CM+ Release 7.0 Release Notes (Build p05)

CM+ 7.0 Release Overview

CM+ 7.0 adds the remaining capabilities to make CM+ the industry's first 4th Generation CM tool. The advance in capabilities enables a much richer role-based user experience requiring less training, more intuitive operation and integrated process guidance.

The main advances in CM+7.0 are in the following key areas:

1. Advanced Configuration Management
2. Repository and Process Engine
3. GUI Capabilities and Ease-of-Use
4. Customization
5. Process Support
6. Optional Add-ons

These areas contain the following list of new features, with detailed feature descriptions following the summary.

1. Advanced Configuration Management

1.1 CMII Process Model Support
1.2 Visual Baseline Deltas
1.3 Baseline tree navigation
1.4 Source code searching, including revision searches
1.5 Item unique numbers on browser data
1.6 Peer Review Process Support
1.7 Formal and Simplified Parallel Checkout Tracking
1.8 Check-in of only unsubmitted files, including for bulkload operations
1.9 Checkout Icon in Tree Reflects Branch Tip Always
1.10 sign Command Support for Comments
1.11 Show Directories/Files that are not in the base tree
1.12 Open Stream Product Function To Start A New Stream
1.13 Timestamps on retrieved binary files now reflect repository date
1.14 Improved Line Count Results and Binary File Handling.
1.15 Release 7 Ease of Use and Increased Functionality
1.16 -get Option on Borrow and Branch Commands
1.17 Elimination of Branch and Borrow Post Command Triggers
1.18 Source Review Staging Area
1.19 Build Dashboard to Manage Build Definitions
1.20 Synchronize Button on Compare to Workspace
1.21 wsdate Operator for Workspace File Date Comparison
1.22 wsdelta WorkSpace Delta Operator
1.23 Flagging a History Chart
1.24 Skip File/Dir Option in Analyse/Bulkload
1.25 Workspace Status Capability
1.26 CM+ Concepts and Overview Document

2. Repository and Process Engine

2.1 Virtual List Field Capability for Inverted Lists
2.2 Resumption of a Transaction on a Restart
2.3 Extended field list capabilities for alias and content sizing
2.4 Verbose delete command for dangling reference identification
2.5 Specific email lists for auto email
2.6 Typed list operator support
2.7 Control Of Folder Separators on PC Platforms
2.8 Commas Optional in Color Triplet
2.9 Graph/Table Prompt uses field list width as a step count
2.10 Graph/Table Prompts support Sub-fields
2.11 Quotes are Stripped off Command Trigger Parameters
2.12 Enable or Disable History with history System Parameter
2.13 CM+ now identifies the suspect field in an invalid field list
2.14 Cross Platform Text Reports
2.15 -txn and -rc Options to the txn-query Command
2.16 refresh$com Macro and Execution
2.17 Exclude Benign Change Commands From Transactions
2.18 XML Import Capability
2.19 Repetition Macro
2.20 Field Lists with Data Macro

3. GUI Capabilities and Ease-of-Use

3.1 Data List Filtering/Searching
3.2 Range element association with colors and browser
3.3 Graph improvements: Color, Data Values, 3D, Stacked, Background, State Shapes, Font Size, ...
3.4 Dynamic Tabbed Views for CM+ Main Output Area, with replace option
3.5 HTML in Tabbed Views
3.5 HTML in Dialog Prompts
3.6 Wildcard tab removal.
3.7 Graphs, Displays, Tables and Charts in Tabs
3.8 Dashboards: Display Panels, Tables, Graphs and Charts in Dialogs
3.9 Dynamic Widget Support for Graphs, Tables, Displays and Charts
3.10 Prominent form Labeling
3.11 Containers and Container Actions
3.12 Color prompts in dialog panels, and simplified state color specification
3.13 Forcing text in a text field
3.14 Command output used to populate note prompt
3.15 Right justification of total fields
3.16 Tree-display in data pane for hierarchies
3.17 Advanced GUI for Linux and Solaris 10
3.18 Sysparm notewrap used for window displays in note drill downs
3.19 Range Color Enabled By Default, Controlled by Sysparm
3.20 Use of Color in Text Panels/Prompts
3.21 Locating Highlighted Text
3.22 SubClass Identification on Forms
3.23 Numeric Field Prompt Navigation
3.24 Display Item Selection
3.25 Selecting Tree Pick List Selects that Browser Tree
3.26 Navigation Control For Dates
3.27 Compound (Complex) Widgets in Forms

4. Customization

4.1 Adding buttons to text displays
4.2 Button macros for automatic button definitions on various panel types
4.3 Button definitions for Tabbed main display panes
4.4 stsgui command to reset popup menus
4.5 Select all and select all but in prompts
4.6 Auto-generation of test commands for GUI testcase generation
4.7 Macro parm list display
4.8 Debug macro and script capability
4.9 Configurable main panel frame - allows longer tree panel
4.10 Separators in Browser Data List
4.11 Context-based Field and Field List Specifications
4.12 Display scale system parameter
4.13 Additional Text Buttons on Text Displays
4.14 Command Output Used To Generate Note Prompt Content
4.15 Trimming of Empty Elements in Browser Data Tracing Tree
4.16 Force Selection Option on Set List and other List Widgets
4.17 Text Line Selection and Popup Menus in Text Tabs
4.18 Secondary Widget Sizing in Prompts
4.19 Font Size System Parameters for Note Fields
4.20 Boolean Values in -gray Option
4.21 N-Up Prompting - beta
4.22 Key Formats in Display Panels
4.23 GUI and Macro Specifications/Documents (new documents)
4.24 Specifier for Wrapping of Note Fields in Prompt Dialogs
4.25 Replacement of Menu Items
4.26 Multiple Field Prompt Modifiers
4.27 Using Table and Graph Widgets as a Master Widget
4.28 Negation of Qualifier for Menu Buttons
4.29 GUI Prompts Component Macros
4.30 Set Enter Key Focus for Panel Buttons
4.31 Optional AccessProt Parameter on Browser Command
4.32 Automatic Panel Titles From Form and Menu Buttons
4.33 Stripping Prefix from Prompt Entries in a Set List Prompt
4.34 Field Selection on Data Type Prompts
4.35 Default Specification on Group Multiple Prompts

5. Process Support

5.1 State Flow Diagram Improvements
5.2 Process flows, gantt charts, and graphs will track state colors
5.3 Process flow in tabs
5.4 On-line process help tree
5.5 Electronic Authorization
5.6 Logging Action As Part of Transition Logging
5.7 Automatic Logging in Short Reply Format
5.8 Automated Application Map Generation - pre beta
5.9 Schema Documentation Generation

6. Optional Add-ons

6.1 Web GUI for CM+ - Option
6.2 DSO (Directory Search Order) Capability - Beta, Option
6.3 VFS (Virtual File System) Capability - Beta, Option
6.4 Live Export Capability - Beta, Option
6.5 Eclipse Plug-in - Beta, Option
6.6 OpenVMS Support on Itanium (HP Integrity) Platform

Detailed Feature Overviews

1. Advanced Configuration Management

1.1 CMII Process Model Support

A CMII Profile plug-in option is added to CM+ so that it can be used to support the CMII Process Model for development. At the time of this writing, CM+ is the only SCM tool CMII certified for Software CM. The CMII Profile is especially useful for organizations which use the CMII Model for hardware development as it allows both hardware and software teams to develop according to a common methodology.

The CMII Profile includes support for the six basic CMII Roles, CMII Process documentation, CMII Work Flow and State Flow, Impact Analysis dashboards, pre-defined tabs and a number of custom buttons and menus, all using natural CMII terminology, extended for software development, as detailed at CMII World 2008. Screen shots, product documentation and a demo are available on Neuma's web site.

1.2 Visual Baseline Deltas

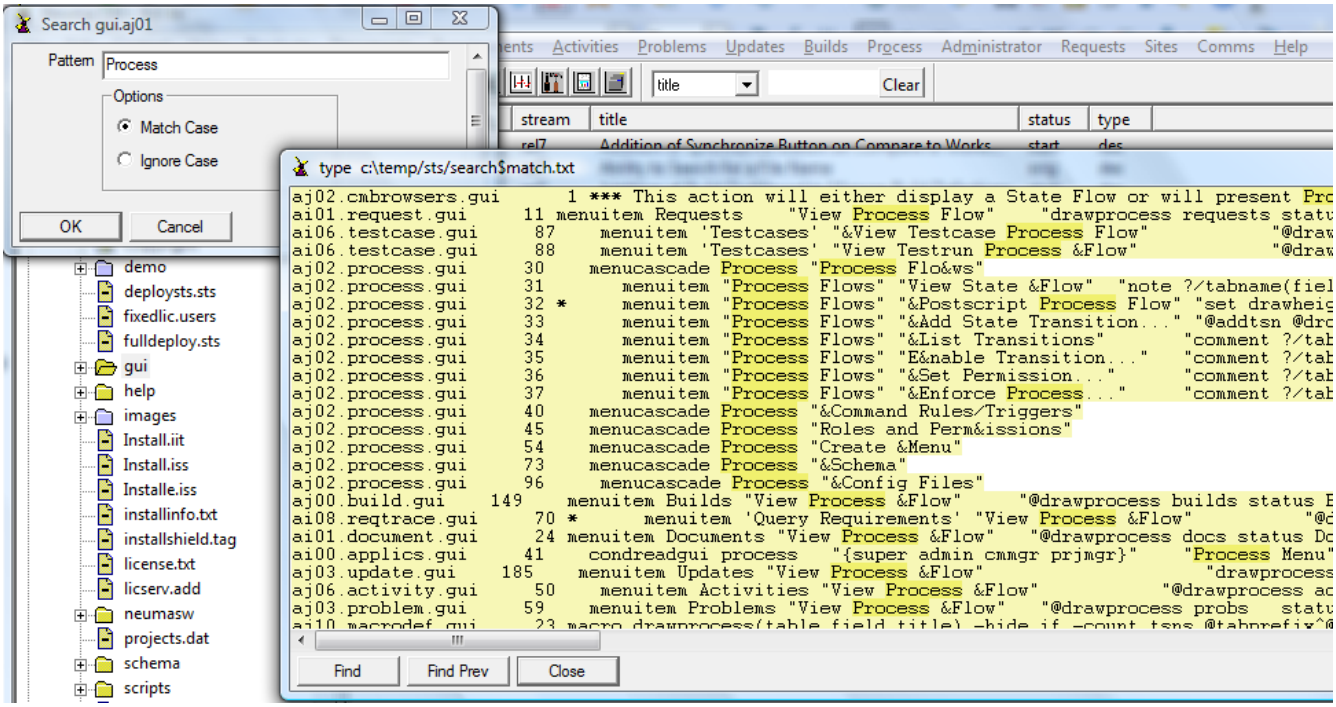
The introduction of the "listview" command allows a comparison of the current context to a set of revisions/updates specified for comparison. The command allows specification of both "old" and "new" revisions, and defines the highlighting that applies to showing them, thus forming a visual baseline delta. As well, highlighting can be applied based on update or build status to help further enhance the visual baseline delta.

1.3 Baseline tree navigation

The data list pane can now display a tree of data for the selected tree browser item, rather than simply the direct members of the selected item. The tree is shown in an indented fashion and expands or collapses portions in parallel with the expansion or collapse of the corresponding tree browser. This allows an easier focus to details of the tree, whether a pure data tree or a context tree, and helps support viewing of baseline deltas.

1.4 Source code searching, including revision searches

The source tree context menu (i.e. right-click pop-up menu) now contains two entries for searching a sub-tree of source for specific text. This is more flexible than the full project searches of popular IDEs, such as Microsoft's Visual Studio (R). In the first case, the source in the specified sub-tree is searched for the specified text. Standard wildcards (i.e. *, $, and a leading !) are permitted in the search text. In the second case, all revisions of the source in the sub-tree, typically all revisions of a single file, are searched. This is very helpful in locating the revision at which a particular function or variable was introduced or first referenced.



1.5 Item unique numbers on browser data

For purposes of identification and auditing, it is now possible to show uniquely assigned item numbers (i.e. unique key values) in the tree browser and data list panes. This is typically useful for the CMII Process Model, or to prove the uniqueness of the CM+ identification mechanism. This is done by enabling the itemnumbers system parameter.

1.6 Peer Review Process Support

A new interactive panel allows a simpler peer review of a software update, and allows the review to be more easily captured against the update. The panel allows scrolling through the delta reports of the individual files of an update, while allowing a consolidated review log to be built and attached to the update. (May require some customization.)

1.7 Formal Parallel Checkout Tracking

CM+ 6.x, and earlier, supported parallel checkouts either informally, or as an alternative to the support of queued checkouts. As well, the support was partial in that parallel checkouts did not appear as part of an update until it was checked in. CM+ now supports a formal tracking of parallel checkouts so that delta reports and checkout lists can be readily reviewed. This capability no longer precludes the operation of queued checkouts and also reduces significantly the pre-existing rules and triggers that were used to support these checkout modes. Two new system parameters are introduced to control the default checkout mode used ('checkoutmode'), and the set of permitted checkout modes ('checkoutmodes'). Allowed modes are "exclusive", "parallel" and "queued". The borrow command now explicitly supports a checkout mode argument to override the default.

1.8 Check-in of only un-submitted files, including for bulkload operations

Often, an update is re-opened to correct a single file, or a portion of the files are checked in prior to modifying the remaining files of an update. On a subsequent check-in operation, all of the files are again checked in. This can be both resource intensive and time consuming, especially if some of the already checked-in files are large. CM+ now provides an option to check-in only those files which have not yet been checked in or have been re-opened (i.e. status is now 'open' for the file revision). This is useful when a large bulkload operation has been done and one or two of the files failed to check in (e.g. because a designated text file had non-ASCII characters within it). Rather than abandoning the bulkload, dealing manually with individual files, or repeating the check-in of all the files, use of this feature allows only those files which have not already been bulkloaded to be checked in, without having to create a separate update.

1.9 Checkout Icon in Tree Reflects Branch Tip

When the promotion level of a context is such that only checked-in files are visible in the view, users still want to know if the current branch of a file is checked out. The checkout coloring (red/green) is extended so that, rather than looking at the status of the item in context, the status of the tip of the branch in context is used to indicate the color. This feature is selectable using a system parameter named "tipstatus". When tipstatus is on, the status color reflects the tip of the branch. When off, it reflects the current context revision's status.

1.10 sign Command Support for Comments

The sign command allows two types of comments to be appended to a signature. By default, the signature simply inidicates "Signed". However, if the [-text QuotedString] is specified, the "Signed" message is replaced with the contents of the QuotedString. If the [-notes FileName] option is specified, the notes in file FileName are appended after the signature line.

```
sign [-remove] Set [-field Fieldname] [-text QuotedString] [-notes FileName]
```

If the [-field Fieldname] parameter is specified (and it is a signature known field), the specified Fieldname is used for adding the user name. Otherwise, the first signature field is used. The signature field may be either a list field or a reference field. In the latter case, the reference field must have a value of NIL when no signature has been applied.

Note that the -text option cannot be used with the -remove option. As well, the quoted string parameter of -text may not contain both the sub-strings "Signature" and "Removed". It may contain one or the other, but not both. This is to ensure the integrity of the signing log, so that only if -remove is permitted will "Signature Removed" appear.

1.11 Show Directories/Files that are not in the base tree

If a directory/file is not in the base tree, but is added as part of the "listview" set, these are shown if their parents are shown. So for example, if a file is changed to a directory and three files are added to it, the 3 files show up in the listview if the original file is visible in the tree. All elements that cannot be selected in the tree, show up in the listview if the lowest level ancestor in the tree is visible.

1.12 Open Stream Product Function To Start A New Stream

The Products | Open Stream... menu item is used to open a new stream. When using branchtracking, the new stream needs to be open first so that the branch tracking will work successfully. Attempting to open an update for a stream before Open Stream has been used to specify the stream heritage will produce a warning message for the user. The Open Stream function branches the root node of the product, and in so doing, captures the branch history for the new stream. This is inherited by all other nodes (files/directories) in the system.

1.13 Timestamps on retrieved binary files reflect repository date

Previously, binary files were always retrieved from the repository with the current date/time as the timestamp. This differed from the handling of non-binary files. CM+ now retrieves binary files with their original repository timestamp unless the "newtime" option or system parameter has been enabled. Although this is the same behaviour as for ascii files, it does represent a change to previous behaviour. Any timestamp-sensitive binary file operations should ensure that the newtime options is specified when retrieving binary files.

1.14 Improved Line Count Results and Binary File Handling.

Line count results now clearly indicate the number of binary and non-binary files and specificallyy indicates that linecounts are exclusive of binary files: An example output line for the right-click Line Count operation is:

Line Count for images.aj02: [98 file(s)] 267 lines, excluding 87 Binary File(s)

1.15 Release 7 Ease of Use and Increased Functionality

In CM+ Release 7, much of the new functionality is brought out to the user interface so that it is available to non-power users. As well, CM+ continues to improve the ease of use through customization of the menus. In many cases successive prompt panels are replaced with smarter dynamic panels which can be used for multiple operations rather than just one.

1.16 -get Option on Borrow and Branch Commands

The -get option of the borrow and branch commands cause retrieval of the borrowed or branched file into the appropriate directory. The appropriate directory tree root is given by the "directory" field of the update, or if that is NIL, then by the current working directory, also known as @cwd. Based on the current tree mode setting, the root is treated as a pool, sub-tree or tree for retrieval.

1.17 Elimination of Branch and Borrow Post Command Triggers

The branch and borrow post-command triggers are removed and replaced with the use of the -get option of the branch and borrow commands. This simplifies the customer environment. Upgrades to CM+ should remove these triggers, if unchanged, or modify them to remove the "get" functionality inherent in them.

1.18 Source Review Staging Area

Normally, source code resides in the user's workspace or in the CM repository. However, often there is a need to review the code or otherwise give public access to it before it is in the repository. As well, there is a need to checkpoint code nightly (i.e. to a server), for backup purposes. "Save" operations, though they do this, increase the revision count of the Source file significantly.

To deal with these issues, a review staging directory is supported. This directory, "staging", appears as a sub-directory of the "txn" directory. Whenever a new txn session is started, the "staging" directory is moved to the new txn directory. Within the "staging" directory are directories with names matching each applicable update key.

The -stage option of the submit command will send user source files to the "staging" directory for the update. The status between "open" and "submit", formerly "reopen", is now named "staged", indicating that there is a staged version of the source code. The first ever submit ... -stage operation will generate the "staging" directory. Local workspace staged files are not renamed, nor are they deleted, as a result of a staging operation.

When CM+ is looking for source code for a revision that is "open", if not found in the workspace directory tree, the "staging" directory is checked to see if the file is staged. The server never uses this directory. However, a submit operation can move source code from the staging area to the txn directory, if the file is not otherwise found in the user's directory.

The get, edit, delta, submit and reconcile commands use the staging area whenever the update is at status staged and source is not found in the workspace directory. Once an update reaches the "submit" stage, the staging directory is removed. A sysparm "stagebackups" controls whether the contents of staged update directories are preserved by moving them to the backup directory in the txn directory. Files in the "staging" directory are always written so that the appropriate number of backups are kept, as determined by the transaction server (backups sysparm). Older revisions in the staging area are not available for general retrieval (ie. this requires explicit access to the txn directory).

For parallel updates the edit, get, delta, and reconcile commands consult the context updates looking for the first update in context that has checked out the revision in question. That update is used to determine which, if any, staging directory to look for.

1.19 Build Dashboard to Manage Build Definitions

The build dashboard manages the CM Builder role of accepting updates that have been readied by developers and integrating them into build definitions in preparation for a build operation. This is the initial version of the build dashboard, which over time will evolve substantially to include, most likely, the launching of builds, and perhaps control over the automation of the build process. The current dashboard is meant for use in place of a fully automated nightly or regular build process. It should be further customized to meet the needs of the project.

1.20 Synchronize Button on Compare to Workspace

The Compare to Workspace function has a Synchronize button which may be used to synchronize your workspace after a comparison. The dialog shows:

1. Compared files which are different (and these are selected)
2. Files missing from your workspace (and these are selected)
3. A list of files which compared differently but which you have checked out

You may wish to manually de-select items in 3. from items in 1., prior to a Synchronize operation.

