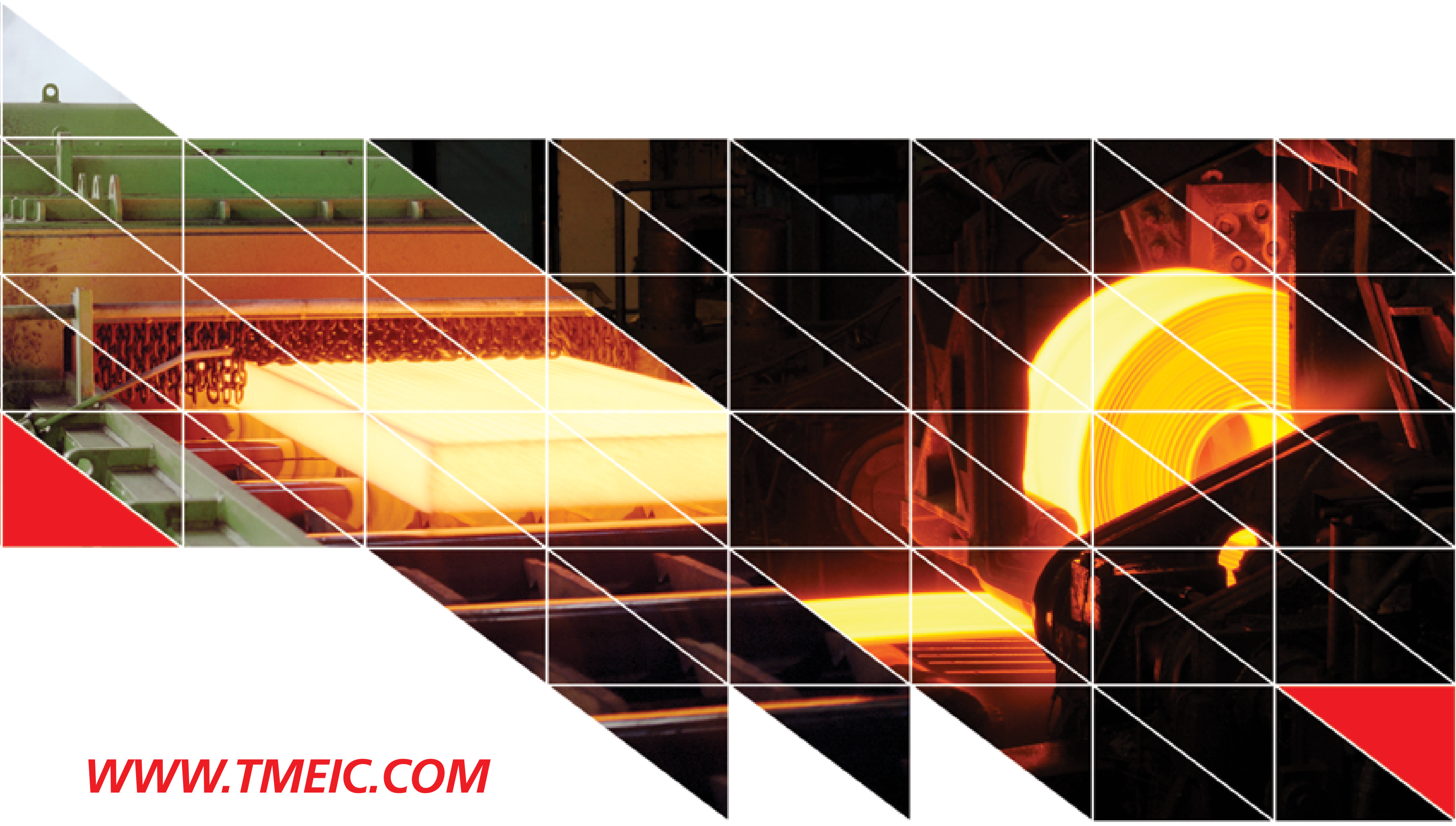
TMEIC
We drive industry

PASolutions
Complete Electrical Systems and Process Automation Solutions with TMACS®



WWW.TMEIC.COM

JAPAN | NORTH AMERICA | SOUTH AMERICA | EUROPE | SOUTHEAST ASIA | INDIA | CHINA | MIDDLE EAST | AUSTRALIA

Process Automation Solutions (PA Solutions)

Fully Configurable Level 2 Infrastructure for Metal Rolling and Processing Control Systems

Wlodzimierz Filipczyk
Manager, Performance Analysis and Development – TMEIC Corporation, Roanoke, VA
1325 Electric Road
Roanoke, Va. 24018 - USA
Tel. 540-283-2106, Fax. 540-283-2395
e-mail: Wlodzimierz.Filipczyk@tmeic.com
John McMillen
Senior Automation Engineer – TMEIC Corporation, Roanoke, VA
1325 Electric Road
Roanoke, Va. 24018 - USA
Tel. 540-283-2174, Fax. 540-283-2395
e-mail: John.McMillen@tmeic.com

KEY WORDS:
Computer Control Systems, Level 2 Automation, Configurable Software, Structured Software Products

Introduction

The traditional hierarchy levels of the industrial control system span from Level 0 (Drives and Actuators) to Level 3 (Production Planning). The traditionally named "Level 2" coordinates the control references and product data distribution to other parts of the system such as Level 1 and HMI. The references can be generated by process models or can come from preset data a.k.a. "stored schedules". Level 2 also includes the gathering and storage of process data as well as the generation of production reports. The set of functions required to support such activities forms Supervisory Process Automation software infrastructure (a.k.a. "Level 2"). Today, since "Level 2" systems frequently expand beyond the traditional boundaries into lower and higher levels of the control system hierarchy, a more appropriate term would be "Supervisory Process Automation". The paper describes a structured approach to implementation of such systems that takes advantage of modern software design technologies while providing multiple benefits to the automation supplier and End Users alike.

Traditional Approach

In the traditional approach to engineering a new supervisory system, after the functional scope and requirements are defined, software from a previous similar application (hot strip mill, reversing plate mill, coating line etc.) is identified to serve as a starting point. A team of several programmers modifies the old programs or writes new customized source code to satisfy the particular requirements of the new project. Much time is spent compiling, debugging and unit testing individual software components. Then even more time is spent integrating the software components, verifying all inter-process communication. An additional level of difficulty is introduced by a multi-piece process environment where coordination of tracking and data distribution is a complex task. Corrections necessitate modification of source code, linking and compiling. Quite often due to the program dependencies, a complete "system rebuild" is required. This causes disturbances to the system test with significant loss of time for the whole software team.

New Approach

During late 80's and early 90's, efforts were begun to address the growing number and variety of automation projects by modernizing the approach to Supervisory Process Automation software. Functionality common to most automation systems was to be "packaged" in re-usable configurable modules with a granularity sufficient to meet scale and variety requirements. At that time, Object Oriented Design (OOD) and Object Oriented Programming (OOP) became attractive concepts for the process control software. As implementations evolved to take better advantage of these concepts, the primary programming languages transitioned from "FORTRAN", to "ADA" and finally to "C" and "C++", following much of the software community at large.

Initial efforts were dedicated to defining application objects and data structures which support the required functionality and create the most universal and flexible software "building blocks". The best candidates for the first structured and configurable applications (products) were "well defined" functions such as material tracking and process data gathering.

Over a period of several years, more and more structured products were developed so that all requirements for Supervisory Process Automation would be covered by the full complement of structured products now called "Process Automation Solutions" or **PASolutions**. As new programming techniques and on-line accessible data structures (Relational Databases) were introduced, the structured products were updated and their functionality augmented to take advantage of the latest software technologies. The ongoing development addressed concerns for increased platform flexibility and scale-ability as well as functionality.

The **PASolutions** approach to satisfying functional requirements of Supervisory Process Automation offers reliability and agility in a fully Event-Driven, fully Configurable suite of software products. The software products are routinely delivered in the familiar form of installation kits.

The same structured software products (executables) are applied to a wide range of applications: Single Stand Cold Mill, Galvanizing Line or Hot Strip Mill, for example. The customization necessary to meet functional requirements of a particular application is achieved by editing configuration files, not by editing source code and recompiling.

For well over a decade, project teams using **PASolutions** have benefited by a significant reduction in the pitfalls of software development and increased focus on application functionality.

