Real Time Computersoftware Ges.m.b.H

RTAPLS V7.2
Real Time Architecture Process Control System

Brief Description

Christian Riedl, RTS
2007/11/09

1 General Information

RTAPLS is a program package used to process data in a process control system. RTAPLS is primarily employed for complex, continuous processes in the chemical industry as well as in the area of energy generation and distribution. In terms of its functional efficiency and data range, RTAPLS is a top of the range program package.

The essential units of information in such a system are the process variables (= PVs). Hence, these PVs are also the objects on which RTAPLS operates. All objects are stored memory-resident in the Real Time Database (RtDb). The RtDb can be distributed to a large number of computers by means of replication. Thus, the RTAPLS objects are, like the PVs, distributed objects.

Typical RTAPLS installations contain 10,000 to 100,000 process variables and can process several thousands of process data modifications per second.

The core functionality of RTAPLS is implemented in C++ for platform independence and highest performance.

RTAPLS based applications can be developed using Microsoft COM, .NET and OLEDB technology. RTAPLS Web Services (W3C conformant) can be used to create intranet applications or may be integrated into enterprise applications according to Service Oriented Architecture (SOA).

RTAPLS supports OPC interfaces to integrate a lot of I/O systems and to make use of widespread Visualization Systems.

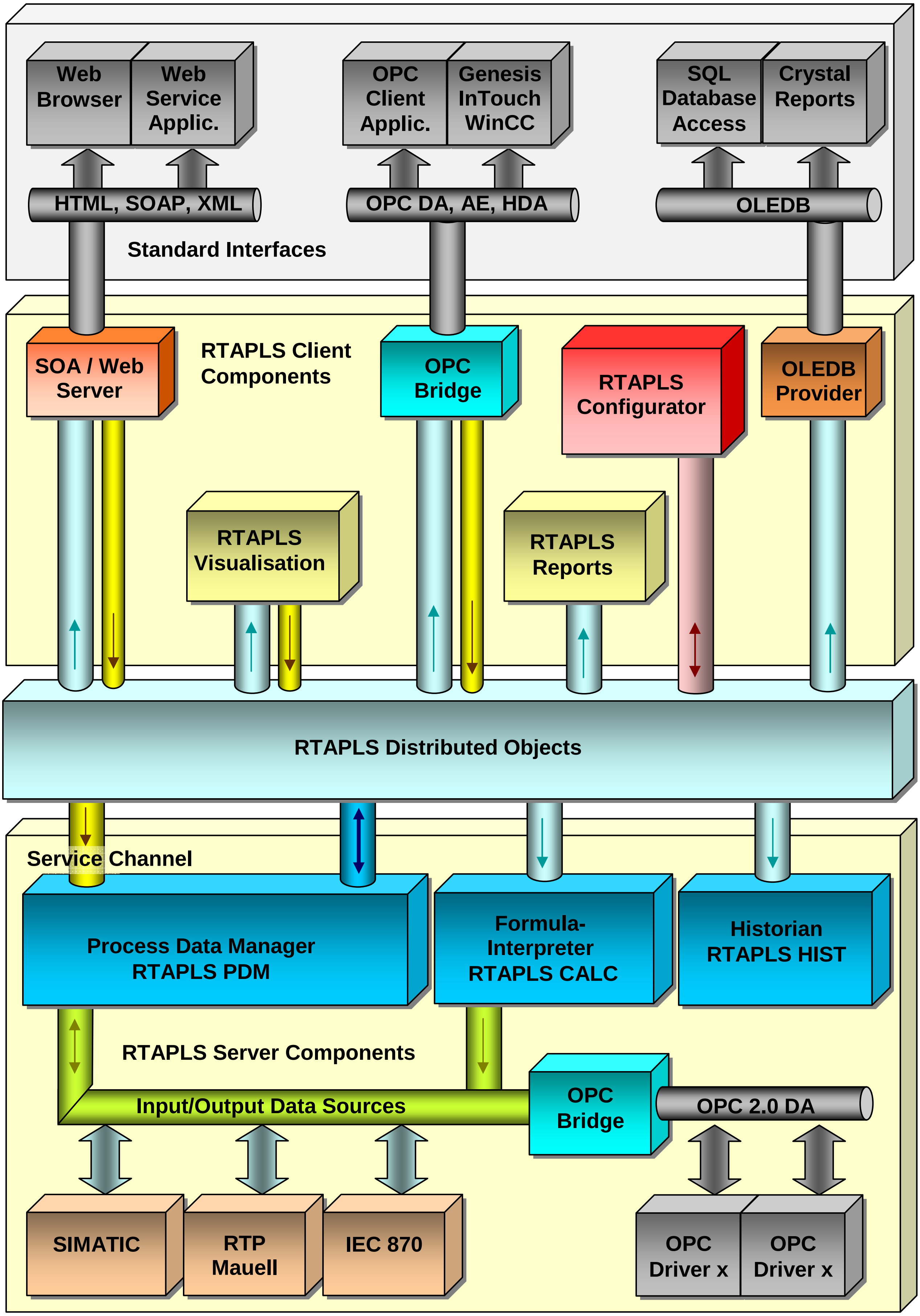
RTAPLS is multi-platform-capable and available on the major UNIX / Linux platforms, on VMS and Windows. Heterogeneous configurations (UNIX or VMS server and Microsoft Windows Clients) are possible, too.