1.21 wsdate Operator for Workspace File Date Comparison

The wsdate operator takes a file revision set, and compares the date on the workspace file to the right hand operand. The right hand operand may be a date, > date, >= date, < date or <= date. In this case, only the date portion of the time stamp is compared. The right hand operator may also be *, in which case the ws file date, (already adjusted for local time), must match the SM issue date, adjusted for local time. Files that are not found in the workspace are identified by a variable named: $wsnofile The content of this variable can only be used prior to another use of the wsdate operator.

1.22 wsdelta WorkSpace Delta Operator

The wsdelta operator accepts a file revision set as a left operand and a workspace root as a right operand. After comparing the file revisions to the source code in the work space, the result is comprised of all files which are either missing or have a source code difference (other than the CM+ inserted header line, if present).

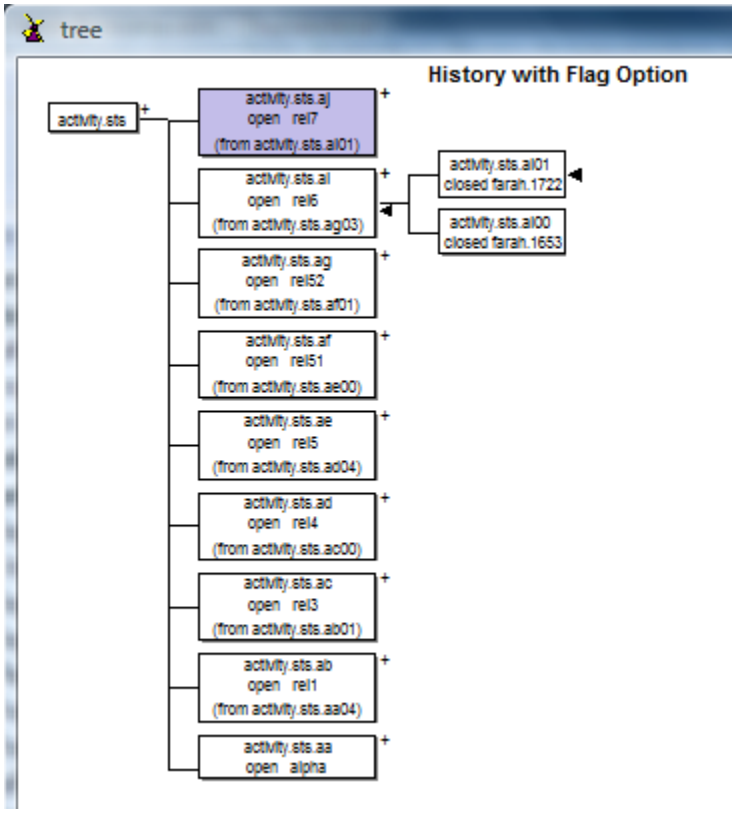
1.23 Flagging a History Chart

When CM+ displays a history chart, it shows the current context (if applicable) shaded. In addition it is possible to draw specific attention to a specific revision, or revisions, on the chart. The [-flag Set] option of browsehist causes two actions.

1. All flagged elments, if/when shown, will have an arrow point to them from right to left, located part-way down the right side of the box containing the element. This allows, for example, a quick and easy way to highlight the version

of the file that is in the workspace currently, or perhaps the revisions pertaining to a particular activity or set of problems.

2. The first element of the Set is made visible as part of the browser. This may include the need to increase the number of revisions shown, from the default value.



1.24 Skip File/Dir Option in Analyse/Bulkload

The analyse procedure includes a selection, when a new directory or file is found, to skip the file or the dirctory. In this case, the skip operation is logged, and on doing the bulkload/update, the file and/or directory is ignored. If a directory is ignored, its entire content is also ignored, with no further prompting for its contents.

1.25 Workspace Status Capability

The workspace status capability allows users to analyze the contents of a workspace against a CM+ context view. The files found are partitioned based on workspace status. Various actions are allowed based on the workspace status (partition) of a file. The user is able to reconcile the workspace contents against the context view in a more straightforward manner. The Workspace Status capability is documented in the CM+ Demo Tutorial Guide.

1.26 CM+ Concepts and Overview Document

A new Concepts and Overview Guide is added to the CM+ on-line documentation. This guide gives an overview of CM+ and Neuma's default customization for use across the ALM functions. This is the initial version of the Concepts and Overview Guide, and is intended to give a mix of CM+ operation overview and CM theory.

2. Repository and Process Engine

2.1 Virtual List Field Capability for Inverted Lists

Because of its efficient design, CM+ has never used index files to optimize access to inverse reference relationships. This is typically done in a relational DB because there are no hierarchical or reference list relationships which lead to better performance, nor is there the smart client caching used by the STS Engine of CM+. For very large data sets, the lack of indexing can produce some queries that are slower in CM+ than in an indexed relational DB. As well, the lack of an inverse field makes it more difficult to display or navigate the inverse relationship.

In CM+ 7.0, the schema administrator can define virtual list fields which are essentially inverted lists, or index files. Such inverted lists are not stored as part of the repository, other than their schema definition, but are automatically generated as needed the first time the inverted relationship is used in a CM+ session.

For example, the default CM+ schema links updates to the problems and activities being addressed by those updates. This is a natural relationship because it is at the creation (or modification) of the update that the reason is most naturally specified. It would be nice to be able to query and display, from a "problems" perspective, which updates were used to address the problem. Without virtual lists, this requires the use of a special menu button, or of a script to generate a non-interactive report. Typically an object-oriented query menu item is provided to trace the problem back to the updates that fix it. If a virtual "changes" field is inserted into the problems table, CM+ will maintain the inverse relationship during the user session if it is referenced. The "changes" field would identify the "problems" field of the "updates" table as the target list to be inverted in order to form the "changes" field. Subsequently, the changes field can be queried or displayed as if it were a normal "list" field of the "probs" table. The virtual field cannot however be modified. Only the original data of the "updates" table (i.e. the "problems" field), can be used to specify the actual problems addressed by updates.

Virtual fields have a field class "vlist" and specify the table and field for which the virtual list is being created. The data type of the virtual field should always be VList, which is a size 0 field. The field name, as with any field name, should be chosen so as not to conflict with existing table names or built-in monadic (unary) operator names [help UnaryOp]. Note that vlist fields may be added either as part of the original data schema, or as a change to the existing library. In the former case, the "field" command cannot be issued until the target table has been added to the repository. So, in the above example, the "updates" table must exist prior to execution of the "field" command for the problems "vlist" field of the "probs" table.

2.2 Resumption of a Transaction on a Restart

Prior to CM+ 7.0, if a power outage, crash or some other action resulted in a pre-mature shutdown of a CM+ client session, on restarting the session, the user would have the option of committing or discarding any transaction that was in progress. In normal operation, where changes are saved automatically, this was a rare event. However, for CM managers or others who typically do not have their transactions automatically committed (i.e. saved), the occurance is less rare. Rather than forcing the user to decide to commit or discard on re-entering the client, post 7.0 CM+ allows the user to resume the transaction so that it may be extended, committed or cancelled on the user's own schedule, just as if the session had not been interrupted.

2.3 Extended field list capabilities for alias and content sizing

Field list specifications for the browser data list pane, the display panel, the show command and for forms, have been extended to allow two new functions. First, an alias may be specified in a field list by specifying "field=alias" instead of "field" for the field to be aliased. In particular, the key field may be aliased by specifying "=alias" as the first field of the field list. Secondly, except for in forms, the data displayed may be truncated by selecting N characters from the start or the end of the data field to be displayed, regardless of the field width

selected (as long as it fits). So, for example, dates.orig="OrigDate":-8 will select the last 8 characters of the date field (yy/mm/dd typically) and display them with a column heading of "OrigDate". This would otherwise display as "orig: yy/mm/dd" with a column heading of "dates". Note that both a field width and a data size can be specified, such as: priority:8+3 to display only the first 3 characters of the priority, but in a field width wide enough to display the heading "priority".

2.4 Verbose delete command for dangling reference identification

A verbose option on the delete command now indicates the references that are being NIL'd out as part of the delete operation. So, for example, if you attempt to delete a file revision that is in an existing baseline, rather than simply being deleted and replaced with a NIL reference (which is why the delete command is not considered a CM command but just a repository administration command), a message is given that it is deleted from the members list of the directory. For example:

```
sts> delete ci.c.ai43 -verbose
jones.1020:  mods modified.  Removed:  ci.c.ai43
stscoms.ai05:  members modified.  Removed:  ci.c.ai43
```

2.5 Specific email lists for auto email

When a transition fires, there are emails that can be sent automatically, without having to code a trigger. Previously, the trigger fires to the mailowner or a maillist against the transition. This behaviour is preserved, but the following is allow performed:

1. Email is sent to all owners and/or sub-owners ("mailowner" option)
2. Email is sent to transition lists, but also, to all users in a "users" list associated with the record ("maillist" option)

So if you have a "reviewers" field (list users), and the "maillist" is selected for a data transition, all reviewers on the list are sent the email.

2.6 Typed list operator support

When a tlist field is used as a monadic operator, the result can be a set of conflicting table types. When this happens, some get ignored. Similarly, for a zero-length argument, the type of the result is unknown. So, for example,

```
    mods (updates take 0)
```

could be an empty set of issues or an empty set of reqissues, etc. This is a problem when used in an expression such as:

```
    list issues take 1 or mods (updates take 0)
```

The "or" operator does not match the right hand side. However, if the operator is qualified with the table name, the result can automatically exclude all results not matching the table specified:

```
   mods[issues] updates
```

results in the mods lists for all updates where the mods type is #issues. No warning is given as it is equivalent to: "mods (updates modtype #issues)". There is no equivalent for the empty set case, and so:

```
   mods[issues] (updates take 0)
```

will result in an empty issues set, rather than an arbitrarily typed set. This permits the following line to succeed because the right side is typed:

```
   list issues take 1 or mods[issues] (updates take 0)
```

2.7 Control Of Folder Separators on PC Platforms

The system parameter "hostdirsep" controls whether the host is used to influence the appearance of directory separators in keys. When disabled, directory separators use the forward slash character. Otherwise, the

separator is determined by the convention of the host OS. In practice this means that, by default (disabled), CM+ keys always appear with forward slashes. This is different than the original default which was to use back slashes on Windows.

2.8 Commas Optional in Color Triplet

A color triplet is now allowed to be specified as, for example, either:
(255, 0, 255) or (255 0 255).
The commas are optional.

2.9 Graph/Table Prompt uses field list width as a step count

When a field width is specified on a field in a prompt dialog for a graph or table widget, the field width is to be used as the "step" value for computing the table/graph display. See the graph/table commands "step" options for more details.

2.10 Graph/Table Prompts support Sub-fields

Graph and table prompts now support sub-field specifications for field names.
?/probs dates.orig > 2000/(dates.orig:32)|Arrival
In this example, the "orig" date is used to determine the Arrival rate of problems. Note the :32 "field width" indicates that dates are to be plotted in steps of 32 days (which is one month, by STS rules).

2.11 Quotes are Stripped off Command Trigger Parameters

Consider the command:

```
change '/sts' -field type to product
```

as opposed to

```
change /sts -field type to product
```

Prior to CM+ 7.0, the quotes were not stripped off of '/sts' before being sent in as parameter 2 of the change triggers. Stripping the quotes is now the default behaviour. Note that stripping quotes is the normal behaviour for passing parameters into any script.

2.12 Enable or Disable History with "history" System Parameter

The system paramter "history" is used to enable or disable the History capbaility. History should normally be explicitly turned off for non-interactive sessions.

2.13 CM+ now identifies the suspect field in an invalid field list

Until CM+ 7.0, erroneous field lists were flagged, but the specific problem was unidentified. The specific field name causing the problem is now explicitly identified as part of the error message:

```
display probs (status prio title note)
              ^^^
```

*** Error Scanning Set: Unexpected name, value or operator in set: **prio**

2.14 Cross Platform Text Reports

In some situations, it is necessary to create output from CM+ for absorption by a cross-platform format. For example, the Open Office data import facility assumes Unix style end-of-line characters. To enable this, a system parameter, textmode, is set, by default to the home environment, Unix or Windows, but can be reset to simulate output for the target system from a cross-platform client.

2.15 -txn and -rc Options to the txn-query Command

The txn -query command returns the log results of a transaction. With the -rc or -txn options, it can also return the return code or the original commands of the transaction.

```
txn -query 22 -txn  returns the .txn original command content (rather than the .log)
txn -query 22 -rc   returns the .rc file contents (rather than the .log)
```

2.16 refresh$com Macro and Execution

The refresh$com macro is used to define actions to initiate after a refresh operation has been done, whether by command or otherwise, but only if the refresh succeeded. If the macro is defined it is executed as a command. If not defined, or is blank or the empty string, it is not executed.

The most important use of the refresh$com is to complete an action on a record whose key was not known until after the refresh operation. For example, when a problem attachment is added, it cannot be submitted until the attachment id is finalized by the server. The refresh$com macro allows a way for this submit operation to occur automatically, without a subsequent user action:

```
e.g. macro refresh$com \+
        if -count (attachment (probs owner @user take -5) body 0) not -0; \+
          submit attachment (probs owner @user take -5) body 0;  end
```

This macro would inspect the last five problems originiated by the user and if any had unsubmitted attachments (i.e. body 0), they would be submitted.

2.17 Exclude Benign Change Commands From Transactions

When a change command executes, it indicates the number of records that have been modified. However, if this number is 0, the change command still becomes part of a transaction, because the client view does not necessarily match that of the server. Although this is proper behavior, when doing upgrades, it is handy to be able to exclude such transaction lines (i.e. when the count of changed records is 0). To do this a system parameter 'txnsuppress' is enabled. By default, txnsuppress is disabled, and is typically only enabled by the upgrade scripts or similar applications.

2.18 XML Import Capability

A "-xml Table [Name] FieldList" option is part of the runfile command used to perform XML data imports. The "Name" is used as the name of the record field tag and the "FieldList" is used to specify the field tags that are to be populated from the XML file. The FieldList may include alias names. In this case, the alias is the name of the XML tag, while the field name is the STS field name it corresponds to. The field names are for documentation purposes only, when alias names are used.

When the CmdList is executed, the values from each XML record, starting with the key value, and then the field values, are passed in as parameters. The output of the fileset -xml command would be in the proper format for use in the runfile -xml command. Note that the input file may have more fields specified than there are values specified in the field list. Also note that the key field, if required, is specified as part of the field list. XML fields may occur in any order within the record tag delimiters.

When a field contains data which is to be placed into a note, the data for the note is placed in the user's current directory under the file name <table>.<field>, and the name of the file passed in as a parameter to the command list. The file is erased prior to processing the next record, and is erased after the last command has been processed.

2.19 Repetition Macro

The repetition macro is denoted by the sequence @do. It takes 2 or 3 parameters.

```
@do  RangeTypeName  Count  'String'   or
@do  Set Count 'String'
@do  {TokenList} Count 'String'
```

The Count parameter indicates the number of elements of the range to be used. A Count of * means all elements are to be used. For example:

```
@do DayOfWeek *  '^rangeord  DayOfWeek ^1 - '
```

The deferred macro notation, ^1, is used to refer to iterative argument. @do passes ^1 into the 3rd argument string as the iteration value. In this case the result of the iteration macro will be:

```
0 - 1 - 2 - 3 - 4 - 5 - 6 -
```

The result is the concatenation of the String argument, with deferred macro substitution performed, for each loop iteration. For example:

```
@do Month 6 "^1 "
```

results in:

```
Jan Feb Mar Apr May Jun
```

2.20 Field Lists with Data Macro

The @data macro is permit a field list to be specified. The 2nd (field) parameter may be a field list enclosed in quotes.

```
note @data p.123 '(priority status title)'
```

would be functionally equivalent to:

```
note @data p.123 priority @data p.123 status @data p.123 title
```

As well, the field width option of a field may be specified:

```
note @data p.123 'priority:9  status:9  title:40)'
```

This causes field truncation or padding.

3. GUI Capabilities and Ease-of-Use

3.1 Data List Filtering/Searching

A field selector and a filter text box at the right end of the tool bar (by default) are used for data list filtering. By default, the field selector is the "title" field, if one exists. When the text box is not empty, the data list is filtered based on records which have the field matching the text string. The text string is a wildcard string, allowing ! (not), * (any number of characters) and ? (one character) wildcard operations. Blanks are significant in the text string. The field list corresponds to the current record type being shown. The field value "key" is one of the fields always available in the field list. The search bar's case sensitivity is controlled by the "ignorecase" system parameter.

3.2 Range element association with colors and browser

The range command is used to associate a colour with a range element.

range -color <Color> TypeName ElementName

will ensure that ElementName is a symbol in the range "TypeName" and will associate the color "Color" with that element. This association will be used by various display tools within STS, including the process, graph and gantt commands. Note that the color stored is a 16-bit color, so that a close approximation to the selected color is used.

The @elementcolor macro is used to query the current color of a range element.
Syntax: @elementcolor <Range> <element>
The result is a colour triplet (R,G,B), unless no color is defined for that range element, in which case an empty string results.

3.3 Graph improvements: Color, Data Values, 3D, Stacked, Background, state shapes, font size, ...

A number of basic improvements have been made to the various charts and graphs produced by CM+.

1. Because a gantt chart may involve several items, the date scale is shown at both the top and the bottom of the gantt chart. The same holds for bar graphs.

2. The "vertfliptext" attribute of a transition allows the text position of the transition description to be flipped from below to above the state transition line or vice versa.

3. A system paramenter, "chartbgcolor" is used to specify the background color used by graphs and charts, including the default background color for the process graph. "chartfgcolor" is used to specify the foreground color used on boxes and process bubbles, etc. These are overridden if otherwise specified on the command (e.g. process command)

4. In a process chart prompt, the field list widths are used to indicate the font size of the text on the chart. The key's field width is used for the size of the transition text, while the "status" (or other appropriate field) width is used for the state's (or data's) font size. So for example,

```
?/#probs/(:11 status:17)!ProbStatusFlow
```

will cause the states to have a font size of 17 and the transition text to have a font size of 11.

5. Bar Graphs allow various possible displays on top of what it currently allows. The "stack" graph mode will stack the bars horizontally for 2D graphs. For both 1D and 2D graphs, bars appear with 3D effects and are more tightly clustered.

6. The STS bar graphs now display the data values within the bar to clearly identify the magnitude of the bar.

7. Larger state flow diagram titles are now used (the larger of size of state text or 16 pts)

8. The stacked graph mode is invoked in a prompt by the use of "||" where the second '|' indicates the GraphMode is "stack" instead of "bar". For example: note ?/probs/(priority status)||ProblemsByPriority

3.4 Dynamic Tabbed Views for CM+ Main Output Area, with replace option

Tabbed views replace the 6.1 (and previous) sliding frames for the Output Log and for Record Notes. Two tabs: Log and Notes are always present and serve the function of the pre-existing frames. As well, new tabs can be created as follows:

command > ["TabName"] - Redirects output of "command" to a tab named TabName
command >> ["TabName"] - appends output of "command" to a tab named TabName

Note that the tab name must be a quoted string to indicate that it is a tab. If the tab does not already exist, it is created; otehrwise it is replaced. Tabs created with []'s are text-only tabs.

When a tab is created, its definition is retained and re-evaluated whenever the tab is given the tab focus. This permits tab summaries to remain up-to-date.

3.5 HTML in Windows and Tabbed Views

CM+ can now render HTML in a separate window or in a tabbed view of the main (lower) panel. When redirection uses {}'s instead of []'s, the content being redirected is treated as HTML and the window or tab view generated is an HTML view. When a quoted title is used within the {}'s, the redirection is made to a tab view, as opposed to a window, in the same manner as for text redirection.

command > {Window Title} - Redirects output of "command" to an HTML window with title "Window Title"
command > {"TabName"} - Redirects output of "command" to an HTML tab view named "TabName"

3.5 HTML in Dialog Prompts

```
note ?<>[Filename]File
```

causes a prompt for a text file in the form of a note field. When a + is added after the closing ">" bracket,

```
note ?<>+[Filename]File
```

the file prompt is changed to an HTML file prompt. This has the effect of displaying the filename content as an HTML file rather than as a text file.

3.6 Wildcard tab removal.

The stsgui command is used to remove one or more open tabs:

stsgui -tabs WildcardString

Any tab names which match the Wildcard String are removed from the display.

3.7 Graphs, Displays, Tables and Charts in Tabs

When the "-title QuotedString" parameter of the display, table, graph, gantt, progress, process, browsehist, and browsetree is specified and the title is enclosed in square brackets (e.g. -title "[My Chart]"), the resulting chart or graph is placed in a tab panel whose name is specified within the square brackets. If the tab panel already exists, the contents are replaced.