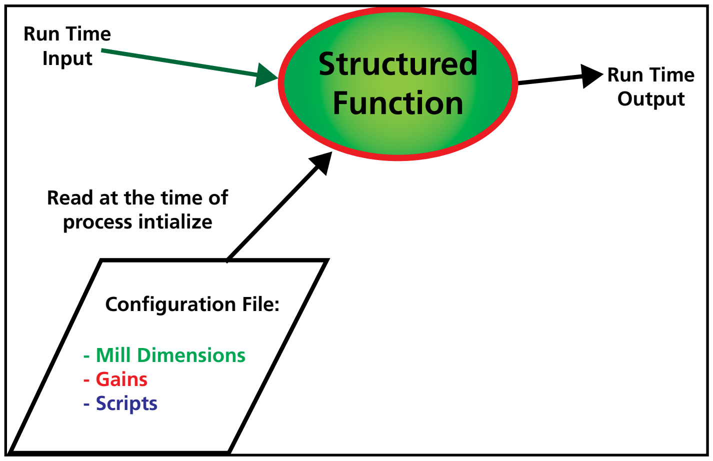


Fig. 1 – Configurable Function

"PASOLUTIONS" Approach to Supervisory Process Automation

Overview
Supervisory Process Automation systems are considered to consist of two primary functions: Process Algorithms Execution and Process Data Storage. The Process Algorithms Execution includes Process Models and Support Functions (Models Infrastructure) software. Process Data Storage includes Process Data Bases and necessary communication interface software. Typical Metal Rolling applications include databases for : PDI (Primary Data Input), Process Models Parameters, Roll Data, and Rolling History.

For smaller applications such as process lines, single stand rolling mills etc. a single computer is used to host both parts. For larger applications such as multi-stand rolling mills, separate computers are often used for better system performance.

PASolutions includes the following design features and characteristics which are explained below in more detail.

- "Packaged" Software Products – Reusable Images as Building Blocks
- Configurable and Maintainable
- Scaleable
- Platform flexibility through Distributed Processing on Multiple Operating Systems
- Open Architecture
- Agility

Software Products – Reusable Images as Building Blocks

The **PASolutions** product suite offers 'building blocks' encapsulating generic functions common to supervisory systems for ALL types of Rolling Mills and ALL types of Process Lines. These separately installable products (almost 30) are 'right-sized' with respect to functional scope and configurability, allowing each project to meet scale and customization requirements without modification of product source code. If there is insufficient scope encapsulated in each product, more components are required to satisfy an application. Thus increasing system complexity and decreasing the advantage of this approach compared to engineering, testing and maintaining source code. On the other hand, a monolithic product attempting to cover too much territory is likely to require modification to satisfy project requirements and may be difficult to scale-down for small applications. PASolutions designers reflect on decades of experience while managing this balance.

'Right-sizing' the products also gives the project engineer opportunities to implement loosely-coupled processes within the supervisory system. In this manner, one may modify and restart a process without unnecessarily impacting other processes.

Each PASolutions product falls into one of the primary functional categories as shown in the figure.
PASolutions software is released periodically for use by project teams. Installation kits are provided for convenient installation on End User computers. The installation kits are managed under version control and support future products upgrades.

A PASolutions Software Development Kit (SDK) is also available so that End User's written applications may be easily integrated to the control system

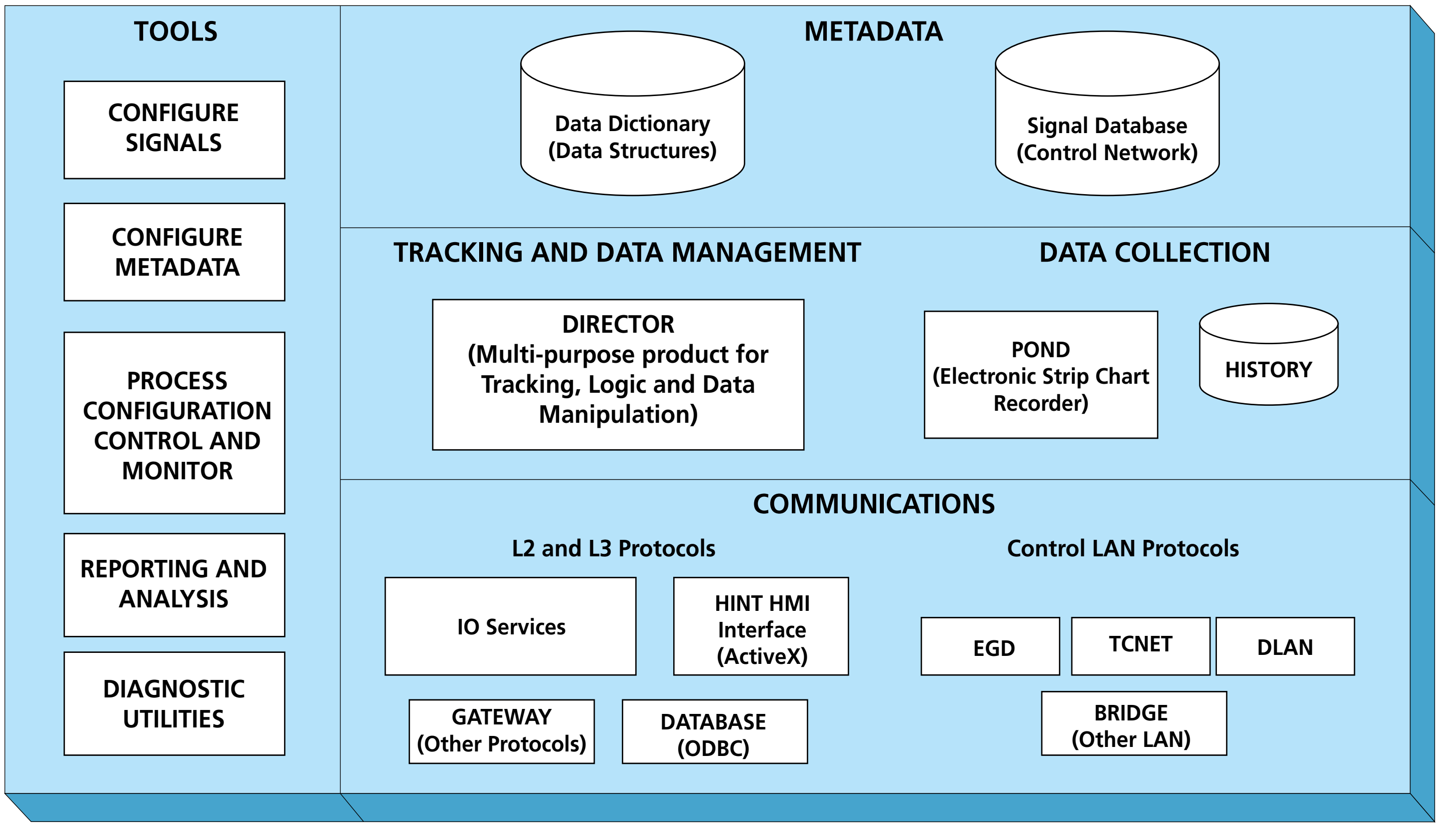

TOOLS
CONFIGURE SIGNALS
CONFIGURE METADATA
PROCESS CONFIGURATION CONTROL AND MONITOR
REPORTING AND ANALYSIS
DIAGNOSTIC UTILITIES
METADATA
Data Dictionary (Data Structures)
Signal Database (Control Network)
TRACKING AND DATA MANAGEMENT
DIRECTOR (Multi-purpose product for Tracking, Logic and Data Manipulation)
DATA COLLECTION
POND (Electronic Strip Chart Recorder)
HISTORY
COMMUNICATIONS
L2 and L3 Protocols
IO Services
HINT HMI Interface (ActiveX)
GATEWAY (Other Protocols)
DATABASE (ODBC)
Control LAN Protocols
EGD
TCNET
DLAN
BRIDGE (Other LAN)

Fig. 2 – Map of PA Solutions Product Suite

Configurability and Maintainability

As mentioned before, in a traditional approach, functional changes require re-linking and re-compiling. When a change has "global effects" such as a change to a "very popular" data record (e.g. PDI record), the inevitable system re-build was required. Variable names were not consistent through all system levels since Level 1 controllers and HMI system usually had

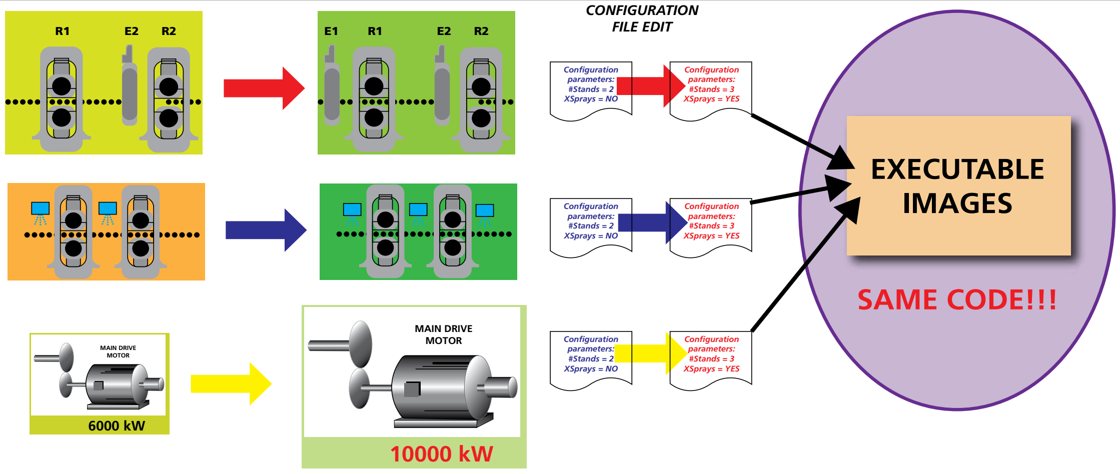

R1
E2
R2
E1
R1
E2
R2
MAIN DRIVE MOTOR
6000 kW
MAIN DRIVE MOTOR
10000 kW
CONFIGURATION FILE EDIT
Configuration parameters: #Stands = 2 XSprays = NO
Configuration parameters: #Stands = 3 XSprays = YES
Configuration parameters: #Stands = 2 XSprays = NO
Configuration parameters: #Stands = 3 XSprays = YES
Configuration parameters: #Stands = 2 XSprays = NO
Configuration parameters: #Stands = 3 XSprays = YES
EXECUTABLE IMAGES
SAME CODE!!!

