# Use of RTX and dPCs in Mission Critical and highly demanding industrial applications

*This article describes a particular process control application where RTX is used as the Reliable Hard-Real-Time Operating System under the dPCs **D**istributed **P**rocess **C**ontrol **S**ystem middleware concept.*
*The application is: Complete Precision Control of KHS Pasteurizer tunnels. Tunnel Pasteurizers are used to pasteurize canned or bottled beverage and food under strict PU (Pasteur Unit)-pickup control conditions:*

- *Too little PU-pickup affects the long-term durability of the product.*
- *Too much PU-pickup affects the taste and flavor of the product.*
- *Quality Control: The PU-Pickup in each product must lie within a defined range.*
- *PU-Control aim: The PU-Pickup in each product is minimal above the lower range bound under all production conditions (Best Quality).*

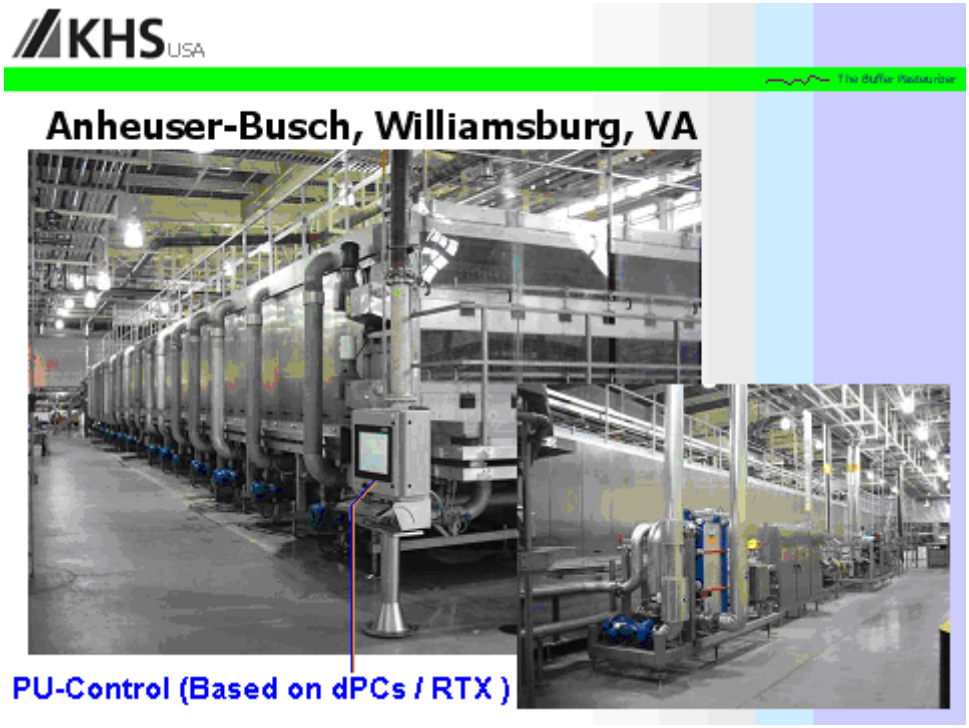



Photo from an Anheuser-Busch Budweiser production line.

## Background: Tunnel Pasteurizers

The KHS Tunnel Pasteurizers are multi Million US$ process machinery that pasteurizes up to over 100,000 products / hour. If a machine unexpectedly stops, this can cause a production loss of up to 30,000 pasteurized products, as each machine re-start may take up to 20 minutes.
The dPCs Concept is also found in Sander-Hansen and KRONES Tunnel Pasteurizers with approximately 1000 installations worldwide.
The dPCs Concept is also found in EMRI Ship Maneuver-Coordinators and Auto-Pilot systems installed on Cruise Ships and Ferries with the requirement for Precision Position Control (Translational and Rotational). This control concept is installed on several hundred ships worldwide e.g. Disney Wonder and its 6 sister ships, all Carnival Cruise ships, as well as Royal Caribbean Cruise ships. Nobody likes its cruise ship to bump into the pier at the port while drinking a cup of coffee...

### Common to all control applications

Process control applications are general mission critical as no manufacturer accepts any unpredictable behavior of their process control system. The process control system must be reliable in order to assure:

- Constant and highest possible product quality 24 hours per day.
- Constant and steady production rate 24 hours per day.

**Some facts about RTX and dPCs**

- The Ardence RTX Windows XP/XPE Hard-Real-Time Operating System Extension has proven over the years to be a reliable platform for mission critical Process Control Systems.
- The ER-Soft, S.A. dPCs Concept has in the same way over years proven to be a reliable Process Control System Framework for mission critical Process Control.