RTAPLS can be configured online without service interruption.

RTAPLS supports redundancy: if the master computer fails, all functions will be provided interaction-free by the back-up computer.

RTAPLS provides an extensive range of diagnostics tools to display and evaluate system status, program status, error scenarios etc.

2 RTAPLS System Architecture



2.1 RTAPLS Components

- RTA: the real time platform
- RTAPLS - DBO: the distributed RTAPLS object model
- RTAPLS – PDM: the process data manager
- RTAPLS – CALC: the formula interpreter
- RTAPLS – HIST: the historian
- RTAPLS – REPORT: the information and report system
- RTAPLS – VIEW: the process Visualization
- RTAPLS – SOA: the Web and SOA Server
- RtOOS: the object-oriented engineering system
- The drivers (interfaces to the I/O systems)

2.2 Integrated Standards

RTAPLS integrates several standard technologies:

- ISO C++
- Microsoft Platform

COM/DCOM	compatible RTA / RTAPLS API
.NET	compatible RTA / RTAPLS API
OLE DB	database access to RTAPLS data

- OPC (OLE for Process Control) Specifications
  - OPC Data Access DA 1.0, 2.0, 3.0
  - OPC Alarm and Events AE 1.1
  - OPC Historical Data Access HDA 1.2
  - OPC Unified Architecture DA 1.0 (SOA Server)

- ORACLE Database
- Data import / export using XML Format
- Visualization based on SVG (Scaleable Vector Graphics)
- W3C Web Service (Service Oriented Architecture)
- SNMP Network Management

2.3 Real Time Architecture – RTA

RTAPLS is based on the Real Time Architecture (RTA). RTA serves as a software platform for the development of distributed redundant real-time systems. All functions relevant for real-time applications are provided based on standards irrespective of the used operating system:

- Real Time Database
- Real Time Interprocess Communication
- Real Time Redundancy Management

RTA supports Unix (HPUX, Linux, Solaris), Open VMS and Microsoft Windows 2000 / XP/ Vista / Server 2003.