Fig. 3 - Examples of Configurable Process Equipment Changes

their own naming conventions. In such systems changes were made with great care, and with no small amount of risk.
In the new Supervisory Process Automation, two key advancements make the project engineer's life less-stressful: the use of metadata and customization by configuration files. When a data record requires modification, the engineer modifies its properties in the Data Dictionary, edits configuration files as necessary (tables, scripts…), and restarts the affected process (or processes).

The cohesion of control system metadata is achieved by use of the centralized System Data Base (SDB) which includes the tag names of all global system variables. The same tag (symbols) names are used throughout the all system levels since all system members are bound against SDB. Should any global signal be changed or added, the symbol server process refreshes definitions from SDB. All affected applications are simply re-started.

A similar concept of acquiring data structure definitions during run-time rather than during compilation time is applied. The special Database called Data Dictionary contains system metadata i.e. the definitions of all data structures (such as data records) required for inter-process data exchanges. If a change to a data record is required, its definition is modified in Data Dictionary DB. Then all affected applications are re-started to acquire actual record definitions.

Even if the controlled process configuration is seriously modified, e.g. Edger stand is added in the mill, no code changes are required. The required data structures are added to DB definitions, and configuration files are modified to reflect the new equipment addition including its physical location, ratings, control characteristics etc. In this way, system upgrades and maintenance is quite simple.

"Scaleability"

The standard products are designed to be used for simple (such as skin pass mill) and complex (such as hot strip mill) applications. A good example is the process data scans where the task named POND (**P**ile **o**f **N**umerical **D**ata) is used. POND is a fully configurable and scalable product, storing collected data on length basis in the circular buffers. POND is usually configured to collect the process data continuously at the certain critical locations of the controlled line e.g. Finish Mill Stand. The data is stored in the shared memory buffers for further use by other applications. In case of simple line, only one POND may be required, for complex applications, several PONDS are used. Individual PONDS are defined in a configuration file, and when started, ru

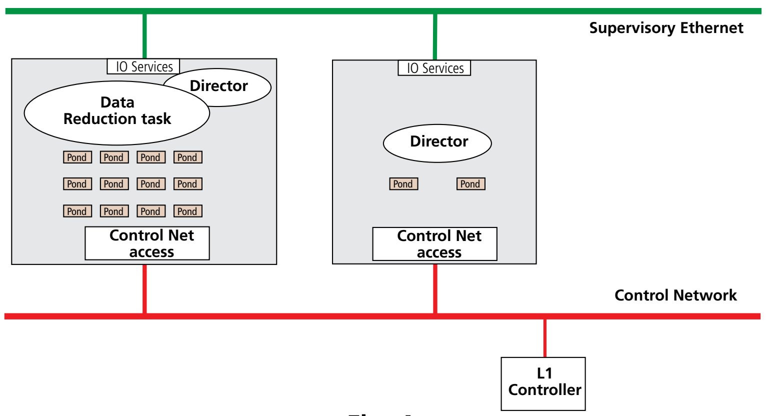


Fig. 4
Example of Complex and Simple Process Data Scans

Platform Flexibility

By decreasing its overall dependence on a specific Operating System (OS) and increasing capability for Distributed Computing, PASolutions offers a greater level of Platform Flexibility.

The C++ source code of the structured products is developed and maintained in the Microsoft Visual Studio® development environment. However, depending on the End User's preferences, the final destination computer can be either Windows® or VMS® or Linux® based. Operating system dependent parts of the source code are minimized and generally separated from the rest which is then largely identical for any of these OS. The Windows® or Linux® executable images are then prepared on transferable media for easy step-by-step installation. For VMS®, the source code must be compiled and linked on the target computer.

The PASolutions structured products may all be installed on one computer or "distributed" among several networked machines with different OS. Thanks to a universal scheme of the inter-process communication (using IO Services), the software products can be moved from one computer to another without negatively affecting the system behavior. This provides the End User with extra flexibility when desiring to change the system configuration, add computers etc.

Open System

Various connections to external systems are supported. For data link to upper levels such as Enterprise level systems three basic transfer mechanisms are supported:

- Messaging (TCP/IP, IBM Websphere, BEA Message Queue are supported by "Gateway" structured product.)
- File transfer Protocol (FTP)
- Open Data Base Connectivity (ODBC)

The connections to HMI utilize standard communication products and ACTIVEX container feature of standard HMI packages.

Since all essential process data is stored in the relational database, End User can apply standard "off-the shelf" products for process data analysis and reports. Any applicable software package which supports ODBC connection can be used to access the data, perform analysis and generate reports. The commonly used applications are: MS Excel, MathCAD, Statistica and MiniTab. "Standard" Excel-based reports are delivered with the system for direct usage as well as for further enhancements.

In addition to supporting industry standard formats and protocols, PASolutions offers a Software Developers Kit (SDK) providing custom programs access to PASolutions "native" communication protocol and data storage format.

Agility

Typical well-formed supervisory systems are composed of a set of processes with an appropriate division of labor and functional responsibility. However, in complex systems, this "appropriate division" is often a moving target during project implementation, commissioning, and sometimes after commissioning. Leveraging the design concepts outlined above, PASolutions allows one to modify and redeploy functional behavior quickly (sometimes without stopping any processes) while limiting disruption to the overall system, affecting only the processes that "care about the change". For example, one can modify a data structure and restart affected processes within seconds, or one can redistribute processes among a set of computers within minutes.

Structured Products Features and Configuration Examples

Overview

Some of the design features within the five major categories of PASolutions are described below.

Category	Product
Metadata	DataDictionary and Signal Database
Communications	IO_Services, Gateway and HINT
Tracking and Data Management	Director
Data Collection	POND
Tools	SSAM

The Figure 5 depicts the fundamental parts of PASolutions which are "instanced" in various combinations to implement any variety of process automation system application. This depicts the general relationships of the categories and products listed.

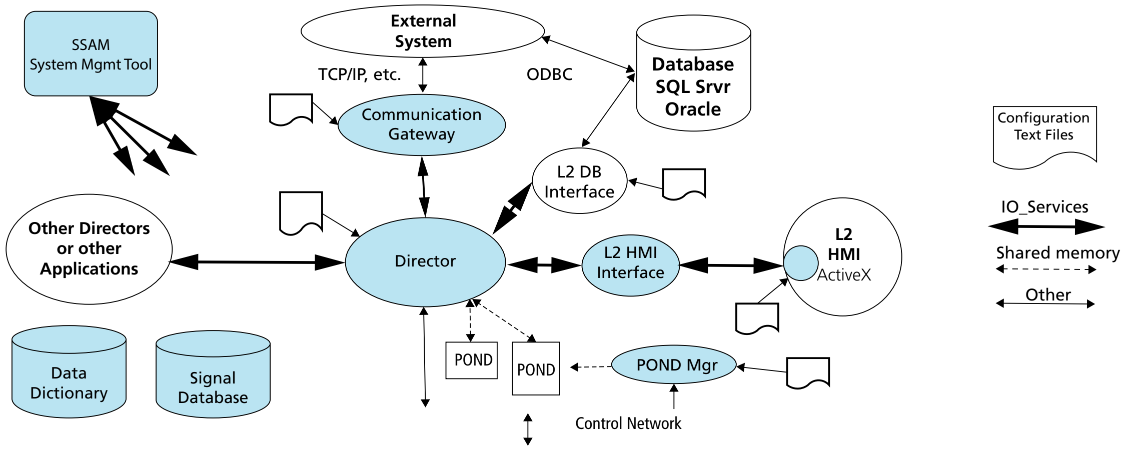
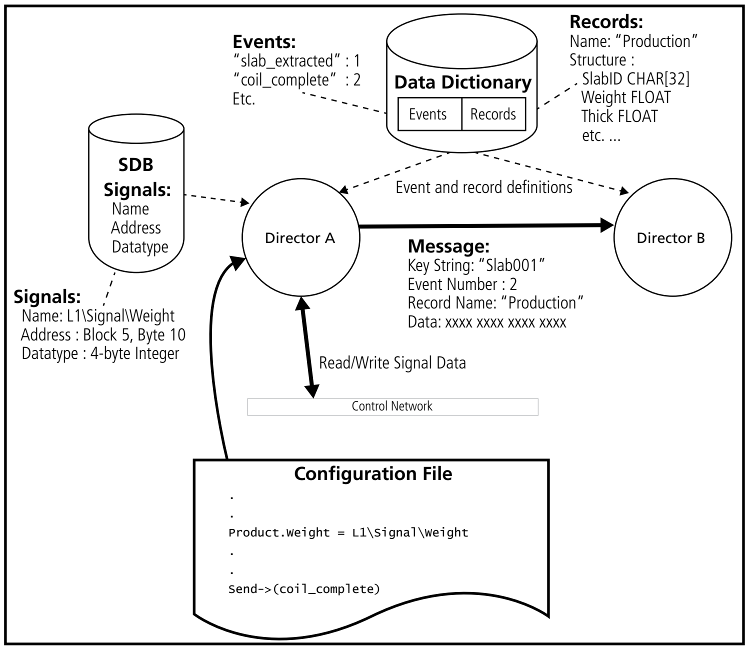