**A brief description of the internals of the RTX / dPCs Control System concept**

There are many differently behaving links in the long chain that spawns a Complete Production System. If the Complete Production System has to be Mission Critical, all the links must be at least as good as the entire system. Special vulnerable and critical subsystems may be designed redundant in order to achieve the reliability level of the total system.

In this article we will only discuss those elements (links) that are related to the Process Control System, as the Production Process related matters are normally not part of the Control System Designer and System Integrator's job. Production Process related matters are assumed to be resolved by the Process Engineers.

**Elements considered in this context:**

- *The applied Process Control System Software and Hardware components. How to develop a reliable Precision Control System using:*
  - *Industrial PC Hardware,*
  - *Windows XP,*
  - *Ardence RTX,*
  - *dPCs.*

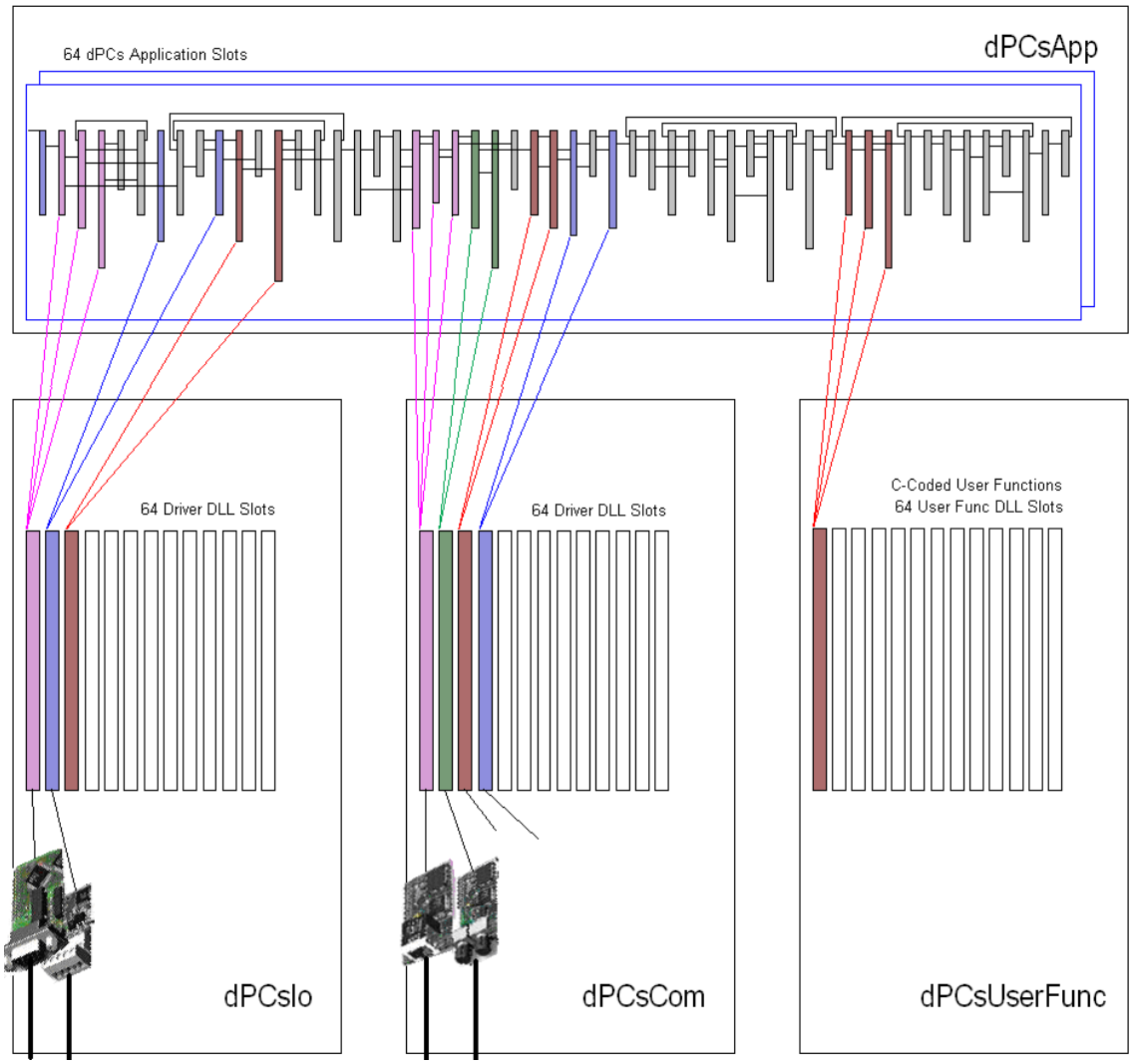
  Application of the components:
  - The first two components in the following are expected to be known. It is expected that the reader understands how to setup a reliable Windows XP computer.
  - The Ardence RTX is in relation with dPCs or other C-coded projects just another Operating System Platform similar to Windows, but offering Hard-Real-Time performance. The use of RTX increases the software complexity as the computer now changes to a Dual-Core Operating System (Win32 and RTX). RTX is rather a Hypervisor, - an Operating System (OS) under Win32 and thus has full control over Windows. RTX offers an OS platform similar to Win32, but offering Hard-Real-Time performance.
  - The dPCs Platform can be installed on both the Win32 and the RTX platform, if available. dPCs Data Communication Drivers and Applications can either be installed on the Win32 or RTX platform (if available) as the user desires.

