Retek® Predictive Application Server™ 11.2

Release Notes

Corporate Headquarters:

Retek Inc.
Retek on the Mall
950 Nicollet Mall
Minneapolis, MN 55403
USA

888.61.RETEK (toll free US)
Switchboard:
+1 612 587 5000

Fax:
+1 612 587 5100

European Headquarters:

Retek
110 Wigmore Street
London
W1U 3RW
United Kingdom

Switchboard:
+44 (0)20 7563 4600

Sales Enquiries:
+44 (0)20 7563 46 46

Fax:
+44 (0)20 7563 46 10

Printed in the United States of America.

Customer Support

Customer Support hours

Customer Support is available 7x24x365 via email, phone, and Web access.

Depending on the Support option chosen by a particular client (Standard, Plus, or Premium), the times that certain services are delivered may be restricted. Severity 1 (Critical) issues are addressed on a 7x24 basis and receive continuous attention until resolved, for all clients on active maintenance. Retek customers on active maintenance agreements may contact a global Customer Support representative in accordance with contract terms in one of the following ways.

Contact Method	Contact Information
E-mail	support@retek.com
Internet (ROCS)	rocs.retek.com Retek's secure client Web site to update and view issues
Phone	+1 612 587 5800

Toll free alternatives are also available in various regions of the world:

Australia	+1 800 555 923 (AU-Telstra) or +1 800 000 562 (AU-Optus)
France	0800 90 91 66
Hong Kong	800 96 4262
Korea	00 308 13 1342
United Kingdom	0800 917 2863
United States	+1 800 61 RETEK or 800 617 3835

Mail	Retek Customer Support Retek on the Mall 950 Nicollet Mall Minneapolis, MN 55403

When contacting Customer Support, please provide:

- Product version and program/module name.
- Functional and technical description of the problem (include business impact).
- Detailed step-by-step instructions to recreate.
- Exact error message received.
- Screen shots of each step you take.

Release Overview

Introduction

This document provides information about the changes and enhancements in RPAS 11.2 and covers the RPAS server, RPAS client, and Configuration Tools. Each feature contains the following information in varying levels of detail:

- Functional or technical definition of the feature and the underlying requirements the feature meets
- How to set up and configure the feature
- How to use the feature in the RPAS client
- Notes, assumptions, and limitations about the feature

Documentation

The RPAS 11.2 GA documentation is being released in draft form. All of the RPAS 11.2 documents will be updated and re-released in an upcoming RPAS 11.2 patch (date TBD).

The following documents have been included with this release and will be updated in an upcoming RPAS 11.2 patch:

- RPAS 11.2 Installation Guide – instructions for installing the RPAS server, client, and Configuration Tools. Oracle / Retek recommends that existing customers running RPAS 11.1 upgrade to 11.2 using the version of RPAS available on Retek's On-line Customer Support site (ROCS). Using ROCS, the upgrade process is primarily automated and will be easier for existing customers.
- RPAS 11.2 Administration Guide – a primarily technical document that describes how to administer an RPAS environment and server, including how to use the RPAS standard administrative workbook templates and the suite of standard RPAS utilities.
- RPAS 11.2 Users Guide – provides an overview of RPAS concepts and how to use the RPAS client.
- RPAS 11.2 Configuration Guide – describes how to configure a solution using the RPAS Configuration Tools.
- RPAS 11.2 Calculation Engine Users Guide – describes how the RPAS calculation engine works, including key concepts and basics about aggregation, spreading, rules, expressions, and so on.
- RPAS 11.2 Rule Functions Reference Guide – contains the detailed syntax and usage of standard RPAS functions, procedures, and modifiers, which are mechanisms of the calculation engine that are used when writing expressions.

Impact and implications of upgrading

The following should be noted about upgrading to RPAS 11.2. Some of these points are important notes while others are unavoidable implications of upgrading due to an architectural change or the implementation of a new feature.

- Like any RPAS patch, workbooks must be rebuilt after upgrading to reflect underlying changes to the domain. Workbooks should be committed before deleting. Administrators are prompted to confirm the deletion of workbooks when the upgrade process begins.
- Limitation for Hierarchy and Dimension Names – must force same names
  - RPAS has a limitation that does not allow a hierarchy to contain a dimension with the same name. For example, the PROD hierarchy cannot contain a dimension named PROD. It is imposed as a “soft” limitation in that RPAS will allow this condition but requires an administrator to force it.
  - This limitation is being imposed to address a specific Internationalization issue in the RPAS Translation Administration workbook template (used for modifying labels in multi-lingual environments). The names of hierarchies and dimensions are stored as positions and the template cannot handle these identical position names.
  - Existing configurations will be checked by the Configuration Tools to detect this condition, and users will be warned in the RPAS client if a workbook is built with this condition (including the Translation Administration template).

Scheduled enhancements for RPAS 11.2 patches

The following enhancements and changes are scheduled for an upcoming patch of RPAS 11.2. The release date had not been determined at the time this document was written.

- Ability to add new local domains to a Global Domain environment – currently the number of local domains must be decided before the initial build of a global domain environment. RPAS will provide the ability to add new local domains after a domain environment has been built.
- Ability to move partitions between local domains – RPAS will provide the ability to move a partition and its associated data from one local domain to another.
- Enhancement to “find next alert” – RPAS will introduce a new way to find the next alert within an alert workbook. The current functionality was built to find the next alert based on a single sorted list, which assumes a user is building workbooks with the positions they wish to see. The enhancement will allow a user to find the next alert based on the current “slice” that is in the z-axis of a window.

11.2 Summary

The following chapter summarizes the features and changes that are available in RPAS 11.2.