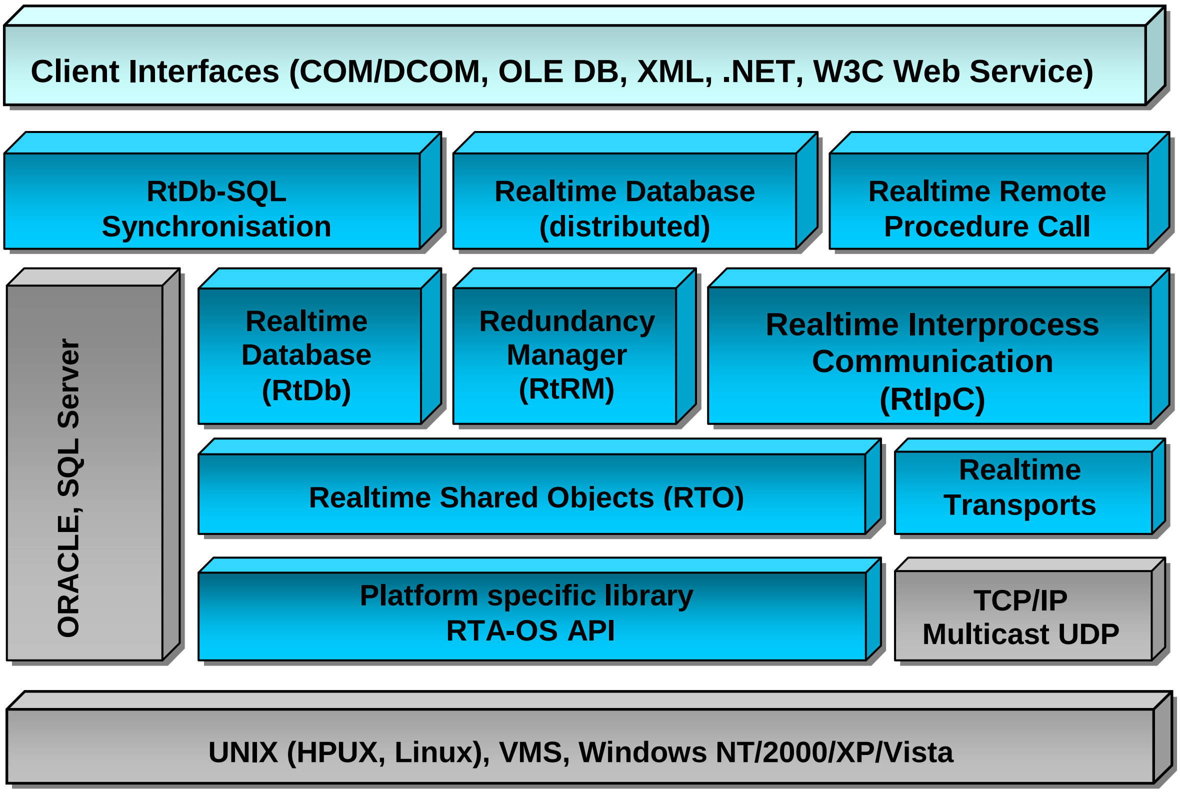
The real-time capabilitis of RTA are ensured by the performance of the Real Time Database and Interprocess Communication. Response times of a few milliseconds can be achieved without any problem.

Synchronization between the Real Time Database and an RDB (relational database) is possible. In this case, RtDb acts as "Cache", the Real Time Database is capable of speeding up the relational database by a factor of 100 (reference value). This means that the user does not have to employ additional hardware and that he can exploit all the advantages of his database technology for applications that, up to now, would have required the utilisation of conventional technologies.

RTA supports redundant configurations by duplicating the relevant servers, and also allows for heterogeneous configurations (e.g. VMS Server or UNIX Server, Microsoft Clients).

RTA is integrated in RTAPLS to support development of projects specific applications seamlessly fitting into the RTAPLS framework.

2.3.1 RTA – Components


Client Interfaces (COM/DCOM, OLE DB, XML, .NET, W3C Web Service)
RtDb-SQL Synchronisation
Realtime Database (distributed)
Realtime Remote Procedure Call
ORACLE, SQL Server
Realtime Database (RtDb)
Redundancy Manager (RtRM)
Realtime Interprocess Communication (RtIpC)
Realtime Shared Objects (RTO)
Realtime Transports
Platform specific library RTA-OS API
TCP/IP Multicast UDP
UNIX (HPUX, Linux), VMS, Windows NT/2000/XP/Vista

2.3.2 Realtime Interprocess Communication (RtIpC)

Local communication is performed using shared memory. This allows local data rates of up to 10,000,000 process messages per second.

In the network, up to 50,000 (100 Mb LAN) process data messages can be sent per second. A special feature is the support of redundant networks to grant uninterruptible operation.

Optimized multicast protocols allow data to be replicated to a large number of recipients.

2.3.3 Realtime Database (RtDb)

RtDb provides uncompromised real-time ability with over 10,000,000 single accesses per second.

The organisation of RtDb is similar to that of a relational database. The data is displayed in tables with rows and columns.