# The Architecture of the dPCs Control System concept

The figure on the next page illustrates the simple dPCs structure.
The dPCs Control System concept is simple and consists of only 4 basis subsystems:

- The topmost of the of the 4 basis subsystems is **the Horizontal dPCs Application Container subsystem** (**dPCsApp** where one or several dPCs Applications can be loaded and execute), this will be described later.
- The other 3 of the 4 basis subsystems are **the 3 Vertical Container subsystems** that each can hold a number of:
  - **dPCsIo:** Fieldbus I/O Data Communication Drivers,
  - **dPCsCom:** Process Data Communication Drivers, and
  - **dPCsUserFunc:** Dll Modules with an unlimited number of different User Functions implemented in C-code.
  - *Note that all Drivers exist in both:*

- o *Win32* ***Dll*** *versions, and*
- o *RTX RTSS* ***RtDll*** *versions.*




The figure illustrates the simple dPCs structure that consists of:

- The 3 parallel **Vertical dPCs Driver subsystem** containers,
- The top level **Horizontal dPCs Application subsystem** container layer.

Detailed description: See “The Architecture of the dPCs Control System concept” on the previous page.

**dPCsApp**

Each dPCs Application consists of an arbitrary number of dPCs Objects:

- One **Application Program Execution Control and Container object** that makes-up the dPCs application unit. On install and start, it is loaded into the dPCsApp container and it is linked to the Operating System for cyclic execution each scan-time (as a new created OS worker-thread and creation of a time reliable OS Clock resource). The dPCs Program object specifies the dPCs application program scan-time and priority. On each program scan it checks its own scan-time and priority, and if changed it adjusts the parameters in the OS Clock and/or worker-thread before the application program execution scan. This feature means that dynamic scan-time and priority changes are supported in dPCs.
- **I/O Data Communication objects** that communicates I/O data through the different loaded vertical High-Speed dPCsIo communication drivers (loaded into the dPCsIo Container subsystem),
- **Data Communication objects** that communicates Process data with DCS and PLCs through the different loaded vertical Medium-Speed dPCsCom communication drivers (loaded into the dPCsCom Container subsystem),

- **User-Func objects** that call C-coded User Functions in one or more User Function Modules (loaded into the dPCsUserFunc Container subsystem), in order to efficiently solve complex process control.
- It is important to understand that the dPCs Application layer offer groups of objects that are developed to be used together with specific Drivers. As an example we look at the Modbus TCP Data Communication Protocol. All applied Modbus TCP Data Communication objects can only link to an installed Modbus TCP Data Communication Driver. Each object can communicate a number of simple data items from a data array using the Modbus TCP Data Communication Protocol.
  I/O Data in dPCsIo Drivers are all accessed in the same way (all drivers offering the same API functions). This means that dPCsIo objects are more versatile, so the same group of dPCsIo Object classes will work with all dPCsIo Drivers.
- *A very important feature **in the dPCs concept is that there are only two places where parameters are found and can be adjusted**:*
  - *Upon Load Install and Start of Drivers, these Parameters are Driver Global Parameters and specified as Driver Parameters.*
  - *All dPCs objects hold all required parameters (constant values) on its input terminals as well as the input values (dynamic process values). This is an important object feature because all relevant values are at hand and visible to the user. No hidden parameter values.*

### The Architecture of the dPCs Control System concept in deeper details

Please dedicate time to read through the whole of the article and not just the final part, as we believe it will be enlightening:

#### dPCsIo

- The **dPCsIo** Container subsystem can hold one or more Fieldbus or Local I/O Data Communication Drivers. I/O Data Communication Drivers can be any Fieldbus I/O Driver, Master and/or Slave Driver such as:
  - **Profibus DP Scanner (Master),**
  - **Ethernet/IP I/O Scanner (Master),**
  - **Devicenet Scanner (Master),**
  - **AS-I Scanner (Master),**
  - **CANopen Scanner (Master),**
  - **Allen Bradley RIO Scanner (Master),**
  - **Modicon S908-S Scanner (Master),**
  - **Profibus DP Slave or Multi-Slave,**
  - **I/O and Bulk Data Gateway Driver,**
  - Etc ...
  - All Fieldbuses and Local I/O Systems are supported by the dPCsIo Driver concept.
  - dPCsIo Drivers are normally based on a Communications Co-Processor interface cards that can process a high number of Fieldbus I/O data frames per second. Fieldbus interface cards are supplied by manufacturers as: **HMS Anybus, Woodhead Software and Electronics, Hilscher, ER-Soft...**

  The above mentioned dPCsIo Driver Design supports the following features:
  - Standard Implicit Fieldbus I/O Data Messaging (The Basic Fieldbus feature),
  - Explicit Messaging, Fieldbus Parameter Data Communication (if defined for the fieldbus).
  - OPC-Server Access to I/O data.
  - Support for FDT / DTM (Field Device Tool) / (Device Type Management), if defined for the fieldbus or the interface card).

  Some other special and interesting features of dPCsIo Drivers:
  - dPCsIo Drivers are High-Speed drivers enabling the fastest possible access to Field I/O data for the dPCs and/or C-Coded User Applications.
  - dPCsIo Drivers all offer Dual-API, which means that they all can be accessed concurrently and reentrant from both Win32 and RTX-RTSS based

applications. The dPCs Dual-API is public and can be called from Visual Basic, VBA, Delphi, C-Coded Applications and any other application that can call external Dll functions. The C-Coded Applications can be designed so they can execute in both the Win32 and/or RTX-RTSS domain with absolutely no application change.

- The dPCsIoMonitor tool enables the user to monitor the functionality of each of the installed dPCsIo Drivers.

**dPCsCom**

- The **dPCsCom** Container subsystem can hold one or more Industrial Data Communication Drivers. The dPCs Data Communication Objects and Drivers communicate elaborated data sets at moderate speed between dPCs Applications and other Process Control Systems as DCS, PLCs and other Dedicated Process Controllers using Industrial Data Communication Protocols as:
  - **Modbus TCP (Ethernet),**
  - **Modbus RTU (Master RS-485 Serial),**
  - **Modbus RTU (Slave RS-485 Serial),**
  - **Allen Bradley DF1 (RS-232**
  - **Allen Bradley PCCC (Ethernet),**
  - **Rockwell Automation Ethernet/IP CIP (Ethernet),**
  - **Siemens Native S7 (Ethernet),**
  - **Siemens ISO on TCP for S5 (Ethernet),**
  - **Siemens 3964R (RS-232 Serial),**
  - **Omron – Fins (Sysmac C, CV and CS1)**
  - **Mitsubishi - Melsec (AnA, AnU, AnS, QnA, QnAS)**
  - **GE Fanuc - SRTP (GE90-30 & 90-70)**
  - **Schneider - Uni-TE (Premium, Micro)**
  - Etc...

  Each of the above mentioned dPCs Data Communication concepts support the following general features:
  - Can have one or more **SendData** Client Data Communication objects that write data to one or different data areas in one or several different remote data consumer devices or controllers.
  - Can have one or more **FetchData** Client Data Communication objects that read data from one or different data areas in one or several different remote data producer devices or controllers.
  - Can have one or more **ReceiveData** Data Server Communication objects that receive write data from one or several different remote data producer devices or controllers.
  - Can have one or more **ServeData** Data Server Communication objects that return read data to one or several different remote data consumer devices or controllers.
  - Offers access to the Data Communication Driver Statistic Data through the **Status** object.

  Note that:
  - The **SendData** client objects correspond to similar **ReceiveData** server objects or PLC database data arrays.
  - The **FetchData** client objects correspond to similar **ServeData** server objects or PLC database data arrays.
  - All dPCsCom objects contain a data array similar to a piece of PLC database data array.
  - All dPCsCom Client objects contain a Data-Flow Control Block similar to a PLC Message Control Block.
  - It is extremely easy to setup well-performing Process Data Communication in dPCs using the indicated simple to use object oriented dPCsCom concept.
  - All the dPCsCom objects offer an extended Data-Validation option.

**dPCsUserFunc**

- The **dPCsUserFunc** Container subsystem can hold one or more User Produced **Dll** or **RtDll Modules**. Each User Produced Dll or RtDll module can contain an endless number of User Functions. The User Function concept has a special and very efficient parameter passing structure that enables the User-Function objects to:
  - Dynamically link to a C-coded User-Function,
  - Efficiently Call the linked User-Function with up to several thousands of Input Parameters,
  - Execute the User-Function code and produce up to several thousands of result Parameters,
  - Return from the User-Function and efficiently return all the produced Output Parameters. (In fact, the produced Output Parameters can be written directly to their destination location while produced, thus saving execution time and temporary result variable space).

