ODABA Releases TODBMS and Tools 13.0.0

The latest version of the Terminology-Oriented Database Management System (TODBMS) ODABA has been released on Monday, August 31, 2015. The new version ODABA 13 comes up with a lot of extensions and improvements. Most essential features are version management for database entries and reimplemented client/server interface supporting full interface functionality with improved event handling. Tools have been improved by extending and improving a number of features.

Several service classes for file and internet access have been provided in order to improve support for OSI development. ODABA functionality as well as ODABAGUI API have been extended. Essential changes have been made for the installation in order to simplify running examples. Several minor improvements have been made for ODABA database and ODE tools.

Finally, many bugs have been detected and removed. More details are described in change logs and in notices delivered with the development databases (ODE tools: **Object/Notices**). Notices delivered with the databases also contain a list of open topics planned for next releases. Notices are stored separately for basic functions (**sos.dev**), database kernel (**opa.dev**), GUI framework (**gui.dev**) and ODE tools (**ode.dev**).

Detailed changes (ODABA)

The most important changes on database kernel are the introduction of flexible version management. ODABA now supports database entry versions following different strategies applied to different versioning scopes (e.g. synchronized versioning, individual versioning etc.). The versioning strategy may be chosen depending on application requirements for a database.

The client/server interface has not only be completed but basically redesigned in order to improve data synchronization between server and client and enhanced event handling by returning server events also to client. The client/server interface is a function-oriented interface, i.e. each database access class function may be executed on an ODABA object server. Executing OSI expressions is possible on client but also on server side.

Several extensions have been made for to improving the usability of the ODABA API. Moreover, OSI functionality has been extended in order to make using OSI more comfortable, but also for simplifying the use of the OSI debugger, e.g. showing broken stacks when running OSI event handlers.

Most of the features result from practical requirements met in several customer projects. Several mixed code applications (OSI/C++) have been build and transferring OSI code to C++ and reverse got more support by appropriate macros and template classes, which allow nearly identical coding in C++ and OSI.

ODABA Database kernel (base)

This version provides several changes and extensions for the kernel functionality:

- Extended support for data versions
  ODABA now supports a large number of versioning strategies for creating versions for database entries (instances and indexes). Automatic data versioning is supported for different scopes and data versions may be synchronized.
- Client/Server interface
  The client server interface has been completely rewritten in order to provide support for all interface functions. Client/server applications use bidirectional data and event cache, i.e. each client/server communication besides data also carries all updates and generated events created an client or server side.
- Support 32-bit enumerator codes
  In order to support 32 bit enumerator codes, a new enumerator encoding type (access type AT_ENUM) has been introduced, which causes enumerator valued to be stored with 32 bit (instead of 16 bit). This type becomes the default enumerator type. Old enumerations still store 16-bit enumerator codes, only.
- System output cache
  System output cache allows caching system messages to be submitted (or suppressed) later on. The feature is used in client server applications in order to return system massages created on server side back to client.

- CheckDB allows restoring deleted entries instead of removing reference to deleted entry depending on new option.
- Initialize application options when opening a database
  Options are initialized from settings in COMMON and user specific configuration (options) when opening a database. This includes options defined in an ini-file, but also options stored in the database. .
- Maximum size for data area in 64-bit applications has been increased to 256 TB
- Allow pointer attributes in transient attributes
- System depending or application defined default line breaks
- Enhanced function initialization
  Instead of replacing option variables, only, function templates may be defined as OSI templates, now, which may include OSI code for more complex expressions.
- Full support for XML configuration files including storing and restoring configurations.
- DATETIME is not anymore considered as atomic type but as complex data type
- Make extended shell calls available in Linux environment
  In order to support the executeShell function under Linux, file associations (extension - application) may be defined in option section FileAssociations (top level).

Data model extensions made for version 13 are compatible with ODABA 12. Some header data has been changed in order to support data versions. When accessing ODABA 12 data bases with ODABA 13 libraries, header will be upgraded and database is not accessible for ODABA 12 software, anymore. Following model extensions have been made on system level:

- SDB_Resource::display_name (transient attribute)
  For ODC_Variables, the display_name is set to the first synonym when being defined for a variable.
- Primary key for ODC_Project and ODC_Module changed to ik_Ident8

ODABA Application Program Interface (base/opa)

Several extensions and changes have been made to the ODABA API:

- Systematically changes from non const string parameters to const string parameters in all interface classes
- Pre-load OSI script files from external directory
  In order to add new or overwrite existing OSI functions, one may provide external OSI script files in one or more directories, which are loaded when calling **`Dictionary`**`::loadLibraries`
- Make extended shell calls available in Linux environment
  In order to support the `executeShell()` function under Linux, enhanced file associations (extension - application) may be defined in option section `FileAssociations` (top level).
- **`Application`**
  - cacheOutput (new)
  - clearOutputCache (new)
  - initialize.client (new)
  - outputCache (new)
- **`Client`**
  - cacheConnection (new)
- **`DataSource`**
  - databaseConnection (new)
  - dictionaryConnection (new)
  - resourceDatabaseConnection (new)
- **`Dictionary`**
  - extentDefinition.nsid (new)
  - loadODLSchema (renamd from loadSchema)
  - loadOSIScript (renamed from loadOperation)
  - loadOSILibraries (new)

- **`Database`**
  - createMajorVersion (new)
  - createVersion (new)
  - lastMajorVersion (new)
  - lastVersion (new)
  - majorVersionBegin (new)
  - majorVersionCount (new)
  - majorVersionEnd (new)
  - resetMajorVersion (new)
  - resetSchemaVersion (new)
  - resetVersion (new)
  - toMajorVersion (new)
  - updateMajorVersion (new)
- **`ObjectSpace`**
  - accessVersion (new)
  - commitTransaction.version (new)
  - createDataVersion (removed)
  - dataVersion (removed)
  - dataVersionBegin (removed)
  - dataVersionEns (removed)
  - executeExpression.server (new)
  - majorAccessVersion (new)
  - resetDataVersion (removed)
  - resetVersion (new)
  - setAccessVersion (new)
  - setDataVersion (removed)
  - setMajorVersion (new)
  - updateDataVersion (removed
  - updateVersion (new)
  - version (new)
- **`Property`**
  - accessVersion (new)
  - checkCollection (new)
  - closeAll (new)
  - collectionLocked (new)
  - dataVersion (removed)
  - eventHandling (new)
  - executeExpression.server (new)
  - extensionProperty.name (new)
  - instanceLocked (new)
  - instanceVersion (removed)
  - lockCollection (changed)
  - lockInstance (changed)
  - nextVersion (new)
  - previousVersion (new)
  - propertyVersion (removed)
  - registerHandle (removed)
  - removeOrder (new)
  - resetVersion
  - save.overwrite (new)
  - setAccessVersion (new)
  - unregisterHandle (removed)
  - updatePropertyVersion (new)
  - updateVersion (new)
  - version (new)
- **`PropertyDefinition`**
  - internalTypeID (new)

    - TypeName (new)
- **`TypeDefinition`**
    - lookupType.name (removed)
    - lookupName (renamed from lookupString)
    - lookupCondition (new)
    - lookupLabel (new)
    - lookupTitle (new)
    - instanceVersioning (removed)
- **`BaseContext`**
    - executeCommand (new)
    - executeProgram.array (new)
    - executeShell (changed/new)
    - parent (renamed from highContext)
    - hasSystemContext (removed)
    - resetSystemContext (removed)
    - systemContext (removed)
- **`DBBaseContext`** (event handling for client/server solved in a better way)
    - notifyClose (removed)
    - notifyCreate (removed)
    - notifyDelete (removed)
    - notifyDirty (removed)
    - notifyInsert (removed)
    - notifyOpen (removed)
    - notifyRemove (removed)

The behavior of **`odaba::String`** class has changed:

- Use code points rather than units for string operations
  Instead of using units with fixed size depending on character encoding types, now code points (characters) are used for text operations. Thus, string operations become independent on encoding type.
- When system encoding under Linux is UTFxx, the default encoder is the appropriate UTF encoder (instead of local encoder).

Several new service classes have been provided especially for improving OSI application features:

- **`File`** (Basic file access functions)
    - Support recursive file functions
    - Support BOM for text files
- **`BinaryFile`** (Special access functions for accessing binary files)
- **`TextFile`** (Special access functions for accessing text files)
- **`IniFile`** (Functions for reading, updating and writing configuration or ini-files)
- **`ZipArchive`** (Functions for reading and writing files from/to zip files)
- **`XMLString`** (DOM tree access to XML strings and files)

More details are described in ODABA online documentation: **Reference documentation/ODABA Application Program Interface.**

ODABA Script Interface OSI

Besides several new service classes and functions, OSI debugging has been improved.

Now, OSI supports implicit file or I/O operations, which can be called from any OSI functions. One way to read data from or write data to a file is the **`File`** object. In order to provide more specific support for special file types, more specific file classes are supported by OSI:

- **`TextFile`** - for supporting text files (ASCII, UTF8)
- **`BinaryFile`** - for supporting binary files
- **`ZipFile`** - Supporting zip archives in C++, .Net and OSI
- IniFile - for supporting ini- or configuration files (classical ini-file or XML configuration file)

In order to access file system directories, a platform independent definition for the **`DirectoryEntry`** data type has been provided, which allows opening directories directly in OSI applications without any extensions in the application's data model.

There are two more OSI interfaces provided for supporting internet functionality.