Each of the individual nodes in a distributed system can keep a local copy of the RtDb tables. In case of changes, these tables will automatically be replicated using multicast protocols.

The trigger mechanisms allow an application to define actions (installation of call-back functions) that will automatically be invoked when an application alters data in the RtDb.

Automatic synchronisation of the RtDb and a relational database (ORACLE or SQL Server) is supported, too.

An RtDb table can be distributed to several (redundant) servers whereby each server or each server pair is responsible for a part of the data (partitioning).

2.3.4 Realtime Redundancy Manager (RtRM)

RtRM controls the restart and – in the event of an error – also the computer fail-over in case of redundant systems.

The continuous monitoring of the vital software functions is very important in the case of redundant systems as this monitoring provides the criteria for a computer fail-over.

The individual components are signalling their errors to RtRM. Additionally, there is some kind of watchdog mechanism that RtRM uses to monitor all processes. Every process has to active this watchdog in a configurable cycle. If a process fails to trigger the watchdog, RtRM assumes that the process is defective. In this case, RtRM performs error handling according to the process specific configuration:

- Generating an error message (in any case)
- Restart the process (N times, up to a configurable restart limit)
- Stop RTAPLS if there is a standby system available and ready for failover
- Stop RTAPLS unconditionally
- Reboot the system

The computers of a RTAPLS system are monitoring each others using cyclic multicast messages. In case of failure of a redundant server, RtRM initiates a failover. The redundancy mechanism handles 2 or even more servers. All timeout values for error detection are configurable.

2.4 The Process Data Manager (RTAPLS - PDM)

PDM is the central component in RTAPLS. It processes all process data based on configuration data described below. The results of this processing are saved in the distributed RTAPLS data and object module.

Here a list of the most important configurable processing features:

- Scaling (linear, curve, free programmable)
- Supervision of 4 pairs of bounds and 2 gradients
- Configurable alarm processing with up to 8 priorities
  Global und local (per Workstation) alarm acknowledgment
- Automatic and manual suspension
- Value substitution
- User specific rights management
- Analog und binary filters
  - Smoothing
  - Jitter suppression
- Derived (computed) values
- Integration, minimum / maximum, average computation
- Integrated, free programmable mathematics package (RTAPLS - CALC)
- Tracking of field time and cache time
- Deadband feature
- Dual plant hierarchy (with automatic computation of status summary)
- Project specific status information
- Comprehensive status information per PV (a subset is OPC compatible)
- Command / Feedback supervision

PDM provides 2 types of interfaces:

- I/O Data Source interfaces for the transfer of process data to and from the acquisition system and PLCs. This interface has been optimised for large volumes of data throughput and supports two different priority levels. This interface is realised by means of configurable queues capable of handling temporary overload situations.

- Service interface for "service functions" and user interaction. The service functions can be called from the distributed RTAPLS object module. The interactions on the server interface are logged as internal system messages.

The output of PDM processing is:

- The entry of the data in the process image (in the distributed RTAPLS object model)
- Optionally (configuration-dependent):
  - Forwarding to a maximum of 4 configurable "channels". The first two channels are short-term historians for binary and analogue information. They also form the interface for the long-term historian and various project-specific "loggers". Each of these channels can be read by any required number of processes.
  - Issueing of output PVs.
  - Triggering of computation of the derived values

In typical RTAPLS configurations, the maximum continuous rate of PV changes is about 40.000 changes per second.

2.5 The Formula Interpreter (RTAPLS - CALC)

RTAPLS – CALC supports programmable computations. Essentially, it is used to generate derived values, but it can also be used for a number of other operations.

RTAPLS – CALC offers a range of standard functions that can easily be extended using project-specific functions.

The programming language used by RTAPLS – CALC is called **RtPL** (Real Time Process Language). In syntactical terms, this is a subset of BASIC. RtPL adds special constructs used for spontaneous and cyclical processing. For sophisticated applications RtPL supports object-oriented features like classes and inheritance.

RtPL interprets a binary intermediate code and reaches a speed of approx. 600,000 operations per second (using a 3 GHz Pentium).