Functional Enhancements

- Measure formatting controlled from the tools – Ability to define measure formatting styles in the Configuration Tools and assign those styles to measures on windows. Provides styling upgrade path when patching domains or rebuilding workbook templates.
- Alert configuration support via the Configuration Tools – Ability to configure alerts in the Tools.
- Rule group simulator – For a chosen Rule Group, allows a user to preview the simulated effects on other measures of editing measures, effectively providing the ability to run "what-if" scenarios by flagging measures as changed and viewing which measures will be calculated, protected, and the calculation paths, without building a domain.
- Rule pooling – Ability in the Configuration Tools to use named rules in multiple Rule Groups. In previous versions, 'identical' rules are duplicated, leading to an explosion in rules, and potential inconsistencies.
- Measure pooling – Ability to define and reuse measures so that they are not duplicated within a solution/configuration.
- Configuration task list – Ability to see a comprehensive list of configuration errors in the Tools, the nature of those errors, and some actions to resolve.
- Printing/reporting capabilities in the Tools – Ability to generate a number of reports about configurations from within the Tools and output them in electronic format. Standard reports include measures, rules, workbook templates, and translation files.
- Function helper – Enhancement to the Rule Tool that will provide information about the syntax and usage of standard RPAS functions when writing expressions.
- Single hierarchical selection widget – A widget/window with a single hierarchical tree structure where you can use hierarchy navigation to drill down the appropriate branch of the hierarchy to the leaf nodes, and select a single position. The widget can be launched from a cell in the UI, and the selected position is pasted into the cell.
- Context sensitive dynamic picklists – The contents of a picklist measure can be based on the contents of another measure, which can vary based on the current context so that different contexts within the same workbook can have different picklist contents.
- Relative calendar in wizards – extended use of position queries to define a relative time position to use in wizards to build workbooks.

Technical enhancements

- New installation process for RPAS server and Tools on Windows – the RPAS 11.2 server and Configuration Tools can now be installed on Windows using an InstallShield installation process. This process also sets the required system environment variables.
- User interface for the RPAS installer – user interface that can be used to build and patch domains in a Windows environment instead of running the RPAS installer utility from a command line. The Installer UI can be launched from a start-up menu selection that is created when installing the Tools. The parameters for the UI Installer are the same as the traditional installer run from the command line.
- Ability to add or remove positions without have to reshape – ability for new positions to be added to a domain and be immediately available for use without having to reshape the data structures in the domain (off-line utility name “reshapearrays”).
- Position repartitioning (a.k.a. reclassification) – Provide ability to automatically move positions (when required due to restructuring) between local domains in a global domain environment.
- Enhanced hierarchy load process – in conjunction with the above two features, RPAS has enhanced the hierarchy load process. This new process allows for a single call to the loadhier utility in the master domain, after which the utility handles the movement of positions below the partition level, then updates the data structures in the domain without having to manually run the old reshaping process (reshapearrays utility).
- The process of reshaping arrays (using **reshapearrays** utility) has been incorporated into the hierarchy load process/utility. Data in all domains of an environment is reshaped, if necessary, after loading a hierarchy that has changed. As such, the rehapearrays utility has been deprecated.
- Centralization and parallelization of data load and export – Centralization and parallelization of utilities for loading and exporting measure data in Global Domain environments. The measure load process now also includes the splitting process. This allows for a single input file to be loaded from the master domain, and RPAS will split the data and load it into the sub-domain where the data resides.
- Ability to remove a user-defined dimension from a domain.
- Ability to run a subset of alerts – previously the “alert finder” (-find argument of the alertmgr utility) evaluated all alert rules in the domain. The utility has been updated to allow a subset of alerts (either a single rule or alert rule group) to be evaluated.

Minor changes and enhancements

The following is a list of minor changes and enhancements made to RPAS and the Tools that are available in RPAS 11.2.

- RPAS Administrative Templates
  - Security Administration – modified Workbook Template Limits worksheets such that undefined values will appear as blank cells. Technically the system stores a value of 1 billion by default but users will no longer see this value and will only see a value if one is defined (other than 1 billion).
- RPAS Client
  - Added “Commit ASAP” button to dialog window when closing a workbook.
- Technical
  - Ability to force the purge routine when loading measure data (**-forcePurge** for **loadmeasure** utility).
  - RPAS server component is compiled using HP’s ACC compiler on HPUX. Using the ACC compiler results in performance gains for certain off-line operations.
- Tools
  - Measures and measure components can now be displayed by the name or label in various locations of the Tools. This property is set in the *Tools Preferences* of the File menu, and can be set for measures and measure components.

Note: If “label” is selected and no label is defined for a given measure, the system will display a blank value in the Tools.

Hierarchy administration and position repartitioning

Overview

There are several new features in RPAS 11.2 that should simplify the administration of Global Domain environments. The following new functionality is included in this release:

- Enhanced hierarchy load process that removes a number of operations that previously required additional utilities to complete
- Ability to add or remove positions in a domain and use new positions without having to "reshape" the data in the domain; new positions are mapped to what were previously 'dummy positions'
- Position repartitioning – Ability to automatically move positions and data when a position has been moved from one local domain to another; this functionality is commonly referred to as "reclassification" by many RPAS customers

In RPAS 11.1, it was possible to have position names longer than 24 characters. This was accomplished with the implementation of phase 1 of Position Name Indirection (PNI) where there is effectively a mapping of internal and external position names.

RPAS 11.2 introduces phase 2 of PNI and provides the ability for a domain to contain dummy positions. These dummy positions are held as a buffer. As new positions are added to a domain in the hierarchy load process they are mapped to the dummy positions, so the domain does not need to be "reshaped," which is a time-consuming process in large domain environments.

Dummy positions

Phase 2 of the PNI implementation provides the ability for a domain to contain dummy positions. RPAS provides the ability for any dimension, other than those in the calendar hierarchy, to contain dummy positions, and the number of dummy positions is a percentage of total positions for a given dimension. For example, imagine the SKU dimension of the PROD hierarchy contains 1 million positions; a dummy position buffer of 1% will allow for 10,000 dummy SKU positions.

New positions are added to a domain through the normal hierarchy load process (RPAS utility **loadhier**). If a position is new, RPAS will map an existing unused dummy position to the newly added position so that it can be used in the domain without having to "reshape" the domain.

Similarly, as old positions are deleted, the external name for the internal position is removed from the mapping table (and data for the positions is removed from the arrays), and the position effectively become a dummy position. Thus deletes can also happen without the need to "reshape" the domain.

As dummy positions are consumed, the number of available dummy positions will be reduced. Dummy positions are held in the "buffer," and the process of updating this is "re-buffering." The buffer can be updated automatically, but unpredictably, when required (based on the lower and upper bounds that are defined for a PNI dimension), but it is recommended that the rebuffering process is scheduled to ensure that batch process windows are predictable. Details of this process are discussed in later sections.

The buffer of dummy positions is defined by a high and low setting for the buffer size that is a percentage of total positions for each PNI-enabled dimension.

Position repartitioning

Position repartitioning allows for positions to move between local domains. This functionality is only available (and relevant) in Global Domain environments. Positions need to be moved between local domains when they are assigned a new parent that exists in a different local domain.