- **`HTTP`** - post ant get functions in order to read data (files) from the internet or request WEB services
- **`Email`** - Sending and receiving emails via OSI applications (extended)

With **`XMLString`** OSI provides a simple interface for reading, writing or updating XML files, which supports hierarchical XML elements ans attributes for each element.

The check feature for checking OSI functions has been improved in order to provide proper error location.

The OSI debugger now also shows "broken stacks". When running OSI applications, OSI functions may call database functions (C++), which again may call OSI context functions. In order to display the complete calling statck for OSI functions, the gap caused by C++ calls has to be bridged.

Open document support

Some minor bugs have been fixed.

Detailed changes (ODE and GUI framework)

Some changes and improvements have been made on existing tools. Minor changes have been made in the GUI framework. Important extensions are:

- Support generic comment generation
  By overloading a few OSI functions with application specific comment styles, one may define kind of style guides for code comments in generated source files depending on programming language.
- Actions for version navigation
  Default actions for navigating to previous or next version for selected instance in a property handle

Besides, some minor bugs have been removed, which are reported in the change log.

GUI Framework (gui)

The GUI framework kernel had been changed slightly. Essential changes are provided by action-log support.

- Update line breaks

  When editing memo properties, line breaks are updated according to system requirements or settings in `DefaultLineBreak` option or environment variable.
- Directory path dialog added

ODE tools (ode)

Ode tools have been changed slightly in for better supporting mixed coding (OSI, C++, C#)

- Collect changes in commit notices (ClassEditor)
  In order to collect changes related to a certain topic of change, a notice can be assigned as current commit notice, which collects all following changes.
- Extended OSI function check features in ClassEditor
- Collect changes in commit notices (ClassEditor)
  In order to collect changes related to a certain topic, a notice can be assigned as current commit notice, which collects all following changes.
- Support generic comment generation
  By overloading a few osi functions with application specific comment styles, one may define kind of stylguides for code comments in generated source files depending on programming language.
- Extended OSI function check features in ClassEditor
  Updated OSI functions and classes are maked in class editor. Cecking OSI class functions always checks all updated functions.
- Actions for import/export current settings in option dialog

- Preload OSI script files provided in external directories when starting ODE tools
- Versioning support
  In order to support database versiong, the Versions menu has been added allowing listing, creating, setting, resetting and changing selected version. Moreover, dersion display and navigation functions have been added to tool and status bar. Version notices can be created in order to document database versions.
- Extended Find/Replace dialog
  The Find/Replace dialog has been extended by a droplist containng last 30 find/replace expressions.
- Support storing tree actions
  When calling an action from a resource tree (**Selection** or **Run Action** in context menu) in Designer or ClassEditor, the defined action may be stored for future re-use. Stored actions may be selected for re-running or update.
- New edit options for customizing code implementation rules.
- Improved generation of OSI interface functions and function templates

ODABA GUI Application Program Interface (gui/ode)

Additional functions in GUI context interface are

- **`ControlContext`**
  - directoryPath (new)

ODABA Documentation

Documentation has been extended. Especially, documentation versioning and client/server have been extended. If anybody needs specific topics or areas to be documented, we will consider this in our documentation priority list.

Installing ODABA

ODABA, including applications and libraries, is available for free under Open Source licenses (GPL). ODABA runs on various hardware configurations, operating systems and works on many desktop environments. ODABA can be obtained as source code distribution and in various binary formats from http://sourceforge.net/downloads/odaba/.

Several features require third party components, which have to be installed before installing ODABA. When the corresponding libraries are not available, one may install ODABA, but the features referenced below will not work.

- libzip - required for LibreOffice document generation
- zlib - required for data compression and database backup and restore)
- curl - required for enhanced email support)
- hunspell - required for spell check in ODE tools, like terminus

Previous Releases

When running ODABA 11.x.x or higher, no upgrade is necessary. When still using ODABA 10.x.x, resource databases and databases referring to ODABA system data types need to be upgraded. Details about how to call a database upgrade are described in the readme file for the ODABA 11.0.0 installation.

With the release of ODABA 13.0.0 we declare the end of live for all previous released ODABA versions. Bug fixes on 12.3.x version are provided on demand.

Important: Running databases with ODABA 13 in write or update mode will upgrade the database header automatically. After upgrading the header, the database cannot be used with ODABA 12 or older.

System Requirements

In order to get the most out of this release, we recommend to use a recent computer with at least 1 GB of memory and 2 GHz CPU or better. In order to install the binaries, about 100 MB are required. Installing sources requires about 50 MB. 80 MB are required in addition, when installing the documentation locally.

About RUN-Software

RUN-Software develops database management system ODABA and tools since 1994. Besides general and particular software solutions, RUN-Software publishes theoretical works about database theory and terminology in connection with data modeling.

See also: www.run-software.com