INTERNATIONAL STANDARD

ISO 15746-2

First edition
2017-10

Automation systems and integration — Integration of advanced process control and optimization capabilities for manufacturing systems —

Part 2: Activity models and information exchange

Systèmes d'automatisation et intégration — Intégration de contrôles de processus avancés et capacités d'optimisation des systèmes de fabrication —

Partie 2: Modèles d'activité et échange d'informations

Reference number
ISO 15746-2:2017(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 184, *Automation systems and integration*, Subcommittee SC 5, *Interoperability, integration and architectures of enterprise systems and automation applications*.

A list of all parts in the ISO 15746 series can be found on the ISO website.

Introduction

As a crucial part of manufacturing systems with increased complexity, the automation and control applications enabled by the advanced process control and optimization (APC-O) methodology and solutions perform the operations directed by production planning and scheduling. This work will initially deal with the specific use of APC-O to enable integration of manufacturing operations management (MOM) with the automation and control of manufacturing processes and equipment.

Automation solutions composed of software and hardware components are provided by different suppliers to accomplish APC-O functions. Due to the diversity of development environments and the variety of demand focus, the automation solutions from various suppliers are isolated and relatively independent. These differences make the integration of automation solutions difficult. Consequently, the customers could purchase different automation solution components with redundant and duplicated functions, resulting in a waste of resources and limited interoperability. The proposed standard offers a reference interoperability framework for advanced process control and optimization. It is intended to maximize the integration and interoperability of automation solutions.

This document provides a consistent framework for integration and interoperability between APC-O systems, parts of an APC-O system, Level 2 automated process control systems, and Level 3 manufacturing operations management systems. It builds on the functional models defined by ISO 15746-1 by defining activity models for APC-O systems and the information exchange requirements needed to support those activity models. The modelled activities operate within Level 2 and Level 3 with interaction between each level.

It is not the intent of this document to suggest that there is only one way of implementing APC-O or to force users to abandon their current way of implementing APC-O.

The target users of this standard include: users and providers of advanced process control and optimization solutions, such as, project solution suppliers, automation systems integrators, production departments of companies, process engineers, independent software testing organizations, implementation and consulting service organizations of advanced process control and optimization software, and relevant government and academic organizations.

Automation systems and integration — Integration of advanced process control and optimization capabilities for manufacturing systems —

Part 2:
Activity models and information exchange

1 Scope

This document defines:

— activity models to describe the dynamic aspects of the APC-O modules;

— information exchange requirements of the dynamic aspects of the APC-O modules;

— workflows and lifecycles of APC-O elements;

— service definitions to support the following information exchanges between:

  — Level 3 and APC-O components;

  — Level 2 and APC-O components;

  — APC-O components within one or more APC-O systems.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 15746-1, *Automation systems and integration — Integration of advanced process control and optimization capabilities for manufacturing systems — Part 1: Framework and functional model*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 15746-1 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at http://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org

3.1
alarm
notification regarding an abnormal condition of special significance to an *APC-O component* (3.2)

EXAMPLE A temperature controlled by an APC module exceeds a high limit or an input to a Soft Sensor module is outside the valid range.

3.2
APC-O component
procedural object in an APC-O system that is an instantiation of an APC-O functional module

Note 1 to entry: An APC-O component can execute various functions involving reasoning and communication with other APC-O components, interaction with simulation and optimization, diagnostic and forecasting, and control.

3.3
APC-O package
group of commercial software products supplied by a single vendor to deliver one or more APC-O functional modules

3.4
APC-O platform
computer-based system capable of managing and executing an APC-O system comprised of one or more *APC-O components* (3.2)

3.5
bias
adjustment made to a calculated value to force it to match actual measurements

Note 1 to entry: A bias is normally applied to account for unmeasured disturbances.

3.6
controlled variable
variable that a controller maintains either at a target or within minimum and maximum limits

Note 1 to entry: It accomplishes this by adjusting the *manipulated variables* (3.11).

3.7
disturbance variable
variable that a controller considers as influencing the *controlled variables* (3.6) but is not adjustable by the controller

3.8
event
detectable change or occurrence that changes the state of an *APC-O component* (3.2) or requires action by an APC-O component

EXAMPLE The mode of a PID loop associated with a *manipulated variable* (3.11) in an APC module changes to REMOTE or a process managed by an APC-O system changes state from “Producing” to “Shutdown”.

3.9
final control element
physical equipment that is actually moved to accomplish control of a process

EXAMPLE Valves, dampers, and variable speed drives.

3.10
hard limit
limit that a controller is not allowed to exceed under any circumstances

3.11
manipulated variable
variable that a controller adjusts

Note 1 to entry: These are typically setpoints of PID loops but could also be direct manipulation of a *final control element* (3.9).

3.12
performance baseline
evaluation and assessment of a set of key performance indicators for a process prior to implementing an APC-O system

Note 1 to entry: The intent is to evaluate performance of the APC-O system against the performance baseline.

3.13
rate of change limit
limit placed on the amount a value is allowed to change over a defined period of time

3.14
recipe
set of ingredients, product specifications, and process settings that define how to produce a product

3.15
soft limit
limit a controller should attempt to respect but may exceed if necessary

3.16
tracking
action of making the value of one parameter the same as the value of another parameter

EXAMPLE A PID controller may track the setpoint to the process value when it is in MANUAL mode.

3.17
trajectory
set of values, typically an array, representing expected behaviour over a defined time horizon

Note 1 to entry: In APC applications, the trajectory of a *manipulated variable* (3.11) is the set of moves planned by the controller and the trajectory of a *controlled variable* (3.6) is the expected changes based on those planned moves and recent history of process changes.

3.18
watchdog
function to determine if a component of the control system or an external system is functioning properly

Note 1 to entry: Typically, a watchdog function will perform some set of instructions if it determines the watched component is not functioning properly.

3.19
workflow
sequence of activities with explicit starting and ending points to describe a task

Note 1 to entry: Workflows may also have branches, decision points, and *events* (3.8). A workflow is a type of activity model.

4 Abbreviated terms

APC	advanced process control
APC-O	advanced process control and optimization
CV	controlled variable
DV	disturbance variable
KPI	key performance indicator
MPC	model predictive control

MV	manipulated variable
OP	output of a PID controller
OPC	open platform communications
OPD	object-process diagram
OPL	object process language
OPM	object process methodology
PID	proportional, integral, derivative
SP	Setpoint

5 APC-O workflow

5.1 Lifecycle concepts

The lifecycle of an APC-O system consists of the following phases:

a) Requirements Analysis

b) Design

c) Development

d) Execution

e) Support

Figure 1 illustrates the workflow of an APC-O system as it relates to the phases of the lifecycle. This and all subsequent illustrations are depicted using the object process methodology (OPM). Table 1 defines the OPM notations used in this document. Object process language (OPL) is the textual counterpart of the graphic OPM system specification. Example of OPL can be seen in Annex B.

Table 1 — OPM notation used

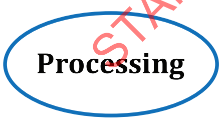
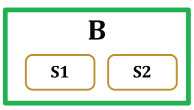
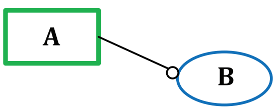
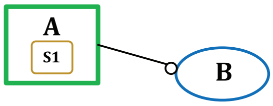
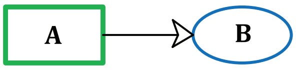
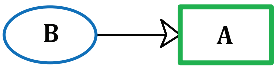
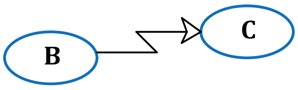
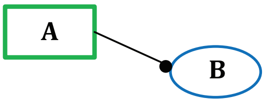
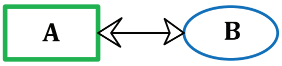
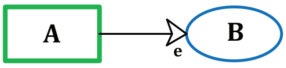
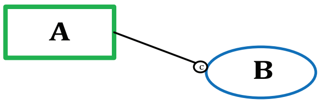
Symbol	Name	Description
	Object	An object is an item that exists or can exist once constructed, physically or informatically. Associations among objects shall constitute the object structure of the system being modelled, i.e. the static, structural aspect of the system.
	Process	A process is an item that expresses the behavioural, dynamic system aspect: how processes transform objects in the system and how the system functions to provide benefit. Processes complement objects by providing the dynamic, procedural aspect of the system.
	State	A state is a situation or position at which an object can exist for a period of time. Object B can be at states S1 or S2.
	Instrument Link	An instrument link is a procedural link that connects a process with an enabler of that process where the enabler is tools, data, etc. Object A enables Process B and Process B cannot happen if Object A does not exist.

Table 1 *(continued)*

Symbol	Name	Description
	Condition Link	A condition link is a procedural link that connects a process with an enabler of that process where the enabler is the state of an object. The process executes if and only if the object is in the certain state.
	Consumption Link	A consumption link is a link that connects a process with an object that is used by, or consumed, as a result of the occurrence of that process. Process B consumes Object A
	Result Link	A result link is a link that connects a process with an object that is constructed as a result of an occurrence of that process. Process B creates Object A.
	Invocation Link	Invocation links are procedural links between invoking processes and invoked ones. The invoking processes activate the invoked processes. Process B invokes Process C.
	Agent Link	Denotes that the object is a human operator. Activating the link triggers the process B.
	Effect Link	Process B changes the state of Object A; the details of the effect may be added at a lower level.
	Consumption Event Link	Existence or generation of object A will attempt to trigger process B. If B is triggered, it will consume A. Execution will proceed if the triggering failed.
	Condition Link	Existence of object A is a condition to the execution of B. If object A does not exist, then process B is skipped and regular system flow continues.

Note the Requirements Analysis and Design phases generally exchange information manually rather than making use of computer based or electronic interfaces. Therefore, this standard only addresses the Development, Execution, and Support phases as illustrated in Figure 1. Figure 1 indicates out of scope processes and objects with a dashed line and in scope processes and objects with a solid line.

Different tools are typically used for each phase due to the different functional requirements of the separate phases. Therefore, this document will describe the Development, Execution, and Support lifecycle phases as separate systems with integration interfaces between them.

ISO 15746-1 identified four functional modules within an APC-O system as Soft Sensor, Advanced Process Control, Optimization, and Performance Assessment. The workflow for any phase is similar for these four functional modules, but not identical. As a result, some commercial APC-O products include completely separate software tools for developing, executing, and supporting the four functional modules while others are well integrated internally. The integration interfaces defined in this standard will facilitate integration between different commercial APC-O products and between APC-O and non-APC-O systems. It does not intentionally favour any particular level of internal integration.