Note: Moving positions at the partitioning level is not currently supported.

For example, imagine Style1 belongs to Sub-Class1 in LocalDomain1. If Style1 is reassigned to be a child of Sub-Class2, which is located in LocalDomain2, RPAS will move the Style1 position, Style1's children, and all corresponding data to LocalDomain2. This process is often referred to as "reclassification" by RPAS customers. RPAS is referring to this functionality as "Position Repartitioning" because it technically does not handle the many complex functional requirements of true reclassification as most retailers define the term to mean.

All of the activities and processes associated with PR will be automatically handled by RPAS and will not require manual intervention by the system administrator.

Note: Position repartitioning relies upon dummy positions. Thus all dimensions at or below a level that may be repartitioned must have dummy positions enabled.

Configuration and administration

There are a number of decisions that must be made regarding the use of dummy positions and the position repartitioning process. Dimensions will need to have dummy positions enabled for the following processes:

- Adding or deleting positions without having to reshape – customers wishing to add positions (most commonly adding new SKU's in the product hierarchy) will need to enable dummy positions for the dimensions.
- Using the automated position repartitioning processes for positions below the partition level.

All dimensions with dummy positions enabled need to have a buffer defined that represents the targeted number of dummy positions. There are high and low values for the target buffer size, which can be defined in the Tools and/or set using an RPAS utility. These buffers are used when "rebuffering" a domain.

Administrators need to determine how they wish to rebuffer the domains. Rebuffering can be scheduled and run manually using the Position Buffer Manager utility or can be handled automatically as part of the hierarchy load process.

Enabling a dimension for dummy positions

PNI buffering is enabled in a domain per dimension. Dimensions are enabled using the Configuration Tools both before and after a domain is built. Two properties were added to the Hierarchy Tool, and these properties can be set for each dimension. These properties are the "PNI Buffer Percent" for both the lower and upper bounds of dummy positions.

New positions are added to a domain by including new positions in the hierarchy data input files, then running **loadhier**. The syntax for **loadhier** remains unchanged. Newly added positions will be immediately available for use (unless all dummy positions have been consumed which launches an automatic rebuffering process); positions that have been assigned a new parent that require movement to other domains will be automatically processed.

Configuring and Scheduling the Rebuffering Process

Administrators need to determine the process by which they wish to rebuffer the dummy positions. Rebuffering can be completed automatically as part of the hierarchy load process (when a domain runs out of dummy positions), or can be scheduled using an RPAS utility.

If it is desired to have a predictable batch window it is recommended that administrators schedule the rebuffering process rather than use the automated process. Scheduling rebuffering will minimize the possibility that the automated rebuffering process occurs during a time critical batch process. The automated rebuffering would then only be used as a backup and would only run in unusual circumstances.

If customers do wish to reschedule the rebuffering, they will need to determine an approach that fits their business and batch processes. One approach might be to schedule all dimensions with dummy positions enabled in all domains to be rebuffered together on a weekly or monthly basis when there is a large amount of system down-time. Another approach could be to rebuffer a few local domains on a cyclical basis (such as a few each day or week).

There are high and low settings for the dummy position buffer. Within the Configuration Tools these settings are the “Buffer % Low” and “Buffer % High.” When executing a scheduled rebuffering process (using the **positionBufferMgr** utility), the buffer for a dimension in a given local domain is updated when the number of dummy positions falls outside the high or low target buffer percentages. The number of dummy positions is calculated by taking the average of the high and low percentages multiplied by the total number of positions of the dimension in the local domain.

Deciding the buffer percentages will depend on a number of criteria. The goal will typically be to have sufficient dummy positions to allow for growth in the local domains without having to execute an automated rebuffering process. Determining the targeted number of dummy positions will be a product of the anticipated growth in a given time period (for instance, 100 SKU’s per week) and the frequency of the scheduled rebuffering process (for instance rebuffering once a week or month).

These buffer settings and rebuffering processes are managed by the **positionBufferMgr** utility.

Hierarchy Load Process

From an administrative perspective, the hierarchy load process is unchanged from RPAS 11.1. All hierarchy files are loaded only into the master domain and contain all positions for a given hierarchy. If a position below the partition level has been added, deleted, or reassigned, additional background processes are executed to perform the required activities. Administrators will only need to use the **loadhier** utility to perform these activities.

RPAS will know if a position in the hierarchy data files is new and will perform a number of activities to “enable” that position in the domain. If an existing position has been reassigned such that it requires movement to a different local domain RPAS will perform a number of operations to move the position and its corresponding children and data from the previous domain to the domain to which it is being reassigned.

Utility Usage

positionBufferMgr

The position buffer manager is used to set a number of parameters when using dummy positions in a domain. See the above write-up for information about using and administering dummy positions.

Usage:

positionBufferMgr -d domainPath [-rebuffer|-report] {-hier hierName}* {-partitionPositions "pos1,pos2..."}

-rebuffer	Adjust all dimensions of the provided hierarchy to have the specified number of positions based on the configuration of the dimension.
-report	Report the count status of positions for all dimensions of the provided hierarchy.

Notes:

- If a hierarchy isn't provided then the -rebuffer or -report argument will occur on all non-calendar hierarchies in the domain.
- If a -partitionPositions is provided then the -rebuffer or -report command will work on just the domains that contain those positions.
- To get this usage text, use -?, -help or –usage.
- To get the version of this utility, use –version.
- To set the logger verbosity level, use -loglevel with values of: all, profile, debug, information, warning, error, or none.
- To disable timestamp header use -noheader.

dimensionMgr

The dimension manager utility is used for setting a number of parameters for dimensions and positions. These parameters are set when using the functionality of Position Name Indirection (PNI). This feature provides the ability to have position names that are longer than the default 24 characters and for a dimension to have dummy positions.

Usage:

dimensionMgr –d pathToDomain –dim dimensionName [COMMAND]

Argument	Description
–d *pathToDomain*	Specifies the path to the domain.
–dim *dimensionName*	Specifies the name of the dimension to which the settings will apply.
–specs	Command that displays the properties of the specified dimension.
–width *widthVal*	Command that sets the width of position names for the specified dimension. The default width for positions of a given dimension is 24 characters. Widths can only be extended and cannot be decreased.
–bufPctMin *minVal*	Command that sets the minimum percent of unused positions.
–bufPctMax *maxVal*	Command that sets the maximum percent of unused positions.