Fig. 5 - Basic Roadmap of PASolutions Products

Use of Metadata

Many of the engineering productivity advantages realized by PASolutions are due to two key elements of infrastructure: metadata (information about data) and a sophisticated communication system. By referencing metadata, structured product software quickly adapts to changes in interface definitions. The communication software, meanwhile, accommodates the use of metadata and provides flexibility and agility required to quickly form new software topologies. Metadata is applied to two primary areas: (1) data structures for messaging and storage and (2) communication on control networks. In the first case, a Data Dictionary contains the definition of data structures and enumerations passed as IO_Services messages between processes or stored in application databases (PDI, HISTORY, etc.). This information is typically managed by "Level 2" engineers. Secondly, a Signal Database contains names and properties of variables on the control network. This information, usually managed by a Level 1 configuration tool, may be cached in the Supervisory Automation System.

In the figure, notice the simple Director script. Recall that this configuration script is read upon initialization, at which time the necessary attributes of the signal variable, event number, and record structure are also obtained. In this way, one need only modify the definitions and restart the processes to adapt to new interface specifications.



```
.
.
Product.Weight = L1\Signal\Weight
.
.
Send->(coil_complete)
```

Fig. 6 - Use of Metadata for Configurable Communication

Communications Mechanisms

PASolutions offers a suite of communications mechanisms with product software serving various areas:

Product	Description
IO_Services	Message-based Internal Communications
Gateway	Protocol Translation
HINT	Interfact to HMI systems
DBServices	Interface to MS SQL Server or ORACLE

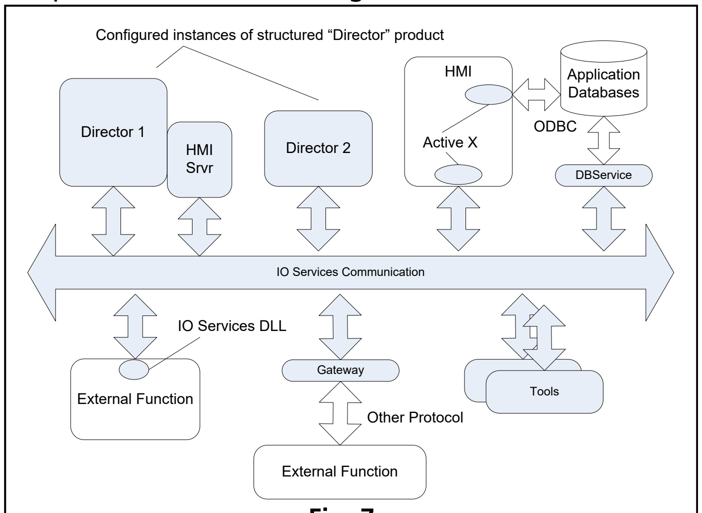


Fig. 7 -
Example use of Gateway for TCP/IP Communcation

IO_Services

IO_Services is a unique message-based communication service forming the backbone of PASolutions systems. Applications using IO_Services connect and communicate through "Interface Objects" (IO's). Based on UDP and implemented for Windows, Linux and VMS, IO_Services unites all Supervisory Automation runtime processes and tools in cross-platform systems.

IO_Services automatically reconfigures its virtual network (a.k.a. "job") allowing computers and processes join and exit the system dynamically. Tools provide convenient monitoring and diagnostics. With a Data Dictionary, monitored messages are translated for display in engineering units.

IO_Services guarantees the delivery and integrity of network messages, while making efficient use of compute resources. A simple API allows custom programs to join and communicate within an IO_Services "job". The table displays additional unique features making IO_Services particularly suitable to agile programming in Supervisory Automation Systems.

Unique Feature	**Positive Impact**
Multiple clients can receive the same message from a single sender	Senders do not require awareness of other programs interested in their messages.
An IO (interface object) is valid for the life of a client, even when other clients restart	Clients do not need to constantly monitor and reconnect to partner programs.
A program can "Attach" to a client before that client has been started	Clients can be started (and restarted) in any order without regard to communication connections
"Handle" names are automatically visible to all all computers with same "job code".	Clients are easily moved from one computer to another without code changes. No name configuration is required.
IO_Services Utility provides a wealth of diagnostic information.	All message traffic between clients can be monitored, captured, saved and replayed as needed. Detailed information about all clients and their objects can be displayed. Statistics on messge traffic, network usage, communications efficiency, etc.
Multiple "jobs" may be started on a single network (or computer).	Processes may be partitioned into communication "groups" which cannot interact with each other.

Gateway

The Gateway product is a conceptually simple application that allows PASolutions applications to communicate with external systems by several commonly used protocols. (TCP/IP, IBM Websphere, BEA Message Queue). The data within each message are not interpreted, but "repackaged". (So the Data Dictionary serves no function here.) In addition to serving as a "protocol translator", a Gateway process manages connections (retry, re-connect logic) and optionally queues outgoing messages. In case of TCP/IP, a single Gateway process can interface to multiple external sockets. So a single Gateway process can service all external connections. Alternatively, separate Gateways may be instanced for dedicated communication to specific external systems.

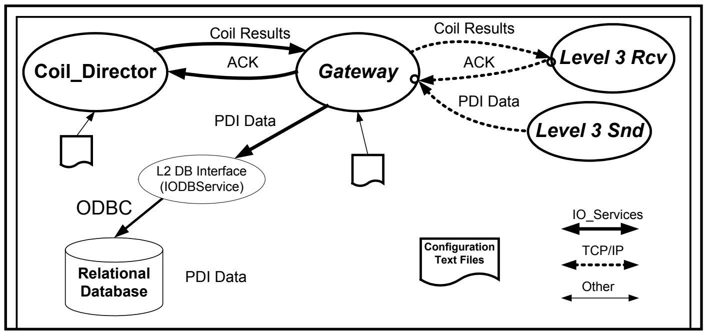


Fig. 8 -
Example use of Gateway for TCP/IP Communcation

Its simple configuration file describes connection and routing information and the necessary basic characteristics of messages expected to pass through the Gateway. Configuration options accommodate common TCP/IP schem

```
                         Example Gateway Configuration
[GATEWAY]
  Debug     5              ; Debug level (0 = off, 1..5 for messages)
  Cycle     5              ; Remote connection retry cycle time (in seconds)
  Port      10022          ; Incoming TCP port for other nodes to 'connect' to
  Interface 191.0.2.40     ; IP address of network interface to use (if more than 1)
  Handle    Gateway        ; Handle name programs send to (and name of this process)
  Max_Data  1024           ; Maximum message size (bytes)
  Ascii                    ; Data type of header len & type info (Ascii | Binary)
  Length    61 4 0         ; Position & Size & offset of msg length field
  Type      0 5            ; Position & Size of msg type field

[INCOMING]  ; From TCP/IP to IO_Services  (From L3 to L2)
;===========================
; MSG     TCP Message Type       IO_Services    IO_Services      IO_Services    Data   Formatted
; Type    Debug Description      Handle Name    Action Name      Selector       Type   RecordName
; ----    -----------------      -----------    ---------------  -----------    ----   ----------
  SH001   "PDI Data"             L3IF_DIR       P_Rcv_L3PDIMsg       ""         101    r_L3PDIMsg
  SH002   "Ack from L3"          L3IF_DIR       P_Rcv_L3AckMsg       ""         101    r_L3AckMsg

[OUTGOING]  ; From IO_Services to TCP/IP   (From L2 to L3)
;===========================
; IO_Services       TCP Message      Que  Port     Primary Node    Secondary Node
; Action Name       Debug Descrip    Y/N  Number   Name / Number   Name / Number
; -----------       -------------    ---  ------   -------------   -------------
  P_Snd_CoilResults "Coil Results"    Y   10021    191.0.2.46      191.0.2.48
; optional additional messages to any port/node
```

Fig. 9 - Example of Gateway Configuration File

HMI Communications with HINT

HINT (HMI Interface New Technology) originated within PASolutions in the mid-1990's as a mechanism to provide internal automation data to various HMI platforms and to allow sophisticated interaction between the HMI screens and the Supervisory Automation System. Because many common HMI systems are ActiveX containers, this technology presented itself as a convenient way to "package and deliver" the PASolutions communications solution for HMI.