A number, up to 2 digits, may precede the title text after the '['. If present and successfully scanned as a value less than 100 this value overrides the drawscale system parameter setting for this particular display.

There are times when the title for the chart/graph, should be different than that for the tab. For example, if the chart is part of process help, it might always be redirected to the "Process" tab. To do this, the title and tab are both specified in the -title option, with the tab title in square brackets, and the chart title following:

```
-title "[Process]My Chart"
```

This would give the title My Chart to the chart, but would place it in the Process tab.

3.8 Dashboards: Display Panels, Tables, Graphs and Charts in Dialogs

The creation of custom dashboards is facilitated in CM+ 7.0 by allowing the appearance of various charts, graphs, tables and displays in a dialog prompt panel. These are read-only widgets which have no resulting value for the prompt, but which are interactive in the same way as their non-prompt counterparts behave, including pop-up menus and zoom-in capabilities. By judiciously arranging the appropriate prompts into a prompt panel, an interactive dashboard results. Note that traditional prompts may also be mixed into the dashboard.

The following prompts are enabled:

Graphs/Tables:

```
pie chart:   ?/set expr/(fields)+PieChart
line graph:  ?/set expr/(fields)/LineChart
bar chart:   ?/set expr/(fields)|BarChart
stackedbars: ?/set expr/(fields)||StackedBarChart
table:       ?/set expr/(fields)#Table
```

Display Panels:

```
display:     ?/Set/(fields)##Display
```

Charts:

```
process:     ?/Table/(field)![class]Process
gantt:       ?/Set/!(fields)=[range]Progress    [! optional, indicates both planned and actual]
progress:    ?/Set/(rangefield)==[range]Progress
```

Tree Browsers:

```
browsetree:  ?/Key/(field,fieldlist)-[orientation]Tree
browsetree -root: ?/Key/(field,fieldlist).[orientation]RootTree
browsehist:  ?/Key/(status update)\[stream]History
```

The parameters of the related commands, for these types of prompts, are encoded into the prompt specifications. So, for example, the table and field for the process prompt are specified as the selection set and the field list, respectively.

For graph prompts, and the table prompt, one, two or three fields may be specified and are take as the <field>, <by field> and <total field> respectively. If the second field has the "tot" attribute, rather than being considered a "by" field, it is assumed to be a -total field. If additional fields occur after the total field, they form a field list as in the table and graph commands. When only two fields are specified, <field> and <total field> are assumed. If a field width is specified on a field, it may be used as the "step" value for that field of the graph or table being produced.

```
bar graph:      ?/Set/[scale](field[:step] [byfield[:step]])|Graph
stacked graph:  ?/Set/[scale](field[:step] byfield[:step])||Graph
```

The widget is reduced to the approximately 10% of the normal display size for a graph. A numeric size modifier can be used to
specify a percentage other than 10. The numeric modifier is located in the usual position (after the prompt 'brackets' - which
in this case are / characters).

A display or table widget is a scrollable widget which shows the full width of the table or the default set of fields for the display, but only as many lines (including total and percentage rows for tables) as are given by the numeric modifier. The default is given by the system parameter "windowlen". E.g. ?/set expr/15(fields)##Display

Note that most of the widgets specified here may use dynamic widget capabilities, so that the content of the widget may depend on the content selected in a controlling widget. This makes for very powerful dynamic, customized dashboards.

3.9 Dynamic Widget Support for Graphs, Tables, Displays and Charts

The Dynamic Widget support is extended to support charts, graphs, tables and display panels. The appropriate widget is redrawn using the same real estate it originally used or a near approximation. Referencing the controlling widget in the prompt set or in the default value causes the normal dynamic widget relationship.

3.10 Prominent form Labelling

The -label parameter of a form (add, modify, form) adds a bold label to the top of the form in a larger bold point size (e.g. 30). The label appears left justified at the top of the form before the fields are shown.

Syntax: -label [String]

If String is not specified, the data type of the object is shown (e.g. Activity Problem, etc.). Otherwise the String is used. A system parameter, "formlabels", is used to indicate that the default label is always to be shown (i.e. even if -label is not specified). This is disabled (off) by default.

3.11 Containers and Container Actions

Normally, when a browser command specifies an action, the action applies when the browser item is double-clicked. If the browser item is a container, the action is applied on a selection (single click). So for keys, a double-click is required to perform a default action, and for containers, a single-click performs the action, which will typically be to display some help information. There is only one container action in effect for all containers, and this is the last specified container action.

This capability is used to help integrate the process documentation into the tree browser pane of the main CM+ window.

3.12 Color prompts in dialog panels, and simplified state color specification

The dialog prompt for colors is ?+Color. Prior to CM+ 7.0 this caused a color selector to show up and after picking a color, the next color prompt was brought up, in series. The revised color prompting causes a color prompt to show up as a colored empty widget/button with a selection button to bring up a color selector. The results of the color selector are immediately shown in the color widget.

3.13 Forcing text in a text field

When a text prompt includes a ! character, ?![]Text, the prompt will force a valid (non-empty) text field prior to allowing a prompt form closing action.

3.14 Command output used to populate note prompt

Normally, for a note prompt, the initial content of the note is filled using the contents of a file, which name is specified in the default field:

```
note ?<>[tmp.fil]OK
```

causes the OK note field to be populated with the contents of tmp.fil. The note result from the prompt is written back to "tmp.fil".

At times, it may be desireable to have the note field content determined dynamically, as, for example, with a dynamic prompt. For example, an update may be selected in one widget with the delta report generated into the notes widget. As you cycle through updates, you would see different delta reports in the note field.

When the default field contains a leading ">", the content of the field is taken as a command rather than a file name, and the note widget becomes "read-only" (in that, the result is not exported to a file), and takes on as its content the standard output of the specified command which follows the ">". For example,

```
note ?/updates status <= ready/(title)[@data 'updates status <= ready take -1' key]Update
?<>[>delta mods ?Update]Delta
```

would generate a delta report area which would (dynamically) be filled based on the content of the controlling Update widget selection. This is primarily useful in generating interactive prompt displays where a dynamically generated text result is part of the display.

3.15 Right justification of total fields

When a field shown in the data list browser has the "tot" (total) attribute, or is otherwise numeric, the data is shown in the field right-justified rather than left-justified.

3.16 Tree-display in data pane for hierarchies

The data display in in the data pane may be an indented data list if the corresponding tree selection is a node of a tree. If the -tree parameter is specified on the browser command which generates the tree in the tree panel, selecting an element of the tree shows the entire sub-tree in the data pane, rather than just the members of the element (default). The "browserindent" system parameter determines how many blanks are used to indent successive tree levels.

3.17 Advanced GUI for Linux and Solaris 10

A new Unix/Linux user interface, very closely resembling the Windows user interface, is made available using the technology known as "wxWidgets". The interface is available for Linux platforms in the 7.0 release and will also be made available for Solaris 10 clients and subsequently other clients in the coming months. Note that the Unix and Linux GUI functionality continues to evolve.

3.18 Sysparm notewrap used for window displays in note drill downs

When note displays are opened as a result of a drill-down from the title, the width of the note field popup is now determined by the setting of the "notewrap" system parameter.

3.19 Range Color Enabled By Default, Controlled by Sysparm

For graphs, charts, and anything else that takes a -rangecolor option, the system parameter, "rangecolor", indicates the default value for that option. If rangecolor is off, then the option can be turned on through the command. If rangecolor is on, then -rangecolor is implicitly specified.

3.20 Use of Color in Text Panels/Prompts

Both text redirection and note prompts allow for text to be displayed in a pure text format. Often it is useful to highlight text. For example in a delta report, or in a search. In CM+ the macro text$color is used to specify the text highlight colors, and optionally, the text font colors, to be used. If text$color is defined, it is of the format:

"pattern" (color triplet) [(color triplet)] "pattern" (color triplet) [(color triplet)] ...

So for example, to highlight the NEW and OLD lines of a delta report:

```
macro text$color "OLD" (255 200 200) "NEW" (200 255 200)
```

would provide for red highlights on old lines and green highlights on new lines, when using the oldnew format. The first pattern match indicates the color to use for a given line. The matched text is highlighted using this color and the entire line is highlighted using half this color (i.e. for each RGB entry, add 1/2 the difference between 255 and the entry. In our example, (255 228 228) is used for OLD lines. When a pattern occurs more than once in a

line, all occurances are highlighted.

In CM+ 7.0, the default definition for text$color is found in the file macrodef.gui:

```
    macro text$color @deltatext$color @findtext1$color @findtext2$color @findtext3$color
    macro findtext1$color @choice "@find$1" "" "" '"@find$1" (240 240 110)'
    macro findtext2$color @choice "@find$2" "" "" '"@find$2" (240 110 240)'
    macro findtext3$color @choice "@find$3" "" "" '"@find$3" (110 240 240)'
    macro deltatext$color "OLD" (240 110 110) "NEW" (110 240 110) "   CHG" (240 110 110)
"___CHG" (240 110 110) "CHG" (110 240 110) "DEL" (240 110 110) "ADD" (110 240 110)
```

This definition makes it easy to specifiy up to 3 search terms, by defining macro find$1, find$2 or find$3. It also sets up a default highlighting for the delta command, using either adddel or oldnew formats. Finally, the text$color macro can be easily modified to include or exclude any of these components.

3.21 Locating Highlighted Text

When text is highlighted in a text window, in the Main window "Log" pane, or in a Note field, it is a common action to look for the next highlighted text. In a text window, Find and Previous buttons are added to find the next or previous highlighted text. On the Main window pane, two magnifying glass icons are used to indicate that the next or previous highlighted text is to be located. For both these cases, as well as for note fields, the F3 (and Shift F3) function key may also be used for the same purpose. Whenever a find request is initiated, the cursor moves to the next highlighted line following at least one (possibly the current) line that is not highlighted. The cursor is always moved to the first character on that line.

3.22 SubClass Identification on Forms

When a table's record has a well known "subclass" field, and the system parameter "formlabels" is enabled, the form label adds the subclass of the form to the form label, if known. For modify and form commands, the label is picked up from the record field. For add/input commands, if the optional [-class String] parameter is specified and String matches one of the values allowed by the "subclass" field, the subclass is correctly defaulted on the form and will appear in the label. For example:

```
    add acts -class ecr
```

would yield a label: ECR ACTIVITY instead of just ACTIVITY.

3.23 Numeric Fields Prompt Navigation

When a numeric field is displayed, the up and down arrows may be used to cycle through the values. If the "numericprompt" system parameter is enabled, or if the "+" is used in the numeric prompt (e.g. ?<Number>+Size), up and down arrow buttons are shown for incrementing and decrementing the field value, according to the native widget and window manager conventions.

Clicking on the up or down arrows of the control increments or decrements the value. Holding it down performs an accelerated operation. Likewise clicking/holding down the Up/Down Arrow keys perform the same way. Releasing the button/key causes the field to update any dependent prompts based upon its value at the time of release.

3.24 Display Item Selection

The Item Selection capability is used to select a number of items for applying a single operation to them. This capability, already present in the data pane of the main window, is extended to display panels. A checkbox appears beside the key of each item of a display panel whenever the "-checkbox Name" parameter is specified on the display command, or whenever

the "checkbox" system parameter is enabled. In the former case, items are selected or deselected by clicking the checkbox. The resulting selection is held in a variable with the name "Name". When the checkbox is inferred by the system parameter "checkbox", rather than the explicit -checkbox Name option, a variable is created by the name <table>$selected, to hold the set of checked records. In either case, the variable can also be used to predefine the set of checkboxes by setting it's initial value prior to executing the display command.

Along with the checkboxes, there are two related capabilities. An "Invert Selection" button appears just in front of the "Refresh" button at the bottom of the display. This is used to invert the current selection so that unselected items are now selected, and vice versa.

As well, whenever ^0 is used in a button definition, it is replaced by the list of selected keys. This allows buttons to be defined which take advantage of the selection.

3.25 Selecting Tree Pick List Selects that Browser Tree

When the pick list is used to make a browser tree visible, that browser tree is also selected. Prior to CM+ 7, the tree panel was adjusted so that the entry was visible. Now it is also selected and the data pane reflects the selection.

3.26 Navigation Control For Dates

Navigation Control is available to support Dates. The up and down arrow keys may be used to navigate the dates. The navigation control is enabled by specifing the "+" indicator and optionally an NxM format. The basic syntax is:

?<Date>NxM+[@date]Date

The N, if specified, represents the width of the date box, while xM, if specified, represents the increment value M as follows:

M not specified: 1 day
M less than 32: M days
M=7: 1 week
M=32: 1 month (i.e only the month is incremented, and year if Dec)
M=96, M=128: 1 quarter (i.e. 3 months, only the month is incremented, and perhaps year)
M=256: 2 quarters (6 months)
M=365, M=366, M=512: 1 year (12 months, only the year is incremented)

Any other value is treated as M days. So, for example, to have a date field which increments by week, you may specify:

?<Date>10x7+[@date]Date

When incrementing annually from a leap day, the year changes and the day is decremented to Feb 28. Use of the spin control buttons or the up/down arrows result in the increments. Note that if the date is specified as YY/MM/-- and monthly or higher increments are indicated. the -- is kept for the day field, similarly with YY/--/-- for annual increments.

3.27 Compound (Complex) Widgets in Forms

When a field such as WeekTimes or some other record or table field of a tuple is shown in a Prompt Dialog, by default, the individual portions of the field (record fields or table elements) are individual sub-widgets. By default, starting in 7.0, this applies to forms as well as dialogs. A Field Attribute "forcetext" may be specified to force the field to be treated as a single text field for all of the value components, as in pre-CM+ 7.0, rather than as individual widgets. The set of sub-widgets are treated as a set of blank separated values.

4. Customization

4.1 Adding buttons to text displays

Whenever a text panel is displayed, it may have additional buttons before the Close button. On clicking the additional button, a text file is generated and the name of the file is passed to the button definition as an argument. The buttons appear on a text panel if the macro below is defined:

macro text$buttons <Button> <QuotedCmd> <Button>...

4.2 Button macros for automatic button definitions on various panel types

The form, add/input, modify, and display commands all take an option -button clause argument. When not specified, the button macro for the command is used, if defined, to add additional buttons to the panel. "add$buttons" is the macro used for both add and input commands. All other commands, including the "table" command, have their own buttons defined. In the case of the table command, as no prior button parameters exist, (the command syntax in is not altered by this activity), support will be needed for buttons which will use the command's parameters as arguments for the button definitions. All other buttons have their standard button arguments.

4.3 Button definitions for Tabbed main display panes

With the introduction of a tab control, as a means of adding tabs instead of popping up new windows for various displays, a means of adding customized command buttons to the Command Bar area is enabled. The Command Bar buttons are modified based on the type of tab shown. Initially, there are 2 types of tabs supporting this feature:

Text Tabs - controlled by text$buttons macro
HTML Tabs - controlled by the html$buttons macro

When a button is selected, the content of the tab window is written to a file (same temporary file name as for a text or HTML stand-alone window), and the file name (full path) is passed in to the command defined for the button.

4.4 stsgui command to reset popup menus

The stsgui command is used to reset popup menus so that any changes to such menus may be picked up in the current session. The syntax is:

```
stsgui -popups
```

The stsgui command is used to remove one or more open tabs:

```
stsgui -tabs WildcardString
```

Any tab names which match the Wildcard String are removed from the display. The stsgui without any arguments is not supported.

4.5 Select all and select all but in prompts

When a prompt is specified that is for multiple selections, the appearance of a * as a single token in the default string should invert the current selections. So,

```
?{a b c}*[a b *]Pick
```

would select everything except "a" and "b". Note that

```
?{a b c}*[a b c b]Pick
```

selects only a and c, as repeating an element toggles its setting. Hence,

```
?{a b c}*[* b]Pick
```

is equivalent, selecting everything (because it was all deselected) and then de-selecting b.

4.6 Auto-generation of test commands for GUI testcase generation

To test the CM+ GUI, or any other configuration of an STS application, is a huge job. Hundreds of buttons, dozens of permutations for each. The test command was created so that the GUI operations could be tested through a command rather than through keystrokes. However, the generation of these commands in themselves is a huge task. The STS engine makes this task easier by allowing a mode whereby it can generate "test" commands as the GUI is being manually tested. The "testmode" system parameter is used for this purpose. When enabled, the file "Testlog.sts", in the user's STSHOME directory, is used to accumulate test commands reflecting the user interface actions. This file can be "read" (i.e. replayed) to simulate the test.

4.7 Macro parm list display

The parameter names, if specified for a macro command, are saved in the current session data, and when help is shown for a macro, (e.g. using the macro command), the parameter names are displayed. As well, the help command is upgraded so that when help is issued on a macro, it not only identifies it as a macro, but display the macro definition in the same way as the macro command does.

4.8 Debug macro and script capability

The debug macro capability is enabled by the debugmacro system parameter. When set (off, by default), debugmacro causes macro evaluation requests to be displayed as:

```
  @> macro(parameters) 'parm 1' 'parm 2' etc.
```

For example, the command note @settable 'acts take 2' would result in:

```
  @> settable(set) 'acts take 2'
```

assuming the parameter list for settable is defined as "(set)". Note that the number of paramters shown is based on the parmcount of the macro, not on the presence or absence of parameter values. When verbose is also enable, the normal @: expansion of the macro would follow the above output. Similarly, when a "debugscript" system parameter is set, any script invoked by CM+ would produce the output line:

```
  $> Script 'parameter 1' 'parameter 2' ...
```

Trigger and Rule scripts will have the same behaviour, but only if triggerecho is also enabled. In that case, the script name will be the key of the trigger along with the field name:

```
  $> c.borrow.precmd 'parameter 1' 'parameter 2' ...
```

For scripts, empty parameters are shown (as '') only if non-empty parameters appear after the empty parameters.

4.9 Configurable main panel frame - allows longer tree panel

By default, the main panel is split upper/lower, and then left/right for the upper pane. The system parameter "verticalsplit" which is by default off, can be set to enable the major pane split to be vertical instead of horizontal. This enables a longer tree panel and less tab space. The main display reacts to this system parameter only at the start of the GUI session.

4.10 Context-based Field and Field List Specifications

When a command is issued which takes the root set of a context, it is sometimes desirable to specify the fields parameter as relative to the current context rather than the root set. For example, in the browser command, the ExtFieldList parameter
cannot be specified for a context tree because the Set specified is typically a set of Files, while the fields to be shown are FileIssue fields. To allow specification of File Issue fields when a File set is specified, the field list begins with a (= rather than simply a (. So (type revs title) would be a valid File field specification, while (= status

change suffix) would be a valid File Issue field spec, even when a File set is specified in the browser command. The closing bracket for the field list may optionally be preceded by the "=" character as well.

Another example, in the browser command, the -field parameter cannot be specified for a context tree because the Set specified is typically a set of Files, while the sons field to be used is a FileIssue field. To allow specification of File Issue fields when a File set is specified, the field is prefaced with an =. So "-field members" would be a valid File Issue field spec, even when a File set is specified in the browser command.

4.11 Separators in Browser Data List

The Listview command allows the insertion of separators into a data list pane associated with a browser. This is done using the optional argument: -sep ExtFieldList. Each field list specified is preceeded by a standard separator which logically divides the data list panel into visual groupings.

The current default values for the separators are designed to be the width of a scroll bar, by default and the colour of the header background color, typically an RGB of (233, 236, 216). The current default for indenting the levels when the -tree option is used on a browser is increased by 50% to 70% (i.e. add half the blanks again, and round up).

System parameters are used to control the customization of these values.
- sepcolor: RGB used to specify the separator color
- sepwidth: Pixel width of the separator
- browserindent: Number of blanks used to indent one level of a tree in the listview

4.12 Display scale system parameter

On some Windows Vista platforms, especially those with widescreen displays, the display size reported by Windows is incorrect under certain circumstances. This can lead to improper formats for dialogs, graphs, etc. The "displayscale" system parameter is introduced to allow some compensation for the scaling problem. The scaling is specified as a percentage.

4.13 Additional Text Buttons on Text Displays

Whenever a text panel is displayed, it may have additional buttons before the Close button. On clicking the additional button, a text file is generated in STSHOME and the name of the file is passed to the button definition as an argument. The buttons appear on a text panel if the macro below is defined:

```
macro text$buttons <Button> <QuotedCmd> <Button>...
```

The text file generated is textbuttondata.txt and is passed as parameter @1 to the Button command.

