.entity_dbid
            and T2.login_name = T3.user_name
            and (T1.dbid=$dbid) and (T1.dbid <> 0) ")
RT_GetEmailAddresses()
```

- ▶ Empty lines are ignored
- ▶ Address duplicates are removed



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Record URL

- Record URL imbedded into message body

- ▶ Notification **Body** field

```
...
To view this record click on the following link:
GenerateURL($dbid, $EntityDefName)
...
```

- ▶ GenerateURL user-defined function (udb\_property):

Name: **GenerateURL**

Value:

```
my ($dbid, $enttype) = @_;
my $db = $session->GetSessionDatabase()
        ->GetDatabaseName();
my $url = "http://mywebserver/cqweb/main?command=
GenerateMainFram&sevice=CQ&Schema=<schema_name>&contex
tid=${db}...";
return $url;
```



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Include Defect History

- Record history included into message body
  - ▶ Just include into Email\_Notification\_Rule **Body** field:  
Defect History  
Action Date New State Old State Login Name  

```
SQL("select distinct
      convert (char (10), T4.action_name)
    + convert (char (12), T4.action_timestamp)
    + convert (char (12), T4.new_state)
    + convert (char (12), T4.old_state)
    + convert (char (10), T4.user_name),
T4.action_timestamp
from Defect T1, history T4
where T1.dbid = T4.entity_dbid
and (T1.dbid=$dbid) and (T1.dbid <> 0)
order by T4.action_timestamp ASC ")
```

Note: MS SQL syntax is used in this example



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Include Defect History

**Subject:** Defect SAMPL00000004 has been updated

Defect SAMPL00000004 has been updated.

```
Defect History
Action Date New State Old State Login Name
Submit Jun 15 2000 Submitted no_value engineer
Open Jul 30 2000 Opened Submitted QE
Repair Aug 1 2000 Resolved Opened lead
Validate Aug 14 2000 Closed Resolved engineer
Re_Open Aug 21 2000 Opened Closed lead
Repair Aug 31 2000 Resolved Opened user
Import Apr 2 2006 Resolved Resolved admin
Import Apr 2 2006 Resolved Resolved admin
Modify Apr 2 2006 Resolved Resolved admin
```

Record history is included into message body



CCMA43

# IBM Rational Software Development Conference 2006

## Notification Rule Examples – HTML Message

- Simple notification in HTML format
  - ▶ Specify Content type in the Header Add-in filed  
Content-Type: text/html; charset=us-ascii

- ▶ You can use HTML tags in the message body

```
<HTML> <BODY>
<H2 ALIGN="left">$EntityDefName $ID</H2>
<TABLE ALIGN="left" BORDER="1">
<TR><TD><b>ID</b></TD> <TD>$id </TD></TR>
<TR><TD><b>headline</b></TD> <TD>$Headline</TD></TR>
<TR><TD><b>Priority</b></TD> <TD>$Priority</TD></TR>
<TR><TD><b>Severity</b></TD> <TD>$Severity</TD></TR>
<TR><TD><b>Description</b></TD>
<TD><PRE>$Description</PRE> </TD></TR>
</TABLE> </BODY> </HTML>
```



## Notification Rule Examples - HTML Message

- The rule on the previous slide generates HTML formatted message:

**Subject:** Defect SAMPL00000018 has been opened  
**To:** pdubovitsky@yahoo.com

### Defect SAMPL00000018

<b>ID</b>	SAMPL00000018
<b>headline</b>	Email notification
<b>Priority</b>	1-Resolve Immediately
<b>Severity</b>	2-Major
<b>Description</b>	Email notification needs to be extended



# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Attachments

- Use attachments with email notification
  - ▶ We need to create 2 functions:
    - **RT\_HasAttachments** (check for attachments in current entity)

```
my ($result, $i);
my $att_fields = $entity->GetAttachmentFields();
# check for attachments in all Attachment type fields
for($i = 0; $i < $att_fields->Count(); $i++){
    my $att_field = $att_fields->Item($i);
    if($att_field->GetAttachments()->Count() > 0 ){
        $result = 1; # attachment exists
        last;
    }
}
return $result; # TRUE if current record has attachments
```



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Attachments

- **RT\_InsertAttachments** (add attachments to the message body)