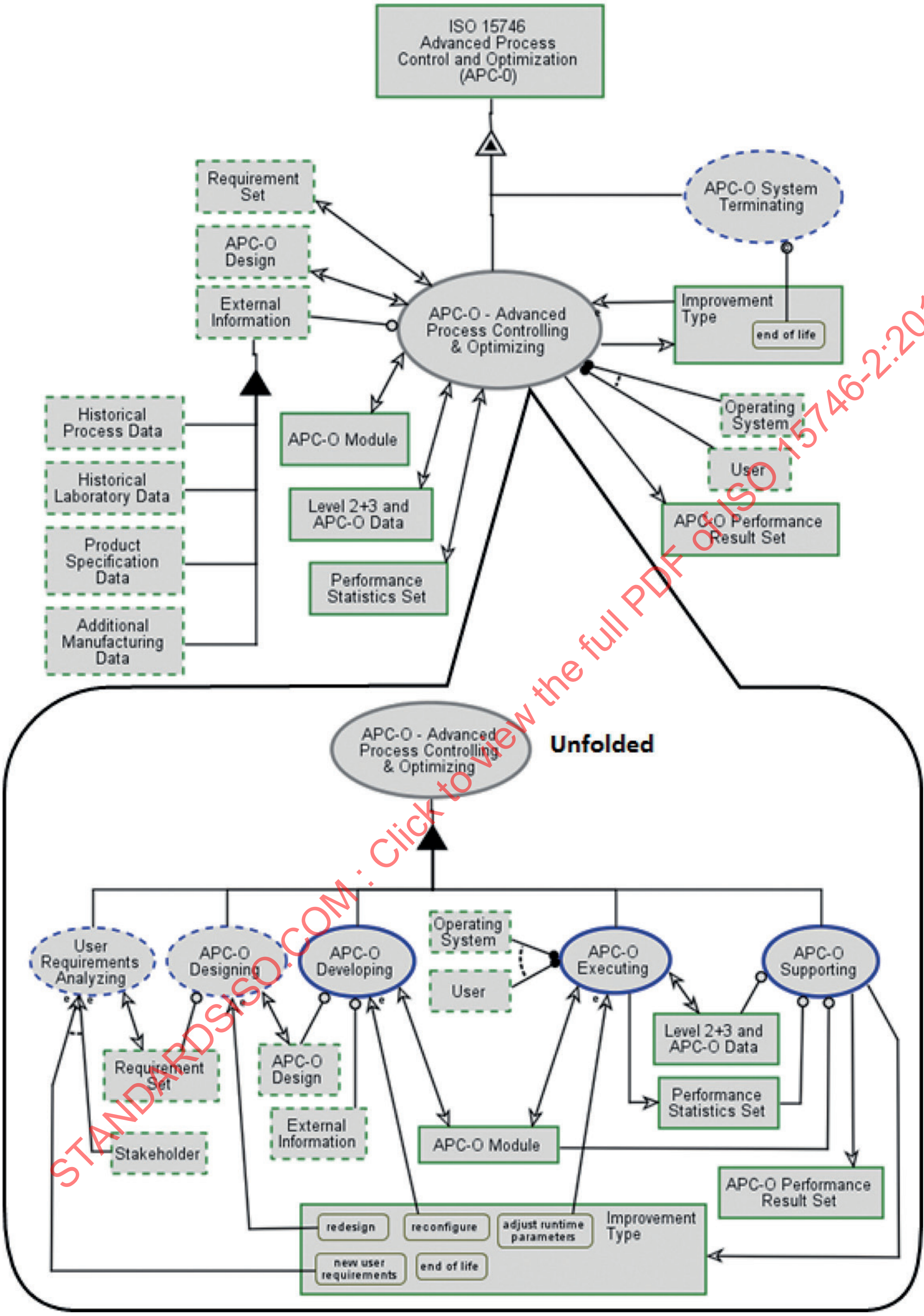


Figure 1 — APC-O Lifecycle Workflow

5.2 Development phase

The Development phase consists of the workflow illustrated in Figure 2. APC-O modules are comprised of one or more components performing specific functions, or jobs. These individual components may be constructed individually then assembled to create the complete APC-O module.

External interfaces used by the Development phase include non-APC-O information models. These could be global resource locators for any level or manually transferred information describing the various systems the APC-O application is expected to interface with in the Execution and/or Support phases.

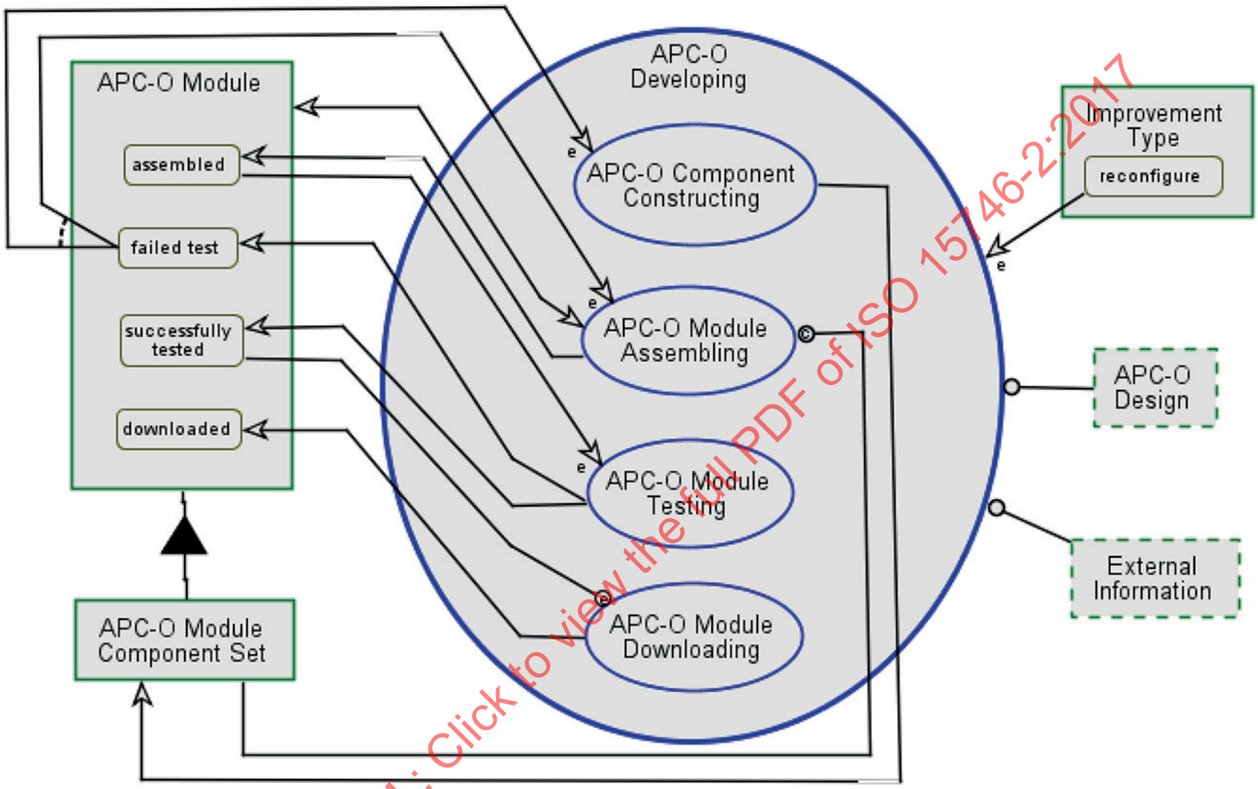


Figure 2 — Workflow of the Development Phase

The Development phase also interacts with historical data from various levels, for example:

— Product specifications

— Historical process data

— Historical laboratory results

— Manufacturing costs

The Development phase shall interact with APC-O systems in the Execution and Support phases by providing APC-O module definitions. These could be new definitions or modifications of existing definitions. Activities in the Development phase may be initiated by completion of the Design phase or by the Support phase signalling improvement of an APC-O system is needed.

5.3 Execution phase

5.3.1 Execution workflow

The Execution phase consists of the workflow illustrated in Figure 3.

Generic activities are defined rather than detailing each functional module separately. While there are differences between the modules as to how these activities are performed, those differences are not relevant to this standard. Furthermore, the details of the Execution phase workflow will differ from one commercial APC-O package to another but those differences are not relevant to this standard. For simplicity, a simple, single-threaded system is used to illustrate the Execution phase workflow. Some commercial packages are designed this way but there are also commercial packages that use fully asynchronous processing and even highly distributed processing.

Figure 3 illustrates APC-O components executing in a continuous loop controlled by a "schedule tasks" process that is only activated if the runtime state of the APC-O module is "running". As shown in the Development phase, an APC-O module comprises one or more component. In the Execution phase, APC-O components may be updated individually as shown in Figure 3 or the entire module could be updated at one time.