Similarly, for HTML panels, a macro may be defined:

```
macro html$buttons <Button> <QuotedCmd> <Button>...
```

The text file generated is htmlbuttondata.html and is passed as parameter @1 to the Button command.

4.14 Command Output Used To Generate Note Prompt Content

Normally, for a note prompt, the initial content of the note is filled using the contents of a file, which name is specified in the default field:

```
note ?<>[tmp.fil]OK
```

causes the OK note field to be populated with the contents of tmp.fil. The note result from the prompt is written back to "tmp.fil". At times, it may be desirable to have the note field content determined dynamically, as, for

example, with a dynamic prompt. For example, an update may be selected in one widget with the delta report generated into the notes widget. As you cycle through updates, you would see different delta reports in the note field.

When the default field contains a leading ">", the content for the note field is taken as he output of a command rather than from the contents of a file, and the note widget becomes "read-only" (in that, the result is not exported to a file), and takes on as its content the standard output of the specified command, or command sequence, which follows the ">". For example,

```
note ?/updates status <= ready/(title)Update ?<>[>delta mods ?Update]Delta
```

would generate a delta report area which would (dynamically) be filled based on the content of the controlling Update widget selection. This is primarily useful in generating interactive prompt displays where a dynamically generated text result is part of the display.

4.15 Trimming of Empty Elements in Browser Data Tracing Tree

When the browser -fields option is used, CM+ allows you to trace through all data associated with a record by expanding the list of traceable fields, and then the contents of those fields. However, quite often the fields are empty, as indicated by a [0] (at least for lists). When there is no data in the field, and the -trim option is specified on the browser, empty fields are not shown.

4.16 Force Selection Option on Set List and other List Widgets

When a set list prompt includes a ! character: ?/users/!*User the user is forced to select a value in the list (or have ome default). If the resulting list set is zero elements, the OK button is disabled and does not close the form, much as when a ! is used to force text in a text widget. This applies whether the list widget is presented as a selectable scrolled list of items, or as a checklist. It forces the one or more selection instead of 0 or more. It applies to all list widgets.

4.17 Text Line Selection and Popup Menus in Text Tabs

Various mechanisms in CM+ allow text information to be displayed. When output redirection to a text tab is performed, it is possible to highlight a line of text (or any text) and right-click to reveal a popup menu which takes the highlighted text as an argument (@2). @1 is always the content of the entire line that was right clicked.

When a popup file is specified between '<' and '>' immediately after the '[' of the redirection tab identifier, that popup menu is attached to the resulting tab window and invoked whenever a user right clicks in that window. For example:

```
show users > [<umenu.pop>"Users"]
```

will create a "Users" tab and will use the umenu.pop file to define the right-click popup menu for the Users tab.

4.18 Secondary Widget Sizing in Prompts

In a dialog prompt, the construct NxM can be used instead of simply N in situations where a second widget size needs to be specified. In particular:

```
?<>5x120[]Note  - Note prompt using 5 lines by 120 characters
?/probs/6x8(status priority)#Table   - Table prompt 6 rows by 8 columns
```

The NxM construct is valid for all prompts, but will not necessarily have semantic meaning for all prompts.

4.19 Font Size System Parameters for Note Fields

When a note field is specified in a prompt, it may be either fixed or variable length fonts. To specify the size of the font to be used, system parameters 'fixedfontsize' and 'varfontsize' are used. These specify the font sizes, in points, for fixed and variable fonts respectively for use within 'note' fields.

4.20 Boolean Values in -gray Option

In popup menus and pull down menu items, a parameter is used to dynamically evaluate whether or not to show the menu item. As the @calc macro returns a 0 or 1 for boolean calculations, the semantics of this parameter is expanded to check for a value of 1 (if set scanning fails). A value of 1 indicates that the menu item is to be shown (i.e. not greyed out or omitted). So the logic is:

First evaluate macros in the parameters
Scan the parameter for a set. If found, a non-empty set is success.
An empty set is treated as failure (i.e. do not show the menu item)
If the set scanning itself fails, scan for a number.
If found, a value of 1 is success, Any other value is treated as failure.

4.21 N-Up Prompting - beta

CM+ allows prompting of multiple values in a single prompt, all of the same type. The use of an N-Up value after the ? indicates that the prompt is to be repeated N times:

?*3<Stream>Stream

produces a list of 3 Stream prompts (i.e. 3 rows). Results are appended to a single result string. Typically % is used to enclose the results in quotes for each prompt:

?*3<Stream>%Stream.

Use of ?*3<Stream>Stream[2] would return only the 2nd value, ?Stream[1] and ?Stream[2] could be referenced elsewhere in the command string. The label appears only for the first row of the prompt. The remaining rows follow closely beneath the row above, with only a thin separator.

This functionality is of limited use in its initial beta format, but will be extended in a future release.

4.22 Extended Support For Key Format Displays

The display panel can display the record key in various formats by indicating the format as part of the key name. For example: display users (uniquekey title)
The key format, if specified, must be part of the first field.

4.23 GUI and Macro Specifications/Documents

New Guides have been written covering GUI customization, and Macros. These are the initial versions of these documents. The GUI Guide is especially meant for those who will be doing GUI Customization work. It details all of the various capabilities of the GUI customization language and syntax. The Macros Guide is intended primarily, at this point, for customization work. In particular, it identifies special macros recognized by CM+ as well as a number of built-in variables that have special meaning.

4.24 Specifier for Wrapping of Note Fields in Prompt Dialogs

When a note field is displayed in a prompt dialog, using the ?<>[file]Note construct, the lines of the displayed file are not wrapped in the note field. Instead, a + sign must be specified as the first character of the default value to indicate wrapping is to occur: ?<>[+file]Note. This syntax is similar to the output redirection syntax.

4.25 Replacement of Menu Items

When a menuitem command is executed for an existing menu item, the new menu item definition replaces the previous one, unless a -noreplace option is specified (which forces a second copy of the menu item). Prior to CM+ 7, -noreplace was the default behavior. If an addmenu or menucascade is issued for an existing menu, the command is ignored (no error or warning message is given). This allows revision of and re-reading of .gui files to reflect the revisions. Prior to CM+ 7, exiting the library was necessary to pick up revisions to a .gui file.

4.26 Multiple Field Prompt Modifiers

In release 6.x, it was possible to modify a field prompt modifier is various ways by adding a modifier immediately after the table name:
e.g. ?(probs-{mask hide})Field
In CM+ 7 and later releases, multiple modifiers may be specified in sequence:
e.g. ?(probs-{mask hide}:class data:class list)

4.27 Using Table and Graph Widgets as a Master Widget

When a table or graph widget (i.e. ?/set/#Table, ?/set/|Table, etc.) is used as a master widget of a dialog panel, it causes a re-evaluation of dependent widgets. The dependent widget references the master widget in a place where a valid set expression may be used.

The dependent widget will generally specify the ?Table as a master only within a set specifier (i.e. ?/?Table/...) . A dependent widget is re-evaluated whenever a cell of the table is clicked or when an interactive portion of the graph is clicked (e.g. a pie slice or a bar). The reference is replaced with the expression the clicked cell/area represents. So for "table changes @user -field stream -by status", if the rel3:open cell is clicked, the reference is replaced by "changes @user stream rel3 status open". This is a dynmaically generated set expression. Similarly, for "table updates by stream" (one field only), if the rel3 cell is clicked, the reference is replaced by "updates stream rel3". Similar behavior applies to graphs, when used as a master widget. This effect may be cascaded. For example, a pie chart may show updates by stream within a dialog panel. A second pie chart widget might show the updates in the selected slice by status, while a third might show the updates in a slice of the second, by class.

For a table as a master widget, when a total field is clicked, the set expression generated reflects the set elements used to compute that total field.

4.28 Negation of Qualifier for Menu Buttons

Sometimes a condition is most easily expressed as the negation of another condition. In a qualifier, this can simplify coding:
change @1 status not open or (change @1 status open user not @user)
is more easily codified, for a qualification, as: !change status open user @user
The "!" character at the beginning of a qualifier, reverses the sense of the qualifier. This applies to both popup menu buttons and the -gray option of the menuitem (etc.) commands.

4.29 GUI Prompts Component Macros

Macros allow reuse of functionality to simplify scripts and commands. For GUI Prompts, macros can also be used. However, normally GUI prompts are executed before macro evaluation. As a result, delayed GUI prompts are needed to use these macros (@Q instead of ?). But this can have other undesireable side effects.

When the "?#<macro>" construct is used (e.g. ?#Stream), prompt time substitution is performed on the macro, to substitute it into the command line prior to prompt evaluation.

The ?#<macro> behaves differently from the @<macro> substitution. @macro substitution recursively substitutes macros until there are none left to substitute. ?#macro substitution does no recursive substitution. However, parameters (@0 to @N) are substituted using the arguments of the ?#macro.

The normal use of ?#macros would be to reuse a common prompt:

```
macro Product ?/products rootnode not NIL/[@cur product]Product
macro Stream ?<Stream>[@cur stream]Stream
```

could be defined as the normal stream prompt and then referenced as:

```
setcontext ?#Product ?#Stream
```

which would become:

```
setcontext ?/products rootnode not NIL/[@cur product]Product ?<Stream>[@cur stream]Stream
```

in a command line.

Note that the ?# substitution is done prior to the prompt substitution. ?# substitution is repeated whenever normal prompt substitution is to be attempted, and is completed before the prompt substitution.

The use of Prompt components in this way helps to ensure consistent prompts for the same information in various commands. It also makes the command lines much more readable (as in the before and after cases above). If the macro Stream is not defined in the above example, the result would be the empty string substitution, just as for normal macro substitution. Although component macro substitution is primarily a prompt simplification mechanism, there is no reason that is cannot be used for non-prompt related substitutions.

4.30 Setting Enter Key Focus for Panel Buttons

When a panel has user-defined buttons added to it, the button name is specified via the

```
?'title'[<button>command]
```

construct. When a ! follows the ">", the button also closes the panel on execution. If a * follows the ">", either before, after or instead of the "!", the Enter Key focus is placed on that button instead of the OK button. This allows the button ?'title'[<Refresh>*] to be defined so that the panel is updated when the enter key is used. Similarly, other non-null actions can be defined for use when the enter key is pressed. This applies to both dialog panels and forms, when buttons are added.

4.31 Optional AccessProt Parameter on Browser Command

The browser command supports an optional AccessProt command as its first parameter. When specified, the browser command is ignored if the current user permissions do not have a match with the specified Access Protection.

```
browser {cmmgr prjmgr} ....
```

will only be executed if the current user has either or both of the cmmgr/prjmgr roles.

4.32 Automatic Panel Titles From Form and Menu Buttons

In specifying a panel, it is necessary to specify the panel title as well. CM+ can do this automatically when no specification is otherwise present. This eliminates work in defining the application and ensures that default behavior is acceptable in a generated application. The panel title is taken from the form or panel button, if a button is selected to bring up the form. If a context menu is used, the panel title is a combination of the context menu button title and the first parameter of the menu definition, that is the object key itself.

4.33 Stripping Prefix from Prompt Entries in a Set List Prompt

When a Set list is presented to the user, it is sometimes the case that the prefix on the key is better omitted. For example, if the set being presented is a list of sysparms (sp.<key>), commands (c.<com>) or data types (dt.<type>), it is sometimes desireable to display a list of these items without the prefix. This is done by specifying the "\" option modifier on the set. For

example,
  note ?/sysparms take 10/*Sysparms
The result of the prompt is the set items, with their prefix values, but the widget display omits them. This option can be applied to both lists and non-lists.

4.34 Field Selection on Data Type Prompts

In order to specify a subset of schema data on a form, whenever a data type prompt evaluates to a record, the datatype may be followed by a field list, in much the same way as permitted for a set prompt. So, for example,
  ?<FieldData>(class wkfield title:64)Data
  ?*4<Problem>(priority assignee owner)Data
Note that this works with multi-prompt fields. Also note that if a field width is specified on the field name, the width for that field is used for the widget prompt (count field, generally the width of the widget).

4.35 Default Specification on Group Multiple Prompts

When a multiple modifier is specified on a group prompt, the default value is normally applied to each element of the group:
    note ?*4<FieldData>[3 probs priority]Test
In the above, "3 probs priority" is specified as the default for each of the 4 entries in the group. When a default prompt for a group is grouped by []'s, each sub-default is applied to succeeding records, with wrap around of the sub-defaults to fill all needed records. So in the following:
    note ?*4<FieldData>[[3 probs priority][2 acts stream]]Test2
The first set of default values would be applied to elements [0] and [2] and the second set of defaults would be applied to elements [1] and [3] of the groupings. This allows a group prompt with a potentially different set of defaults for each entry.

To support a simpler case, where perhaps the multiple is a single data element (but also, for potential use with records), the interior []'s may be omitted. In the example above, this would not work since only a portion of the default is specified. However all of the following would be valid:
    ?*3<Date>[[9/9/11][9/10/22]]Dates
    ?*3<Date>[9/9/11 9/10/22]Dates
    ?*3<Color>[255 0 0 255 255 0 0 255 255]Colors
    ?*3<Color>[[255 0 0] [255 255 0] [0 255 255]]Colors
    ?*5<Color>[[255 0 0] [255 255 0] 255 255 255]Colors
In the latter case, the last set of values, not enclosed in []'s would be used to repeat the defaults for the last 3 of the 5 Colors. Results can be referenced as follows:
    ?Dates - all the elements
    ?Dates[1] - the second element (0-origin)

5. Process Support

5.1 State Flow Diagram Improvements

State flow diagrams have had a number of improvements made to them to permit better diagrams.

The shape of the state boxes are, by default on Windows, more square, similar to the Unix/Linux defaults. States are grouped more tightly when there are a larger number of states that need to fit a diagram. The following system parameters control these and other properties:

stateheight	The height, as a percent of diagram, of process state bubbles
statehsep	The horizontal separation between process states as % of diagram
statestyle	The square/oval factor of process state bubbles: 0 for square
statevsep	The vertical separation between process states as a % of diagram

statewidth	The width, as a percent of diagram, of process state bubbles

Note that the "chartbgcolor" and "chartfgcolor" system parameters may also be used to configure colors used in the state flow diagrams.

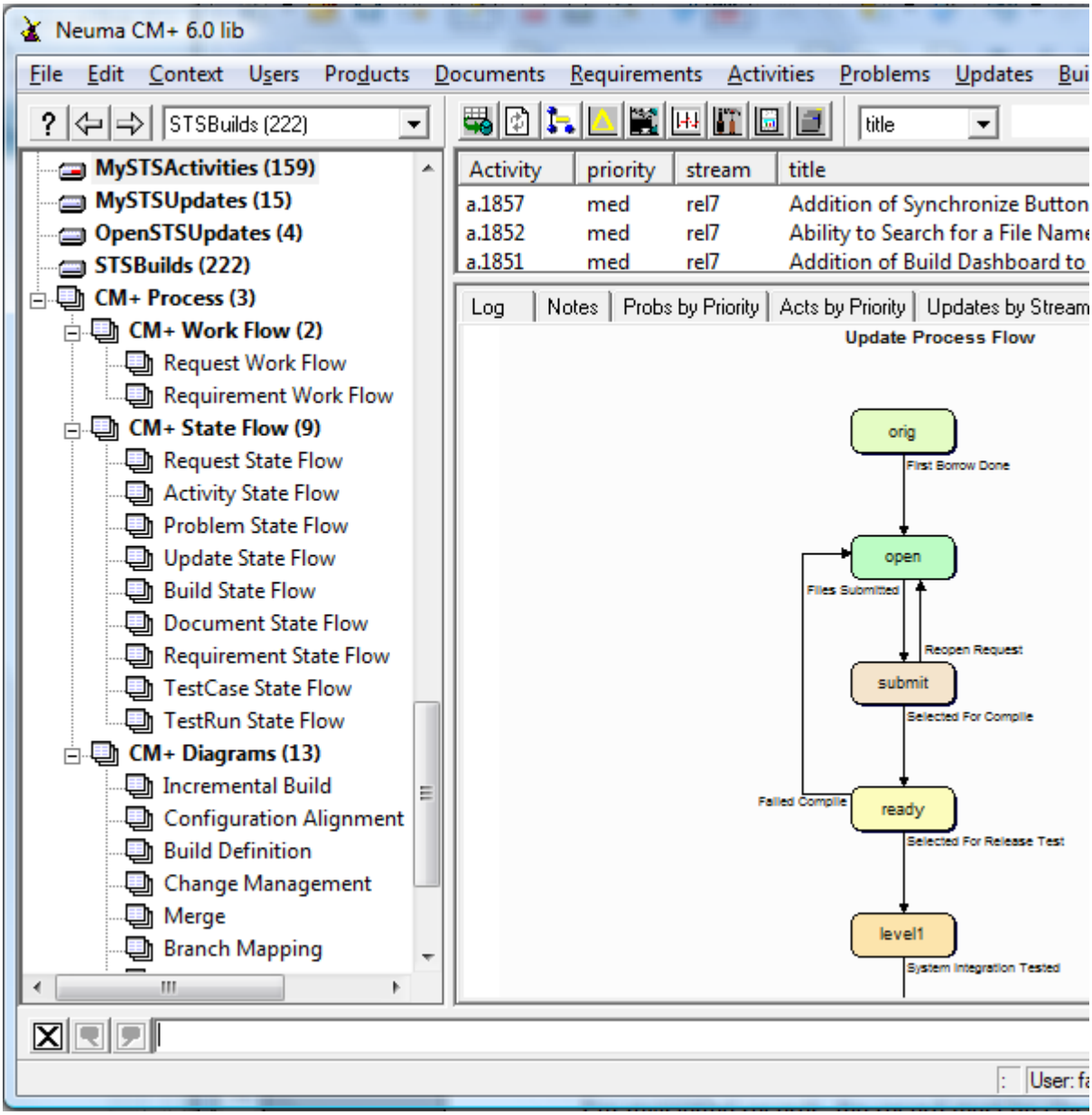
5.2 Process flows, gantt charts, and graphs will track state colors

Gantt charts can make use of the the status range colors used to define the chart. When -rangecolor is specified, each segment of the actual gantt bar is color coded with the range color assigned to the status at the begining of the segment. If there is no range color assigned, the default actual bar color is used for that segment.

The optional parameter -rangecolor on the process command instructs the process drawing to appear with the colors of the state bubbles corresponding to the colors specified for that range element, if any. If there is no color specified for that range element, the color appears as if -rangecolor were not specified for that state.

5.3 Process flow in tabs

CM+ now supports process flow diagrams displayed in tabs. These are fully interactive displays and will generally be accessed through the State Flows section of the Process Help tree, although they continue to be accessible through menus for display in separate panels.



5.4 On-line process help tree

CM+ now supports a process tree in the tree browser that can be used as guidance in using both CM+ and the project methodology. The process tree contents are easily customized. The initial process tree contains a subset of CM+ process, and will be expanded over time by Neuma. The default configuration is specified in the "gui" file "cmbrowsers.gui".

5.5 Electronic Authorization

CM+ allows the tracking of Electronic Signatures against a record. Records which allow electronic signature must have a 'signature' well-known field. This field is a "list users" field. The command level syntax for signing is:

sign [-remove] <Set>

This allows a set of objects to be signed with a single signature. A user cannot sign a record unless the user has

a non-empty password specified for that user. If so, the user is prompted for a the user password, and if confirmed successfully, the user's name is added to the signature field of the records specified by <Set>. If a "reply" field is associated with the record, a time/date/user stamp is appended to the record:

-------jones ----------07/07/12 12:16:22-------- Signed

If -remove is specified and the user has already signed the record(s), the signature is removed (i.e from the signature field) and a time/date/user stamp appended to the reply field of the record:

-------jones ----------07/07/12 12:16:22-------- Signature Removed

For the -remove option to work, the system paramter "unsign" must be enabled. Otherwise the user is given an error message: Signatures may not be removed from records.

"signature" fields are expected to have both "prot" and "mask" attributes as well as the "ro" attribute enabled.

For revisioned records, the record must be closed (or in "submit" state and it will be moved to closed) in order to sign the record. The signature applies to the revision.