The HINT product includes three object classes as depicted. All three objects take advantage of the Data Dictionary to interpret the data records and events used in messaging. HINT DATA objects interact with HINT Server objects to automatically provide latest tracking or piece (coil) information on an event-basis. Each object "establishes" a contract with the server regarding the type of information to be sent. HINT DB objects provide access to application databases such as PDI database. Both HINT DATA and HINT DB manage multi-user situations, giving operators the opportunity to recognize when others are modifying the same data and take appropriate action. Normally invisible to the operator, HINT DATA and HINT DB objects can provide visible dialog windows to edit data. These canned pop-ups display the Data Dictionary record name as well as maximum and minimum allowed limits. Finally, HINT COMM objects provide a straight-forward mechanism for HMI to send IO_Services messages to pass data and trigger events on the Supervisory System cor

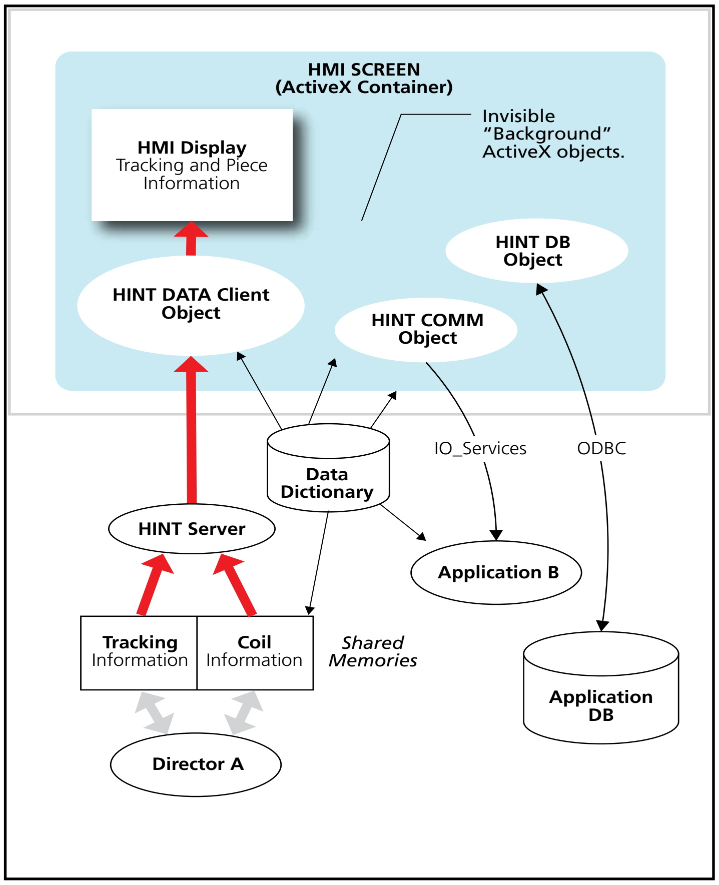


Fig. 10 - Configurable HMI Commnications Objects

Director

The Director structured product (a.k.a the "Piece Director") is the centerpiece of most PASolutions implementations. Applied mostly for tracking and data handling of "pieces" (coils, slabs, strips, etc.), the director offers a generic set of powerful functions making it useful for a wide variety of tasks within the Supervisory Process Automation systems. Therefore it is often the case that many instances of a director are configured to run within a single system – spanning a range of complexities from the mundane gauge interface to mill tracking and data analysis.

As shown in the text box, the director is "scripted" with syntax similar to the familiar "C" language. The Director is "event driven" so one configures, through scripting, the Director's response to each expected event. Director script syntax includes the expected compliment of features (macros, ASCII, Standard C Math functions, etc.). But what make the Director particularly useful for automation engineers are the many additional built-in objects that automatically integrate a Director within the PASolutions environment and perform standard functions common to automation systems. For example, a Director can inherently "speak IO Services", read and write from the Control Network, and access POND sample data. With the "basics" already provided for, project engineers are able to concentrate on configuring the custom aspects of a particular solution and more quickly develop a working system.

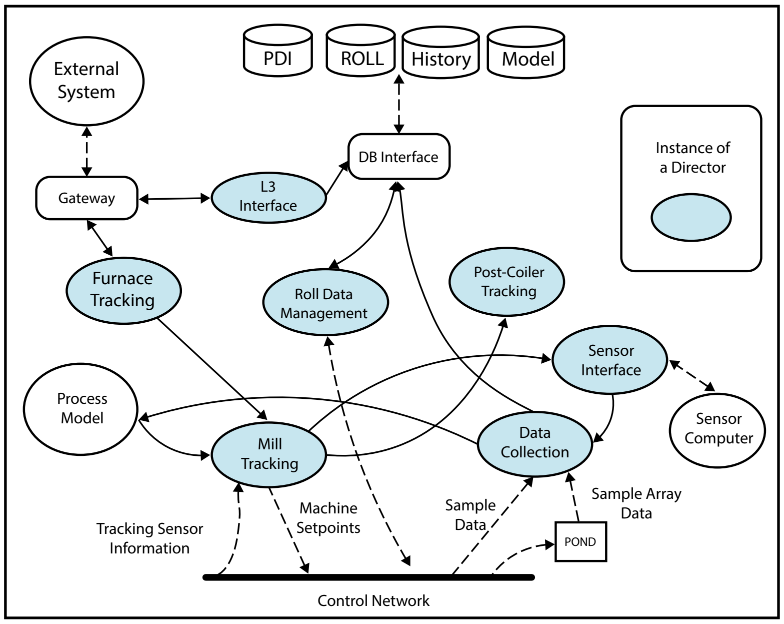


Fig. 11 - Multiple Instances of Director Applications

As records (data structures) are defined in the data dictionary and control network signals are described in the signal database, aided by the metadata infrastructure of PASolutions, the director script can quickly utilize the definitions referring to records and signals by name.

Director Scripting Example

```
! Respond to incoming IO_Services message
EVENT = coil_accept;

  ! Create Coil Object data structure
  Piece_Director->Create(new_coil, piece_id, type_entry_coil);

  IF ( piece_id <> 0 ) THEN

       ! Modify data in the coil object
       new_coil->TrackingRecord.accepted = TRUE;

       ! Read array of POND samples and copy into Coil Object
       My_POND->Read( My_Signal_Name, My_Data_Samples );

       new_coil->Coil_Data = My_Data_Samples;

       ! Notify other by sending IO_Services message
       Other_App_IF->Send(new_coil_event);

       ! Notify others by setting a Control Network variable
       TRACK\ENTRYCOIL\ACCEPT = 1;

  ELSE
       ! Issue alarm message
       s_AlarmText = "Failure Message";
       Alarm->Send (
            s_AlarmID,
            i_AlarmClass,
            s_AlarmResource,
            s_AlarmText,
            n_AlarmOutput ) ;
  ENDIF;

END_EVENT;
```

Director Built-in Objects

Piece Director	Create & manage "piece objects"
Zone Director	Manage piece membership in Zone objects.
IO Serv	IO Services communication
Pond	Access POND shared memory
Calc	Sample Set Manipulation
Serial Interface	Client side TCP/IP connection

Others set Internal Timers, Monitor other processes, Manipulate unformatted Blocks of memory, conduct File IO and send Alarms.

A multi-purpose software tool, the Director and the rest of PASolutions have been successfully applied to automation projects ranging from Slab Yards, to Process Lines to Rolling Mills of all types.

In the typical automation system application the "Director" product is instanced multiple times such as: "Piece Director", "Tracking" and "Gauge Interfaces"

POND Data Sampler

Gathering data samples is an essential and ubiquitous activity in Supervisory Process Automation systems. Necessary for process model feedback and adaptation, for quality control and diagnostics, data collection is equally important as tracking. It is also an activity which is common to nearly all types of automation systems. PASolutions "packages" this function in a configurable product called POND (a whimsical acronym for "Pile of Numeric Data"). Essentially, this product can be described as an Electronic Strip Chart Recorder. The name "POND" also refers to a single dataset stored in shared memory as depicted.

When taking large quantities of samples, precision of the sampling instant is a concern. By isolating this activity in a separate dedicated process (the POND Manager), the Operating System may give preferential treatment only to this intense activity. When a Director, for example, wants high resolution sampled data, it simply maps to an appropriate POND and reads the data.