The following data sources (variables) can be used by RTAPLS-CALC:

- Process variables together with their status information and other statistical and dynamic attributes.
- Other RTAPLS objects
- Any required RtDb data records and RTA objects
- RtOOS – objects
- System objects (time,...)
- Global and local variables

For every computation, RTAPLS PV-status information is also combined in additional to value operation.

2.6 The Historian (RTAPLS - HIST)

RTAPLS - HIST is used to save process data with a high time resolution over a longer period of time, i.e. several years.

Binary and text data is stored with a 100 nanosecond, analog values with 1 millisecond resolution. Bound violation alarms of analog values are stored similar to the binary record format including the analog value.

Furthermore, each analogue PV may be individually configured for data compression according to the “dead-band” or “swinging-door” algorithm. The goal of these algorithms is to eliminate storage of data values that “differ hardly from their predecessor” or that “can be predicted from the PV’s value trend”.

Independent of any compression algorithm, a further optimization stores analogue value records in an incremental fashion. The “deltas” of value, PV state or field time stamp are usually (much) smaller than the full values.

The required disk storage is arranged as daily, weekly or monthly partition files. These partitions can be individually exported and imported to / from any backup media.

By specifying the PV name and a point in time (or time interval), Historian client application programs may access archived data. These data are either “raw values” (just as written to the Historian), or “calculated accumulation values” (mean value, minimum, maximum, standard deviation,... per time interval according to OPC-HDA standard).

Optionally, RTAPLS - HIST may be configured to calculate and store pre-accumulated data (for all or selected analogue PVs). Thereby, access to frequently used accumulation results is significantly improved (e.g. one full year of hourly average values may be retrieved in a few seconds).

As PV configuration data can change over time, RTAPLS - HIST saves all modifications to these configuration data. When PV values are accessed in RTAPLS - HIST, the fitting configuration record is delivered and/or considered, too.

RTAPLS - HIST has been optimised with regard to the efficient writing of process data with ascending time stamp. However, data can be inserted, updated or even deleted at a later point in time. In typical configurations, the continuous load of 50,000 analogue value changes per second is supported. In specialized configurations (storing raw integer values, fixed intervals), a multiple of this load can be handled.

RTAPLS – HIST redundancy architecture grants uninterrupted operation even in case of failovers.

2.7 The Information and Report System (RTAPLS-REPORT)

The RTAPLS objects (stations, PVs, user groups,...) and the objects of the RTAPLS Historian form an object hierarchy. All information that can be provided by RTAPLS has a fixed position in this hierarchy.

2.7.1 OLE DB Provider (Microsoft Platform)

The RTAPLS OLE DB Provider provides a database view of these objects. This view contains a table of binary PVs, analogue PVs, stations etc. The individual objects (PVs) form the records in this table, and the attributes of the objects form the columns in this table. The RTAPLS OLE DB Provider implements a subset of SQL e.g. to access this data using SQL SELECT instructions.

All standard – tools that can be used in connection with OLE DB providers will be able to access all RTAPLS objects and attributes.

RTAPLS is often employed to create reports with Crystal Reports (http://www.crystaldecisions.com) or Microsoft Excel.

RTAPLS can also be used as a web server for intranet applications and ad hoc queries.

Exporting data from the RTAPLS OLE DB Provider to a Microsoft Access or SQL Server database is also very easy.

2.7.2 XML Provider (Linux Platform)

2.8 The Visualization (RTAPLS – VIEW)

RTAPLS – VIEW is the process visualization of RTAPLS used for comfortable creation of graphical user interfaces for process control systems.

RTAPLS –VIEW consists of:

- **Dynamic plant images** based on extensible symbol libraries, showing the status of the plant in a schematic display.
- **Basic static HTML content** (Documentation, Help function)
- **List displays**, for example, to display alarms that are requesting attention.
- **Dialogues** employed for user interaction (e.g. define set point).

RTAPLS - VIEW makes use of the vector graphics standard SVG. One key advantage of SVG is the availability of a lot of editors and converter tools to reuse existing images. SVG is platform independent and available on Microsoft as well as on Linux platforms.

Additionally, RTAPLS – VIEW integrates a web browser to display HTML content.