  The execution of the User-Function C-code is performed directly from the User-Function object. This quality enables the user to execute any complex calculations in the dPCs Application scan as if they were programmed and executed in native dPCs objects. C-coded User Functions allows for total programming flexibility in the problem solution.
  It is important to mention that the dPCs system is natively Crash-Proof !
  NB: The User-Function feature is intended for skilled programmers !

**dPCsApps** (continued)

- The **dPCsApp** Container subsystem can hold one or more dPCs Applications executing in parallel. dPCs Applications are basically a list of dPCs objects (object instances) linked in the sequence that they are listed in the dPCs application. Each dPCs Application is executed cyclically, which means that object instance's main method is executed for each object in the linked list of objects. All dPCs Objects have some Input Data and Output Data terminals. The output data terminals are located on the objects right side and are output variable allocations that offer access to the objects calculated data. All the objects in a dPCs Application offer public access to all its output terminals. Any input terminal on any object in the dPCs application can be connected to any output terminal in the dPCs Application or to a Constant value. The dPCs concept offers a generalized version of an Object Network. Compare it to the one found in Neural Networks. The Object Network concept is extremely powerful.

  The first object in an application must be the Program object. When a dPCs Application is installed and started, the following actions take place:
  - The dPCs Application is loaded into a memory block. A new Operation System worker-thread and a Clock-Tick object are created, and the dPCs Application is linked to them.
  - The Scan-Time specified on the Program object controls the Clock-Tick time by which the dPCs Application is cyclically executed. The dPCs Time variable has a granularity of 0.1 micro-second (hecto-nano-second) resolution.
  - The Priority specified on the Program object controls the priority of the worker-thread which controls the dPCs Application execution.

**Classes**

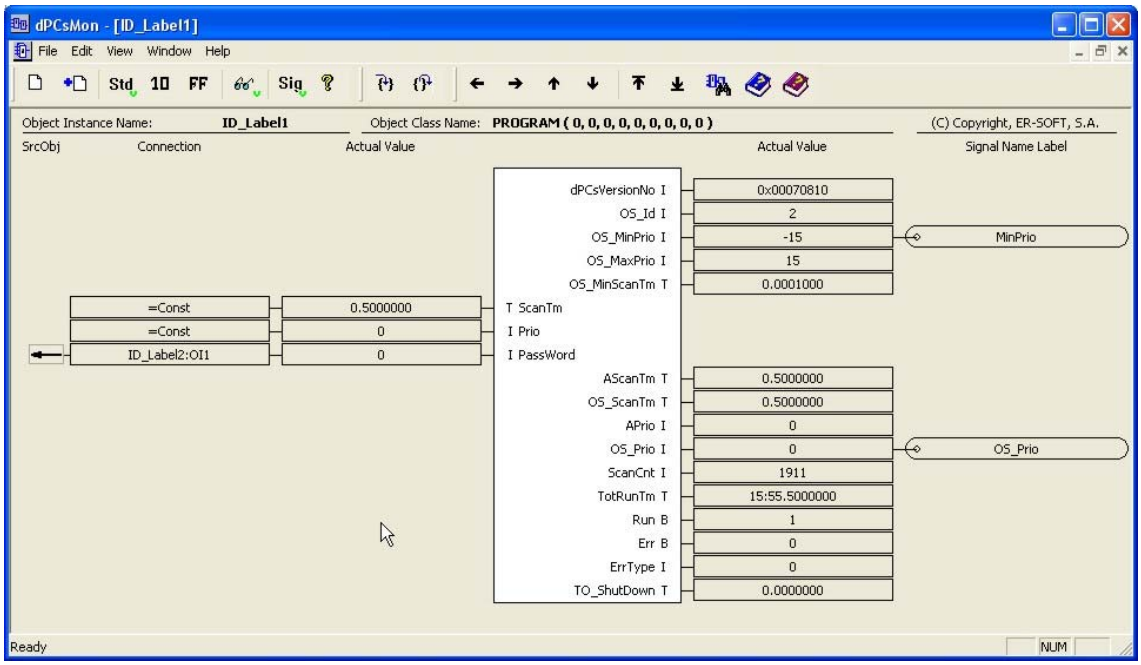
It has already been briefly described how dPCs Objects are linked together and executed cyclically in order to form a dPCs application. However, it has not yet been explained what dPCs object classes are available for dPCs Application construction. The available dPCs object classes can be roughly be classified into the following main classes:

- The **dPCsApp object class: Program** which defines a dPCs Application Program.