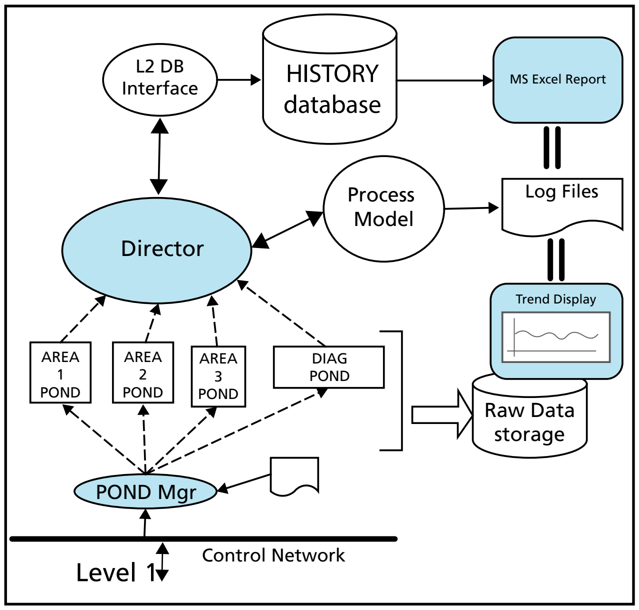
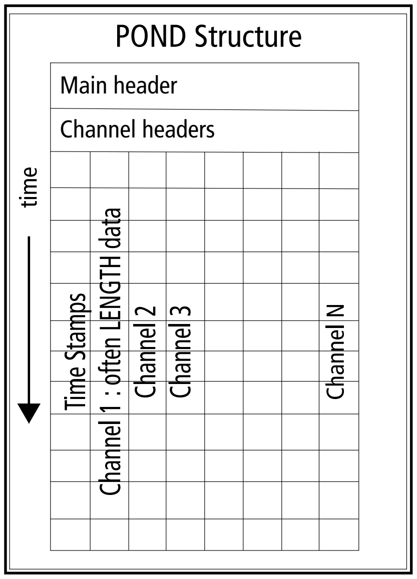


Fig.12 – POND Data Flow

All PONDs are managed by a single POND manager process according to a single configuration file. Utilizing SDB metadata, each POND may be configured with names of "control signals" and a list of signals to be sampled (one signal per CHANNEL). The control signals indicate when to Start and Stop sampling. Each POND is also configured to take samples in one of three styles : Time-based sampling, Length-based sampling (based on speed integration) or Event-based sampling (based on a "trigger" signal). A POND may buffer data in the so-called 'circular' style or not circular ('linear').

In batch Rolling Mills, most PONDs are configured to be Length-Based, Linear PONDS and collect data related to the strip passing a particular point in the mill. A Circular POND with many channels is also configured to provide diagnostic data from throughout the mill (in case of emergency stops, for example). In a full Hot Mill scenario, there may be more than a dozen PONDs.



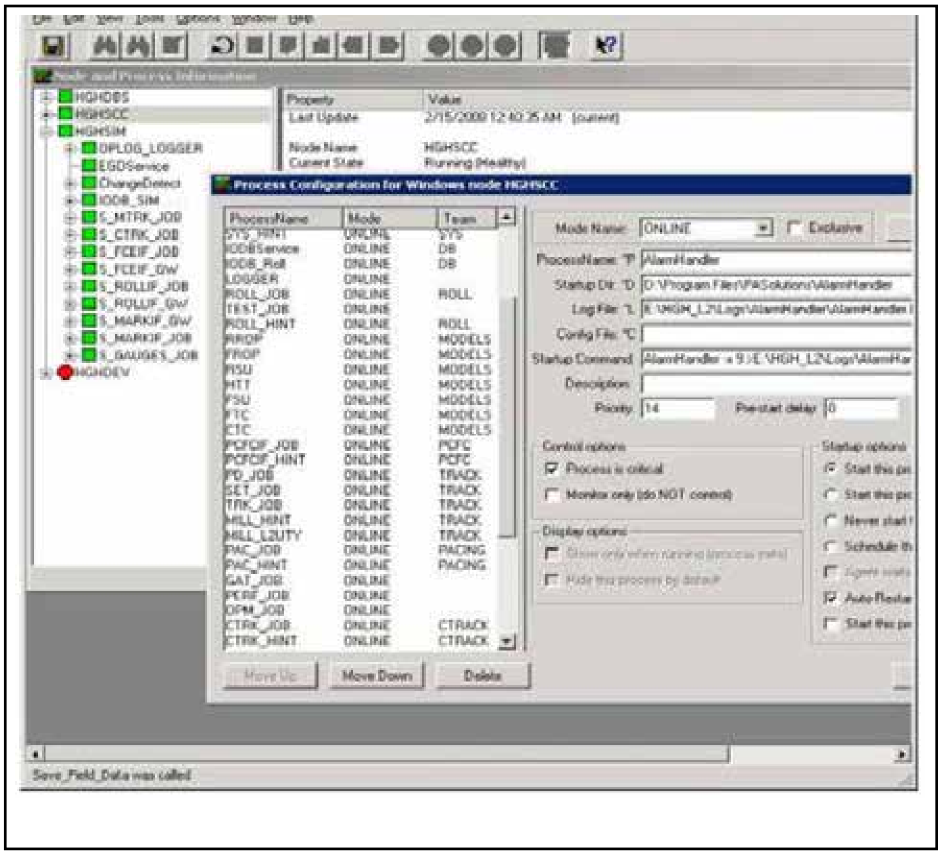
Example POND configuration

```
Diagnostics:     3           ! Enable diagnostics messages (0 -> 9)
Speed_Format:   MPS          ! MPM, FPM, MPS, FPS
CD_RATE:         0.030       ! seconds

! ----------------------------------------------------------------------
! POND NAME   : F1_POND
! TIMING      : Between FM1 ON and OFF / every quarter meter
! ----------------------------------------------------------------------
!                 <name>              <type>      <num samples>
POND:             F1_POND             linear      3000

    LEN_SCALE:   0.25

    START:       TN\L_SUC_F1LRX      PU_DO
    SPEED:       TN\L_FM_F1_SFB              ! F1 Exit Strip Speed

    CHANNEL:     TN\L_AG1_F1CTGAPFBK        ! F1 Center Gap [mm]
    CHANNEL:     TN\L_F1H_TLRLFCFBK         ! F1 Total Rolling Force [kN]
! other channels removed
    CHANNEL:     TN\L_F1LP_FCFBK            ! NO.1 Looper Force [N]
    CHANNEL:     TN\L_FM_F1TRANSLEN         ! F1 Strip Transfer Length [m]

END POND;
```

In Continuous Rolling Mills or Process Lines, PONDs are generally configured as Event-Based Circular PONDs, taking samples when a Length counter signal is incremented. When a finish cut occurs, a specifically configured Director gathers the appropriate data for the cut material found in multiple PONDs from the beginning to the end of the process.

Supervisory System Analysis and Monitoring (SSAM)

Because IO_Services offers distributed cross-platform communication for the runtime processes responsible process automation, it is available also for management and diagnostics of these same processes. The SSAM product utilizes IO_Services and operating system functions to provide a single interface to project engineers interested in the managing the complete multi-node Supervisory Process Automation system.


HGHDBS
HGHSCC
HGHSIM
- DPLOG_LOGGER
- EGDService
- ChangeDetect
- IODB_SIM
- S_MTRK_JOB
- S_CTRK_JOB
- S_FCEIF_JOB
- S_FCEIF_GW
- S_ROLLIF_JOB
- S_ROLLIF_GW
- S_MARKIF_GW
- S_MARKIF_JOB
- S_GAUGES_JOB

HGHDEV

Property	Value
Last Update	2/15/2008 12:40:35 AM (current)
Node Name	HGHSCC
Current State	Running (Healthy)

Process Configuration for Windows node HGHSCC

ProcessName	Mode	Team
SYS_HINT	ONLINE	SYS
IODBService	ONLINE	DB
IODB_Roll	ONLINE	DB
LOGGER	ONLINE	
ROLL_JOB	ONLINE	ROLL
TEST_JOB	ONLINE	
ROLL_HINT	ONLINE	ROLL
RROP	ONLINE	MODELS
FROP	ONLINE	MODELS
RSU	ONLINE	MODELS
HTT	ONLINE	MODELS
FSU	ONLINE	MODELS
FTC	ONLINE	MODELS
CTC	ONLINE	MODELS
PCFCIF_JOB	ONLINE	PCFC
PCFCIF_HINT	ONLINE	PCFC
PD_JOB	ONLINE	TRACK
SET_JOB	ONLINE	TRACK
TRK_JOB	ONLINE	TRACK
MILL_HINT	ONLINE	TRACK
MILL_L2UTY	ONLINE	TRACK
PAC_JOB	ONLINE	PACING
PAC_HINT	ONLINE	PACING
GAT_JOB	ONLINE	
PERF_JOB	ONLINE	
OPM_JOB	ONLINE	
CTRK_JOB	ONLINE	CTRACK
CTRK_HINT	ONLINE	CTRACK

Mode Name: ONLINE Exclusive
ProcessName %P: AlarmHandler
Startup Dir %D: D:\Program Files\PASolutions\AlarmHandler
Log File %L: E:\HGH_L2\Logs\AlarmHandler\AlarmHandler
Config File %C:
Startup Command: AlarmHandler -s 9 -l E:\HGH_L2\Logs\AlarmHa
Description:
Priority: 14 Pre-start delay: 0

Control options
- [✓] Process is critical
- [ ] Monitor only (do NOT control)

Display options
- [ ] Show only when running (process stats)
- [ ] Hide this process by default

Startup options
- (●) Start this pr
- ( ) Start this pr
- ( ) Never start
- ( ) Schedule th
- [ ] Agent waits
- [✓] Auto Restar
- [ ] Start this pr

Save_Field_Data was called

File Edit View Tools Options Window Help
Node and Process Information
Move Up
Move Down
Delete

Fig.15 – SSAM UI – Configuring Process Startup Parameters

The SSAM product consists of two parts: a "back end" agent and the GUI "front end". SSAM Agents are installed and run on the same computers running the automation applications. The SSAM User Interface (SSAM UI) runs on any Windows or Linux computer within the same IO_Services "job code".