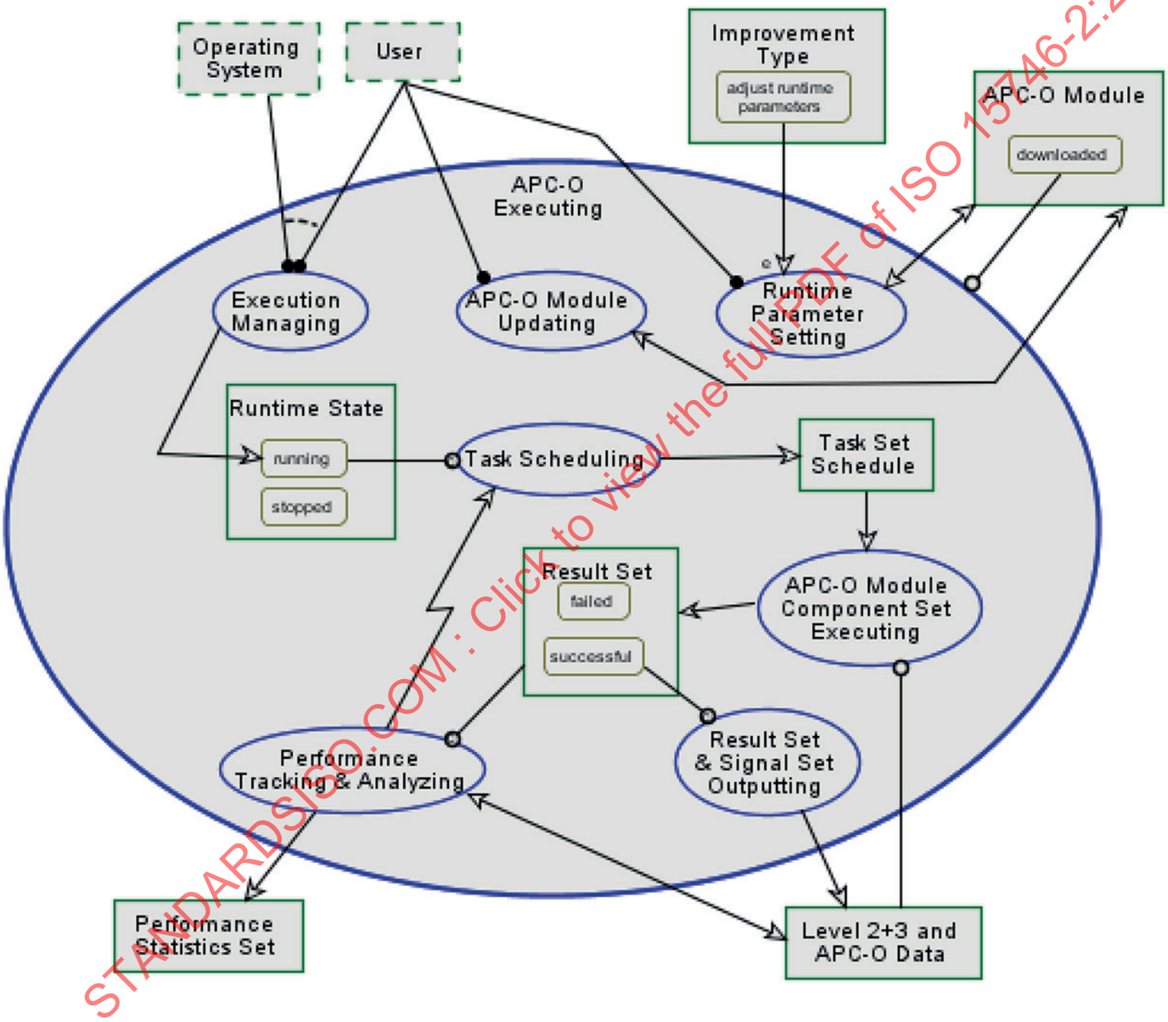


Figure 3 — Workflow of the Execution Phase

The following non-APC-O interfaces to the Execution phase are identified here and described in detail in Clause 6:

a) Level 2 Information Exchange for data such as:

1) PID Loop parameters

2) Final control elements settings such as valve positions and drive speeds

3) Process measurements from instruments such as flow and temperature sensors

4) General control system tag values

5) Alarm and event signals

b) Level 3 Information Exchange for data such as:

1) Product specifications

2) Recipe based process settings

3) Laboratory results

4) Manufacturing costs

The following interfaces between APC-O Components are also identified here and described in detail in Clause 6:

a) APC-O Information Exchange, for example:

1) APC-O data, alarms, and events

2) APC-O performance statistics

3) Execution requests and results

4) APC-O Module Definitions

5.3.2 Distributed APC-O system

Figure 4 illustrates how distributed APC-O components interact. In the example system, six APC-O modules interact to perform several interrelated functions. The six modules are identified as:

— Module 1 – a soft sensor providing one or more data elements for Module 3

— Module 2 – a second soft sensor, different from Module 1, providing one or more data elements for Module 3

— Module 3 – an APC module performing advanced control functions on a manufacturing process wholly within a level 2 environment

— Module 4 – an APC module performing advanced control functions on a manufacturing process in a way that affects the manufacturing process itself in level 2 and also one or more MOM functions in level 3

— Module 5 – an optimization module performing process optimization functions wholly within a level 3 environment

— Module 6 – a performance assessment module analysing, tracking, and reporting performance of modules 1 - 5

Various interactions between modules are illustrated in Figure 4. Modules 1 and 2 simply provide data for Module 3 in a similar manner as physical instruments installed on the manufacturing process. Module 3 also consumes one or more outputs from Module 4. These outputs could be setpoints written from manipulated variables in Module 4 and used as targets for controlled variables in Module 3. Likewise, Module 4 consumes one or more outputs from Module 5. Since Module 5 is an optimization module, a good example would be targets for controlled variables in Module 4 where these targets are determined by Module 5 to be the optimum targets based on business objectives as gathered from various MOM systems at level 3.

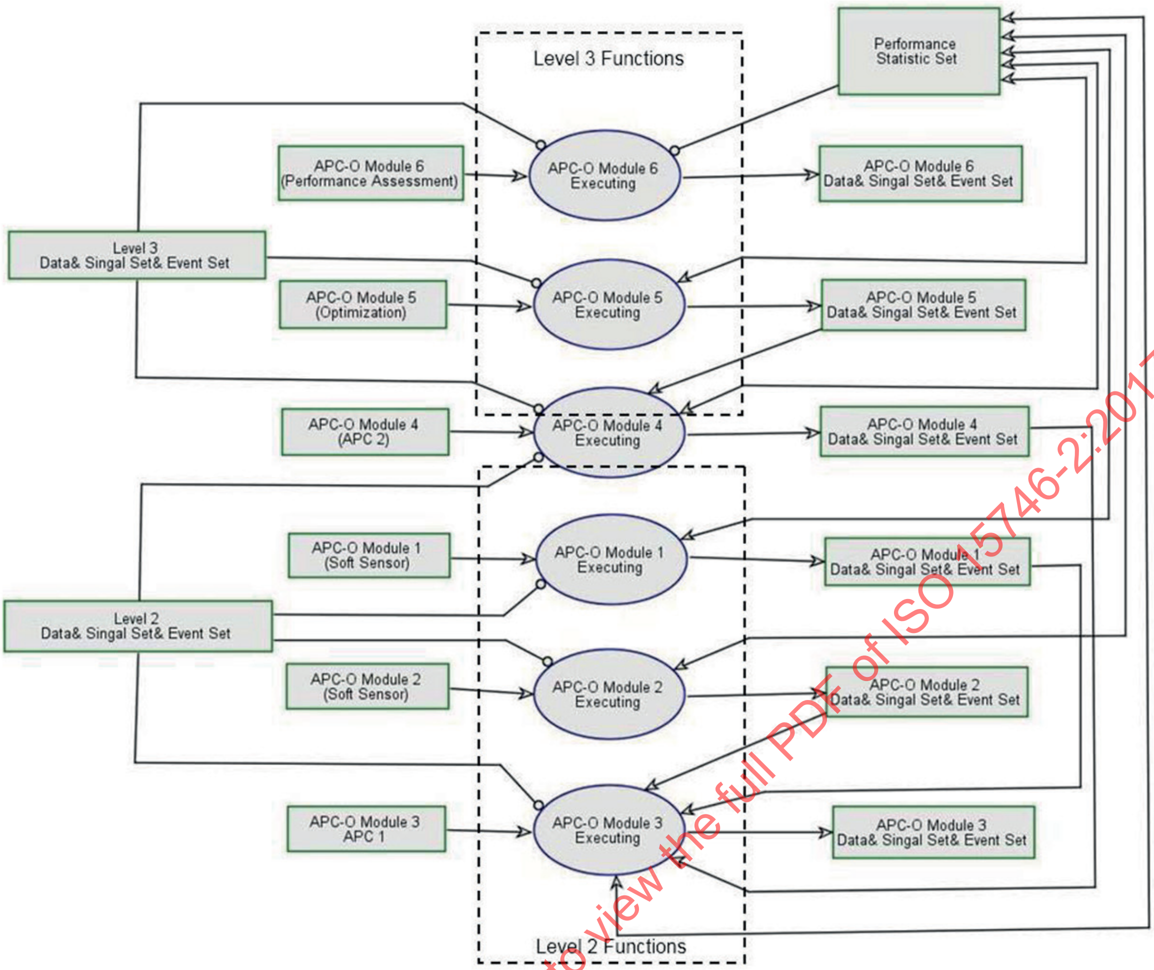


Figure 4 — Example of Distributed APC-O Interaction

5.4 Support phase

The Support phase consists of the workflow illustrated in Figure 5. The Support phase workflow shall make use of interfaces identified in preceding phases.

The Support phase is often a manual activity to enhance and maintain the performance of an APC-O system. In a support phase that is executed manually, decisions by engineers are commonly based on information and analysis from a performance assessment module. The other three modules shall provide the performance assessment module with performance statistics gathered through internal tracking and analysis functions. A support phase may be fully automated using a performance assessment module that is also fully automated.

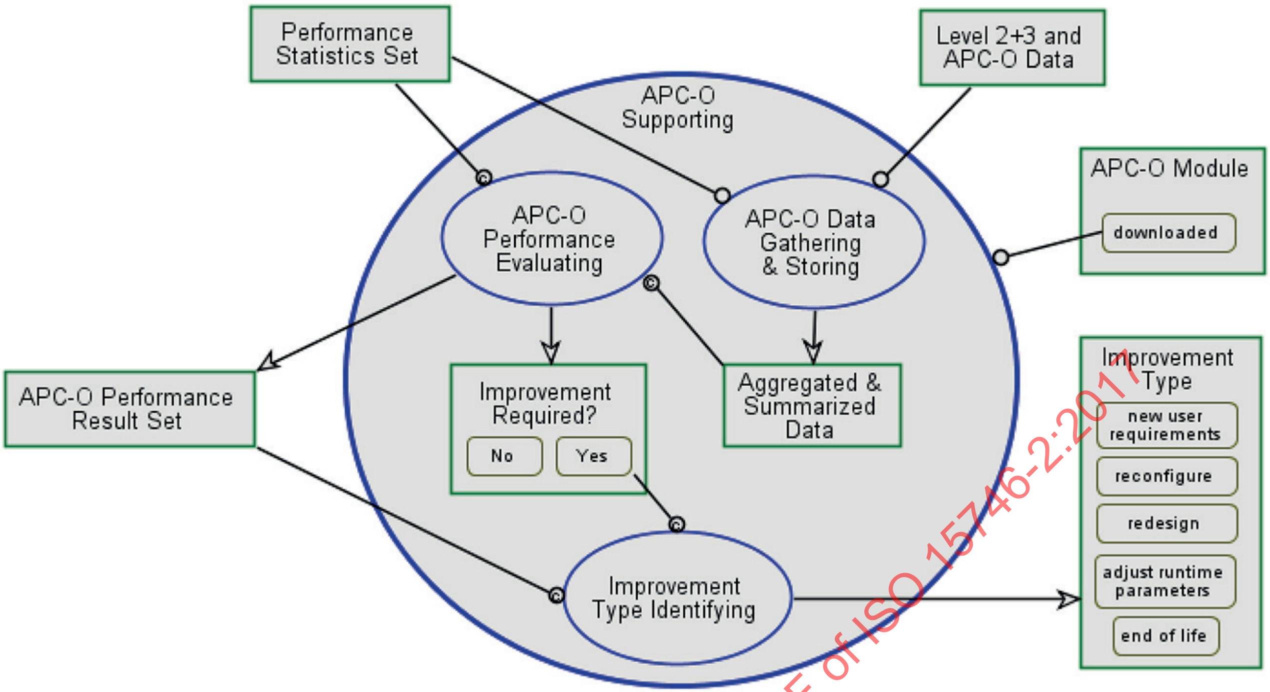


Figure 5 — Workflow of the Support Phase

6 Information exchange

6.1 Overview of information exchange services

The APC-O lifecycle workflows define the following non-APC-O interfaces that APC-O systems shall support:

a) Level 2

1. Global resource locators to determine the availability of and capabilities of non-APC-O resources. Examples include:

   i) PID Loops

   ii) Final control elements such as valves and drives

   iii) Process instruments

   iv) Alarm and event signals

2. Information exchange for non-APC-O data. Examples include:

   i) PID Loop parameters

   ii) Final control elements settings such as valve positions and drive speeds

   iii) Process measurements from instruments such as flow and temperature sensors

   iv) General control system tag values

   v) Alarm and event signals

b) Level 3

   1) Global resource locators to determine the availability of and capabilities of non-APC-O resources. Examples include:

      i) Product specifications

      ii) Recipes

      iii) Historical Process data

      iv) Laboratory results

      v) Production orders

   2) Information exchange for non-APC-O data. Examples include:

      i) Product specifications

      ii) Recipe settings

      iii) Historical process data

      iv) Laboratory results

      v) Manufacturing costs

In addition, the following internal interfaces between APC-O Modules are defined in sublevel workflows to support integration of commercial software products to create a heterogeneous APC-O system. This reduces the industry's dependency on every commercial APC-O product supporting every possible technology. APC-O systems shall support these interfaces.

a) APC-O Modules definitions

b) APC-O data

c) APC-O alarms and events

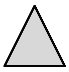
6.2 Information model

6.2.1 General

All interfaces are based on an overall information model for APC-O systems shown in Figure 6. An APC-O system comprises one or more APC-O Modules. An APC-O Module shall be identified by Name and Type and may also have one or more vendor-specific attributes that are provided through a discovery service interface.

The following symbols are used in information models in this document:

Table 2 — Information model symbols

Symbol		Description
Object	object	An object is an item that exists or can exist once constructed, physically or informatically. Associations among objects shall constitute the object structure of the system being modelled, i.e. the static, structural aspect of the system.
	aggregation-participation relation link	A fundamental structural relation. Aggregation-Participation is a source item that aggregates one or more other participant items, the destination items, into a meaningful whole.
	Exhibition-characterization relation link	A fundamental structural relation. Exhibition-Characterization means that an item exhibits, or is characterized by, another item. The Exhibition-Characterization relation binds a source item, the exhibitor, with one or more destination items, which shall identify features that characterizes the exhibitor.
	Generalization-Specialization relation link	Generalization-Specialization relations extend the inheritance concept to both objects and processes. A specialization item has at least the same structural relations and procedural relations as the general item.
	Classification-Instantiation relation link	Classification-Instantiation relations connect classes to their instances.

APC-O modules SoftSensor, APC, and Optimization shall have a Definition Type that is an object type with subtypes to define the specific instantiation of the APC-O module. Examples of APCDefinition Type are MatrixMPC, ExpertSystem, and TransitionProcedure. Examples of SoftSensorDefinition Type are Equation and NeuralNetwork. Examples of OptimizationDefinition Type are SteadyStateOpt, DynamicOpt, and ExpertSystemOpt. An APC-O module shall provide its specific Definition Type structure through a discovery service interface.

A PerformanceAssessment module does not have a Definition Type but shall have KPISets, which are sets of KPIs used to evaluate performance of the APC-O system. Specific KPIs and their definitions shall be provided by an APC-O system through a discovery service interface.

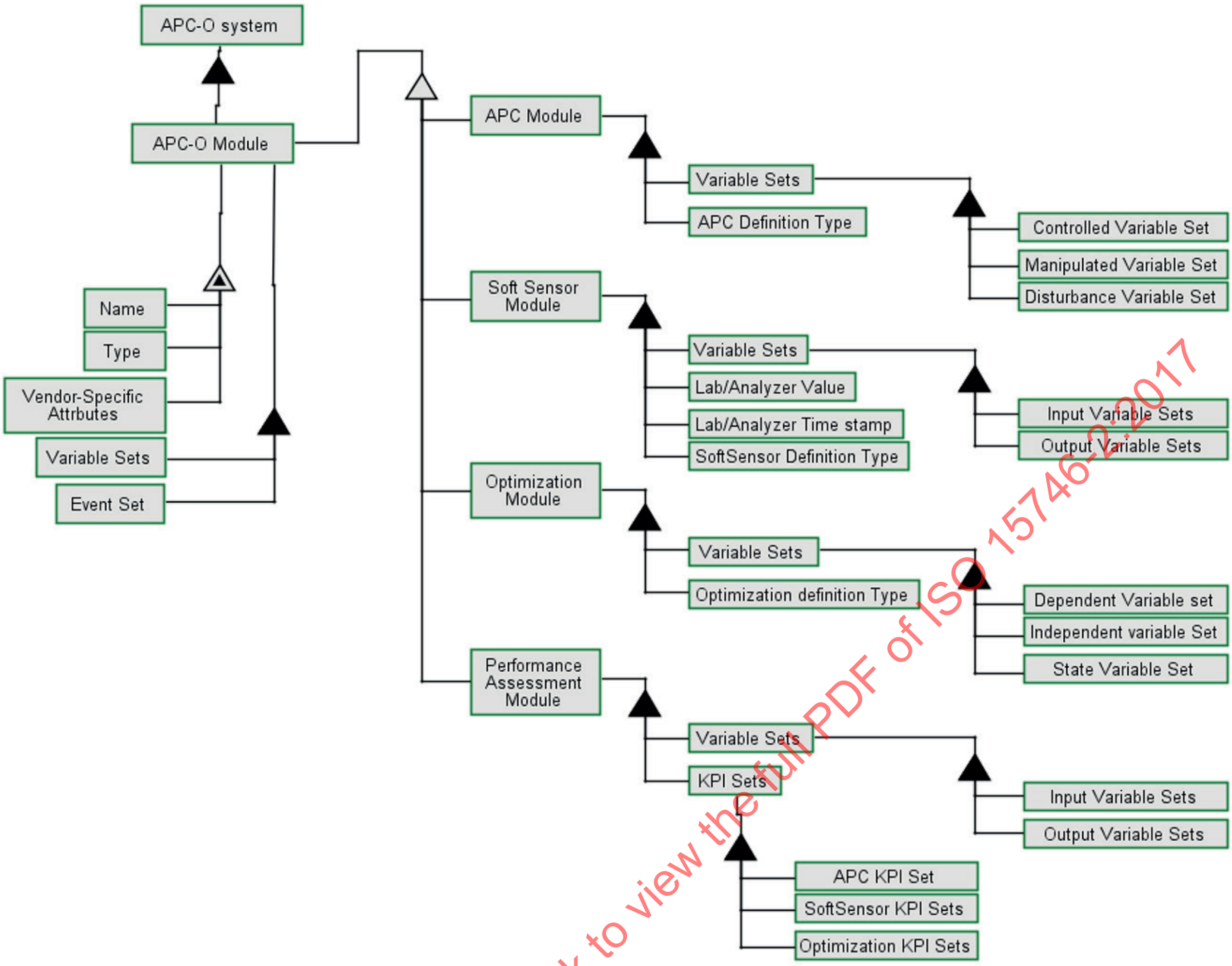


Figure 6 — APC-O Information Model

6.2.2 Properties of APC-O event type

An APC-O Module shall contain an Event Set that is a group of events the module will monitor or generate. The exact events depend not only on the type of APC-O Module but also on the type of manufacturing process the module is applied to. Events in an Event Set are objects of type APC-O Event Type. Specific events and their definitions shall be provided by an APC-O system through a discovery service interface. Communication Timeout, Process On Product, and Product Grade Change are examples of events of interest to an APC-O system. Events may trigger action, for example, a Product Grade Change event could trigger loading product specific targets and limits into an APC module.

APC-OEvent Type is an object type defining common attributes of events used in APC-O. Figure 7 shows the information model for APC-OEvent Type and the subtypes defined. Annex A provides an example event set.

Properties of APC-OEvent Type are:

— Source - A reference to the object that generated the event

— Time - The time when the event occurred

— Type - The type of event

— EventCategory - The defined grouping of events, such as Process Events or System Events

— Severity - The urgency of the event

— Vendor-Specific Attributes - Additional attributes defined by the specific APC-O package. These attributes will be provided by a discovery service interface.

A ConditionEvent is a subtype of APC-OEvent Type used to indicate changes in some condition such as the alarm state of a process measurement or the status of a system communication link. A ConditionEvent has additional properties as shown below:

— ConditionName - Name of the condition

— NewState - The new state of the condition

A TrackingEvent is a subtype of APC-OEvent Type used to indicate actions such as initiating a product grade transition or entering a new laboratory measurement. A TrackingEvent has no properties other than those defined for an APC-O Event Type.