Notes:

- Multiple command arguments are allowed.
- Buffer Minimum and Maximum size are specified as a percentage of the total size of the Dimension. For example, a Dimension with 200 real positions and a Buffer Mininum of 5 and Maximum of 20 could have between 10 and 40 extra buffer positions at any given time.
- If set, the Buffer Maximum must be greater than the Buffer Minimum and less than 10000. To turn off buffering for a Dimension, set both Minimum and Maximum to zero.
- To get this usage text, use -?, -help or –usage.
- To get the version of this utility, use –version.
- To set the logger verbosity level, use -loglevel with values of:

  all, profile, debug, information, warning, error, or none
- To disable timestamp header use -noheader.

Functional enhancements

Single hierarchical selection widget

Feature overview and scope

A new ‘single hierarchy selection widget’ has been introduced to allow the selection of a single position from a potentially large number of positions for a measure. The selection is made in a window with a single hierarchical tree structure where the user can use hierarchy navigation to drill down the appropriate branch of the hierarchy to the leaf nodes and select a single position. The user will be able to view either position names or descriptions in this widget.

The hierarchy selection widget can be launched from a cell (by using a new control in the cell) for a normal string measure, and the cell will be filled with the (external) name of the position selected. The measure will not be manipulable other than through launching the widget to select a position.

Usage

In the Measure Tool, the user declares that the measure is using the single hierarchy selection widget, and the hierarchy and dimension from which the position will be selected.

The user can launch the single hierarchy selection widget from a cell for the measure in the RPAS client. The widget will contain the entire hierarchy that the position to be selected belongs in, down to the defined dimension, as in the scope of the workbook, which will therefore normally be a subset of the hierarchy as a whole.

When the widget is launched, the position that is the current value of the cell it is launched from will be selected in the widget, which will show just the hierarchical rollup in the active window from which the widget was launched (if the hierarchy is not present in the active window from which the widget was launched, a default hierarchical rollup will be used). The user may change the roll up being used using functionality similar to that on a spreadsheet window. When the user selects a position, any previously selected position in the widget is unselected. That is, there can only be a single position selected at any time.

Note: The value pasted into the cell is the (external) name of the selected position.

Since the measure is an ordinary string measure, not a picklist, the UI displays the contents of the cell (for instance, the name). If it is required that the user see the label (description) of the position, a further measure will be required that holds the label of the position, calculated using the attribute function.

Relative time period position queries in wizards

RPAS 11.2 provides the ability to select positions from the calendar hierarchy based on a relative offset from the current date (as defined by the system or overwritten by an RPAS environment variable). This functionality is available in all 2-tree wizard processes on the wizard page where the calendar hierarchy selections are made. This offset is evaluated every time the wizard process is executed, which allows for the calendar positions to “roll” across time.

Users will see this functionality in the calendar 2-tree of all wizard processes. Users can specify a start period and end period that is calculated as an offset from the current period, and all periods between the ‘start’ and ‘end’ are selected. Use a negative value to specify a period before the current time and a positive value to specify a period after the current time period. The end period should be a value that corresponds to a time period after the start period; however, if it actually corresponds to a time period before the start period, the system will switch the values around, so that the section of the calendar horizon between the two values will be selected.

The start and end periods can be saved as a position query (in the same manner as explicit user selections could be saved in RPAS 11.1). These position queries can be saved for an individual user or for all users (world access), and can be used (loaded) in other wizard processes, and used in autoworkbook builds.

Context sensitive dynamic picklists

Overview

RPAS 11.2 introduces the ability to have picklists where the values displayed for a given cell/intersection can vary based on the context of the current selection. These enhanced “dynamic” picklists follow the same concept as the dynamic picklists that exist in RPAS 11.1, but ‘look up’ the picklist values dynamically each time the picklist is opened, rather than looking up the values dynamically once during workbook build, but then having static picklists inside the workbook. The feature allows for the measure in which the valid picklist values are held to be at any level in the hierarchies and the values are effectively “looked up” using normal ‘non-conforming measure’ handling.

Since the valid values for a picklist for a cell are referenced from a measure dynamically, it is possible, if required, for the valid values of picklists to change during the life of the workbook, as a result of calculations or end-user edits. The value used will always be as at the last calculate, so direct or indirect (through calculation) edits to the picklist value measure are ignored when a calculation is pending.

Usage

In the Configuration Tools, the configurer will declare whether a picklist measure will be a dynamic picklist or a context sensitive picklist, and supply the name of the measure that holds the valid picklist values.

From the perspective of the end user, the use of context sensitive picklists will be identical to dynamic or static picklists.

Configurable measure formatting

To ease the effort of defining formatting options for measures, the Tools now provide the ability to define styles that can be assigned to measures. These styles have an inheritance structure such that each style inherits the settings of its “parent” similar to the way styles work in Microsoft Office products.

Styles are created in the Configuration Tools and are defined for a project. These styles are then assigned to measures in the Measure Tool (as an inheritable property), and can be further overwritten within a workbook window.

Note: A defined style for a given measure will overwrite any existing formatting in a given workbook.

The following formatting can be defined in a style within the Tools. These settings will be applied to the label and/or each cell of the measure (as applicable):

- Prefix – String to display as a prefix to values in cells. For example, enter “$” to indicate the monetary nature of a sales measure.
- Suffix – String to display as a suffix to values in cells. For example, enter “%” to indicate the value is a percentage.
- Scale factor – Integer value to act as a scaling factor for the measure. When you enter data for a measure that contains a scaling factor, the value you enter is multiplied by the measure’s scaling factor to obtain an internal value that is recognized by the server in data calculations.
- Precision – Formatting option to specify the number of digits to be displayed to the right of the decimal.
- Separator – Specifies the character to use as a separator between every three digits of a value in cells (usually either a comma or period).
- Text Font – Font in which the text of the measure’s label and/or cells will be displayed.
- Text Style – Text can be displayed as regular, bold, italic, or both bold and italic.
- Text Size – Size of the text.
- Text Color – Color of text for the selected measure’s label and/or cells.
- Background – Background color for the selected measure’s label and/or cells.
- Alignment – Specifies left, center, or right alignment for the selected measure’s cells.
- Border Style – Specifies the type of line (solid, dotted, or dashed lines) and to which edges to apply the border.
- Border Color – Specifies the color of the specified border for cells.