```
require MIME::Base64;
my ($result, $tmp, $attachments, $attachment, $file, $tmpfile,
    $nload, $i, $k, $num);
# get all fields of attachment type
my $att_fields = $entity->GetAttachmentFields();
my $dbid = $entity->GetFieldValue('dbid')->GetValue();
# create unique boundary separator
my $boundary = "Boundary_($dbid)";
# get temporary directory name where to extract attachments
$tmp .= '/' if (($tmp = $ENV{TMP}) ne '' ||
    ($tmp = $ENV{TEMP}) ne '');
$tmp = '/tmp/' if ( $tmp eq '' && $^O ne 'MSWin32' );
# continue on the next slide
```



CCMA43



# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Attachments

```
for($i = 0; $i < $att_fields->Count(); $i++){  
  # for all attachment fields:  
  my $att_field = $att_fields->Item($i);  
  if( ($num= ($attachments =  
    $att_field->GetAttachments()->Count()) > 0 ){  
    for($k=0; $k < $num; $k++){ # for all attachments in the field  
      $attachment = $attachments->Item($k); # get attachment  
      $file = $attachment->GetFileName(); # get the file  
      if( $file =~ /[\\\/]/ ){  
        # if the file has path delimiters - it has not been loaded  
        $tmpfile = $file; # we can use it original location  
        $file =~ s/[\\\/]//; # remove path delimiters  
        $nload = 1; # set 'do not use load' flag  
      }else{  
        # otherwise - create a temporary file name  
        $tmpfile = "$tmp$dbid.$k.txt";  
        $nload = 0; # unset 'do not use load' flag  
      } # continue on the next slide  
    }  
  }  
}
```



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Attachments

```
# load attachment from record into temporary file  
if( $nload || $attachment->Load($tmpfile) ){  
  # open the attachment file  
  open(TMPFILE, $tmpfile)  
    or die "Cannot open file '$tmpfile'\n";  
  # set file handle to binary mode  
  binmode TMPFILE;  
  # unset record delimiter  
  local $/ = undef;  
  # encode content of the file with Base64 encoding  
  my $content = MIME::Base64::encode_base64(<TMPFILE>);  
  # close the file  
  close(TMPFILE);  
  # and delete it, if it was a temporary file  
  unlink($tmpfile) unless $nload;  
} # continue on the next slide
```



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Attachments

```
# add header and encoded attachment to the result
$result .= "
--$boundary
Content-type: application/octet-stream
Content-transfer-encoding: base64
Content-disposition: attachment; filename=$file

$content\n";
}
}
}
return $result; # return encoded attachments
```



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Attachments

### ▪ Create Email Notification Rule record:

#### ▶ Header Add-in field

```
IF ( RT_HasAttachments (), "MIME-Version: 1.0
Content-type: multipart/mixed;
boundary=\"Boundary_\"($dbid)\"")
```

#### ▶ Message Body field

```
IF ( RT_HasAttachments (), "--Boundary_\"($dbid)
Content-type: text/plain; charset=us-ascii")
```

```
Defect $ID
Headline: $Headline
Description: $Description
```

```
RT_InsertAttachments ()
```



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Attachments

- Result is a MIME/multipart message:

- ▶ Header

```
To: some@email.address
Subject: Defect SAMPL00000010 has been updated
MIME-Version: 1.0
Content-type: multipart/mixed; boundary="Boundary_(33554442) "
```

- ▶ Body

```
--Boundary_(33554442)
Content-type: text/plain; charset=us-ascii

Defect SAMPL00000010
...
--Boundary_(33554442)
Content-type: application/octet-stream
Content-transfer-encoding: base64
Content-disposition: attachment; filename=Run_time_error.JPG

/9j/4AAQSkZJRgABAQEAYABgAAD/2wBDAAgGBgcGBQgHBwcJCQgKDBQNDAsLD
```



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples - Attachments

**Subject:** Defect SAMPL00000010 has been updated

Defect SAMPL00000010  
Headline: logout button should be disabled during sale  
Description: logout button should be disabled during sale

### Attachments

#### Files:

Run\_time\_error.JPG (26k)

The multipart message is recognized by email client



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Change Notification

- What has changed in the ticket?  
Notification requirements:
  - ▶ Notification rule should create a message in HTML format
  - ▶ Record is represented as HTML table
  - ▶ The table columns are: Field Name, Field Value and Filed Original Value
  - ▶ When field value is changed during the action, the field name font should be changed to bold
  - ▶ We should be able to configure fields that are included into the table



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Change Notification

- RT\_Record\_as\_HTML\_table function (create HTML table)