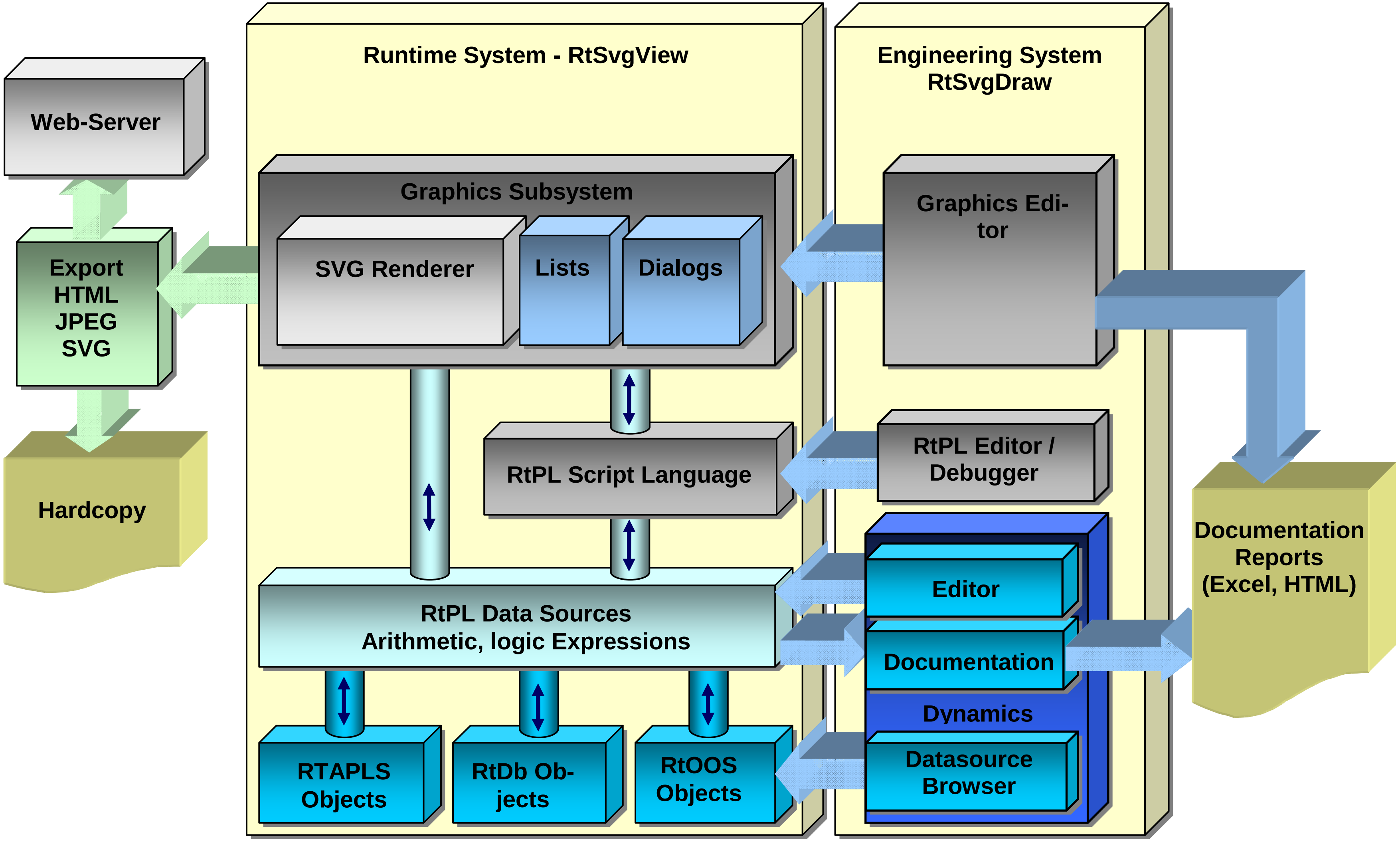
A Visualization system also contains “sequences of logic”, e.g. to control navigation through the images. Implementation of project specific, complex logic requires a programming language. For such purposes, RtPL, the language of RTAPLS – CALC, is integrated in RTAPLS – VIEW. Additionally to the RTA and RTAPLS objects (PVs, RtDb tables, …), all graphic objects and their attributes are accessible inside RtPL programs. RtPL is Visual Basic compatible and therefore easy to learn.