Note: The measure formatting background color for a measure takes priority in the RPAS client over the ‘read/write’ background color. Thus if styles are used through the tools, the ‘read/write’ color will not be seen. If a specific read/write color is desired, set that as the background color of the default style, and do not override it for any other styles. The ‘read only’ background color, on the other hand, takes priority over the measure formatting background color, so that ‘protection processing’ will be visible.

When building or patching a domain, the user can decide whether or not to apply the styles that have been defined.

Configurable Alerts

RPAS 11.2 provides the ability to configure alerts in the Tools. In RPAS 11.1 and prior versions, alerts had to be configured using a combination of the measure registration utility (regmeasure) and the alert manager utility (alertmgr).

Alerts are an exception management tool for users. An alert is a measure that evaluates a business rule (returning a value of true or false). RPAS then notifies users of the “true” conditions and allows users to build workbooks to resolve the scenario that drove the alert.

Alert measures are first defined for the domain. These measures are of type Boolean, which means they have a value of true or false. These measures are configured as regular measures in the Measure Tool.

Next, a rule (expression) or rule group is associated with a given alert to define the business rule or rules that are used to evaluate the alert. These rules and/or rule groups are defined in the Rule Tool like regular rules. They are then associated with the alert measure in the Measure Tool by clicking on the *Alert Expression* measure property and selecting the defined rule or rule group to use for evaluation.

Once the registration process is complete, the alert utility is run to “find” the alerts in the domain. See the RPAS 11.2 Administration Guide for additional information. After the alert finder has been run, the user can view the identified alerts in the Alert Manager in the RPAS client.

Rule pooling

RPAS has introduced the concept of “rule pooling” in the Configuration Tools that allows a named rule to be defined only once and reused in multiple rule groups. Rules will be created in a “pool” within the Tools and managed by unique names. These rules can then be added to rule groups, and subsequent changes to a given rule are then reflected in all instances where that rule is used.

To help minimize the possibility of a duplicate rule, the Configuration Tool warns the user when attempting to create a rule that has the same measures as an existing rule, regardless of the ordering or the calculation. The user then has the ability to use an existing rule or create the new one.

To summarize, the following capabilities have been provided:

- Create and name a unique rule that is added to a pool.
- Add a rule from the pool to a rule group.
- Edit a rule and have changes reflected in all instances of that rule.
- Remove a rule from a rule group (does not delete from the pool).
- Delete a rule from a rule group (deletes the rule from the pool and all rule groups where it is used).
- Validation – warns the user when creating a rule that has the same measures as an existing rule.
- Provide the capability of visually inspecting the contents of the pool with the ability to search by rule name starting string or rule measure content. This also includes displaying the list of rule groups that use a given rule.

Measure pooling

RPAS has introduced the concept of "measure pooling" in the Configuration Tools that prevents the use of duplicate measures with different characteristics or measure properties. This is particularly valuable when wanting to 'share' a measure between solutions in the same domain. For example, making a financial plan target visible in an item planning solution.

Rule group simulator

Background

The RPAS calculation engine is powerful, complex and configurable. The 'rule group' approach means there are very many potential calculation paths. However, during any configuration exercise, there is a significant design verification cost to ensure that the behavior is 'as would be expected' by an end user. In the absence of any rule group simulation functionality in the configuration tools, verification of the behavior of a rule group (between measures, not the calculations themselves) can only be undertaken by physically building domains and workbooks from the configuration, and then testing. With a large and complex workbook, the time spent configuring, building domains and workbooks, testing and then reconfiguring, and going round the cycle again may be significant.

Overview

RPAS 11.2 introduces the Rule Group Simulator, which is integrated into the workbook tool. When launched from the workbook tool, the simulator uses all the measures used in the rule set in the workbook, which may be more than those mentioned in the rule group.

Note: The rule group simulator is not currently able to simulate the expressions that will be evaluated as the result of a rule group transition, or simulate the calculations that will follow if a rule group is evaluated in 'full' mode, such as when evaluated from the mace utility, or the evaluation of the load rule group when a workbook is built.

Users of the rule group simulator are expected to understand the calculation cycle, especially with respect to measure protection processing and the process that determines which expressions will be evaluated. For more details see the calculation engine user guide.

The feature is provided in a separate window with two areas: a measure table, and a 'tree window' with upstream and downstream panes.

The measure table shows all the measures in the scope of the simulation. The first column of the table gives the measure name and (by color coding and a 'tool tip') the status of the measure. All measures can be shown, or the list of measures can be filtered according to status (such as only those that will be calculated, those that are read only, or those that are on a specific worksheet), or by name (those that include a specific string). Measures that are currently manipulable can have their status toggled (to or from 'edited') by clicking on them, and the simulator immediately updates all statuses, calculations, and trees.

The second column of the table shows the 'calculation' for each measure. For those measures that would be calculated if the user issued a 'calculate' with the current collection of edited measures, the expression that would be used to calculate the measure is shown. For measures that would not be calculated, this column displays the status of the measure. Selecting a measure in this column of the table makes it the 'active measure' that is used as the focus of the tree windows.

The tree window shows, in separate panes, the upstream and downstream measure relationships (for instance, the expressions that will be evaluated) for the measure with focus, which can be changed by clicking on the calculation column for the measure or the measure in a tree pane. If the measure with focus is calculated, the upstream pane shows the expression to calculate it, and, recursively, all measures that it are dependent upon (calculated from) with their expressions if appropriate. The downstream pane shows measures that are dependent upon (calculated from) the measure with focus, if there are any.

Usage

The simulator can be invoked from the Workbook Tool by first selecting a Rule Group from the Workbook Properties table. Right click the mouse to bring up the context menu and select "Simulate Rule Group" to bring up the Simulation dialog.

The status of each measure, given the current set of 'changes', is shown in the simulator measure table through color coding, and tool tips. The color scheme is as follows:

Color	Meaning
Yellow	Edited. If it is a recalc measure, it will be calculated by indirect spreading of another measure through a mapping rule and recalculation at aggregated levels. If the measure has another aggregation type, it will be calculated by spreading and aggregation.
Pale Grey	Editable. However, although the measure is not forced, and thus is still editable, it will be calculated through the calculation engine having to select an expression in an affected rule.
White	Editable. Will not be calculated, and thus will not change at all.
Pale Blue	Protected by protection processing. Although the measure is protected (usually this will be because it is the measure on the LHS of the only expression in a rule), it is not 'forced' because none of the RHS measures are changed, so it does not need to be calculated, and will not change at all.
Mid-Blue	Protected by protection processing. Is 'forced' and so will be calculated.