```
# define function variables
my ($tbl, $fldname, $func, $curr_val, $orig_val );

# define functions to create HTML tags
sub TBL { return "<TABLE BORDER=\\\"1\\\">\\n${_}[0]</TABLE>\\n"; };
sub B { return "<b>${_}[0]</b>"; };
sub TR { return "<TR>@_</TR>\\n"; };
sub TH { my $width = "WIDTH=\\\"${_}[1]\\\""; if defined(${_}[1]);
         return "<TH $width>${_}[0]</TH>\\n"; };
sub TD { return "<TD>${_}[0]</TD>\\n"; };
sub PRE { return "<PRE>${_}[0]</PRE>"; }; # preserve multiline format
sub stub{ return "${_}[0]"; }; # stub - return original value

# translate HTML special characters
sub ESC { my $s = shift;
         $s =~ s/\\&/\\&amp;/g; $s =~ s/\\\"/\\&quot;/g;
         $s =~ s/\\</\\&lt;/g; $s =~ s/\\>/\\&gt;/g; return $s; }

# continue on the next slide
```



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Change Notification

```
# include the following field types only  
# other field types will be ignored  
my $fld_funcs = {  
    $CQPerlExt::CQ_SHORT_STRING    => \&stub,  
    $CQPerlExt::CQ_INT             => \&stub,  
    $CQPerlExt::CQ_MULTILINE_STRING => \&PRE, # display as multiline field  
    $CQPerlExt::CQ_REFERENCE       => \&stub,  
    $CQPerlExt::CQ_DATE_TIME       => \&stub,  
    $CQPerlExt::CQ_STATE           => \&stub,  
    $CQPerlExt::CQ_STATE_TYPE      => \&stub,  
    $CQPerlExt::CQ_ID              => \&stub,  
    $CQPerlExt::CQ_REFERENCE_LIST  => \&PRE, # display as multiline field  
    $CQPerlExt::CQ_DBID            => \&stub };  
  
# continue on the next slide
```



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Change Notification

```
# create table header  
$tbl = TR(TH('Field Name', '20%'), TH('Field Value','40%'),  
          TH('Field Original Value','40%'));  
  
# for each field name that is in the function parameter list  
foreach $fldname (@_) {  
  
    # if the field type in the list of eligible types  
    if( defined($func = $fld_funcs->{$entity->GetFieldType($fldname)}) ){  
        # get field current value  
        $curr_val = $entity->GetFieldValue($fldname)->GetValue();  
  
        # unset original value variable  
        $orig_val = undef;  
        # try to get the field original value  
        $orig_val = eval {$entity->GetFieldOriginalValue($fldname)->GetValue(); };  
        $@ = ''; # and ignore errors  
  
        # continue on the next slide
```



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Change Notification

```
# create new table row
if( $curr_val ne $orig_val ){
  # if current value <> original value
  # put both values and make field name font bold
  $tbl .= TR(TD(B($fldname)), TD($func->(ESC($curr_val))),
            TD($func->(ESC($orig_val))));
}else{
  # if current value == original value - put current value only
  $tbl .= TR(TD($fldname), TD($func->(ESC($curr_val))), TD());
}
}
}
# return table
return TBL($tbl);
```



CCMA43

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Change Notification

### ▪ Email Notification Rule fields

#### ▶ Header Add-in