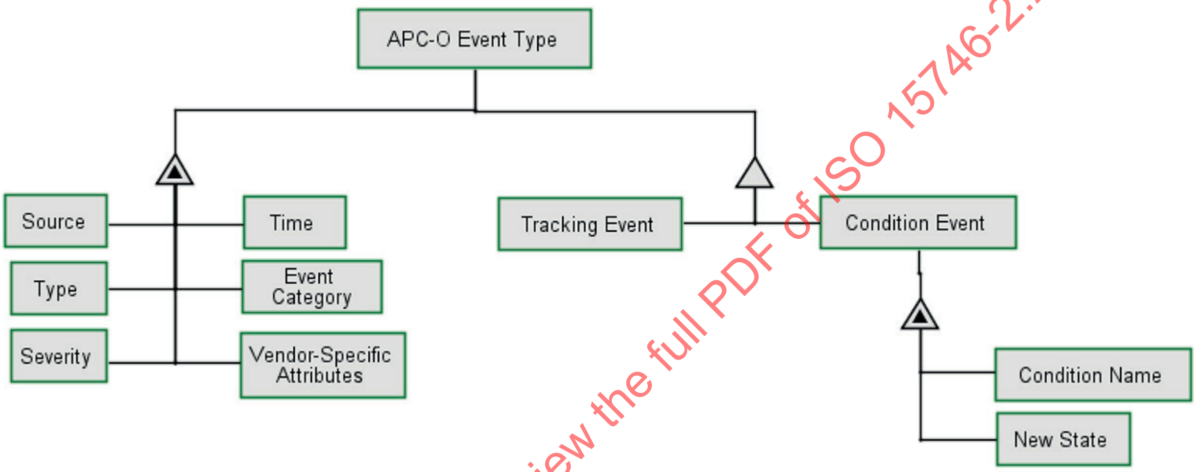


Figure 7 — APC-OEvent Type Information Model

6.2.3 Properties of APC-O variable type

All APC-O Module types shall have VariableSets of base type APC-OVariable Type. VariableSets are sets of variables used by the APC-O system. Defined sets are shown for each APC-OModule type. Variables in each of these sets are subtypes of APC-OVariable Type. Other sets may be defined by an APC-O system and exposed through a discovery service interface. InputVariableSet and OutputVariableSet are sets of variables of the base type APC-OVariable, and there are no Behaviour Specific attributes.

APC-OVariable Type is an object type defining common attributes of all variables used in APC-O. Figure 8 shows the information model for APC-OVariable Type and the subtypes defined.

Attributes of APC-OVariable Type are defined as follows:

— Variable Value Set consisting of:

  — ProcessValue – current process value read from Source

  — Quality – data quality of ProcessValue

  — TimeStamp – the date and time associated with ProcessValue

— Variable Attribute Set consisting of:

  — Name – descriptive name of the variable

  — VariableID – identifier to uniquely identify the variable, format will be vendor-specific

  — DataType – data type of ProcessValue, for example REAL, INT, BOOL, etc

- ModuleType – type of APC-O module the variable is associated with, for example APC or SoftSensor
- Source – a reference to an external source for the data. VariableSource subtypes include PIDLoop, FinalControlElement, CalculatedVariable, and SoftSensor. These subtypes are described in 6.3 and 6.4.
- EngineeringUnits - units of measure defining the value
- VariableIsActive - flag to indicate whether or not the variable is included in the component calculations
- VariableStatusEvent – an event to signal a change in the APC-O variable's status
- Vendor-Specific Attributes - Any additional attributes provided by the specific APC-O package
- Behaviourspecific – subtypes of APC-O Variable exhibiting additional behaviour

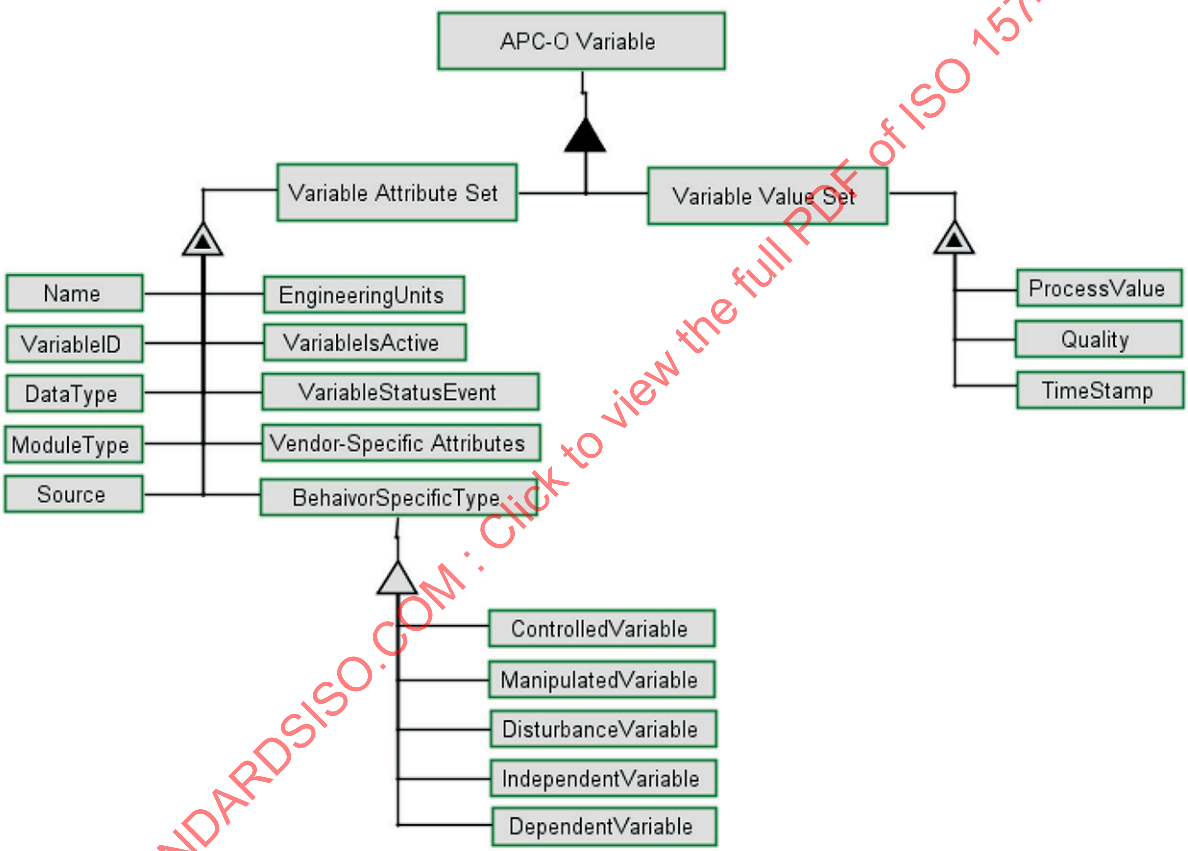


Figure 8 — APC-OVariable Type Information Model

6.2.4 Module definition types

Three object types are defined for the APC, SoftSensor, and Optimization modules.

An APCDefinition Type specifies the object for performing Advanced Process Control. The definition of the object depends on the type of APC used. Examples are MatrixMPC, ExpertSystem, and TransitionProcedure. Other types of APC exist and these will have unique object definitions. The specific APC-O package shall provide the definitions for APCDefinition Types it uses.

A SoftSensorDefinition Type specifies the object for implementing a Soft Sensor. The definition of the object depends on the type of Soft Sensor used. Examples are Equation and NeuralNetwork. Other types of Soft Sensors exist and these will have unique object definitions. The specific APC-O package shall provide the definitions for SoftSensorDefinition Types it uses.

An OptimizationDefinition Type specifies the object for implementing an Optimization. The definition of the object depends on the type of Optimization used. Examples are SteadyStateOpt, DynamicOpt, and ExpertSystemOpt. Other types of Optimization exist and these will have unique object definitions. The specific APC-O package shall provide the definitions for OptimizationDefinition Types it uses.

6.3 Non APC-O system interfaces

6.3.1 General

The following subclauses define the interfaces between an APC-O system and non-APC-O systems.

6.3.2 Level 2 data and events

6.3.2.1 General

While in the Execution phase, an APC-O system will need to read several data artefacts from the traditional control system, or Level 2 systems. Much of this data are simple tag/value mappings and existing standards built on the Open Platform Communications (OPC) specifications, such as OPC Data Access (IEC 62541), or OPC-DA, are sufficient. Improvement is needed in more complex interactions.

This interface builds upon existing communication standards such as OPC Unified Architecture (IEC 62541), or OPC-UA, by creating information models specific to an APC-O client. This allows APC-O application development software to make use of discovery functions to create data links to be used by the APC-O execution engine at run time. This also provides a standard method for the APC-O execution engine to access data and events at run time. Information exchange should support both synchronous and asynchronous methods.

All variables in an APC-O system shall exchange data and events with external systems through their Source, which is a generic mapping to an external data source of type VariableSource Type. A subtype of VariableSource Type could be as simple as a data connection to the process value of an instrument in the underlying control system. Two classes of VariableSource Type that apply to external systems are complex enough and common enough to be of interest in this document. These classes are PIDLoop and FinalControlElement.

6.3.2.2 PID loop

PIDLoop is a representation of a PID controller in a Level 2 control system. Figure 9 is an information model for a PIDLoop containing information of interest to APC-O. Many features and parameters of an actual PID controller have been omitted in this information model in order to focus on the requirements of APC-O. Properties of PIDLoop are defined as follows:

— Name – descriptive name of the variable

— ProcessValue - the value read from an instrument and controlled by the PID controller

— Setpoint - the target that the PID controller should control to

— RemoteSetpoint - the setpoint supplied by a higher level controller such as a master in a cascade arrangement or an APC-O system

— SetpointLimits - a convenient grouping of parameters that define limits on the setpoint. These include Minimum, Maximum, and Rate of Change

— Output - the output signal sent from the PID controller to a final control element such as a valve

— RemoteOutput - the output supplied by a higher level controller such as an APC-O system

— OutputLimits - a convenient grouping of parameters that define limits on the output of the PID controller. These include Minimum, Maximum, and Rate of Change

— Vendor-Specific Attributes – Any additional attributes provided by the specific APC-O package