Color	Meaning
Dark Blue	Read Only. The measure is set as being read only in the measure properties, and will not change at all (a measure that is read only, but is going to be calculated will be shown as mid-blue, as that status takes priority over read only).

Note: The status of a measure encapsulates two concepts that are not as closely linked as may appear at first sight: whether or not the measure is manipulable (shades of blue = no, white/grey/yellow = yes) and whether or not the measure will be calculated. It is perfectly possible for a measure to currently be manipulable, but, nevertheless it would be calculated if a calculate were issued. Similarly, it is perfectly possible for a measure to be protected by protection processing that would not be calculated if a calculate were issued.

Measures that are currently manipulable and not already changed (white and grey) may have their status changed to edited (yellow) by clicking on their names in the measure table. All other measure statuses, calculations and the tree panes are updated. Similarly, a measure that is currently edited (yellow) may have that edit 'removed' by clicking on the measure in the measure table. The measure reverts to another status (white or grey), and all other measure statuses, calculations, and the tree panes will be updated. In addition, a reset button is provided so that the user can start a new simulation from a clean slate with no measures set as changed.

The simulator provides a mechanism for filtering the measures. Using a drop down box, the user may opt to show:

- All measures
- Only those that will be calculated (arranged according to order of calculation)
- Only those that will not be calculated
- Read only measures.

Furthermore, there is a "Contains String" filter, which can be used in conjunction with a search text field. Selecting this filter and typing a string to filter by updates the table to display only measures that contain the filter string. If the user selects the "All" filter and types, a search string on the text field, the first measure found containing the search string becomes the selected measure. A worksheet allows the user to view only the measures in a given worksheet.

The tree panes show measure dependencies through expressions that will be evaluated based on a single measure that has focus. The measure with focus can be selected by clicking in the calculation column of the measure table for the measure, or by clicking on the measure name in a tree pane. When the focus changes, the tree panes are redrawn as appropriate to the new focus, and the measure table scrolls so the measure with focus is shown.

If the measure with focus is calculated, the 'upstream' pane shows the measures that it is dependent on (calculated from, directly and indirectly). This is shown by a 'tree' structure with the measures used to calculate an individual measure showing as 'children' of it. If such children are also calculated, they will be shown with their dependent measures, and so on, recursively. Thus the 'upstream' pane fully explodes all the measure relationships that affect the measure with focus. If the measure with focus is not calculated, the 'upstream' pane will display that measure as a leaf by itself indicating that it is not dependent on any other measures.

The downstream pane shows measures that are dependent upon (calculated from) the measure with focus, if there are any. Measure relationships (expressions) are again shown in a parent-child ‘tree’ structure. Since there may be several measures that are immediately dependent on the measure with focus (that is, the measure with focus may be on the RHS of several expressions that will be evaluated), these relationships cannot all be shown at the same time in a simple tree structure. A ‘VCR control’ allows the selection from several immediate ‘dependent measures’ by paging through their trees in the downstream pane. Since the multiple dependent children problem may occur at all levels, below the first level of ‘dependent measures’ the tree for just the first dependent measure is shown. Thus the ‘downstream’ pane does not, indeed cannot, fully explode all the measure relationships that are affected by the measure with focus. The full explosion of detail can only be viewed by ‘walking down the tree,’ changing the focus to the desired ‘dependent measure,’ and selecting the sub branches through the VCR control.

The simulator displays multiple result expressions by indicating this beside the rule’s name in the second table column. The second table column also displays the entire expression showing all the left hand measures that comprise the multiple results. If a measure that has focus is one of the multiple result measures, it will be shown in the tree displays as MeasA [+MeasB][+MeasC] where MeasA is the measure in focus and MeasB and MeasC are the other multiple result measures. Measures that are calculated in a ‘cycle,’ which typically includes BOP and EOP inventory values, are indicated with an ‘*’

Configuration task list

RPAS 11.2 has introduced the concept of a “task list” that provides a single location for all configuration issues encountered when configuring a solution. When configuring a solution, it is common to run into situations where a given configuration setting is invalid. This can be for a wide range of causes and can often be difficult to troubleshoot. The Task List is intended to ease this troubleshooting process by providing a location where all outstanding configuration-related issues can be seen.

The task list is a partition in the bottom half of the main window of the Configuration Tools and appears by default when opening the Tools. The window can effectively be minimized by clicking on the “Task List” button of the window, or shrunk or enlarged by dragging the divider between the top and bottom windows.

The Task List can identify two types of configuration errors: local errors and remote errors. A local error is one that is caused when actually making an edit, such as a syntax error or a duplicate name. A remote error is the result of an edit where the edit itself is not necessarily invalid, but where the edit causes another configuration setting to become invalid.

The configuration errors checked by the Task List are based on validation within the Tools and validation provided by the interface between the RPAS server and the Tools. The number of possible configuration errors is large and is therefore not explicitly listed in this document.

Reporting configuration information

The Configuration Tools provide the ability to generate reports about a configuration in RPAS 11.2. These reports contain a variety of information about a configuration and are output to a text file that can be read into other applications for documentation or inspection purposes.

The Report Generator is run from the *Utilities* menu of the Configuration Tools. Once inside the Report Generator, you must select from the following reports (note that additional reports will be added in future releases):

- Measures
- Data interface
- Measure descriptions for translation purposes
- Measure labels for translation purposes
- Rules
- Rule group labels for translation purposes
- Workbooks
- Workbook group labels for translation purposes
- Workbook labels for translation purposes
- RPAS messages for translation purposes

Miscellaneous changes to existing tools

Rules

- Previously, adding a Rule to a Rule Group resulted in the Rule being added at the end of the list in the GUI. The Rule is now inserted at the currently selected location.
- Up/Down buttons are now available on the toolbar to make moving Rules easier. Previously, if you wanted to move a rule several places up or down, you would have to click once on a small arrow beside the Rule name, and then mouse over to the new location of the arrow for the next click. The tool bar buttons still require multiple clicks, but there is no longer a need to move the mouse before each click.
- Autoscrolling is now available whenever moving Rules in a small window.

Measures

- If the database field is populated for a measure, the Tools will prepend these values with “data\” in the configuration file (cfg).