With any SSAM UI, one may manage processes throughout the system, local or remote, start and stop them, even monitor message traffic. The SSAM UI communicates with the SSAM Agents, sending commands and receiving process information for presentation to the user.

This powerful time-saving tool adds another dimension to the "configurability" and "manageability" of Supervisory Process Automation based on PASolutions.

SSAM Features

Configure	Programs, Services, Batch Files Startup Parameters, Priority Startup Sequence
Control	Start / Stop Complete Node Start / Stop Individual Processes Manual, Automatic Restart Schedule batch executions
Monitor	IO_Services message activity Other OS process statistics Application Log Files Automatic Alarm Generation

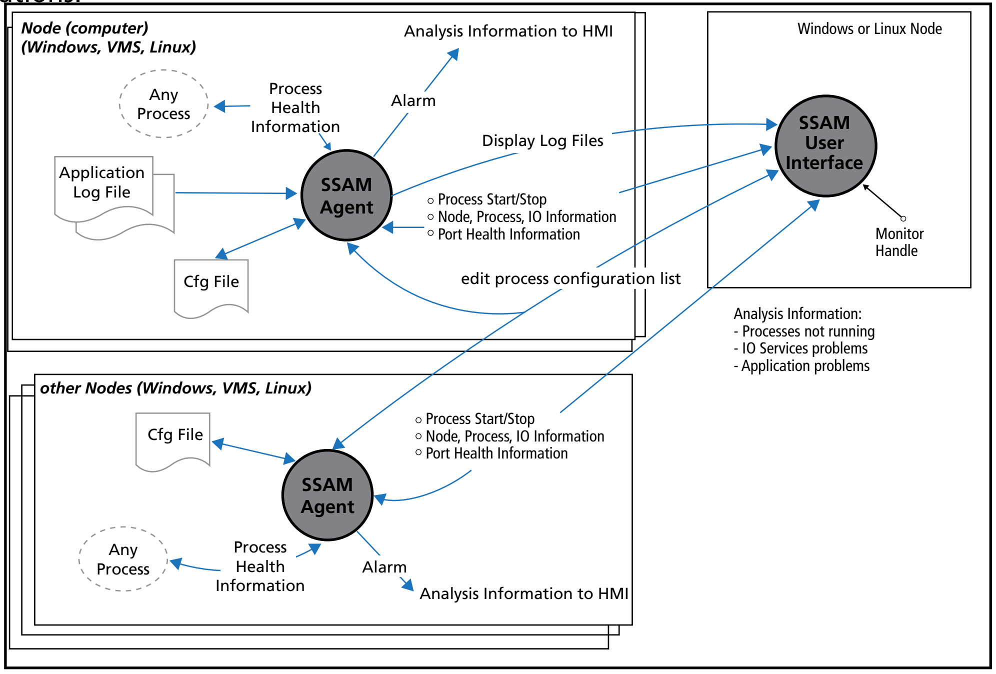
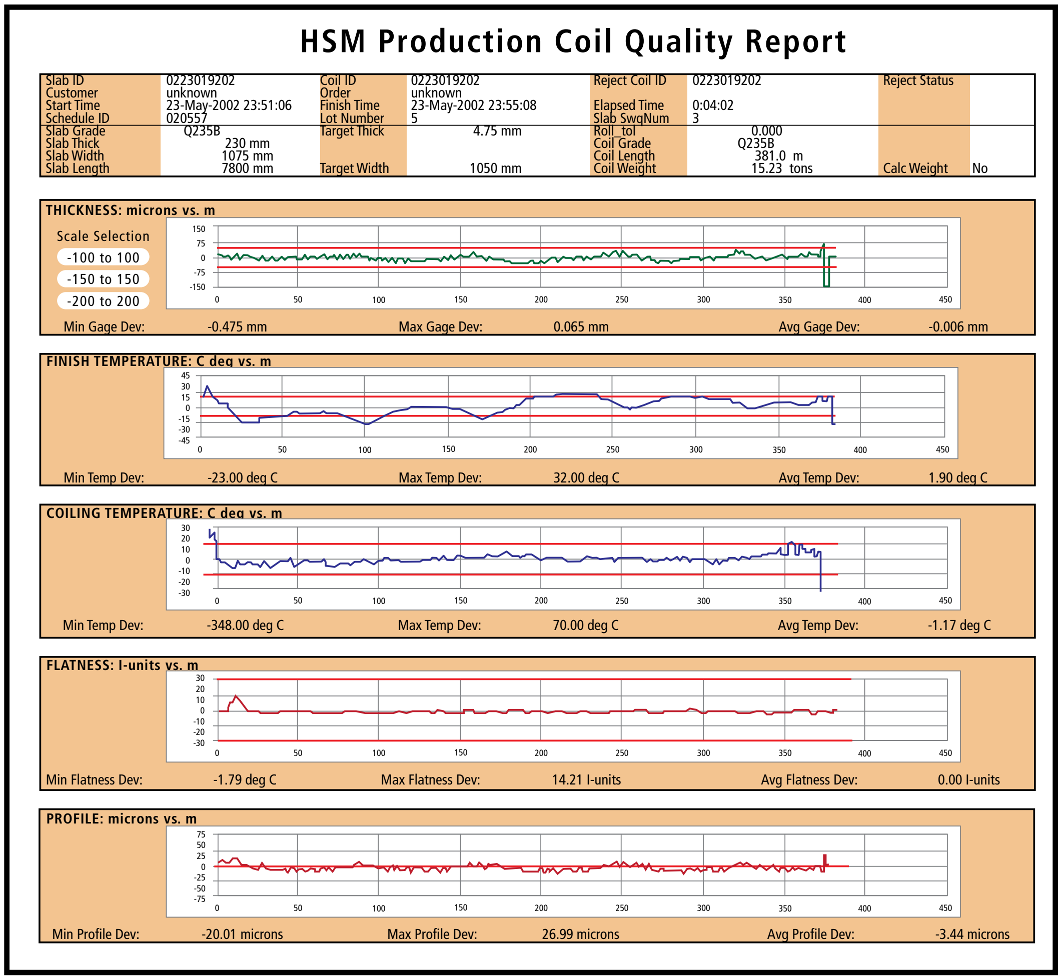


Fig.16 – SSAM Software Architecture

Reports

As mentioned before, the data storage is implemented based on the standard software package such as Oracle Database or Microsoft SQL Server database. Several separate databases are implemented for Metal Rolling application systems, the biggest and the most important being "Rolling History" DB. This DB includes data of all rolled products, including quality and performance evaluation. Due to the openness and flexibility of data access to relational database, the customized coded report/logs application are not longer required.
Most of the office type applications such as Microsoft EXCEL, VISIO etc. include remote access features (ODBC) to the external databases as well as possibility to create own queries to retrieve DB data.
The reports customized for the particular needs can be then easily created and modified. The textual and graphical formats can be also combined in single report. Reports data processing and data presentation can be augmented by including VB scripting into the application.



HSM Production Coil Quality Report

Slab ID	0223019202	Coil ID	0223019202	Reject Coil ID	0223019202	Reject Status	
Customer	unknown	Order	unknown				
Start Time	23-May-2002 23:51:06	Finish Time	23-May-2002 23:55:08	Elapsed Time	0:04:02		
Schedule ID	020557	Lot Number	5	Slab SwqNum	3		
Slab Grade	Q235B	Target Thick	4.75 mm	Roll_tol	0.000		
Slab Thick	230 mm			Coil Grade	Q235B		
Slab Width	1075 mm			Coil Length	381.0 m		
Slab Length	7800 mm	Target Width	1050 mm	Coil Weight	15.23 tons	Calc Weight	No

THICKNESS: microns vs. m

Scale Selection: -100 to 100 / -150 to 150 / -200 to 200

Min Gage Dev: -0.475 mm Max Gage Dev: 0.065 mm Avg Gage Dev: -0.006 mm

FINISH TEMPERATURE: C deg vs. m

Min Temp Dev: -23.00 deg C Max Temp Dev: 32.00 deg C Avg Temp Dev: 1.90 deg C

COILING TEMPERATURE: C deg vs. m

Min Temp Dev: -348.00 deg C Max Temp Dev: 70.00 deg C Avg Temp Dev: -1.17 deg C

FLATNESS: I-units vs. m

Min Flatness Dev: -1.79 deg C Max Flatness Dev: 14.21 I-units Avg Flatness Dev: 0.00 I-units

PROFILE: microns vs. m

Min Profile Dev: -20.01 microns Max Profile Dev: 26.99 microns Avg Profile Dev: -3.44 microns

Fig. 17 – Example of Coil Quality Report for HSM application

The Process Models software can also be considered as a part of Supervisory Process Automation, although they are usually categorized as a "Process Technology Applications". Vast improvements to process modeling technology in the recent years have been made. Main emphasis was given to reduce empirical or tabular solutions by replacing them with theoretical and physical math solutions. Among these improvements, has been development of advanced force and power models explicitly including coefficient of friction and flow stress and phase transformation modeling that use actual piece
chemistry. These solutions often require iterative mathematical computation because of the implicit multi-variable equations. They have been combined with advanced mathematical and control theory techniques to create accurate and stable online adaptation techniques which result in superior product quality within a minimal number of pieces. The fundamental changes in process modeling algorithms ignited also the concepts of new software structuring which could be implemented for these applications.