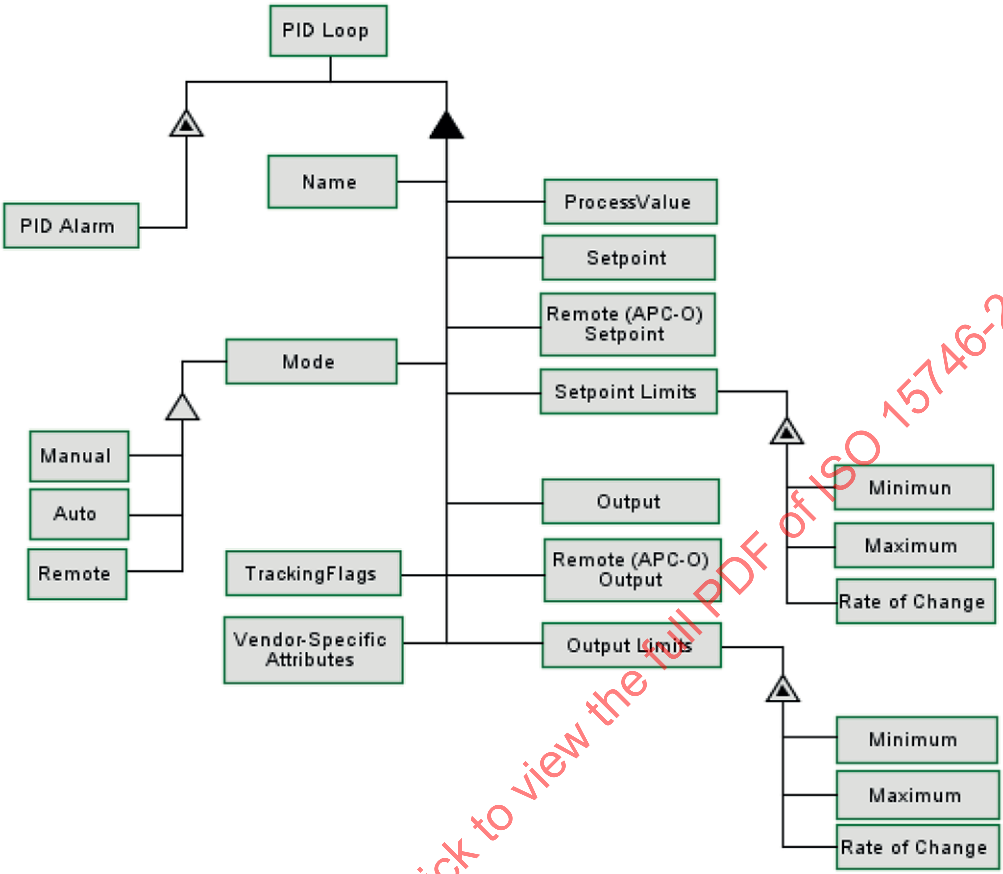


Figure 9 — PIDLoop Information Model

Mode and Tracking affect the behaviour of the PID Loop. Mode can be MANUAL, AUTO, or REMOTE. When in MANUAL, the loop does not calculate an output, operators set it manually. When in AUTO, the loop calculates the output to provide control. When in REMOTE, the loop behaves as if it is in AUTO except the setpoint is provided by a higher level control. This setpoint is written to the RemoteSetpoint parameter and the loop is responsible for copying the value to the actual setpoint. A common feature of PID Loops is to track the setpoint to the process value when the loop is in MANUAL so that the process is not disturbed when the loop is placed in AUTO since the process value will be at the setpoint. A similar necessary feature to support bumpless transfer in and out of REMOTE mode is to track the remote setpoint to the setpoint when not in REMOTE. TrackingFlags are attributes used to manage these features and their exact definitions are determined by the APC-O system vendor.

PIDAlarm is defined as an object type consistent with standards such as OPC-UA to allow the APC-O system to register for events such as alarms and failure events.

Proper interaction with the underlying PID controller at run time is critical to the performance of most APC-O applications. This interaction is illustrated in the SAMA diagram in Figure 10. SAMA diagrams are based on symbols and diagramming conventions developed by the Scientific Apparatus Makers Association (SAMA) for describing and documenting control strategies. Table 3 shows SAMA symbols used in this document.

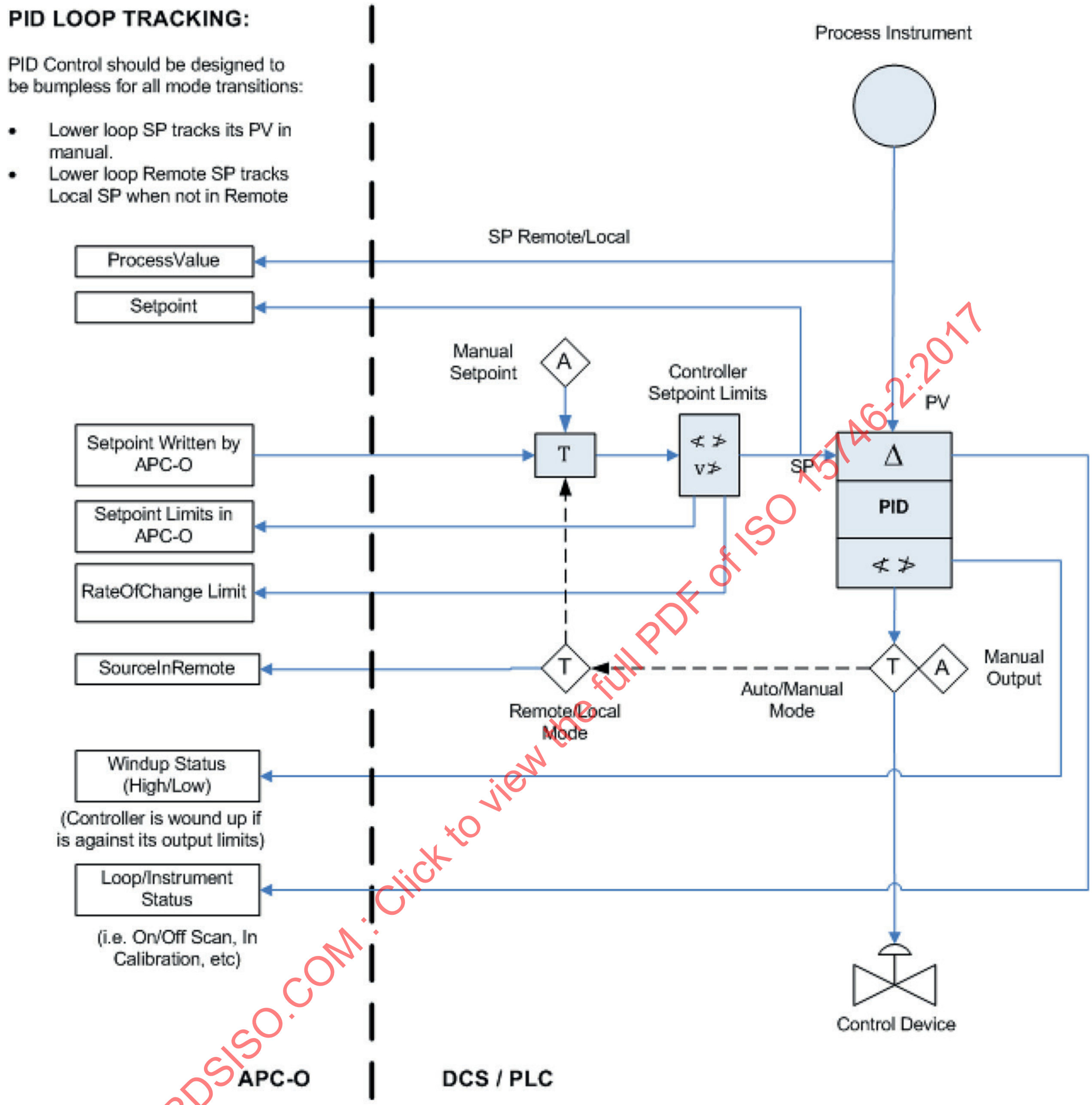
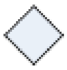


Figure 10 — APC-O interaction with traditional PID loop

Table 3 — SAMA symbols

Symbol	Description
Δ	Difference
PID	Proportional-Integral-Derivative algorithm for control
≮	Low limiting
≯	High limiting
V≯	Velocity limiting (rate of change)
	Automatic signal processing
	Manual signal processing
A	Adjustable signal
T	Transfer signal
	Continuous signal
	On-off signal

6.3.2.3 Final control element

It is common for some APC-O applications to manipulate final control elements directly, rather than through a PID controller. FinalControlElement is a representation of a final control element such as a valve, damper, or variable speed drive in a Level 2 control system. Figure 11 is an information model for a FinalControlElement containing information of interest to APC-O. Many features and parameters of an actual final control element have been omitted in this information model in order to focus on the requirements of APC-O. Properties of FinalControlElement are defined as follows:

— Name – descriptive name of the variable

— Level 2 Output - the setting sent by the Level 2 system to the final control element. This could be the output from a PID controller.

— RemoteOutput - the output supplied by a higher level controller such as an APC-O system.

— OutputLimits - a convenient grouping of parameters that define limits on the setting of the final control element. These include Minimum, Maximum, and Rate of Change.

— Vendor-Specific Attributes – Any additional attributes provided by the specific APC-O package

Mode and Tracking affect the behaviour of the FinalControlElement. Mode can be MANUAL or REMOTE. When in MANUAL, operators set the output manually. When in REMOTE, the Output is provided by a higher level control. This value is written to the RemoteOutput parameter and the block is responsible for copying the value to the actual output. A necessary feature to support bumpless transfer in and out of REMOTE mode is to track the remote output to the output when not in REMOTE. Vendor-specific tracking flags are used to manage these features.

FinalElementAlarm is defined as an object type consistent with standards such as OPC-UA to allow the APC-O system to register for events such as alarms and failure events.

Proper interaction at run time is critical to the performance of most APC-O applications. This interaction is illustrated in the SAMA diagram in Figure 12.