The system parameter "batchsignsize", which defaults to 100, is used to indicate the maximum number of records that can be signed in any sign command. Exceeding this amount will cause no records to be signed, and an error message displayed:

Signature Batch Record Count Exceeded: <batchsignsize>

5.6 Logging Action As Part of Transition Logging

When a state transition fires, if "lognote" is enabled on the transition, a note is appended to the "reply" field of the object, identifying the transition:

Previously: ====> 09/09/03 11:15:40 qa1 : status changed from: Defined to: SCCB

Now: ====> 09/09/03 11:15:40 qa1 [Send to SCCB] : status changed from: Defined to: SCCB

The Action Title, as defined for the transition, is included in the message as shown above.

5.7 Automatic Logging in Short Reply Format

A system parameter "shortreply", default value 40, is introduced which identifies the maximum amount of text in a reply -text string that will cause the reply to remain on the same line as the time stamp. When the textline is within this limit, the log line looks like this:

-------qa1 ----------09/09/03 11:17:44 - Recommend CR is Analyzed

Otherwise, it looks like this:

-------qa1 ----------09/09/03 11:17:44--------
Recommend Change Request CR.115 Be Further Analyzed

When shortreply is 0, the two-line logging is always used. When shortreply is a large number, one-line logging is always used.

5.8 Automated Application Map Generation (beta)

Documentation generation now includes the generation of an Application Map. Each application map contains a number of tables as well as links between appropriate tables. The command appmap is used to generate these drawings:

appmap <Table Set> [-to <Table Set>] [-from <Table Set>] [-in <TableSet>] [-links <FieldSet>] [-lists] [-refs] [-subtables [Orientation]] [-layout Number]

Each table is represented by a rounded box. Sub-tables are indicated by a set of overlapped boxes if -subtables is specified. If Orientation is specified, subtables will appear to the specified direction from their parent, with the best layout being the default.

Links are shown using an arrowed line between boxes, with a double arrow for list links, and a single arrow for reference

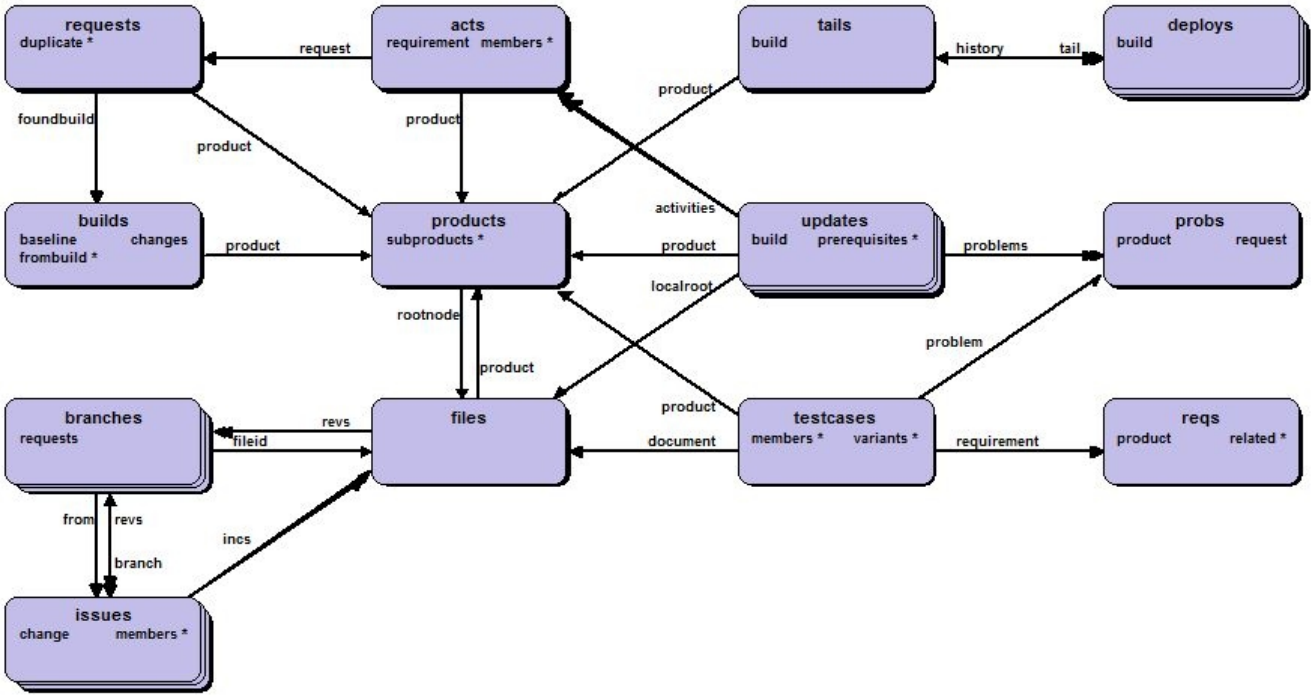
links. Links are identified by the field name as a text label for the line, positioned close to the box from which the link proceeds. If a "vlist" field is present, it is not drawn as a separate link. However, the vlist field name is used as a label on the other end of the inverse link that has been drawn, and arrows are added in both directions. For subtable links (parent field and subtable field), a single link line is drawn with the appropriate field name at each end. A single arrow appears on the parent side, and a double arrow on the subtable side.

Hierarchical links (i.e. links to within a table) are drawn as a 3/4 arc with arrows at one of the unused corners of the table box, and labeled on the outside of the arc. All of the tables in the Table Set are illustrated on the application map. By default, all links are also shown. However, whenever the -to option is specified, only links to the specified tables are shown.

Similarly, when -from is specified, only links from the specified tables are shown. When -links is specified, only the specific set of links, as indicated in the Field Set, are shown. When -lists is specified, only list links are shown. When -refs is specified, only reference links are specified. The parent and subtable field links are always shown for subtables.

When the -in option is specified, links to tables in the specified Table Set appear as field names listed within the "from" table box, without any lines drawn. In particular, if the set specified includes all of the tables in the appmap table set, no links are drawn, but link fields are listed within the table boxes. The table name should be prominent with link field names listed in one or two columns. The layout option is used to force a layout with 1, 2 or 4 tables in the center. (Default is 1 for 9 or less, 2 for 16 or less and 4 otherwise). Only links between adjacent boxes are drawn (up/down, left/right, diagonal). The other links are drawn outward and a "<table>" label inside a compact box is drawn to indicate where it terminates.

A number of system parameters are used to help control layout: mapboxheight, mapboxwidth, maphorizontal, mapvertical. These are used to size boxes and spacing between layers.



5.9 Schema Documentation Generation

To help generate application documentation, data schema charts can be created for each table. This is done through the schema command when the -display option is specified. With reportmode set to "html", the schema charts are generated in HTML. Otherwise, they are generated as a graphical object.

Top Box:

Module Name: mmmm | Key Type: kkkkkk | Record Type: rrrrrrrrrr

Heading Box:

Field Name | Type | Size | Property | Class | Table/Field | Options | Width | Title

Subsequent Rows:

field name | data type | size (bytes) | wkfield | class | [table] | HPMLAXR | width | title

HPMLAXR are the letters present if Hide Prot Mask Lognotes Auto Xsn Ro options are set. The first letter is shown for each set option. The Heading box is shaded.

TABLE: probs	KEY: ProblemId	RECORD: Problem						
Field Name	Type	Size	Property	Class	Table.Field	Options	Width	Title
type	ProbType	1	subclass	data				Type of object with problem (doc, sw, etc)
status	ProbStatus	1	status	data		M X		Indicates action to be taken and by whom
date	Date	2	statdate	data		A R		Date of last status change
owner	UserId	2	owner	ref	#users	A		Person responsible for problem correction
assignee	UserId	2	subowner	ref	#users	A		Person assigned to fix the problem
priority	Priority	1		data				Urgency indicates timeframe required for fix
product	ProductId	2	product	ref	#products	A		Product under which problem was discovered
stream	Stream	1	stream	data		A		Product stream to which problem relates
request	RequestId	4		ref	#requests			"Request that Spawned this Problem"
title	Textline	4	title	textline				Brief, Unique description of the problem
reviewers	List	4		list	#users			Users responsible for reviewing work
originator	UserId	2		ref	#users	A		Person who initiated the problem report
from	UserId	2		ref	#users	H M		Previous owner of the problem
location	Textline	4		textline			32	Brief description of problem location
category	ProbCategory	1		data				Categorization of problem resolution
effort	Shortnum	1		data				Effort taken to resolve problem
forecast	Date	2		data				Latest forecast fix date
dates	StatusDates	16	statdates	appltab		M R		Record of dates each status was achieved
attachment	List	4		list	#attachments	HPM		Attachments to this problem
notes	Note	4	reply	note				Problem description and all (datestamped) replies

6. Optional Add-ons

6.1 Web GUI for CM+

The Web GUI for CM+, CM+Web, an optional CM+ add-on, is a fully customizable Web Interface for using CM+ for most operations. The user interface is very similar to the native client interface, including tree browsers and a status bar. Menus may be defined in much the same way as they are defined in the native GUI. For normal Data Management aspects of the CM+ suite, the Web GUI is adequate to provide remote operation given only the use of a browser. This includes the ability to navigate data, raise problem reports or add other data records, modify existing data, and perform configuration manager functions. For version control, there are a number of limitations that make the native GUI the preferred option.

The Web GUI interface definition may be role-based or user-based. In this way, customers may have access to your repository for raising requests and viewing their progress, while restricting their ability to see any engineering data. Although CM+Web users do not require a separate installation of any CM+ software, use of CM+Web still consumes a client license.

The current release of CM+Web is primarily a CM+Web toolkit, with only a portion of the CM+ menu customization completed by Neuma. However, menu customization, including dialog prompts, work much the same way as for the native user interface. CM+Web requires a server counterpart. At this point in time, an Apache Web Server running on Linux, is the only certified server for CM+Web, although other servers may work as well. The web server may reside on a virtual machine and requires very little configuration.

6.2 DSO (Directory Search Order) Capability (release delayed until 2009)

The DSO add-on is an optional, Windows-only add-on to CM+ which allows you to collect a set of directories and view them as a single directory, as if the set of directories were stacked one on top of another. Looking down the

stack, some files of the composite directory are found in the top-most (first) directory of the search order stack. Files not found there might be found further down in the stack. Only if the entire directory search order (i.e. stack) is exhausted without finding the file, is it deemed not to exist in the composite directory. Similarly, a file in the second directory, which is also in the first directory, will be invisible because the copy in the first directory will always be used in the composite DSO directory.

The first directory in the search order may be a R/W directory. Any write operations to the composite DSO directory will be physically performed to the first directory, if it is a R/W directory. All other directories in the search order are treated as R/O directories.

Directory Search Orders are useful for performing incremental builds or for accessing shared files while allowing you to overwrite or append to the list of files. For example, if the second directory is a shared directory containing all of the release 2 source code, and the first directory is a R/W directory, the user may edit files found in the DSO and the changes will appear only in the R/W directory, not in the release 2 directory. However, the DSO directory will appear to be the entire release 2 contents with the modified files. Others may also share the release 2 directory, using their own DSO directory definition, and make changes that only they can see. All users will only have the modified source files in their physical directories, while the remaining files of the release, although appearing to be in the same directory, are shared by all users.

The DSO capability is modelled after the similar "search order" functionality of the OpenVMS (r) operating system.

A primary use of the DSO directories is to all the use of shared object code directories. For very large projects where users are making changes to a small number of low impact files, object code sharing becomes an effective means of performing change without the need for each user to manage a full object code directory.

6.3 VFS (Virtual File System) Capability (release delayed until 2009)

The Virtual File System capability of CM+, CM+VFS, is an optional product add-on that allows files within the CM+ repository to be visible from the Windows operating system. By establishing views to the CM+ repository, source trees may be mounted at the client, which act like read-only directory trees, from the file system perspective, while the source code actually remains fully and completely in the CM+ repository.

CM+VFS is available only on Windows XP and Vista (at this time), but the mount point may be a shared directory which is shared, through NFS, to Unix systems. The capability here is similar to the VOB functionality of another CM tool, with the exception that the mount points are always client-side, eliminating any server performance issues, and their corresponding scalability impact.

6.4 Live Export Capability (release delayed until 2009)

The Live Export capability of CM+ is an optional product add-on that allows you to define a report from CM+ into a file, and to have the content of that file to be always up-to-date. The report can be a text file, an HTML file, an XML file or a CSV file. It may be a composite of multiple queries or a single query. Effectively, upon opening the file, the query which produced the report is executed to produce an up-to-the-minute version of the report. This allows access to up-to-date data for those with CM-tool-phobia. After configuring the report, one needs only to open the file to view or process its up-to-date content.

The Live Export capability is only available on Windows, as an optional item. However, it should be possibly to access such a Windows file, if shared, from a non-Windows platform.

6.5 Eclipse Plug-in

An CM+ Eclipse plug-in is available for users of the Eclipse platform. The design supports the basic check-in and check-out capabilities as well as difference reporting, change packages, and the possibility of customization to support additional functionality. The CM+ Eclipse plug-in is an optional component.

6.6 OpenVMS Support on Itanium (HP Integrity) Platform

The OpenVMS version of CM+ is available on the HP Integrity platform. The user interface is the standard X-window interface for CM+ in the current release. Version 8.3 of OpenVMS is supported. The OpenVMS version of CM+ is an optional add-on to the base product.

7. WorkSpace Status Specification

The WorkSpace Status Dashboard is used to help analyze the content and state of a user's workspace or a portion thereof. First available with CM+ 7, the dashboard will have variations that evolve both over time and across projects. The Workspace holds the source code used for "local" builds, as well as for the edit-compile-test cycles which accompany such builds. Some terminology:

Source Code:	A file which is generally compiled or otherwise used as part of a run-time or build environment.
Workspace:	The root directory, and sub-directories, containing source code required for a product builds.
Checkout:	Registering a file in CM+ for a change, with an option of retrieving the file to the workspace.
Get:	A retrieval operation to place source code in the workspace.
Checkin:	The submission to the CM+ repository of an Update containing previously Checked Out files.
Context:	The "current" view of repository files in CM+, as specified by the user.
Synchronize:	The operation of bringing the workspace in line with the context view.

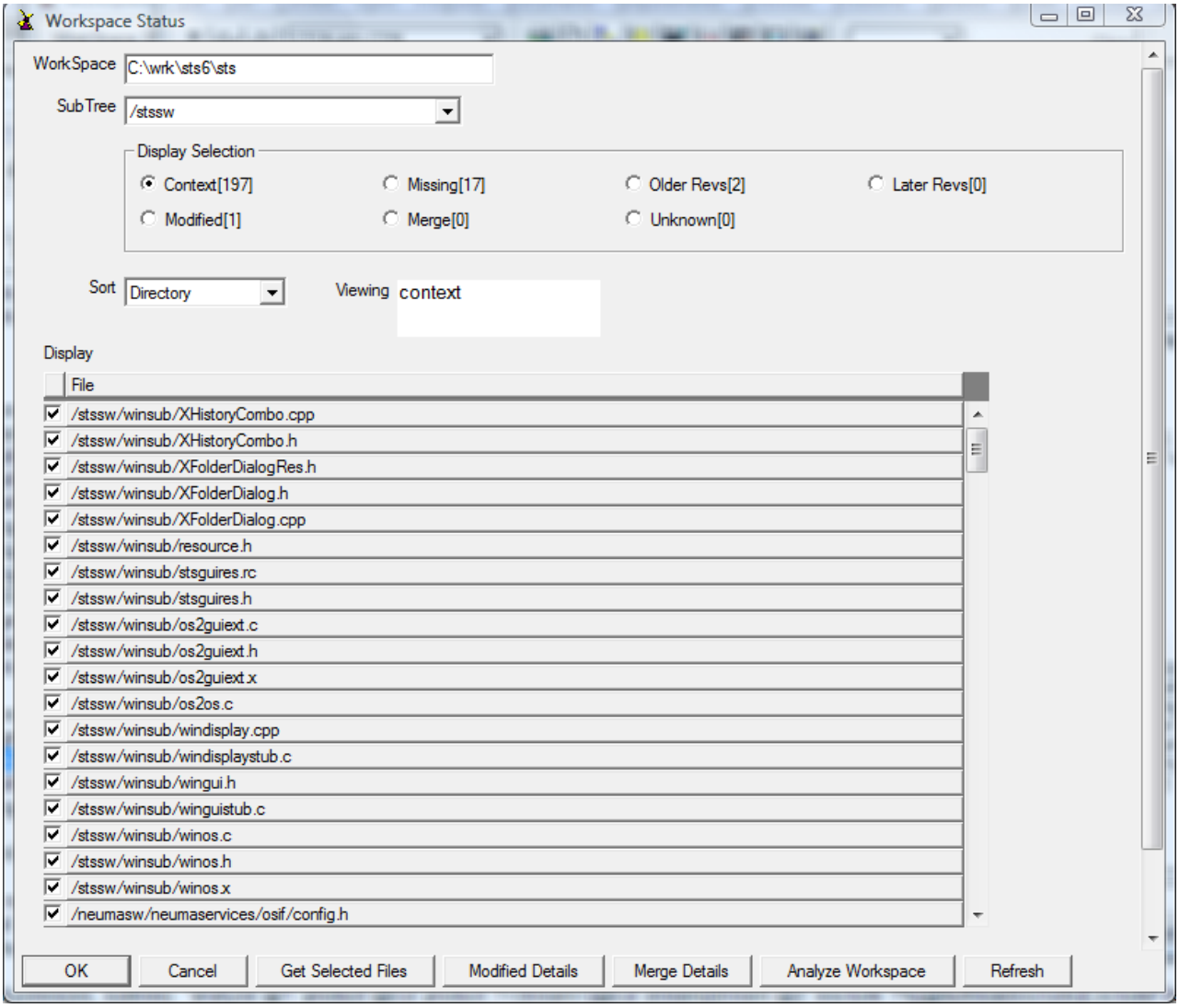
[Note: An important distinction needs to be clarified because of the different common usages of the word "Checkout". In CM+, a Checkout operation registers a file for change (i.e. against an Update), typically reserving that branch of the file until the changes are Checked In. The term used for retrieving source code to the Workspace is "Get" or "Get Source". A Checkout operation may or may not indicate that source code is to be retrieved.]

7.1 Workspace Status Operation

A CM+ user may right-click on any directory in the source tree and select "Workspace Status". This will bring up a dashboard that outlines the various states of the files in that part of the source tree. The states are identified as follows:

Context:	Workspace file matches (date and code) in current CM+ context.
Missing:	File in the CM+ context is missing from the Workspace.
OldRev:	Workspace file matches an older revision than the one in the CM+ context.
Later:	Workspace file matches a later revision than the one in the CM+ context.
Modified:	Workspace file differs from the CM+ context, with changes on top of the context revision.
Merge:	Workspace file differs from the CM+ context, and was derived from an earlier revision.
Unknown:	CM+ does not know the origin of the file in the Workspace.

The count of the number of files in each of these states is displayed in the dashboard:



These are the various fields of the Workspace Status dashboard:

Workspace: Your current workspace setting, to which this is being applied. (Restart this dashboard if you change your workspace). This is a read-only field for information purposes only.

SubTree: The sub-tree of your product source tree on which the workspace status operation is being applied. This may be the entire product tree, but in larger projects, this may result in some significant delays. If you change your sub-tree selection, you must select Refresh to have the information updated.

Display Selection: This selects the workspace status which will be displayed in the Display area. Whenever you switch workspace status, all of the files with that status will be selected (for use in Get operations).

Sort: The Display of files may be sorted by the Directory tree (same order as in the tree), by filename, or by file type.

Viewing: (currently not used - information purposes only).

Display: This is the set of files in the context tree whose corresponding workspace status matches the Display Selection. The "Get Selected Files" button works on the displayed set of files which have check marks beside them.