```
Content-Type: text/html; charset=us-ascii
```

#### ▶ Body

```
<HTML>
<BODY>
<H2 ALIGN="Center">Defect $ID </H2>
RT_Record_as_HTML_table(
  'ID',      'Headline', 'State',    'Priority',
  'Severity', 'Owner',   'Symptoms', 'Description')
</BODY>
</HTML>
```



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Notification Rule Examples – Change Notification

**Subject:** Defect SAMPL00000010 has been closed

### Defect SAMPL00000010

Field Name	Field Value	Field Original Value
ID	SAMPL00000010	
Headline	logout button should be disabled during sale	
State	Closed	Resolved
Priority	3-Normal Queue	
Severity	4-Minor	3-Average
Owner	engineer	
Symptoms	Missing Feature	Missing Feature Documentation Issue
Description	logout button should be disabled during sale.	logout button should be disabled during sale

Change notification email



CCMA43

IBM Rational Software Development Conference 2006



## Agenda

- Introduction
- Email Notification solution features
- Implementation details
  - ▶ UDBProperties package
  - ▶ EmailNotification package
- Configuration Examples
- **Delivery**
- Timed Notifications
- Installation
- Questions and Answers



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Delivery – Message Fields

- The package introduces en\_email\_message stateless record type to save outgoing email messages for reliable delivery
  - ▶ An en\_email\_message record contains expanded email fields:
    - **From**
    - **Recipients**
    - **Header**
    - **Body**
  - ▶ Users with special rights can update message content
    - SuperUser
    - Members of email administration group specified in EN\_AdminGroup property



CCMA43

IBM Rational Software Development Conference 2006



## Delivery – Control Fields

- Another set of fields in en\_email\_message are control fields
  - ▶ **Record Type**            which email rule was base on
  - ▶ **Rule Name**             the rule name that triggered notification
  - ▶ **Record ID**             ID of record that triggered notification
  - ▶ **Status**                 current status of email message
  - ▶ **Entered By**            user that entered email message
  - ▶ **Entered Date**         Date/Time stamp of the message creation
  - ▶ **Attempt**                delivery attempt number
  - ▶ **Action Date**            last action date
  - ▶ **Error Message**        last error message



CCMA43



# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Delivery – Message Status

- Email Message Status
  - ▶ **Ready**
    - Email message is ready for delivery
  - ▶ **Delivering**
    - Email message delivery has been initiated
  - ▶ **Delivered**
    - Email message successfully delivered
  - ▶ **Error**
    - Delivery error occurred
      - Status=Error and delivery Attempt=1000 means a fatal error.  
For example, a run-time field evaluation error



CCMA43

IBM Rational Software Development Conference 2006



## Delivery – Fatal Error

The screenshot shows a window titled "Main" with a "Message" tab. It contains the following fields:

Record Type:	Defect	Rule Name:	New Defect
Record ID:	SAMPL00000001	Status:	Error
Entered By:	CQ Administrator	Entered Date:	3/26/2006 12:28:06 PM
Attempt:	1000	Action Date:	3/26/2006 12:28:06 PM

Below these fields is an "Error Message:" section with a text area containing the following text:

```
field 'msg_body' eval error:  
Error in function 'MyFunction'  
Undefined subroutine &main::MyFunction called at (eval 8) line 1.
```

Run-time field evaluation error  
Undefined function name in the message body field



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Delivery Modes

- Delivery Mode is set by **EN\_DeliveryMode** property
- The following delivery modes can be used:
  - ▶ **Light**
    - Email messages are stored in the queue only when delivery error occurred.
    - Eligible for delivery saved messages are delivered from the queue on the next action
    - External delivery script can be used to process message queue.
    - Pro: speed and disk space
    - Cons: lack of audit trail



CCMA43

IBM Rational Software Development Conference 2006



## Delivery Modes

- ▶ **Deferred** (Default mode)
  - All email messages created by email notification rule are stored in the queue at the first phase of notification hook
  - Eligible for delivery saved messages are delivered from the queue at the second phase of notification hook
  - External delivery script can be used to process message queue
  - Pro: audit trail
  - Cons: hook execution time



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Delivery Modes

- ▶ **Queue**
  - All email created by email notification rule are stored in the queue
  - No delivery attempt is performed by notification hook
  - External delivery script must be used
  - Pro: no delivery attempted on CQ client, speed
  - Cons: external delivery script is required
- ▶ Queue delivery mode is useful to develop and debug notification rules in a test database
- ▶ Queue delivery mode can be used to implement non-SMTP delivery mechanisms in custom delivery script



CCMA43

IBM Rational Software Development Conference 2006



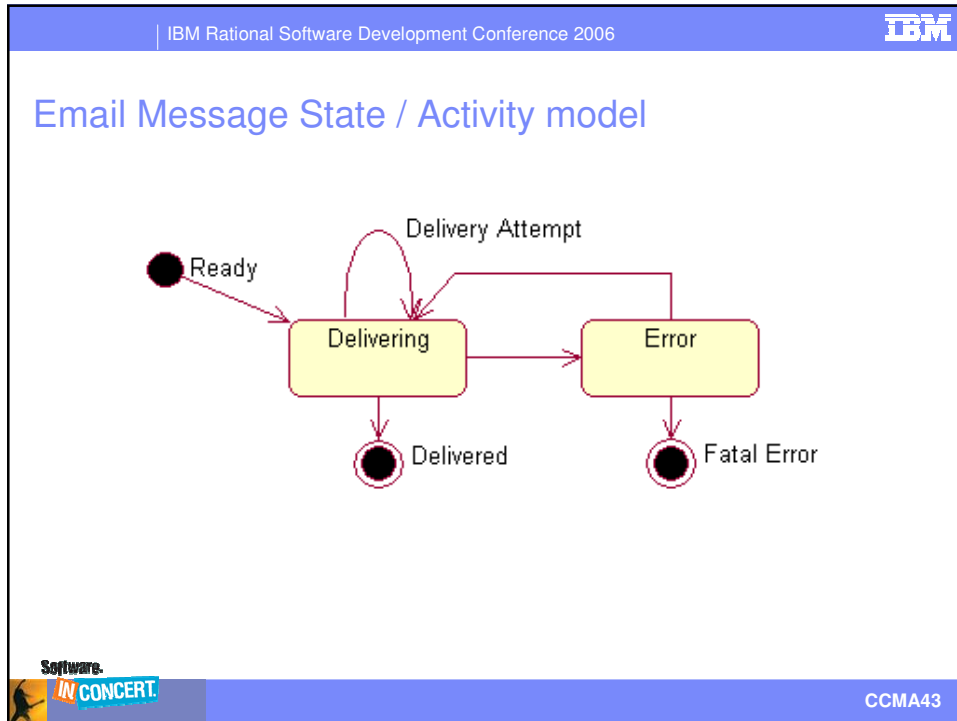
## Message Queue Processing

- Eligible for delivery messages:
  - ▶ Message **Status = Ready**
  - ▶ Message **Status = Delivering** and taking too long for delivery
    - Default timeout is 300 second (5 minutes)
    - Default value can be overwritten by setting **EN\_DeliveryTimeout** property to desirable timeout in seconds
  - ▶ Message **Status = Error** and maximum number of delivery attempts is not exceeded
    - Default number of delivery attempt is 5
    - Default value can be overwritten by setting **EN\_DeliveryAttempt** property



CCMA43

# IBM Rational Software Development Conference 2006



- IBM Rational Software Development Conference 2006
- ## Email Message Actions
- Regular Actions
    - ▶ **Submit**
    - ▶ **Update**
    - ▶ **Remove**
  - Record Script Alias Actions
    - ▶ **Send Message**
      - Re-delivers selected en\_email\_message entity via SMTP  
current status is ignored
    - ▶ **Process Queue**
      - Re-delivers all eligible for delivery messages via SMTP  
Delivery Mode setting is ignored
- Software. **IN CONCERT.** CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Multisite Considerations

- Notification hook, script alias actions and delivery script process en\_email\_message records mastered by local replica only
- Global property values can be overwritten on site basis
  - ▶ **<SiteName>PropertyName** can be set to new value.  
For example, you can specify **<SiteName>email\_host** property to overwrite global **email\_host** property at the SiteName
  - ▶ **<SiteName>PropertyName** can be set to 'EN\_NONE' value to disable global property at the SiteName
  - ▶ Note: SiteName is a value that returned by `$session->GetLocalReplica()->GetFieldValue('Name')->GetValue()`



CCMA43

IBM Rational Software Development Conference 2006



## Agenda

- Introduction
- Email Notification solution features
- Implementation details
  - ▶ UDBProperties package
  - ▶ EmailNotification package
- Configuration Examples
- Delivery
- **Timed Notifications**
- Installation
- Questions and Answers



CCMA43

# IBM Rational Software Development Conference 2006

## Timed Notifications

- It is possible to use email notification rules for Timed Notifications.
  - ▶ External script must be used  
TimedNotification.pl script is bundled with the package.
  - ▶ The script is executed using UNIX cron or Windows scheduler