- The **dPCsIo Data Access object classes** that can communicate I/O Data with the installed dPCsIo Drivers.
- The **dPCsCom Data Communication object classes** that can communicate Data with other Process Controllers, DCS and/or PLCs through the installed dPCsCom Drivers.
- The **dPCs-User-Function object classes** that can call C-Coded User-Functions that efficiently solve special and complex problems.
- The **general purpose data processing object classes**, which counts more than 280 different object classes classified into 20 different groups. See the dPCs Documentation for more details.
- *It is important to understand that all **dPCs Object Classes are scalable in one or more dimensions**. This important feature makes the adaptation of an Object to a certain problem solution simple and exact. The dimensions are controlled by the **Object Construction Parameters**, and offers up to thousands of object variants of each dPCs Object Class.*

### Online Monitoring

The **dPCsMon**, the dPCs Online Object Monitor utility enables the user to monitor the functionality of each of the object in any installed and executing dPCs Application.




The above screen dump shows the dPCsMon utility. The dPCsMon tool shows a dPCs Application Program Object Instance in each open Window. Each window shows online the values of the object terminals, and it shows the connection of the input terminals. Each Window can freely browse all the Objects in the entire dPCs Application following either the Object Execution sequence (Up or Down) or any of the Input Terminal connections. The dPCsMon tool quickly enables the dPCs user to find any problem in the process or the application program.

## dPCs Conceptual Features

dPCs offers some outstanding conceptual features:

- The **dPCs Object Oriented Application Programming (using the dPCs Foundation Classes)** or rather Configuration Language is of very high level. Virtually no programming skills are required. Skills in modeling of Industrial Control and Data Communication System solutions is of much higher importance.
- The **dPCs Compiler produces highly optimized runtime code that is natively crash-proof** and furthermore, it's code **generally performs better**

**than any C-coded implementation**. Thus the user does not pay any penalty in the form of a poorer performing total system. This means that the user only gains in sense of better structured and better performing total system.

- The dPCs coded application even **allows for online monitoring** (using the dPCsMon tool without affecting the system) **while the dPCs application is executing at full performance**.
- It is proven that **dPCs can solve most current Industrial Control and Data Communication** problems in an **efficient** and **economical** manner.
- As an example: A data communication Gateway between two or more different data communication networks and/or fieldbusses can be setup using as few as 3 to 5 dPCs objects. Only a few minutes programming / configuration work.
- The dPCs Concept **is 100% open** for integration with other PC software application programming systems such as: **Visual Basic, VBA, Delphi, Visual C++** etc...
- The dPCs Concept **is 100% open** for integration with other DCS and PLC systems as it communicates the DCS and PLC system Data Communication protocols.