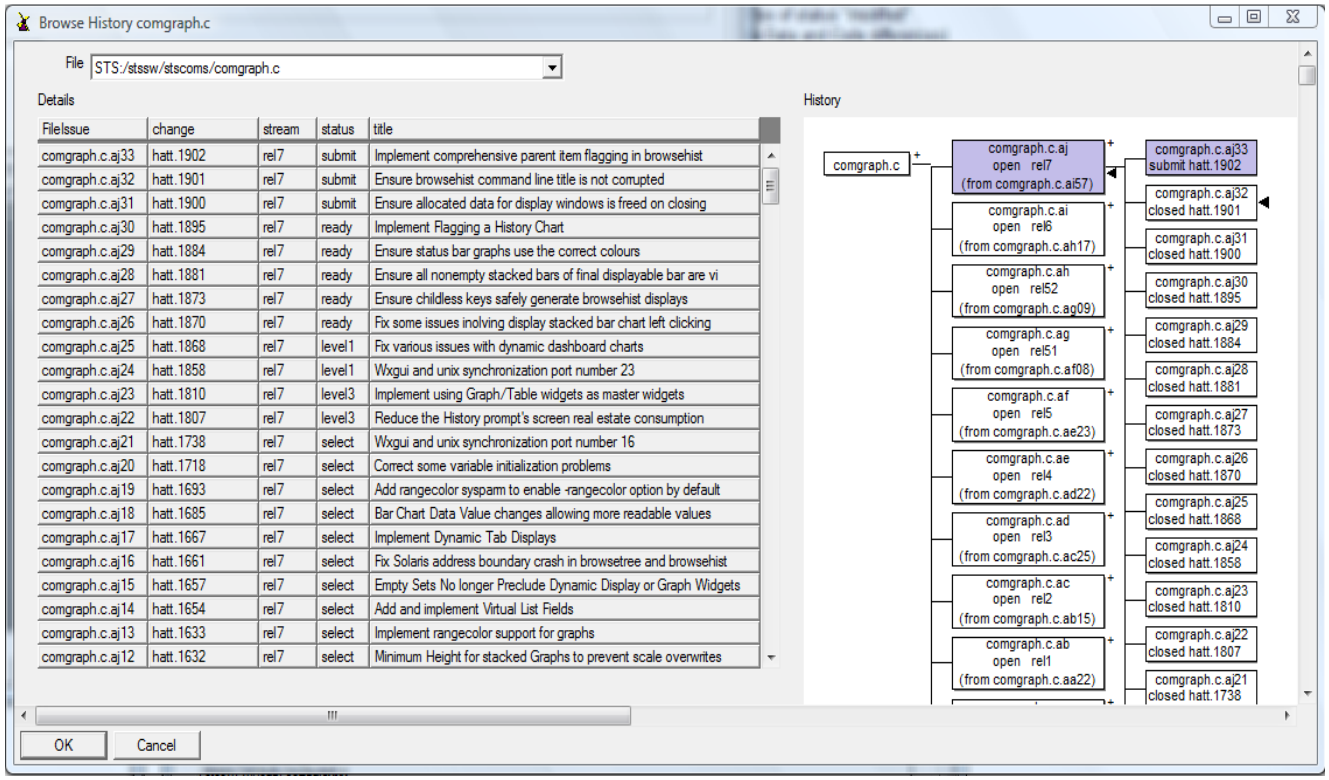
Action Buttons: There are a number of action buttons which may evolve or be customized over time.

OK - Exit the dialog
Cancel - Exit the dialog (not really different from OK - included for GUI consistency)
Get Selected Files - Dialog for retrieving the currently selected files in the Display area (i.e. same status).
Modified Details - Dialog showing the Code vs. Meta-Data differences for files of status "modified"
Merge Details - Dialog showing the various partitions of "merge" files (Meta Data and Code differences)
Analyze Workspace - Dialog to identify New (not in CM+) files with options to perform on the workspace.
Refresh - Refreshes the dashboard to take into account workspace, repository or Sub-Tree selection changes.

Note that generally, actions taken in other dialogs may require a Refresh operation before they are visible on the Workspace Dashboard.

As well as the array of buttons at the bottom of the dashboard, there are also right-click (context menu) operations that are available to further analyze the workspace files. These include "Browse History", "Get Source" and "Compare to Workspace" functions, which may be executed on individual files by right-clicking on them.

The "Browse History" operation will bring up a dashboard which shows an interactive display of file revisions in a scrollable panel, as well as a history chart. On the history chart, you will see the File, the list of Branches, and the list of Revisions in the currently open Branch. The shading of a single Branch and Revision, indicates that they correspond to the current context setting within the user's CM+ view. The arrows, if present, indicate the Branch/Revision which may be found in the workspace. In some cases (modified and merged files), subsequent modifications may have been made to the workspace file. So, in this case, the arrows represent the revision from which these modifications were started.

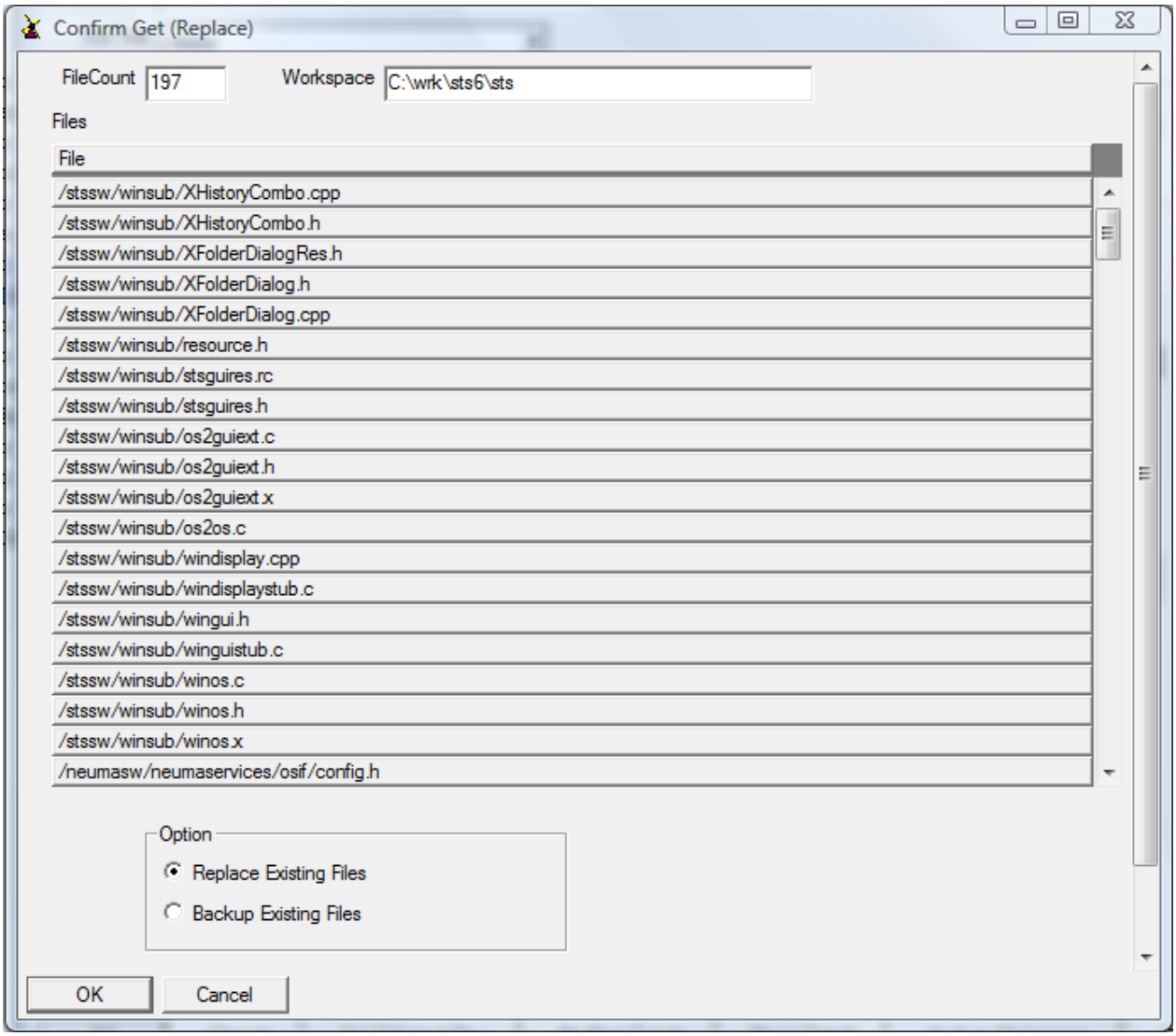


FileIssue	change	stream	status	title
comgraph.c.aj33	hatt.1902	rel7	submit	Implement comprehensive parent item flagging in browsehist
comgraph.c.aj32	hatt.1901	rel7	submit	Ensure browsehist command line title is not corrupted
comgraph.c.aj31	hatt.1900	rel7	submit	Ensure allocated data for display windows is freed on closing
comgraph.c.aj30	hatt.1895	rel7	ready	Implement Flagging a History Chart
comgraph.c.aj29	hatt.1884	rel7	ready	Ensure status bar graphs use the correct colours
comgraph.c.aj28	hatt.1881	rel7	ready	Ensure all nonempty stacked bars of final displayable bar are vi
comgraph.c.aj27	hatt.1873	rel7	ready	Ensure childless keys safely generate browsehist displays
comgraph.c.aj26	hatt.1870	rel7	ready	Fix some issues inolving display stacked bar chart left clicking
comgraph.c.aj25	hatt.1868	rel7	level1	Fix various issues with dynamic dashboard charts
comgraph.c.aj24	hatt.1858	rel7	level1	Wxgui and unix synchronization port number 23
comgraph.c.aj23	hatt.1810	rel7	level3	Implement using Graph/Table widgets as master widgets
comgraph.c.aj22	hatt.1807	rel7	level3	Reduce the History prompt's screen real estate consumption
comgraph.c.aj21	hatt.1738	rel7	select	Wxgui and unix synchronization port number 16
comgraph.c.aj20	hatt.1718	rel7	select	Correct some variable initialization problems
comgraph.c.aj19	hatt.1693	rel7	select	Add rangecolor sysparm to enable -rangecolor option by default
comgraph.c.aj18	hatt.1685	rel7	select	Bar Chart Data Value changes allowing more readable values
comgraph.c.aj17	hatt.1667	rel7	select	Implement Dynamic Tab Displays
comgraph.c.aj16	hatt.1661	rel7	select	Fix Solaris address boundary crash in browsetree and browsehist
comgraph.c.aj15	hatt.1657	rel7	select	Empty Sets No longer Preclude Dynamic Display or Graph Widgets
comgraph.c.aj14	hatt.1654	rel7	select	Add and implement Virtual List Fields
comgraph.c.aj13	hatt.1633	rel7	select	Implement rangecolor support for graphs
comgraph.c.aj12	hatt.1632	rel7	select	Minimum Height for stacked Graphs to prevent scale overwrites

Note that both the Details panel and the History chart are interactive. You may right-click on any item and select one of the operations to be performed. To compare two arbitrary revisions, right-click on the "old" revision first and "Select Delta File". Then right-click the "new" revision and "Delta vs. Selection".

7.2 Get Selected Files

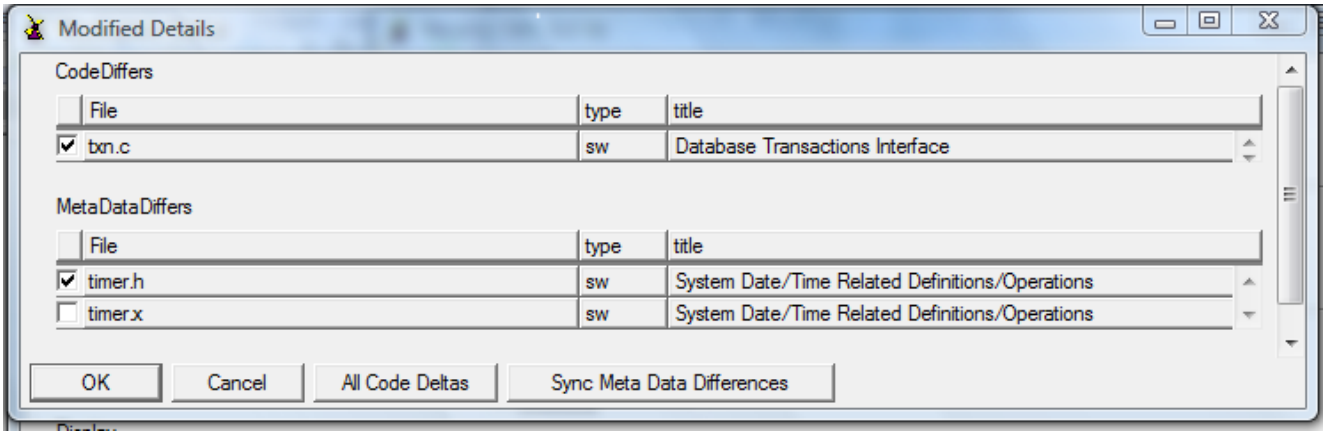
When the "Get Selected Files" button is clicked, a dialog appears which looks like the following:



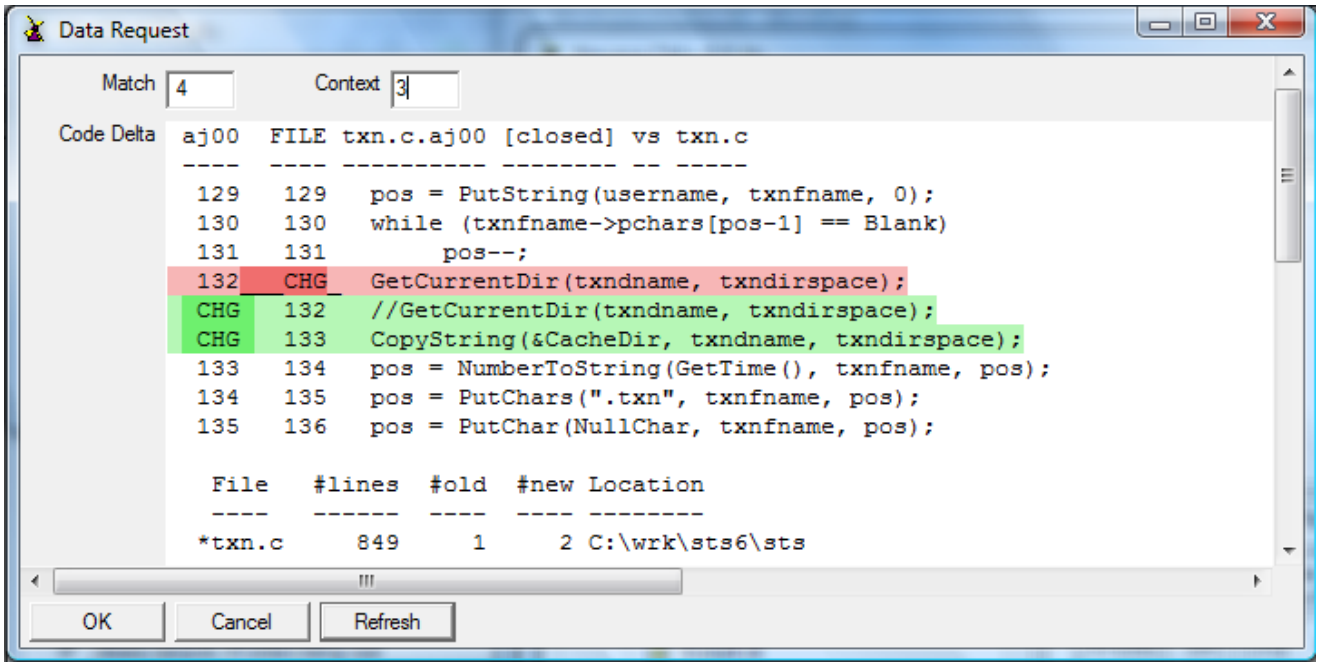
This is a confirmation dialog which also allows you to indicate what happens to existing files. The File Count indicates the number of files that were selected at the time you clicked the "Get Selected Files" button. The Workspace identifies the root directoy of the workspace to which these files will be retrieved. The list of Files are shown in a scrollable list. If you wish to change this list, return to the previous dialog first, by selecting Cancel. The Option field allows you to select between overwriting (i.e. replacing) existing files, and backing up existing files by appending the "backup" character to them (typically "X"). By default, two backups (X and XX) will be kept. When you click OK, the retrieval will begin immediately.

7.3 Modified Details

When you select the "Modified Details" button, a dialog appears showing which files differ in Code, and which differ only in Meta Data (such as the date stamp, or the identification information).



The files in the "CodeDiffers" list have actual source code differences between the Repository Context revision and the Workspace revision. You may right-click and select Compare to Workspace for an individual file, or you may view the code deltas (i.e. differences) for all of the files by selecting the "All Code Deltas" button. The following dialog then appears.

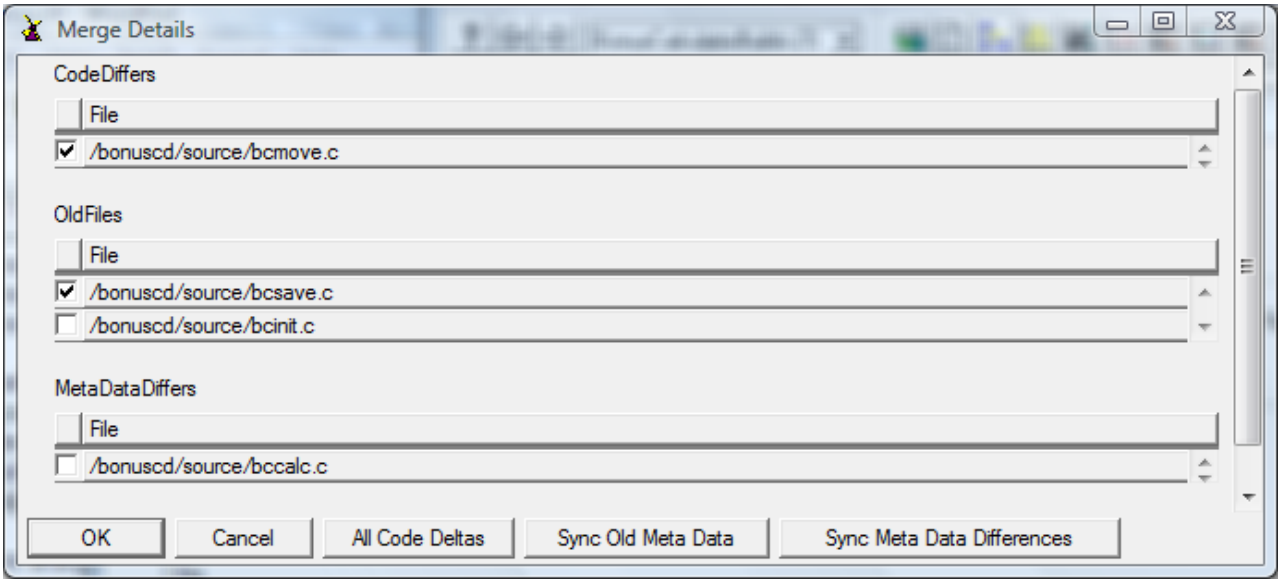


This will show the set of file deltas for all files and will give a summary at the end. By default, deletions will be shown in red and additions in green. You may adjust the number of context lines or the number of lines that must match prior to a change being considered a separate change block. When you do, however, CM+ will recompute the delta report. In the case a large or a large number of files, this may take a few seconds.

If you select the "Sync Meta Data Differences", CM+ will perform a "Get" operation by bringing up the same dialog as it does when you click "Get Selected Files" in the workspace status dashboard. Again, the operation will be restricted only to those files that have been selected in the "MetaDataDiffers" list of the Modified Details dialog. Files which actually differ in code will NOT be retrieved by the Sync operation. You may do so (and overwrite your modifications) individually using a right-click "Get Source" operation, or in batch from the workspace status dashboard, using the "Get Selected Files" button. Note that you may also perform individual Deltas by right-clicking on a file and selecting "Compare to Workspace".

7.4 Merge Details

Workspace files with a workspace status of merge are different both from the current context view in CM+ and from earlier revisions. The differences may be due to code changes, or to meta data (e.g. date stamp, identification info). The Merge Details button launches a "Merge Details" panel which further analyzes these files to partition them into one of three groups: a valid merge may be required; the file matches an Older Revision even though the meta-data indicates it may have changed; the file matches the Current Context Revision even though the meta-data indicates it may have changed.