  - ▶ The script
    - Narrows the record scope for Email Notification Rule Evaluation
    - Sets pseudo-action name that is used to configure email notification rule
    - Invokes Notification hook for each record in the scope
    - Email Notification hook evaluates rule conditions and creates email messages

## Timed Notification Setup

- Notify manager about defects that passed a due date
- Setup a task that runs script daily  
*cqperl TimedNotification.pl cqlogin passwd db dbset <PseudoAction>*
    - ▶ Pseudo-Action is a unique name that is used for notification configuration. For example, **DailyNotification**
  - Enable Defect record type for notification
    - ▶ **TN\_RecordTypes** property contains names of entities eligible for Timed Notification, one entity per line.  
We need to create the property and add **Defect** value.

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Timed Notification Setup

- Limit rules evaluation scope
  - ▶ By a query saved in a public workspace using the following properties
    - TN\_Query-{record type}
    - TN\_Query-{record type}-{pseudo action name}for example:  
**TN\_Query-Defect-DailyNotification** property set to  
'Public Query\Admin\DailyNotificationQuery' value
  - ▶ By dynamic query with excluded states using the following properties
    - TN\_QueryExcludeStates-{record type}
    - TN\_QueryExcludeStates-{record type} -{pseudo action name}Where excluded state names are listed one entry per line



CCMA43

IBM Rational Software Development Conference 2006



## Timed Notification Setup

- Create an Email\_Notification\_Rule
  - ▶ **Actions** field set to pseudo-action name, for example, **DailyNotification**
  - ▶ Condition field contains a notification condition, for example  
`$Due_Date ne " &&  
DateTimeDiff( $Due_Date, GetDateTime() ) < 0`
  - ▶ Other notification fields can contain related information, for example  
Subject:  
Defect \$ID is late  
Body:  
Defect \$ID passed its due date for  
`DateTimeDiff( $Due_Date, GetDateTime(), 'd' ) days.`  
To view this record click on the following link:  
`GenerateURL($dbid, $EntityDefName)`



CCMA43

# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## Agenda

- Introduction
- Email Notification solution features
- Implementation details
  - UDBProperties package
  - EmailNotification package
- Configuration Examples
- Delivery
- Timed Notifications
- **Installation**
- Questions and Answers



CCMA43

IBM Rational Software Development Conference 2006



## Package Installation

- Download the latest version of distribution
- Apply package to the test schema using Package Wizard or command line interface
  - You have to use command line packageutil enablerecord command if you would like to enable the package for a stateless record type
- Create email\_host property (and optionally others)
- Create notification rules
- Set email\_enabled property to 1



CCMA43



# IBM Rational Software Development Conference 2006

IBM Rational Software Development Conference 2006



## License Terms

- The package was released in 2004 under GNU **Lesser** General Public License
  - ▶ It also known as a GNU library license
  - ▶ You can copy, distribute and modify the package
  - ▶ You can combine the package with a "work that uses the Library" to produce a work containing portions of the package, and distribute that work under terms of your choice
  - ▶ No warranty
- Current location is <http://www.geocities.com/pdubovitsky/ClearQuest/>



CCMA43

IBM Rational Software Development Conference 2006



## Acknowledgements

- Richard Scire            Richard.Scire@Mozelle.com
  - Patrick Pitsinger       pitsinger@us.ibm.com
  - Reto Hersiczky         Reto.Hersiczky@infocopter.com
- Thanks to all customers who tested the package in different environments, submitted bug reports and provided me ideas to improve the package



CCMA43


# IBM Rational Software Development Conference 2006

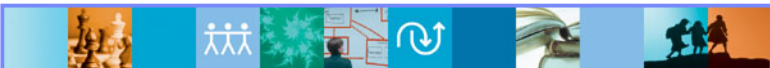
IBM Rational Software Development Conference 2006 



# Questions


Software  CCMA43

IBM Rational Software Development Conference 2006 



# Thank You

*Pavel Dubovitskiy*  
*[pdubovitsky@yahoo.com](mailto:pdubovitsky@yahoo.com)*

Software  CCMA43