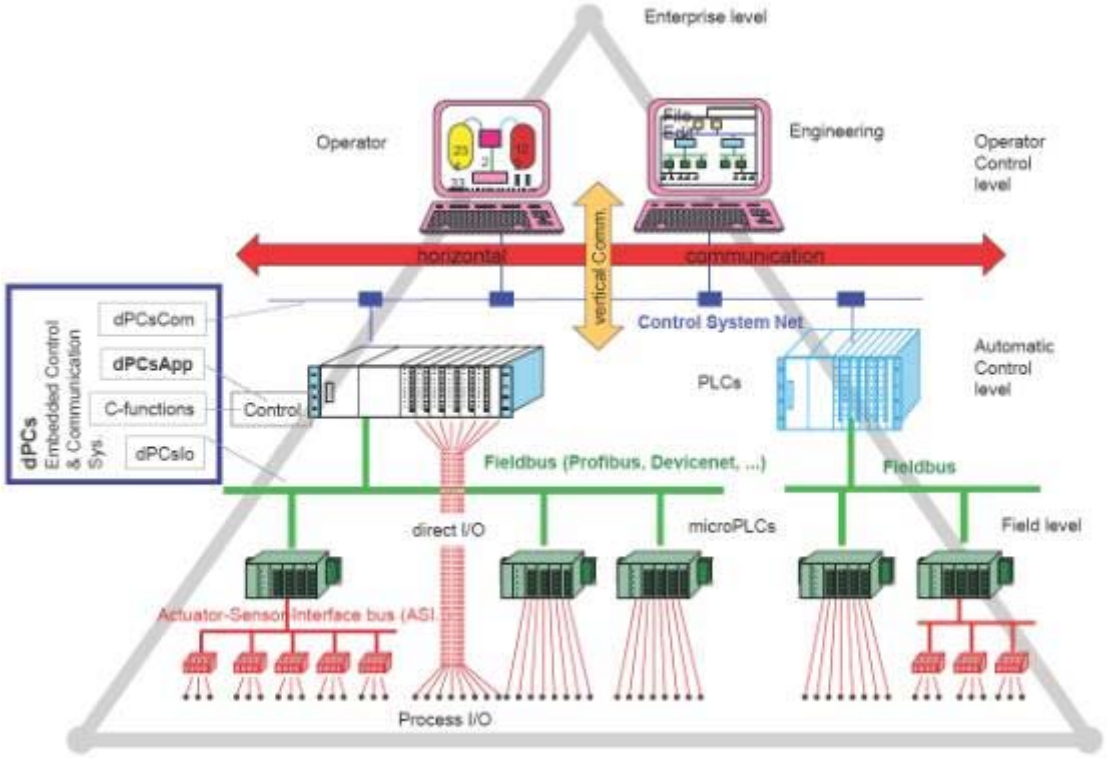



Illustration of dPCs in the Pyramid Hierarchical Control System Structure.

- The dPCs Concept **is 100% open** for being **Automatically Coded** from **CAE / CASE** tools. This is a feature that is often lacking in modern DCS and PLC systems. Many systems offer nice Windows based programming tools, but for manual use. dPCs offers for all its programming and configuration “Structured Text” files. This means that there is a well defined text file format that describes any dPCs configuration.

## Getting started with the dPCs Concept

**Required system components**

- **PC or Industrial PC** (Client supply or by ER-Soft),
- **Installed Win XP + SP2** ( Microsoft product, Client supply or by ER-Soft )
- **dPCs Runtime + Development systems** (ER-Soft)
- ***MS Visual Studio for C-Code Dll development*** *(Optional Item: Microsoft product, Client supply or by ER-Soft )*

- ***RTX Runtime for HARD-Real-Time Driver and Application Execution*** *(Optional Item: Ardence product, Client supply or by ER-Soft )*
- ***RTX SDK + Runtime (Requires MS Visual Studio) for HARD-Real-Time C-Code Dll development*** *(Optional Item: Ardence product, Client supply or by ER-Soft)*

**Possible Hardware Platforms**


Fan-less Industrial PC Hardware Solution

## Conclusion on the dPCs Concept

The described concept is general, and at the same time powerful and can basically solve any process control application if the computer hardware can cope with it in sense of hardware performance.

- dPCs Applications generally enforce cleaner application structure as the dPCs objects keep functionality and data together. At the same time dPCs Applications nearly always execute faster than the same Application implemented in a low-level and execution efficient C-code. The exception to this is the high-level programming language systems, that offer high application program productivity, which in general perform poorer than low-level programmed applications coded in C or Assembler.
  This is not the case with dPCs Applications, because dPCs objects are small visible “top of the iceberg”, while a lot more may exist directly under the visible part. This fact is especially true for complex dPCs objects.
  In other words: If there exist a C-coded solution for the control problem, there most probably exist a much more structured dPCs solution for the same problem (on the same PC hardware platform) and presumably at a cost less than 5% of the cost of the pure C-coded solution.
- dPCs Object Methods are implemented in Speed Optimized C-Code.
- dPCs Applications are compiled down to highly efficient binary code that generally execute in the ultra fast computer cache memory.
- dPCs is easy to work with for people using systematic working methodology. Sporadic Ad Hoc programmers may find dPCs a bit heavy to work with, as the “the Quick and Dirty” option does not exist. The “Good and Clean” solution may take only 5% longer time (in application structural work and documentation).
- The dPCs concept performs well on all industrial control and communication projects, and even better while projects are growing, as dPCs is fully scalable.

- A simple dPCs application of only 2 to 10 objects is able to emulate a PLC in sense of data communication:
  - Siemens Simatic S5 and S7 with Ethernet TCP, or Serial 3964R / RK512
  - Modicon PLC with Modbus TCP or Modbus RTU, or Modbus PLUS,
  - Allen Bradley Control-Logix with Ethernet/IP CIP, or Serial DF1
  - PLC-5 or SLC Ethernet PCCC, or Serial DF1
  - Any other PLC or DCS industrial data communication protocol …
  - This way a dPCs station can work as a "Complex Problem Solution Provider" station in industrial networks (eventually programmed in a combination of dPCs and C-coded User Functions). The initial dPCs control and communication system framework can be setup in a couple of minutes … If C-coded User Functions are used, the setup may take a couple of hours …