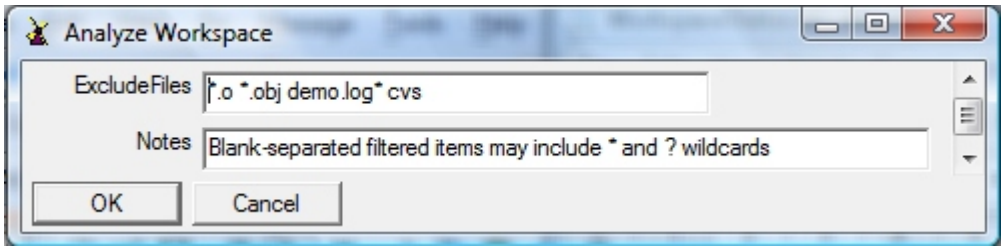


The "All Code Deltas" button presents a code delta report for all of the files in the "CodeDiffers" list, similar to the one for Modified Details, above. The "Sync Old Meta Deta" button will replace files which match Old Revisions (i.e. those in the "Old Files list), with those actual file revisions so that the meta data now matches. The "Sync Meta Data Differences" button will replace files in the MetaDataDiffers list, with current context revisions of those files, to ensure that the meta data differences disappear.

Right-click on a file in the CodeDiffers list should reveal a Workspace Merge operation. This will merge the current context with the file in the user's directory. If there is a common ancestor, a three-way merge is performed, otherwise, a two-way merge is done.

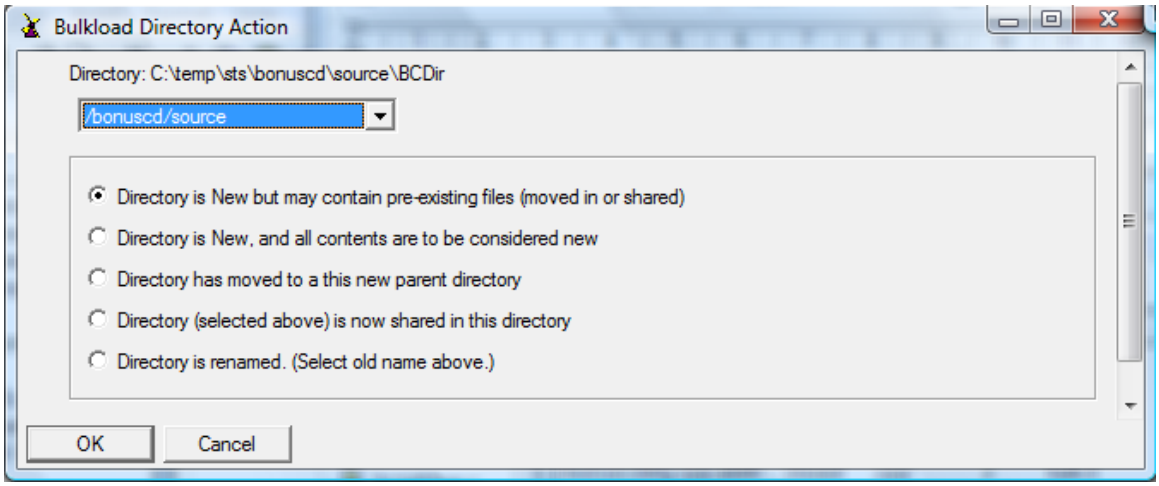
7.5 Analyze Workspace

When you analyze a workspace, CM+ goes through the workspace and compares the files it finds with those in your current context. It ignores files which match the list of excluded files, and otherwise identifies new files and directories, as well as files that match the context source tree.

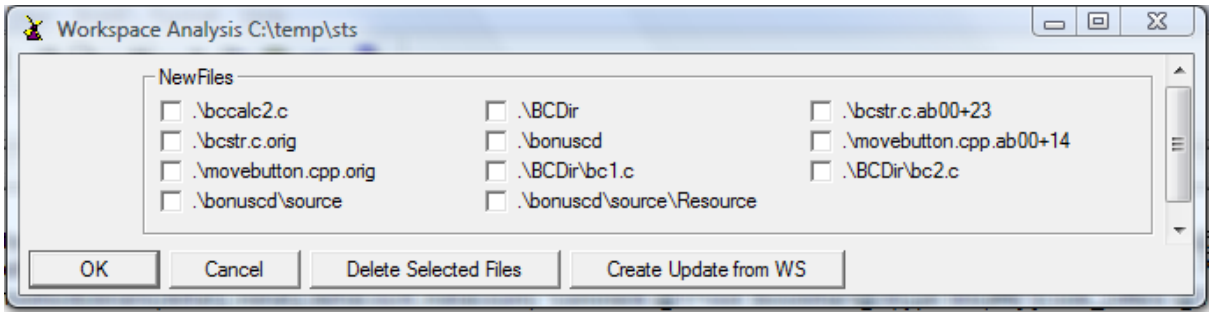


The analysis is performed after presenting a dialog panel for specifying the list of excluded files. If you use your

workspace directory as your build directory, you might have exclusions such as *.exe and *.dll. If you allow backups, either through CM+ or through an editor, you may have exclusions such as *X and *%, depending on what backup characters are used.



As CM+ analyzes the workspace, it may come across directories that don't match the context source tree. In such cases it will ask you how to treat them. Perhaps you've created a new directory and moved some files into it. Or perhaps you've renamed a directory. By indicating what you've done, CM+ can match files from the source tree to the workspace. Otherwise, it has to assume that the files found in "new" directores are themselves new files with no prior history. The top line of the Action panel indicates the item that is causing CM+ to prompt for information. The selector may be used with some of the responses, from which you select one, but is otherwise ignored.



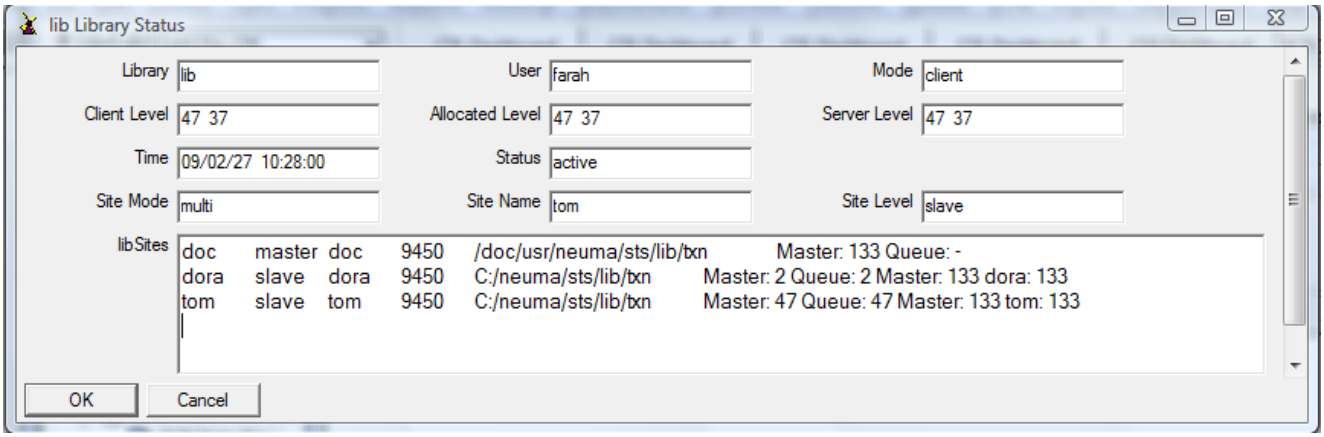
On completion, CM+ identifies the set of new files and directories. You may select files to be deleted from the workspace. You may also select directories, if empty. CM+ will not update the display until a subsequent analysis is requested. CM+ also gives you the option of creating an update based on the differences it finds in the workspace. This assumes that you have first reconciled your workspace to your context. Otherwise, you may find some unintended files as part of your Update. In such a case, you will need to perform a Cancel Checkout operation.

Neuma recommends that you check out (i.e. register against an Update) files as you need to change them. This both helps with the granularity of your traceability, and makes it easier for you and others to track your Update. The Create Update from Workspace operation is a tool that can be easy to use if you work on sub-trees that are wholly "owned" by you, so that the workspace sub-tree does not need constant re-basing (i.e. synchronization with other changes). Prior to using it on a project repository, get a feel for it by using it on a testbed, or on a copy of your production repository.

8 CM+ 7 File Menu Changes

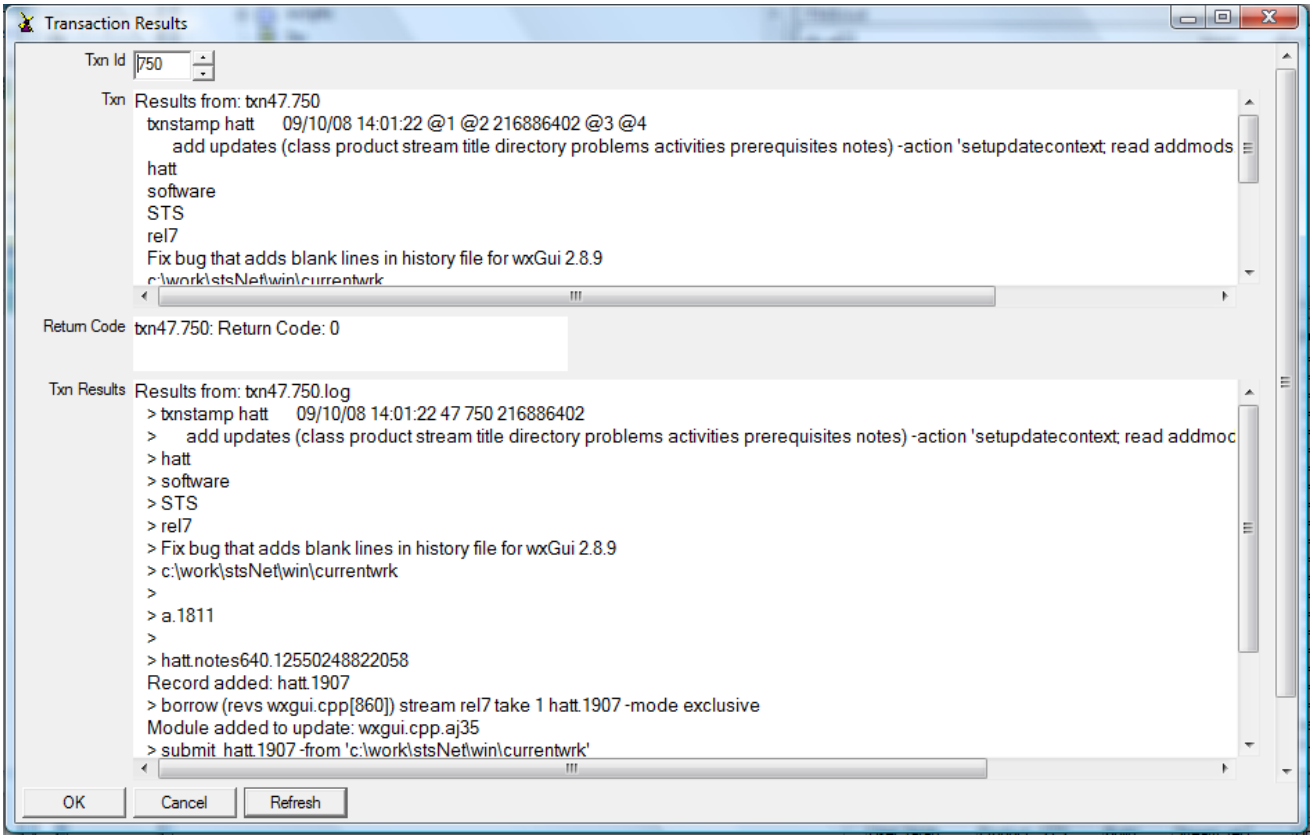
8.1 Library Status Window

The Library Status Window now shows additional information, including the status of a CM+MultiSite installation.

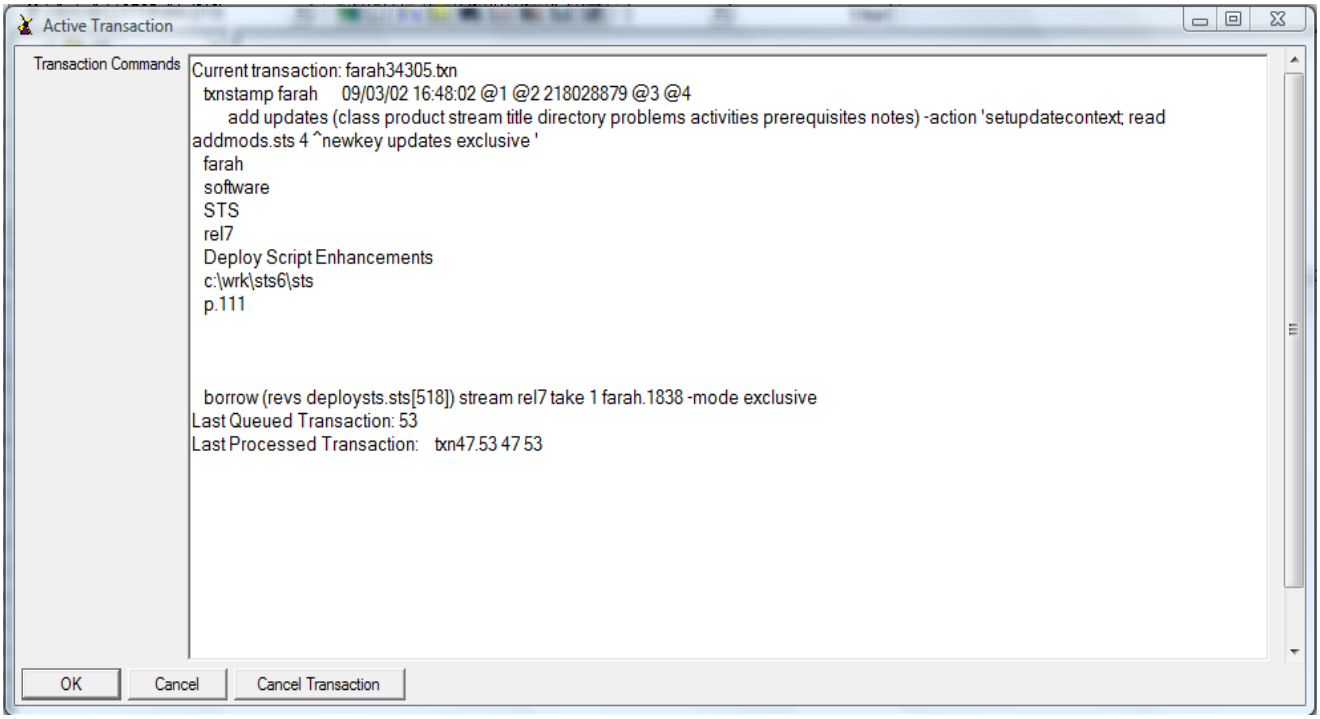


8.2 Viewing Transaction Results

You may now specify a different transaction and view additional logs using the same window. The up/down arrow keys as well as the scroll buttons may be used to view a different transaction, or a different transaction id may be typed into the Txn Id widget.



8.3 Cancelling a Transaction from the Active Transaction Panel



9 New System Parameters

bardataratio	"Minimum width % needed to display data values inside a chart bar"
browserindent	"Number of spaces a data pane tree is indented for each level"
chartbgcolor	"Default background color for a drawn chart/graph"
chartfgcolor	"Default foreground color for a drawn chart/graph"
charttextcolor	"Default text color for a drawn chart/graph"
checkbox	"Check Boxes appear on display panels by default"
checkoutmode	"The default checkout mode for a borrow (check out) operation"
checkoutmodes	"The set of checkout modes currently allowed"
debugmacro	"Enable tracing macro execution with parameters"
debugscript	"Enable tracing script execution with parameters"
displayscale	"Scales the display (typically to avoid Vista display scale error)"
fieldsep	"Separator used by the data macro (quote to quote complex tokens)"
fixedfont	"The name of the fixed font to be used for notes, etc."
fixedfontsize	"The size of the fixed font to be used for notes, etc."
formlabels	"Enable a form label showing the type of the form"
history	"Enable command history (disable for batch use)"
hostdirsep	"Use default host dir separator instead of / (windows)"
mapboxheight	"Height of application map boxes"
mapboxwidth	"Width of application map boxes"
maphorizontal	"Horizontal separation between application map boxes"
mapvertical	"Vertical separation between application map boxes"
maxvlists	"Maximum number of vlist fields allowed"
numericprompt	"Spin Controls enabled for numeric fields by default"
rangecolor	"Enable -rangecolor option by default on commands allowing it"
sepcolor	"The color used for data pane column separators"
sepwidth	"The width, in pixels, used for data pane column separators"
shortreply	"Maximum reply length to enable 1-line time-stamped text replies"
stagebackups	"Maintain staging backups on txn rollover when staging is used"
testmode	"Enable test mode to record command/gui actions in a test script"
textmode	"Enable use of foreign or local CR/LF conventions: unix/windows"
tipstatus	"Tree item colors reflect branch tip, instead of context, status"
txnsuppress	"Suppress from transactions, change commands reporting 0 changes"
varfontsize	"The size of the variable width font used for notes, etc."
variablefont	"The name of the variable font used for notes, etc."
verticalsplit	"Split the primary main window pane vertically"

10 Problems Addressed – Summary by Priority/Class

10.1 Low Priority

Problem	prio	class	title
p.3832	low	misc	File named: square Appears on Building new Library
p.2607	low	gui	Activities Custom Report Cannot Have Spaces in Filename
p.3858	low	cm	Comment Insertion Attempted Without SourceId Attribute
p.4127	low	db	Sections Appltab Display is Irregular on Show
p.3052	low	applic	Find By Activity should Filter to Activities with Updates
p.3837	low	report	Show Command Show - for $-0.01 for a Cents field

10.2 Medium Priority

Problem	prio	class	title
p.4046	med	misc	Access violation: right-click Properties from Products list
p.3584	med	gui	Switch User Not working on Linux - Prior Instance Hangs Around
p.4076	med	gui	frame sizes not preserved between invocations of CM+
p.3990	med	gui	Requirements dialog has title, buttons cut off
p.3594	med	gui	Right Click Window on Process Flow Graphs are Too Small
p.3946	med	gui	Long Gantt charts need to place tracking dates at chart top
p.4035	med	gui	Clicking Ok on a change form causes output area to scroll to top
p.4159	med	gui	Delta Recalulation Slows Closing of a Dialog/Dashboard
p.4149	med	gui	Multi item default strings not handled correctly by autodefault
p.4145	med	gui	Updating dynamically defaulted browsehist gives Nil Tree Result
p.4135	med	gui	Force Selection Character Causes Warning Msg in Prompt
p.4108	med	gui	Colour Prompt with no default has random initial colour
p.4073	med	gui	Button Prompt Scanning Confused by Nested Braces
p.4054	med	gui	Default Not Remembered With Substitution Buttons
p.4057	med	gui	Deferred Macro Substition Not Occuring on Prompt Button Commands
p.4049	med	gui	Trailing Blank in Radio ButtonDefault Causes Incorrect Selection
p.4043	med	gui	Apply Button Misbehaving
p.3890	med	gui	Problem Arrival Rate Prompt Graph Comes Up Blank
p.3696	med	gui	Cascading Prompt Fails When No Parent Default Specified
p.3680	med	gui	Drag And Drop From Activity Data to Tree Not Refreshing
p.4132	med	gui	Session refresh colors open directories green (user checked-out)
p.3878	med	gui	Gui Browser "Erratic" expansion and item selection on refreshes
p.3953	med	gui	ListView Trace gives duplicate trace items for same issue
p.4152	med	gui	Record access with insufficient permission locks output area
p.4045	med	gui	Form Command With Browse Key Gives Empty Set Error
p.3923	med	gui	Listview doubleclick generated forms can have fields overlaping
p.4025	med	gui	An empty container that is first in the treeview terminates CM+
p.3970	med	gui	Radio Button Selector Allows Tuning Off All Options
p.3956	med	gui	Pane Sizes Sometimes Undesirable On CM+ Start Up
p.3929	med	gui	Sub-tree already exists Message Does Not Give Browser Name
p.2690	med	gui	Fix Problem Pop-up Does Not Allow for Adding Problems
p.3989	med	gui	Commit action shouldn't show error if no transaction in progress
p.3825	med	sets	CM+ Should Indicate Invalid Field When For Bad Field Lists
p.3720	med	sets	Product Filtering Slows Tree Expansion in Main Tree Panel
p.3039	med	coms	Use of editdelay for edit -read
p.2819	med	cm	WorkSpace Handling When No Prior Workspace Set

p.3896	med	cm	CM+ Shows Checkout Option if Branch is Checked Out
p.4138	med	cm	Analyse Messages for directories are confusing
p.3621	med	cm	Timestamps On Binary Files After a Get Operation
p.3793	med	cm	CM Rename Does Not Work With Binary Files
p.4104	med	cm	No application Associated with extension is noise
p.3462	med	db	Reference Field that Exceeds maxlist Does Not Default to NIL
p.4133	med	db	Unable to Access Text Line Message Too Intrusive
p.2299	med	os	Piping output on Windows 2000 Gives Funny Resource Message
p.3824	med	txn	Non-Interactive Session Should Not Prompt for TXN Recovery
p.3457	med	applic	Change all build mapping operators to ^build
p.3133	med	applic	Analyse Function Dialogs should Excluded Products with No Source
p.3413	med	applic	Change Precmd Trigger Fails For Members Field and No -to Clause
p.3414	med	applic	Existence of previous issue required for reconcile
p.3578	med	applic	Deploy Source Should Stop On Error With Target Directory
p.2879	med	applic	Add User is Sending Two Transactions - Autocommit Problems
p.4153	med	applic	Project Activity Selection Should Show Most Recent First
p.3588	med	applic	Transition Trigger from Open-Submit for Update Should Not Be
p.4122	med	applic	Yank Should Not Appear for Open Updates
p.4112	med	applic	Mark Not Ready Appears for CMMgr For Updates Not Ready
p.3579	med	applic	Identify the Need to Select Files on Updates Popup: Get Source
p.3973	med	applic	In Set Context, Workspace has an extra Blank, so Browse Fails
p.4103	med	applic	Some Default Ports Conflict with Well Known or Registered Ports
p.3583	med	macro	Edit Preferences Options Sometimes Fails With Macro Error
p.3700	med	macro	Min operator of Calc Macro Fails
p.3886	med	macro	Quotes are Not Stripped off Command Trigger Parameters
p.3367	med	license	Library Server Does Not Recognize New License Key Automatically