Workbooks

- When creating position queries in the tool, the user can now bring up all the Boolean measures with more than one dimension in their intersection. There is no longer a need for the measure to be in the Viewable Profile before it can be selectable as a position query measure. Validation was added to prevent selection of a measure whose intersection dimensions did not comprise a subset of the sheet intersection dimensions. Validation was also added to flag a previously defined position query measure whenever a change in sheet intersection makes the measure invalid for the given sheet.

Technical enhancements

Centralized and parallelized measure loading and export

The RPAS utilities **loadmeasure** and **exportdata** have been updated so that they only need to be run in the master domain of a Global Domain environment.

The **loadmeasure** utility has also been enhanced to incorporate the process of splitting input files so that the data is loaded into the appropriate sub-domain when the measure loading utility is called. Previously, the splitting process was completed using the **splitmeasuredata** utility, which has been deprecated.

Both utilities have a new argument to run multiple processes in parallel across local domains. The –processes argument should be included in the call to the utility and determines the number of additional child processes that are run in addition to the main process (in the master domain). The number of processes should be less than or equal to the number of processors on the server.

Loadmeasure will take an input file and split it into multiple files based on the mapping of positions to their respective domains. Data for positions that belong to the master domain (for instance, for measures with a higher base intersection than the partition level) will be loaded in the master domain.

Note: It is possible to load a single input file or to split the data into multiple files before running **loadmeasure**.

exportdata allows some arguments to be specified in a file in the domain instead of the command-line via the '-params' argument (and entering the arguments individually). If run in a Global Domain environment, the array specified by this argument must be in the master domain, not the local domains.

Note: The –array argument is not supported in Global Domain environments.

New utilities

The following list of utilities new to 11.2 is documented in a different section of this document:

- dimensionMgr – included in Hierarchy administration and position repartitioning

Managing settings for hierarchies – hierarchyMgr

The hierarchy manager utility can be used to set the following properties:

- Enabling/disabling position-level security for a dimension in a hierarchy (other than calendar)
- Allowing the use the same name for hierarchies and dimensions

  Syntax: **hierarchyMgr** –d domainPath –h hierarchyName [COMMAND]

Enhancements to existing utilities

Ability to define a named rule using calc engine utility – mace

RPAS has added the ability to add a named rule to the domain using the calculation engine utility **mace**.

Syntax: **mace** -d domainPath -newRule {-ruleName ruleName}

Filtering out values when exporting data – exportData

An enhancement has been made to the RPAS utility **exportData** (new parameter **–precision**) that allows a user to ignore (not export) values that differ from the NA value less than a specified value.

Note: This feature was also released in a patch to RPAS 11.1.

The purpose of this enhancement is for the export utility not to export values that vary only slightly from the NA value, most commonly very small values close to zero. For example, consider a numeric measure with a NA value of zero. If the precision value is specified as 0.01, any number smaller than that is ignored, so 0.0034 would be ignored while 0.34 would be exported.

The precision value must be less than one. If a value greater than one is provided, the utility returns a warning.

Evaluating (finding) a subset of alerts – alertmgr

The alert manager utility (alertmgr) was enhanced to allow the evaluation of a subset of alerts. The utility can be used to evaluate one or more individual alerts, or one or more alert categories (assigned when configuring the alerts).

Syntax: **alertmgr** –d domainPath –findAlerts {-alerts "a1 a2 …" | -categories "cat1 cat2 …}

Force the purging of data when loading measures – loadmeasure

A new parameter has been introduced (**–forcePurge**) for the **loadmeasure** utility that forces data to be purged when running the utility. Using this flag when running **loadmeasure** will ensure that the purge routine is run even when no new data is loaded. This will allow an administrator to "touch" an empty .ovr file and then run **loadmeasure** to just purge the measure without loading anything.

Reconfiguring the partitions of a Global Domain – reconfigGlobalDomainPartitions

Note: A number of changes were made to the reconfigGlobalDomainPartitions utility in RPAS 11.2. As such, the complete usage has been provided for clarification.

It is common for many customers to regularly add, remove, or change the parent-child relationships for positions in hierarchies, most commonly for positions in the product hierarchy. While this movement/reassignment of positions is normally handled with the loadhier utility, a special process must be followed for positions at the partition level of a Global Domain environment.

The following information is only relevant if an environment has been built as a Global Domain.

For example, consider a Global Domain environment that is partitioned at the "department" dimension/level in the product hierarchy. This environment could handle the addition or removal of SKU's, classes, or divisions without having to follow a special process. However, if a new department is to be added to the environment an administrator must run a special RPAS utility to add this position and load it into the hierarchy.

The following process must be followed to add or remove a position at the partition level of a Global Domain environment:

- Administrator must be notified in advance that a new position is being added to or removed from the system.
- Run the utility **reconfigGlobalDomainPartitions** to add/remove the positions by specifying the sub-domain to which the positions do or will belong.
- This utility calls the loadHier utility at the end of the reconfiguration process to apply the hierarchy changes to the domain. When adding positions (using the –add argument) the updated hierarchy file must be available in the input directory when the reconfig utility is called (otherwise this utility will fail). Updated hierarchy files are not required to remove positions (using the –remove argument).

Note: The use of this utility is only required for positions at the partition level. Positions can be added or removed below or above the partition level by being added or removed from the hierarchy file.

reconfigGlobalDomainPartitions –d pathToMasterDomain –add posName1,posName2, ... –sub pathToSubDomain

reconfigGlobalDomainPartitions –d pathToMasterDomain –remove posName1, posName2, ...

reconfigGlobalDomainPartitions –d pathToMasterDomain –input pathToInputDir

Argument	Description
–d pathToMasterDomain	Specifies the path to the master domain in a Global Domain environment.
–add posName1, posName2,	Adds one or more positions to a specified local domain. The path to the sub-domain must follow the list of positions to add.
–remove posName1, posName2, ...	Removes the designated positions from the sub-domain to which the positions belong.
–sub pathToSubDomain	Specifies the path to the sub-domain to which new positions are being added.

Argument	Description
–input pathToInputDir	Specifies the path to the input directory that contains an xml configuration file (reconfigpartdim.xml) to specify what positions to add. The file must have all the information to run the process including the command name, position names to add, and paths to the local domains. This option is useful for adding positions to multiple local domains. For now, only adding positions is supported with this argument.