The attributes of the graphical objects of a SVG image must be connected dynamically to the real time objects of RTAPLS (PVs, RtDb content, …). Typically, such connections also need some kind of transformation (for instance value to colour). RtPL arithmetic and logic expressions as well as transformation tables can be used to configure these transformations.

RTAPLS – VIEW can be integrated in web servers to make plant images available in the office environment using standard web browsers.

An IEC-1772 conformant symbol library is part of RTAPLS - VIEW.

2.8.1 RTAPLS VIEW – Architectur



2.9 SOA / Web Server (RTAPLS – SOA)

RTAPLS - SOA / Web Server provides integration of RTAPLS into the world of enterprise and office software applications. W3C defined Web Services and Software Oriented Architecture (SOA) are the future-proof technologies to implement RTAPLS services completely platform independent using XML and firewall friendly HTTP.

RTAPLS - SOA / Web Server integrates several functions:

- RTAPLS Web Server
    - HTML Content
        - Help system
        - Online RTA / RTAPLS status monitoring
    - XML Content
        - RTA / RTAPLS Objects (delivered in XML format and rendered using XSLT style sheets).
        - RTAPLS configuration data
        - RTA state overview

- RTAPLS - SOA Server (HTTP based)
    - RTA and RTAPLS specific, W3C conformant Web Services
    - OPC Unified Architecture Data Access 1.0

- RTAPLS RPC Remote Server (TCP/IP)
    - Optimized, binary protocol to all RTA and RTAPLS Objects

2.10 Object Oriented Engineering (RtOOS)

Object-orientation and class-formation are the tools required to reduce the engineering workload in extensive projects and to model the projects. Viewing a process or a plant as a homogeneous, unstructured quantity comprising many thousands data points has proved to be too confusing and time consuming. The human mind is accustomed to - and quite successful in - thinking in hierarchies.

The essential feature of RtOOS is its capability of breaking down a plant into hierarchical classes. A "class" is an abstract, generic term for objects that are, to a large extent, very similar. The behaviour of these objects is described within the class (= ***methods***). An object has static and dynamic ***attributes*** to describe its condition. ***Relations*** exist between the objects. Some of the ***relations*** describe the object hierarchy (an object is *a part of another* object. In other words, an object *contains* sub objects). PVs, which are also classified as being objects, are typically found on the lowest level of the hierarchy.

Realisation of classes and objects:
- The attributes of the objects within a given class and the relations are realized using RtDb tables.
- The methods of the classes are programmed with ***RtPL***.
- Engineering tools are available for construction and test.

2.11 System Administration (RTAPLS – CONSOLE)

RTAPLS offers a comprehensive Diagnostics System with the following functions:

- Error Logging
  All processes produce entries in a central Log file. A new log file is created on a per hour basis. Under Windows, entries are also written to the Windows Event Log. A distinction is made between three different log entries:
  - INF Information entries in the case of important status changes
  - ERR In the case of non-fatal errors
  - FAT In the case of fatal errors (these terminate the program and result into a program restart or computer fail-over)

- Trace System
  The Trace System controls the output of trace information in various processes. Tracing can, for example, be activated in the drivers and in RTAPLS PDM up to PV level. This facilitates the tracing of individual PVs.

- Task and Network Monitoring
  RTA administers information relating to the program (errors, time of the last and first error occurrence,...). Important resources such as queues and the Real Time Database are also monitored permanently.

- Statistics
  Programs such as PDM and HIST provide statistical information relating to data throughput and modification frequency - in some cases even up to individual PV level. This data can, for example, be used to optimise deadband configuration.

- Process data explorer
  A tree view is used to access all process variables, inspect their attributes and manipulate them (depending on the user security system)

- User and rights management

- Online signal views
  Static and animated signal lists depending on selection criteria:
  - Static: PVid, Signal text, description, plant segment, PLC address, priority
  - Dynamic: Value, signal quality, alarm state, suspension

  Sorted by: PVid, field time, text, signal number, priority
  Displays controlled by PV groups:
  - Analog value - bars
  - Analog value - charts
  - Mosaic view (compact view of binary PVs)

- PV group manipulation