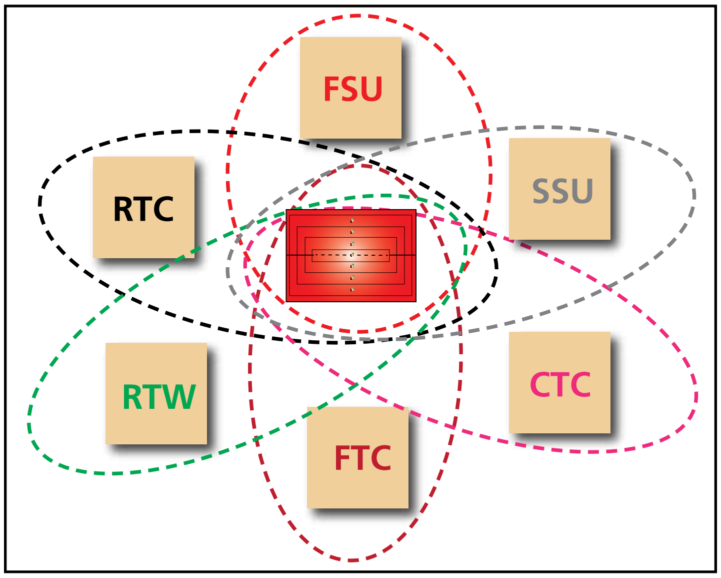


Fig. 18 – "Building Block" concept for Process Modeling

Process Model software is implemented under true "Object Oriented" standards. Software infrastructure is highly configurable and easy to use. The process, mill equipment and specific rolling practices can be pre-set and/or modified through change of text based configuration file. No code change is required for most of modification. Also, control and adaptation behavior can be affected through change of function parameters and gains.

Application code is written in C++ language, however due to extensive use of configurable software structure, End User is rarely aware of code details. Code simplification and reduced maintenance has been achieved through maximization of code sharing. The "building block" concept through shared communications software, shared functional blocks (like force and temperature algorithms) is now widely used in software. The diagram illustrates this concept showing all Rolling Process Models using the same finite difference "building block" for all temperature calculations. Taking an advantage of object structured software and applying concepts of "building blocks" results in the code unification across the whole spectrum of Metal Rolling applications. The same code is used for single stand reversing mill or for 6-Stand continuous mill .The differences are only in configuration files.

SUMMARY AND CONCLUSIONS

This approach to Supervisory Process Automation software can be summarized as a suite of fully structured, configurable and installable products which fulfill the functionality required for any Metal Rolling and Processing control system.

The advantages for System Suppliers are obvious due to increase productivity. It takes much less engineering time to create the software for a particular application since reusable product images are applied. It takes much less testing and integration time due to the proven inter-process communication and standardization of data structures. As a result, Process Automation engineers can concentrate on the process requirements and control specifics rather than "struggling" with code design, compilations and debugging. Many Process Control engineers working on various metal control applications reported that they have not needed to write a single line of code during whole project life - from functional design to full implementation.

The End User benefits can be listed in several categories:

- Better Process Knowledge
  - Process Automation Engineers focus on process engineering NOT on software engineering
- Fast Training Curve
  - Clearly defined and documented software products
  - Standard "installation kit"
  - No compiler necessary
  - Flexible configuration "scripts" - many based on C-language syntax and SQL statements
- Commissioning
  - Shorter period with fewer engineers.
  - Serious software problems already avoided by well tested foundation software.
  - Extensive monitoring and diagnostic tools accelerate problem diagnosis and solution.
  - Functions reconfigured and restarted independently.
- Easy Operation & Maintenance
  - Unified L2 control panel for easy monitoring of system-wide status
  - Event based communication drives low CPU usage
  - Greater Up-Time due to: reliable base software, easy online intervention and modification, functions reconfigured and restarted independently.
- Expansion and Upgrades
  - Open architecture for simplified interfaces
    - TCP/IP, ODBC, SDK for Native Communication
  - Interoperability with VMS, Linux, Windows.
  - New releases of PASolutions foundation software are managed for backwards compatibility

REFERENCES

M. Foster, R. Gillmore - IO Services Users Guide (2008) – Internal document of TMEIC Corporation, Roanoke, VA -USA

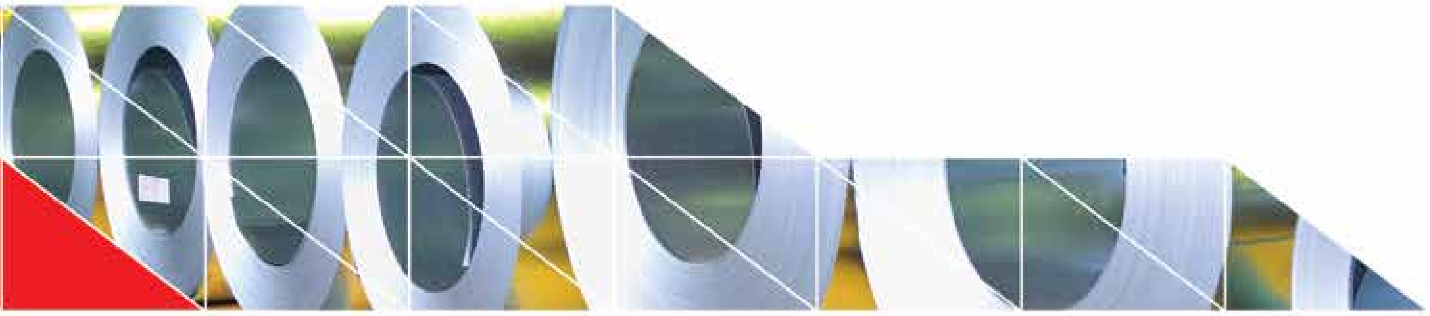
R. Mullet - Director Users Guide (2008) – Internal document of TMEIC Corporation, Roanoke, VA -USA

R. Gillmore - Gateway Users Guide (2008) – Internal document of TMEIC Corporation, Roanoke, VA -USA

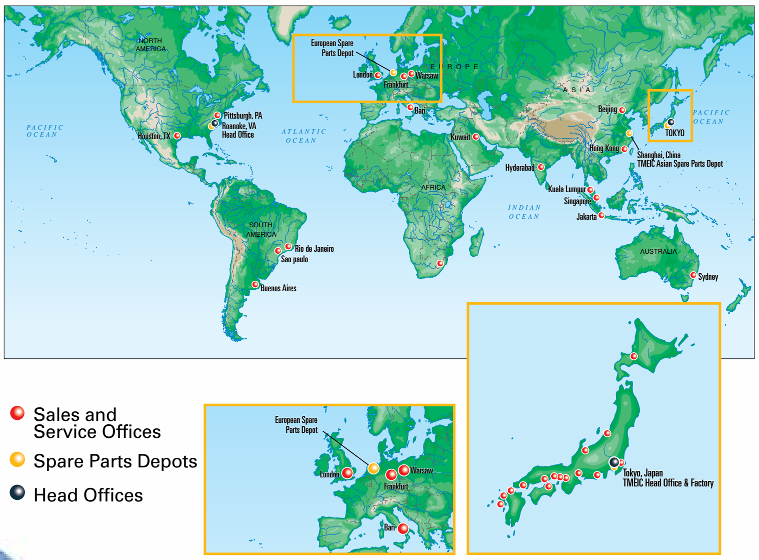
R. Gillmore - POND Users Guide (2008) – Internal document of TMEIC Corporation, Roanoke, VA -USA

R. Mullet - SSAM Users Guide (2008) – Internal document of TMEIC Corporation, Roanoke, VA -USA

W. Mulcahy - HINT Users Guide (2008) – Internal document of TMEIC Corporation, Roanoke, VA -USA



Global Support



🔴 Sales and Service Offices

🟡 Spare Parts Depots

⚫ Head Offices

Wherever You Are, We Are Right Next Door

TMEIC has the capability to provide world-wide service support with trained field service engineers. Spare parts depots are strategically located close to main industrial centers.

In Asia & Pacific: Customers are supported by TMEIC service personnel and the TMEIC factory in Japan.

In North America: Customers are supported by TMEIC factory service personnel from Roanoke, Virginia.

In Europe: Customers are supported by TMEIC European service personnel.

TMEIC Corporation Americas | Roanoke, Virginia | Houston, Texas | WWW.TMEIC.COM

I-2003
Revised June 2021