10.3 High Priority

Problem	**prio**	**class**	**title**
p.3829	hi	misc	WXgui TCP query mode does not display textlines and notes.
p.3868	hi	misc	Modify and input button parameter 1 contains gibberish
p.3964	hi	misc	CM+ Session Teminates with stack overflow on menu File/Exit
p.3828	hi	misc	WXgui fails to launch TCP query mode gui client
p.3930	hi	misc	Disabling interactive for eclipse sessions prevents autorefresh
p.4029	hi	misc	Failed ScanName call in ScanTableOperand terminates unix session
p.3962	hi	misc	Initializing virtual list during library creation aborts session
p.3963	hi	gui	Changing context results in multiple dialogs displaying
p.3931	hi	gui	Left double click of CM+ listview subitem terminates session
p.3918	hi	gui	Clicking a Reference in an Extended Field List Gives Bad Result
p.3827	hi	gui	HTML window require user interaction before rendering is visible
p.3839	hi	gui	Cursor Moves from Listview to Command Area with Arrow Keys
p.4160	hi	gui	Right-Click Remove Functionality Does Not Show up on Linux/Unix
p.4124	hi	gui	Left click on stacked bar chart display has inconsistent results
p.4125	hi	gui	Left click on unix display bar charts terminates session
p.4156	hi	gui	Linux session terminates with output redirection to window
p.4137	hi	gui	Graph Using Wrong Range Colors
p.4136	hi	gui	Last item or items in a Stacked Graph Not Showing Up
p.3981	hi	gui	Displays tables should disable scrollbars on full visibility
p.3965	hi	gui	Bringing up a treeview on Solaris terminates session
p.3922	hi	gui	Table tab windows lose focus for user generated tab windows
p.3836	hi	gui	Solaris browse hist core dump
p.4158	hi	gui	Checkout in Updates Menu Highlites Files Automatically
p.4162	hi	gui	Attempt to perform browse history in wxWindows 2.8.9 terminates
p.4120	hi	gui	Browser -replace on Container Should Not Give Error
p.3951	hi	gui	Interleaved trace and revision items result in reversed ordering
p.3985	hi	gui	Right click of a browse tree item terminates session in unix gui

p.4157	hi	gui	CM+ Crash on add prob (stream request) when there's no requests
p.4148	hi	gui	Prompt Match Failing When Space in Value
p.4142	hi	gui	Scroll Bars Disappear in Dashboard Display Panel with Checkboxes
p.4143	hi	gui	CM+ Crashes with Sub Fields and Checkboxes.
p.4139	hi	gui	CM+ Crashes On Invert Selection Button of Display Panel
p.4116	hi	gui	Arrival Rate Graph Not Following Parent Widget Properly
p.4115	hi	gui	Unix Dashboard prompt windows do not account for display windows
p.4051	hi	gui	Incorrect Substitution in Dynamic Prompt
p.4026	hi	gui	Forced selection textline widgets can terminate CM+
p.3900	hi	gui	Invoking a process chart prompt terminates CM+ session
p.3873	hi	gui	SetContext dialog no context; stream selection crashes session
p.3867	hi	gui	Clicking on browse button of an empty list field terminates CM+
p.3687	hi	gui	File Based Prompting Lists Default Incorrectly When No File
p.3866	hi	gui	Popup Menu Button Protection Has No Effect
p.3851	hi	gui	Multiple Warning/Error Messages/Prompts on Invalid GUI Prompt
p.2939	hi	gui	Incomplete dragdrop operation leads to application crash
p.3683	hi	gui	Menus For Delete/Paste Showing up in Duplicate
p.3684	hi	gui	Dragging a Node into A Child Node Crashes CM+
p.4163	hi	gui	ListCtrl selection on wxWiindows 2.8.9 gui requires double click
p.4161	hi	gui	Clicking field that updates dashboard terminates wxgui session
p.4155	hi	gui	CM+ Crash on Entry - Linux with 0 Main height/width
p.4129	hi	gui	Browser display on unix after bulkload subtree
p.4033	hi	gui	Empty history file causes wxgui to hang on startup
p.3924	hi	gui	Filtering listviews which reflect an expanded subtree has issues
p.3952	hi	gui	Right click on key item is heterogeneous tree terminates session
p.3958	hi	gui	Container Browser Does Not Work Until Non-Container Tree Added
p.3840	hi	gui	Listview Field Expands to 255 When Adjusted to Be Hidden
p.3854	hi	gui	Browser Traps When Expanding Field Set
p.3732	hi	gui	$Selected Set Not Set - Tree Not Resetting Properly
p.3757	hi	gui	Requirement Added To Leaf Node Does not Show Up in Tree
p.3898	hi	gui	Left click of Graph bar crashes wxgui session
p.4128	hi	gui	Unix session terminates on bulkload
p.4141	hi	gui	Invert Selection Inoperative After Clear Button Defined
p.4151	hi	gui	Clicking on Problem Dashboard button can terminate session
p.4060	hi	gui	Crash On List Transitions from Dashboard
p.4147	hi	gui	Failed data transition change can terminate CM+ session
p.4146	hi	gui	Selecting a dashboard display record terminates CM+ session
p.4081	hi	gui	Reference fields of empty tables should not be editable
p.4082	hi	gui	Attempt to modify browseable form note field terminates session
p.4078	hi	gui	Vista Panels and Forms Not Long Enough On Some Displays
p.3919	hi	gui	Selecting a display record with an undefined tlist teminates CM+
p.3089	hi	gui	Modify forms not working when Invalid Field - No Error Msg
p.3842	hi	gui	Redirection of Simple Code To HTML Fails
p.4080	hi	gui	Path, FullPath Forms of Filename Macro Fail with Relative Paths
p.4121	hi	gui	Disabled display panel vertical scrollbar not enabled on resize
p.4117	hi	gui	Display Panel Showing 1 Fewer Item Than It Should - No Scroll Ba
p.4053	hi	gui	Double Clicking Checked Out File Does View Instead of Edit
p.3984	hi	gui	Display tab window's close button leads to session termination
p.3863	hi	gui	Display Drops Last Record Until a Resize is Done
p.3838	hi	gui	The title in the Notes tab clobber the 1st line of the note
p.3835	hi	gui	Click on graph bar terminates CM+ session when tablebrowser set
p.4090	hi	gui	Erroneous Tree View Deltas when DeployMode is tree
p.3986	hi	gui	Selecting a listfield of a prompt display terminates session
p.4140	hi	sets	CM+ Crash When Using Text Operator
p.3841	hi	sets	show probs.status Crashes CM+
p.3831	hi	sets	Set Evaluation Problems - Brackets Change Result and Should Not
p.2656	hi	coms	Bulkload Post Command Not Checking Parameters Before Using Them
p.3913	hi	coms	Modify Action Gives Errors or Crashes on Parameter Substitution
p.4087	hi	coms	Borrow with get corrupts the command line for postcmd trigger
p.4079	hi	vc	Visual Studio SCCAPI Does Not Receive Full Relative Path
p.3380	hi	cm	Failed Submits Cause Change of File Class to .bin on Bulkload
p.4075	hi	cm	Submit Pre-Command Trigger Gives Errors On File Submit

p.3850	hi	cm	Submit Problem when Context Switching and Connects are Unapplied
p.4024	hi	cm	Frequent Crashes on Refresh after Transaction Commit
p.3877	hi	cm	Non-tip Branch on Align Causes Invalid Branch if Stream Differs
p.3897	hi	cm	Branch Tracking Can Invert Branch Labeling Causing Duplicates
p.3865	hi	cm	Rename Popup on Context Tree Does Only Full Historical Rename
p.4130	hi	cm	View History in Updates Menu Crashes Sometimes
p.3405	hi	cm	Directory Checkout With Get Files Selected Fails to Get Files
p.3887	hi	cm	Working Directory in P23 (and perhaps earlier) Not Remembered
p.4109	hi	cm	Search Always Returns Lines in Upper Case
p.4126	hi	cm	Submit of Appltab for Activity Fails...
p.4083	hi	cm	Get Crashes on Field Notes When Treemode is Tree
p.3940	hi	cm	Adding a New File Using Updates | New | Add File Fails
p.3826	hi	cm	CM+ Adds a Duplicate Root Node on New | File by default
p.3852	hi	cli	Pre Command Trigger Not Echoing
p.3880	hi	db	SearchStreams Do Not Always Behave With More than 16 Streams
p.4034	hi	db	Schema Command Missing Comment Symbol for DAD Range Titles
p.3954	hi	db	Show or Display of Link Field Fails
p.3939	hi	db	Default Data Values Fail for Product Field
p.3903	hi	db	Modify Checking Permissions on Read Only Fields
p.3830	hi	db	Buffer Overrun if Textline Segment Count is NIL
p.4084	hi	db	State Commands Dumped More than Once With Subtables
p.4131	hi	db	Schema Dump of Triggers Crashes CM+
p.3938	hi	db	Replies Giving Funny Transaction Behavior/Errors
p.3967	hi	os	Directory Browse Button Causes Change in working directory
p.3857	hi	os	Comments Attribute Causes Writing to a Txn File
p.3971	hi	doc	Help Menu For Custom Process Guide is Invalid
p.3960	hi	txn	Transactions servers are not shutting down on request
p.3902	hi	txn	Zero-size Txn Appearing In MultiSite Txn Dirs
p.3211	hi	txn	CM+ Does Not Detect Empty Transaction Files
p.4119	hi	txn	With Autocommit Enabled, Cannot Reply Again to a Problem
p.3233	hi	applic	Ownership Prevents Assignee from Changing Activity Status
p.3892	hi	applic	Windows Hosttype Has Too Many Variations
p.4134	hi	applic	Reporting Macros are Defined In an Optional Module: Problem.gui
p.3378	hi	applic	Workspace Directory Limited to 64 Characters
p.4123	hi	applic	Sometimes There Is No Drop Down For Set Context Stream
p.4041	hi	macro	Data Macro Gives Warning When Set is Empty
p.3882	hi	macro	Functional Macros Do Not Handle Embedded Brackets
p.3861	hi	macro	Prompt Being Stripped of Parenthesis by Macro
p.3966	hi	macro	post Command For a Macro in Status Bar Causes Crash
p.4118	hi	arch	Active Transaction Give Multiple Errors On Bad Txnid
p.3856	hi	report	Note Command Hangs When Invalid Attributes Specified
p.3580	hi	integ	New IDE Project allows project name to be same as the product

10.4 Emergency

Problem	**prio**	**class**	**title**
p.3925	emer	txn	Library server erroneously believes duplicate replies are sent

10.5 Requests

Problem	**prio**	**class**	**title**
p.3916	req	db	Color Triplets Should Work With Or Without Commas
p.3934	req	process	State Flow Titles Should Be Larger

11 CM+ 7 Upgrade

For those with any recent version of CM+ installed (CM+ 5.2 and later), the upgrade to CM+ 7 is very straightforward.

11.1 Basic CM+ Upgrade Steps

The upgrade to CM+ 7 (from CM+ 5.2 or later) proceeds as follows:

(1) Backup your Library checkpoint file, copying/renaming it.
(2) Run the "pre-upgrade" script using the old installation (i.e. before shutting down your servers).
(3) Stop the library servers, move the library to the new installation, and restart the new servers.
(4) Run the post upgrade scripts, after perfrorming rule and trigger verification.
(5) Re-Apply your project GUI and Scripting customizations.

After step (3), you should perform step (4) prior to any other library transactions.

Step (1) Backups. Generally, it's a good idea to perform an upgrade right after backups have been run. In any event, you should create a copy of the checkpoint file for each library you are upgrading. The checkpoint backup may be maintained in the library directory. It provides an opportunity to role back to the old release should any service-affecting problem arise, although Neuma has been diligent in ensuring such problems will not arise.

Step (2) The PreUpgrade script. The PreUpgrade script will check your library status as it performs the upgrades so that only required steps are performed. If you receive errors during the upgrade step (2), perhapse because of your customizations, you should clear these prior to committing the upgrade step. This step will run the pre-upgrade script but will NOT commit it. You must commit the pre-upgrade transaction yourself after ensuring that there are no outstanding errors of concern.

Step (3) Move to the new Release. This step is fairly straightforward, although you may be moving (or copying) the library to the new server location. Make sure that you first stop all your servers. Note that during this time, read access to all library data, including files, continues without issue.

The are various ways to proceed with this step. Keep in mind that you do not have to move your libraries physically, though you may wish to. The projects.dat file in the CM+ installation directory (e.g. /usr/neuma/sts or C:\Neuma\sts) contains the locations of your library directories. By default they are created within the installation directory, but may be created or moved elsewhere as you see fit.

The idea is to get your new installation up and tested prior to switching over to it. So you might prefer to have a completely separate directory for CM+ 7 (or perhaps a new directory such as sts7 underneath the Neuma directory). In this way, you can test the installation prior to upgrading your production libraries. To do this, you would install the new upgrade into a separate directory, copy one or more library over for testing purposes, and run steps 2 through 5 on this test installation.

11.2 Establishing a Test Installation

To run your test installation, you should:

1. First run your pre-upgrade scripts in your existing libraries to prepare them for an upgrade.
2. Copy the following files from the old installation to the new one:

STS.key, fixedlic.users, licserv.add, and projects.dat.

3. Copy one or more libraries to the test installation (e.g. under the install directory)
4. Edit the projects.dat entries to point to copies of your libraries that you have created for testing.
5. Start a CM+ Administrator client for one of your test libraries and make sure that you are using the new image (Help | About CM+ should show “aj02” or later) and the test library (Context | SearchOrder should show you the directories for the image and for your library).
6. Start your servers in the test environment using the menus in the Administrator menu. You may need to “Stop” your library servers in this environment first, if the servers were running when you copied them. (Or you can erase the “inservice” file from the top directory of each of the test library copies.)
7. Go through the process of verifying, and if necessary, upgrading a copy of the trigger files provided in the installation (in the “schema” directory, the *precmd* and *postcmd* files). If you have made changes to any of these triggers, you must now apply those changes to the files in the “schema” directory. Note that as Neuma has continued to simplify these triggers (and rules), some have been eliminated. In most cases, clients have not modified these rules and triggers or have made minimal changes to them. This step becomes trivial in these cases.
8. You must refresh your client prior to running the PostUpgrade script. Run this script to actually read the modified rule and trigger files (i.e. precmd/postcmd) from your test environment schema directory. It will also update some on-line help information. This needs to be done on a per-library basis.
9. If you have modified the original “gui” or “coms” files, you will have to port these changes to the new installation by repeating (or merging) these changes. However, rather than doing so, you may wish to take advantage of new functionality which allows you to modify default GUI menu items by placing the modifications in a separate, customer or library specific file. CM+ 7 now allows you to remove menu items or modify menu items by running a file containing the revisions. CM+ 7 will replace existing menu items with the revisions. In this way, your original distribution can remain largely unchanged, and your customizations can be kept in a single file, while applying to multiple .gui files. The recommended name of this file is, for customer specific changes, “STS/gui/customer.gui”, and for library specific changes “STS/<library>/<library>.gui”, where STS is the installation directory, and <library> is the name of a specific library. Customization may also be done on a per user basis in the “STSHOME/<user>.gui” file.
10. After testing your customizations, and your upgrade, you can switch your production library over. Normally this would be done at this point by stopping the servers, moving the libraries to this new installation, then renaming the previous installation directory (e.g. To “sts61”), and then renaming the new installation directory to the previous name (e.g. From sts7 to sts). At this point, you can restart the servers, and, if necessary, copy any library specific customizations from the test libraries to the production libraries. You would also run the post-upgrade scripts on the production libraries.

11.3 Additional Upgrade Notes

Step (4) of the overall upgrade process involves a review of all of your command rules and triggers (precmd and postcmd files), as these may have changed significantly since you last upgraded.

The best way to review your rules and triggers is to first identify changes you made to the triggers in the original delivery and be prepared to apply these to the new set of triggers. If you do not have a record of your rule and trigger changes and do not have the originals, Neuma can provide the latter to you. Likely, you made very few, if any, changes to the Neuma provided rules and triggers.

In CM+ 7, in the default configuration, there are fewer command rules and triggers, and generally they are significantly smaller, as additional CM+ functionality has been added to eliminate the need for operations within the triggers and rules. If you are confident that the rules and triggers have not been modified from Neuma's defaults, you should end up with the following triggers only. These will be changed, automatically, by the PostUpgrade.sts script.

c_add_precmd.sts
c_add_postcmd.sts

c_align_precmd.sts

c_analyse_precmd.sts
c_analyse_postcmd.sts

c_borrow_precmd.sts

c_bulkload_precmd.sts
c_bulkload_postcmd.sts

c_change_precmd.sts
c_change_postcmd.sts

c_connect_precmd.sts
c_connect_postcmd.sts

c_context_postcmd.sts

c_password_postcmd.sts

c_submit_postcmd.sts
c_submit_precmd.sts

All other precmd and postcmd rules/triggers, should be eliminated by setting their value to NIL (for example):
> change c.branch -field postcmd -to NIL

To get a list of your currently active rules and triggers:
> list precmd $coms
> list postcmd $coms

Any trigger which is not in the above list should be reviewed to ensure that the rules/triggers are specific to your environment. Neuma does not require any rules or triggers other than the ones above. However, the customer may choose to add in their own rules and triggers.

Once you have modified your command rules and triggers, you may want to compare them to the demo library (that comes with the CM+ 7 distribution). You can do this by using the menu item:
Process | Command Rules/Triggers | Get Rules and Triggers
and placing each in separate directories for a “diff -r <directory>” operation (Unix), or a similar windows operation.

If you have made other customizations to CM+ you may want or need to apply these.

1. GUI customizations. First look at the new functionality. Many of your customizations may no longer be required because of additional default functionality.
2. Scripts: Pay attention to any changes you may have made to your object state names. These may have to be reflected in the new scripts. In particular, if you are making use of the the "reopen" state of an Update, this state is no longer available. Use the "open" state instead. The "reopen" state has been renamed to "staged" for use in a new feature (beta availability in this release) allowing staging of updates before they are checked in.

As well, as applying your GUI customization, you may want to consider changing the way that you apply them. In CM+ 7, if you redo a "menuitem" command, it will replace the current definition of the menu item.

That means, you may want to place all "changed" menu item customizations into a separate file to be run after the initial GUI definition files. In this way, you will not have to merge these changes into the existing Neuma-delivered files, and subsequent upgrades can be done more easily.

One other tip... as you may want to run this upgrade on multiple libraries, or more than once on the same library, it may be a good idea to create a concise point-form repeatable guideline to replace these instructions. You also may want to review the Upgrade scripts prior to the upgrade to get a feel for the type of operations being performed, and the conditions on the operations. In some cases, you may want to restrict these, though if you do, please check with us to ensure they are optional (they're usually commented if optional).