Notes, assumptions, and limitations:

- Position names are separated by commas and must be valid **external** position names without the prefix of a dimension.
- The path to the sub-domain does not need to be specified if removing positions.
- When using the –input argument, the file must be in a particular format and must contain the “add” command, the path to each local domain to which positions are being added, and the list of positions for each local domain. The file must be XML and named “reconfigpartdim.xml.” Below is a sample file:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
  <rpas>
      <command name="add">
          <subdomain>
              <subpath>C:\Domains\test\ldom0</subpath>
              <subpositions>30</subpositions>
          </subdomain>
          <subdomain>
              <subpath>C:\Domains\test\ldom1</subpath>
              <subpositions>40,50</subpositions>
          </subdomain>
      </command>
  </rpas>
```

- The utility backs up the required data and will automatically restore the domain to the original state in case of failure.
- Positions can only be added or removed in EXISTING local domains.
- You have to specify the local domain to which the positions are being added.

- The remove option will NOT allow the removal of all positions at the partitioning level from a sub-domain.
- Multiple positions can be added to **a single sub-domain** in a single call to the utility using **–add** option.
- Multiple positions can be added to **multiple sub-domains** in a single call to the utility using **–input** option.
- Currently the –input argument only supports the addition of positions.
- Positions can only be added or removed in a single call to the utility (the –add and –remove argument cannot be used in the same call).
- When adding positions, an updated hierarchy file must be available in the input directory as the loadhier utility is called after adding positions at the partition level. If the updated hierarchy file is not present in the input directory when attempting to add positions, the utility will fail.
- No updated hierarchy file is required when removing positions (technically the utility creates an empty hierarchy file). If a hierarchy file is in the input directory, the utility will back up this file.
- A log file (loadHier.log) will be created in the root directory if loadHier fails.

Deprecated utilities

The following utilities are no longer available in RPAS 11.2. They are either no longer required or have been replaced by other utilities.

- reshapearrays – The reshaping process has been incorporated into the hierarchy load process. A new utility **defrag** has been provided to be used when it is necessary to defragment or clean up a domain.
- inithier – The inithier utility was used for enabling or disabling position-level security and for extending the width of positions for a dimension (to allow for position names longer than 24 characters).
  - Position-level security is now set using the **hierarchyMgr** utility
  - The width of position names can be extended using the **dimensionMgr** utility
- splitmeasuredata – In RPAS 11.1, the process of splitting the data of an input file for loading into local domains was completed using the splitmeasuredata utility. This process was added to the **loadmeasure** utility so that a single input file can be used to load data into the master domain of a global domain environment.

Miscellaneous technical information

RpasInstall Logging

A verbosity setting was added to rpasInstall (-verbose). Currently, this setting only relates to information output by createRpasDomain. The default is that the parameter is not set, which means only errors and warnings will be output. If -verbose is provided, the utility will output information in addition to warnings and errors.

RPAS lock timeout variable

RPAS introduced a new environmental variable named “RPAS_LOCK_TIMEOUT.” This variable was added to allow an administrator to set how long they want to wait before timing out when there are lock contention issues. This most commonly occurs when two users attempt to simultaneously commit/save the same data.

This environment variable is set on the back end to the number of milliseconds to wait for a file lock before returning a lock contention error. The default value is 60000 milliseconds (one minute). As with any environmental variable, the variable must be set prior to starting the process that uses that variable. The variable was introduced for use with the RPAS database server, which means that the variable is set for the DomainDaemon.

For example, the two lines below indicate how an administrator would tell RPAS to wait two minutes before returning a lock contention error with the RpasDbServer after launching the client and logging in. Any client that connects to that domain daemon would see lock contention after a two minute delay.

```
Export RPAS_LOCK_TIMEOUT=120000

DomainDaemon -port 55123  -start -debug &
```

New Application Programming Interfaces

Disabling wizard page controls

RPAS now provides the ability to programmatically disable wizard page controls in custom wizards.

The class WizardPage has a new method enable(String, bool) that allows an application developer to enable or disable a control through a C++ API. The first parameter is the control name, and the second parameter is 'true' for enabled and 'false' for disabled.

Date control for wizard pages

Wizard pages now support date control. Below is a list of types with definitions and examples.

- Date picker control. The parameters are identical to that of addEdit(...)

  ```
  bool addDatePicker(
        const String& controlName,
        const String& styles,
        long xStart, long yStart,
        long width, long height);
  ```

- Use the following to specify the initial value of the control. If this method is not called, the wizard page framework will set the initial value to the previously selected value. If there is not a previously selected value, the control defaults to the current date.

  ```
  bool setDate(
          const String& controlName,
          const SimpleDate& date);
  ```

- The method below allows the date picker to be ranged. The dates specifed are inclusive with respect to the range.

```
bool setDateRange(
        const String& controlName,
        const SimpleDate& startDate,   // inclusive
        const SimpleDate& endDate);    // inclusive
```

- The method below allows a wizard page to get the value "picked" by the GUI.

```
bool getDate(
        const String& controlName,
        SimpleDate& date) const;
```

Hiding positions in a worksheet

A new API was added that allows an application developer to hide positions in a worksheet. The information below describes this new feature and how it can be utilized.

Note: This feature was original introduced in RPAS 11.1.6.

A new interface “addWindowPositions” was added into WorkbookTemplate class. This function can be used together with preexisting function “setVisibleWindowPositions” to selectively hide window positions for newly built workbooks.

Application teams may use this function in their template to specify all window positions first and then to set the visible position by invoking “setVisibleWindowPositions.” All unmentioned window positions will be hidden.

```
void addWindowPositions(const String& windowName,
                        const String& dimName,
                        const String::SetT& positionNames);
void setVisibleWindowPositions(const String& windowName,
                               const String& dimName,
                               const String::SetT& positionNames);
```

Below are some answers to some anticipated questions about this feature:

1. What happens if the positions already exist when I call addWindowPositions?

   Existing positions will not be added again, and retain their previous state. Basically, this function creates the window position array (arrayName = winName + dimName) if it does not exist and add positions if they are not already in.

2. When I call addWindowPositions, are the positions hidden or visible by default?

   The positions added by “addWindowPositions” are visible by default. However, calling “setVisibleWindowPositions” will set mentioned positions visible and all other unmentioned existing positions hidden.

3. If you have already saved some styles, what is the effect of calling these two functions? Does it just affect visibility of the positions, or does it override some other kind of format too? (for instance, width or something else)

   It just affects visibility of the positions.