  Please note that all the protocols can execute in parallel, so dPCs offers the platform system for eminent industrial data communication gateways. dPCs Concept, due to its flexibility, efficiently can handle any industrial data communication protocol.
  NOTE: Request for demonstration of any of the above solutions if you have doubts about how it works!
- ER-Soft offers software design and implementation engineering services in order to make dPCs able to easily handle special functionality, as for example industrial data communication protocol implementation into the dPCs Concept.
- The dPCs Concept is built around the dPCs Foundation Classes (the complete set of dPCs Object Classes). It is fundamental that the dPCs Foundation Classes are Normalized and well structured in their design, in order to have their design survive over time. This fact makes the dPCs engineering process costly, but the users benefit greatly from this extra design effort.
- An important dPCs design feature to mention is that it is designed in such a way that it's framework constantly can be expanded without affecting the already implemented parts. We realize that "Rome was not built in a day", and the same happens to dPCs. There are still a lot of functionalities that easily can be handled by dPCs, but this is only a matter of available engineering resources.
- ER-Soft also sells parts of or the entire dPCs concept for integration in or implementation on special hardware platforms and Operating Systems. This service may offer a head start for some control and communication system providers that need a proven flexible framework design to start build on.
  NOTE: Request for quotation on any of our services!



## Testimonials about the dPCs Concept

- **KHS, AG, Germany, KHS Inc, USA, - Karsten Frydkjaer:**
  KHS uses all the features offered in the dPCs concept.
  dPCsIo Drivers for different Fieldbus I/O networks,
  dPCsCom Drivers for different Industrial PLC Data Communication networks,
  dPCsUserFunc where Finite-Element based solution algorithms are implemented in C-Code (in RTX RtDll library modules) and called from the User-Function dPCs Objects.
  The KHS projects are described in Excel sheets and executing a script generates all the dPCs program and configuration files. Bug-free ready to compile and use dPCs applications are produced, with huge long-term savings on engineering.
  Interface system is implemented in MS Access and interfaces to dPCs using VBA scripts.

- **Sander-Hansen, Krones Group: - Dr. Falko J. Wagner**
  Sander-Hansen is using some of the features offered in the dPCs concept (both

dPCs16 and dPCs32).
dPCsIo Drivers for different Fieldbus I/O networks,
dPCsCom Drivers for different Industrial PLC Data Communication networks.
The dPCs Components are used by their public API, and called from either C-or Pascal coded applications. The dPCs data components are used in order to integrate the embedded control system into their clients plant data communication systems (Siemens, Allen Bradley, Modicon, and other PLC systems).

- **EMRI, Denmark, - J. C. Nortoft Thomsen:**
  1989 The company EMRI formed part of an investigation committee.
  A Modular Analog Computer was required in order to perform the simulation of a ship movement over a certain time period. The ship data and public known weather data was fed into the model in order to calculate the ship's route.
  The ER-Soft 16-bit dPCs concept was chosen as a very cost-effective alternative to other computer systems.
  The result turned out so well that dPCs afterwards became the core if the advanced EMRI Ship control system for Auto-Pilot and Maneuver-Coordination.

- **Loten Molle, Norway, - Bjorn David Bratlie:**
  I have to make a decision soon about our new control system.
  We have used the ER-Soft 16-bit dPCs concept since 1991 and I want to say:
  "***It is the best system I know***"
  Note: The decision to use dPCs on RTX was taken, Jan 2008.

## About this Article

This article has been written on demand by MD Fabrice Boisset, Ardence Europe. The author of the article is MScEE Ronn Andreasen, Chief Designer of the dPCs Concept. The article contains information supplied and with permission by Lead Engineer MScME Karsten Frydkjaer at the Pasteurizer Competence Center at KHS AG, Dortmund, and KHS Inc, Waukesha, IL, USA.
Ronn Andreasen is the creator of Object Oriented dPCs concept since 1987 and creator and CEO of the ER-Soft, S.A. company.
The core of the new 32-bit dPCs Concept has the same basic functionality as the former 16-bit dPCs concept developed in 1987 to 1997. The porting of the dPCs to 32-bit dPCs was started in 2001 and has been both an interesting and positive experience. The 32-bit dPCs has been substantially extended in sense of flexibility, functionality and performance. It has proven to be reliable and perform to a very high level. The actual platform offers solutions for many common Control and Data Communication problems. There are still many new classes of Control and Data Communication problems that can be well resolved by the dPCs Object Oriented Concept, so dPCs keeps developing. The dPCs Object Oriented Concept enables System Integrators another level of productivity than known so far...

Madrid, April 2008

**ER-Soft, S.A.**
Av. Constitucion, 4
E-28230 Las Rozas
Madrid Spain

Tel: +34 916-408-408
Fax: +34 916-408-409

web site: www.er-soft.com
e-mail: info@er-soft.com