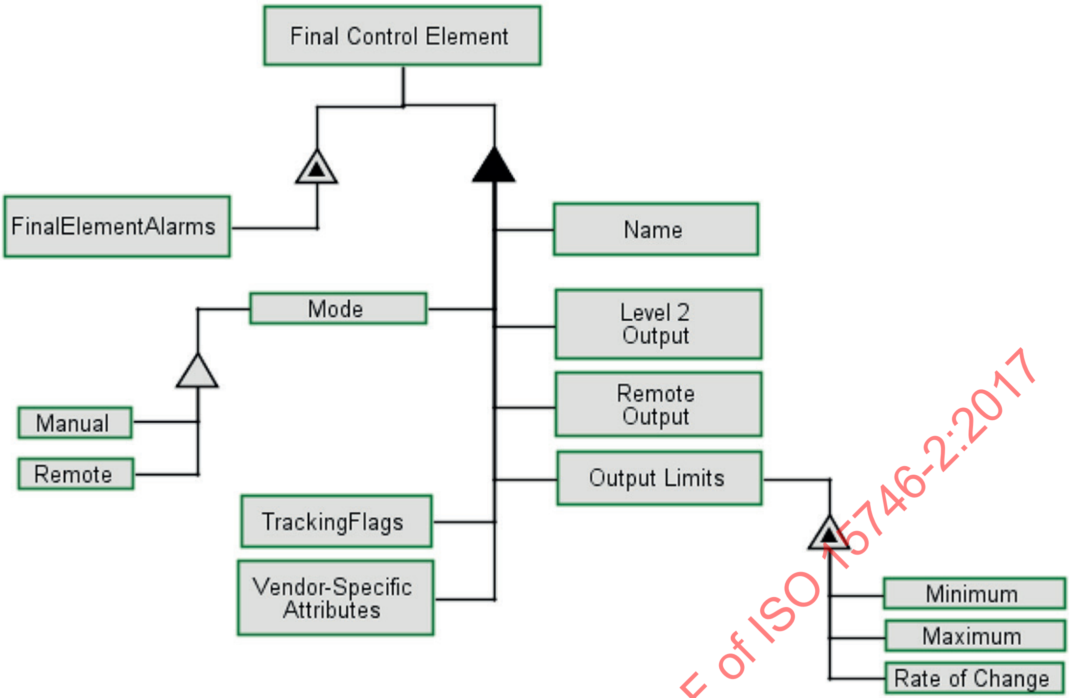


Figure 11 — FinalControlElement Information Model

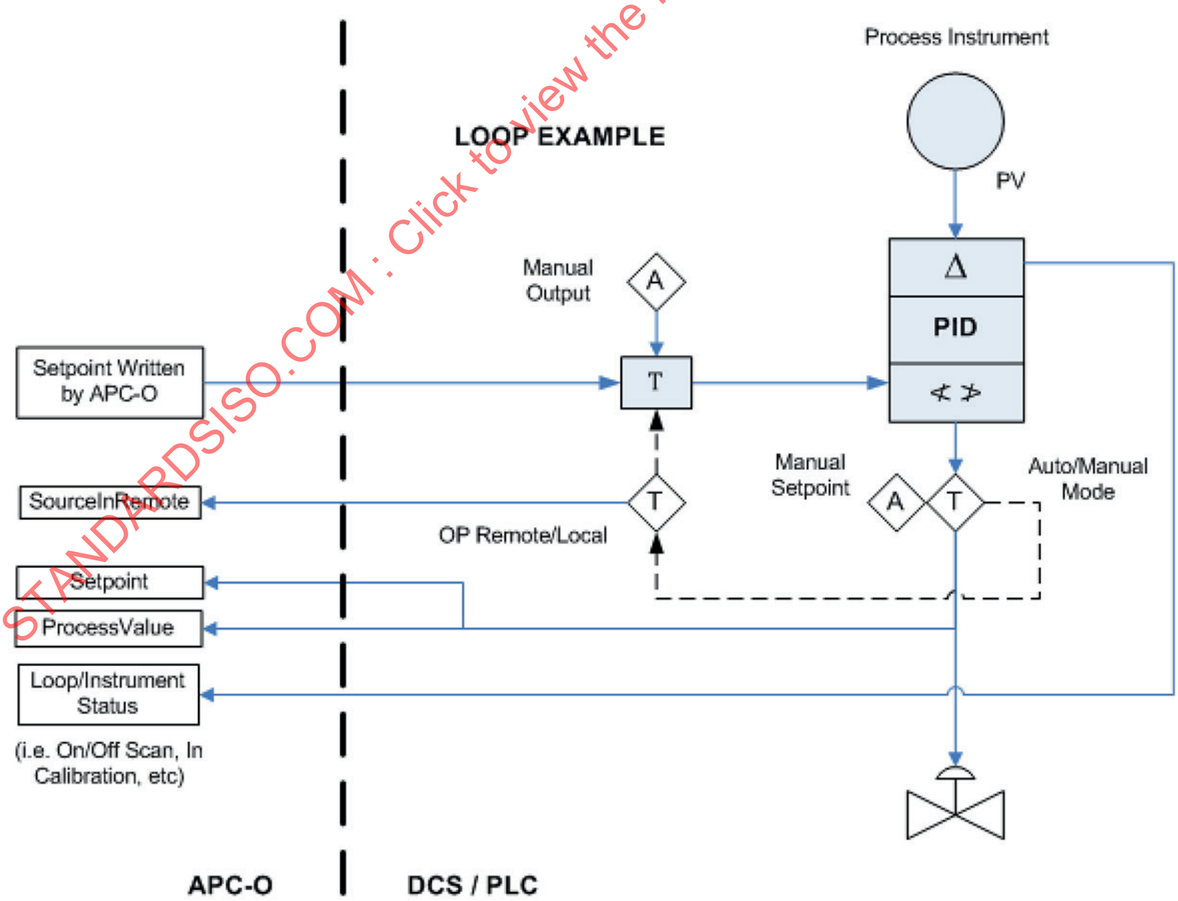


Figure 12 — APC-O interaction with final control element

6.3.3 Level 3 data and events

6.3.3.1 General

As with Level 2 data, this interface builds upon existing standards by creating information models of interest to an APC-O system. Interaction with Level 3 is based on the assumption that changes are initiated from the Level 3 system external to the APC-O system.

6.3.3.2 Laboratory systems

Results of laboratory analysis are used for feedback by soft sensors and may also be used as inputs to calculations or DVs in the APC-O model. The analysis is a set of data elements:

— Result – the result reported by the laboratory

— Timestamp – the date and time the sample was collected from the process

Additionally, there should be an indication that a new result is available. This could be implemented as a third data element that the APC-O system samples on a fixed frequency, or it could be implemented as an event that the APC-O system would register for.

Existing data interface standards such as OPC and SQL are sufficient for this interaction.

6.3.3.3 Recipe systems

Typically, recipe systems manage product specifications and process settings for sites that produce multiple products. APC-O data that potentially could originate in a recipe system include:

— MVConstraints

— MVObjectives

— CVObjectives

One method of interacting with a recipe system would be to simply map each parameter, such as the Target in CVObjectives, as individual data elements reading a corresponding data element that is the value for the currently active recipe.

An alternative method of interacting with a recipe system would be to specify that the recipe system write new values asynchronously when a recipe change is initiated.

A third alternative method of interacting with a recipe system would be to implement recipe changes as procedures in the APC-O system. A separate procedure could be used for every recipe with a copy of the settings contained in the procedure or a single procedure could be used where the procedure queries the recipe system for the correct settings.

Existing data interface standards such as OPC and SQL are sufficient for this interaction.

6.3.3.4 Historical data

This interface may be satisfied by the OPC Historical Data Access standard.

6.4 Intersystem and intrasystem interfaces

6.4.1 General

The following subclauses define the interfaces between APC-O Modules. These interfaces apply to information exchange between modules within a single APC-O system and between modules residing in different APC-O systems.

6.4.2 APC-O data and events

6.4.2.1 General

This interface builds upon existing communication standards such as OPC-UA and Extensible Markup Language (XML) by creating information models specific to an APC-O client. This allows APC-O application development software to make use of discovery functions to create data links to be used by the APC-O execution engine at run time. This also provides a standard method for the APC-O execution engine to access and provide data and events at run time.

The following interfaces are based on the high level information model shown in Figure 6. Subclause 6.2 describes the APC-OVariable Type, which includes subtypes of ControlledVariable, ManipulatedVariable, DisturbanceVariable, DependentVariable, and IndependentVariable.

6.4.2.2 Properties of controlled variable

ControlledVariable, or CV, is a variable that an APCModule attempts to keep at a target or between limits. The information model for a CV is shown in Figure 13. Only properties extending APC-OVariable Type are shown. Properties of ControlledVariable are defined as follows:

— Bias - adjustment to the model based on plant feedback

— Target - target to maintain the CV at

— MinimumSoftLimit - Minimum value to attempt to maintain the CV above

— MaximumSoftLimit - Maximum value to attempt to maintain the CV below

— Price - Value associated with the CV

— OptimizationScale - Scaling used to determine the relative importance of a deviation from target or limits

— PerformanceStatistics - One or more PerformanceStatistic variables

— Vendor-Specific Attributes - Any additional attributes provided by the specific APC-O package

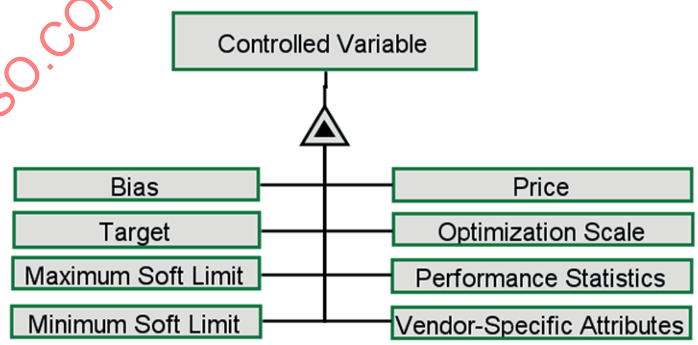


Figure 13 — ControlledVariable Information Model

6.4.2.3 Properties of manipulated variable

ManipulatedVariable, or MV, is a variable that an APCModule can move to satisfy CV objectives. The information model for an MV is shown in Figure 14. Only properties extending APC-OVariable Type are shown. Properties of ManipulatedVariable are defined as follows:

— Setpoint – setpoint of Source

— SourceInRemote – flag to indicate Source is in REMOTE mode

— MinimumHardLimit - minimum limit that the MV is not allowed to exceed

— MaximumHardLimit - maximum limit that the MV is not allowed to exceed

— RateOfChangeLimit – maximum amount Setpoint is allowed to change in a single move

— MoveSuppression - penalty for moving the MV

— OptimizationScale - scaling used to determine the relative penalty applied to a specific move size by MoveSuppression

— Price - Value associated with the MV

— PerformanceStatistics - One or more PerformanceStatistic variables

— Vendor-Specific Attributes - Any additional attributes provided by the specific APC-O package

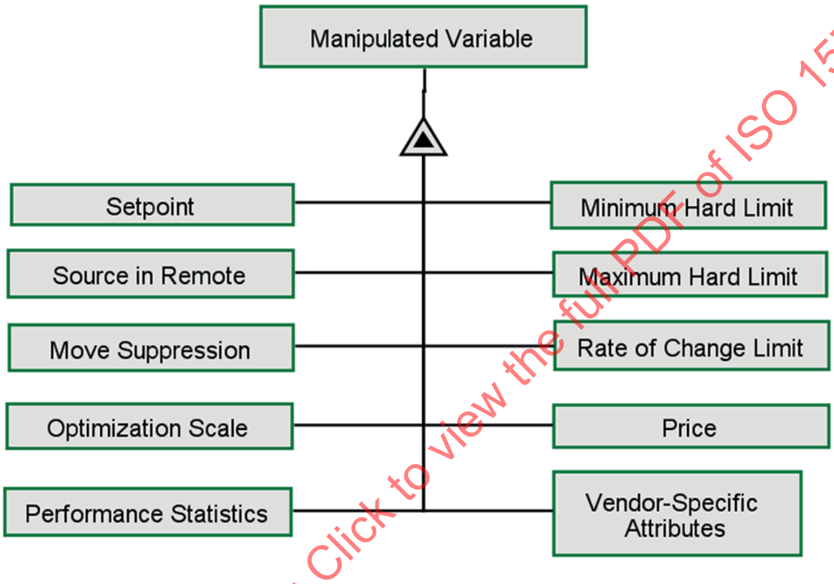


Figure 14 — ManipulatedVariable Information Model

6.4.2.4 Properties of disturbance variable

DisturbanceVariable, or DV, is a variable that affects one or more CVs or outputs of a model but an APCModule cannot move it. The information model for a DV is shown in Figure 15. Only properties extending APC-OVariable Type are shown. Properties of DisturbanceVariable are defined as follows:

— PerformanceStatistics - One or more PerformanceStatistic variables

— Vendor-Specific Attributes - Any additional attributes provided by the specific APC-O package

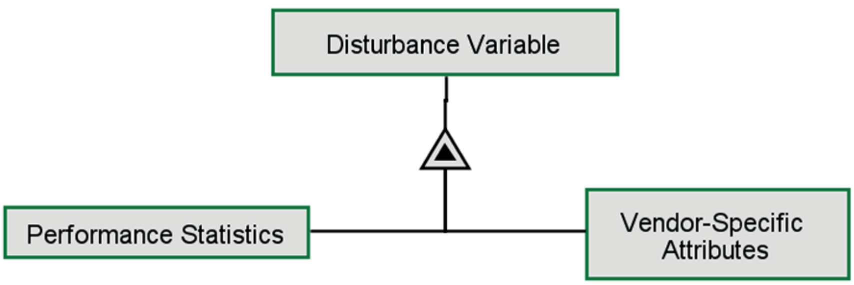


Figure 15 — DisturbanceVariable Information Model

6.4.2.5 Properties of dependent variable

DependentVariable is a variable that an OptimizationModule keeps at a target or between limits. The information model for a DependentVariable is shown in Figure 16. Only properties extending APC-OVariable Type are shown. Properties of DependentVariable are defined as follows:

— Bias - adjustment to the model based on plant feedback

— Target - target to maintain the variable at

— MinimumLimit - Minimum value to maintain the variable above

— MaximumLimit - Maximum value to maintain the variable below

— Price - Value associated with the variable

— OptimizationScale - Scaling used to determine the relative importance of a deviation from target or limits

— PerformanceStatistics - One or more PerformanceStatistic variables

— Vendor-Specific Attributes - Any additional attributes provided by the specific APC-O package

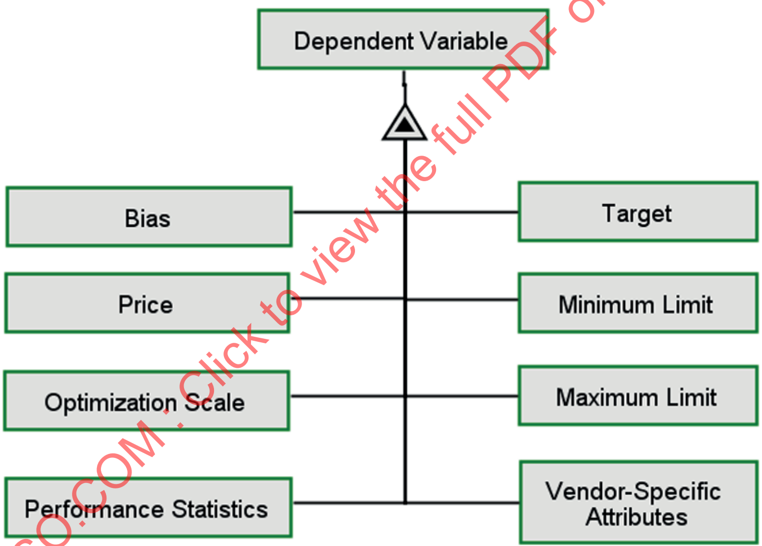


Figure 16 — DependentVariable Information Model

6.4.2.6 Properties of independent variable

IndependentVariable is a variable that an OptimizationModule can move to satisfy optimization objectives. The information model for an IndependentVariable is shown in Figure 17. Only properties extending APC-OVariableType are shown. Properties of IndependentVariable are defined as follows:

— Setpoint – setpoint of Source

— MinimumLimit - minimum limit that the variable is not allowed to exceed

— MaximumLimit - maximum limit that the variable is not allowed to exceed

— Price - Value associated with the variable

— PerformanceStatistics - One or more PerformanceStatistic variables

— Vendor-Specific Attributes - Any additional attributes provided by the specific APC-O package

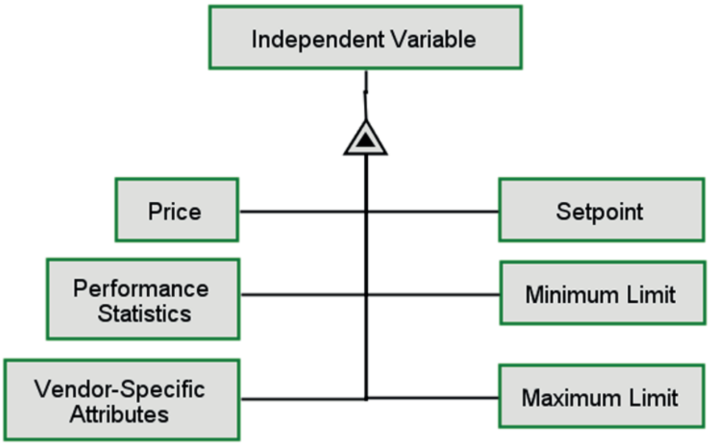


Figure 17 — IndependentVariable Information Model

6.4.2.7 Properties of performance statistic

PerformanceStatistc is a statistical measurement of actions taken by the APC-O module as they apply to the variable that the PerformanceStatistic is related to. Example PerformanceStatistics include mean error from setpoint for a ControlledVariable, standard deviation of a ControlledVariable, and percentage of time a ManipulatedVariable is limited by a constraint. A set of PerformanceStatistics may be provided to a PerformanceAssessment module. The information model for a PerformanceStatistic is shown in Figure 18. Properties are defined as follows:

— Value – the value of the statistic

— TimeBasis - time period over which the statistic is calculated, for example, hourly

— Vendor-Specific Attributes - Any additional attributes provided by the specific APC-O package

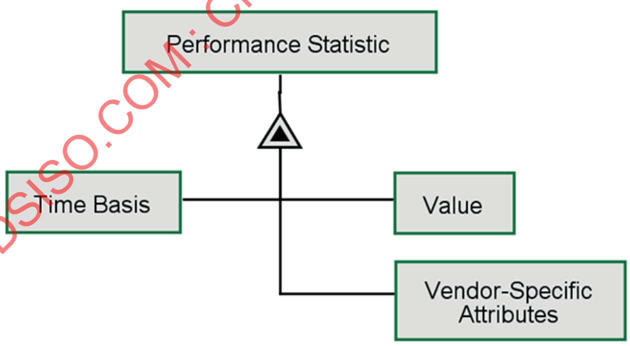


Figure 18 — PerformanceStatistic Information Model

6.4.3 APC-O module definitions

6.4.3.1 General

As described in 6.2.4, APC-OModule is an object type with subtypes APCModule, SoftSensorModule, OptimizationModule, and PerformanceAssessmentModule. A Definition Type object is included in the structures for APCModule, SoftSensorModule, and OptimizationModule. These objects define the specific instantiation of the APC-O Module.

6.4.3.2 APC definition type

There is no generic APCDefinition Type, every commercial package shall provide its own unique structure. Figure 19 contains three examples of APCDefinition Types. These examples do not represent specific technologies in their entirety but rather illustrate how an integration object might look. The attributes shown are those that might be of interest external to the specific instantiation of the APCModule.

A MatrixMPC object represents a type of APC known as “Model Predictive Control”, or MPC. An MPC controller represents the process with a matrix of models where each model describes the response of a CV from a change in an MV or DV. An MPC controller then uses an optimizer to determine a set of control actions that minimize an objective function over a finite control horizon. Properties of a MatrixMPC object are:

— PredictionHorizon – the amount of time into the future the internal models project output responses to changes in inputs

— ControlHorizon – the portion of PredictionHorizon used for control calculations

— TimeUnits – units used for time-based parameters, such as seconds or minutes

— MoveBlocks – specific points in the ControlHorizon where MV moves are calculated

— ModelPairs – models describing the relationship between an MV/CV or DV/CV pair. Each ModelPair has the following properties:

  — MV or DV – an object representing a manipulated variable or disturbance variable

  — CV – an object representing a controlled variable

  — IsIntegrator – indicates if the model is a self-regulating relationship or an integrating relationship.

  — ModelParameterSet – the set of parameters required to express the model. There are many different types of models but only two examples are shown. A Parametric Model Parameter Set includes the parameters needed to define a continuous-time transfer function. A few common parameter names are shown for illustration. An FIR Model Parameter Set includes the parameters needed to define a finite impulse response (FIR) model.

An ExpertSystem object represents a type of APC that uses logic or sets of rules to adjust the process. An ExpertSystem may contain process models or other mathematical representations of the process, but this is not required. Properties of an ExpertSystem object are:

— Rules – a set of if-then rules used to make decisions regarding control actions. Each rule consists of:

  — Rule – the rule itself in a vendor-specific format. This could be a simple if-then rule such as “If temperature X is greater than 500 then open the cooling water valve by 5%” or something more complex combining several if-then structures or even applying the rule to one or more objects.

  — Initiation Criteria – criteria that determines execution of the rule. Some systems may execute on a time frequency and other systems may execute when signalled that new data are available.

A TransitionProcedure object represents a procedural APC that executes specific actions on the process to change from one product grade to another. One method of implementing a TransitionProcedure is to use Sequential Function Charts as defined by IEC 60848. Properties of a TransitionProcedure object are:

— FromProduct – the product that the process has been making. The process will transition from this product.

— ToProduct – the product that the process will make after the transition is complete

— ProcedureStatus – the state of the executing procedure, for example “ready to start”, “waiting on step 3”, and “error on step 4”.

— Steps – a set of actions defining the TransitionProcedure

  — Actions – functions performed in a step, such as “turn on pump”

  — CompletionCriteria – signal or logic defining when a step is complete. For example, “FeedTankLevel > 20% AND PumpRunning = TRUE”.

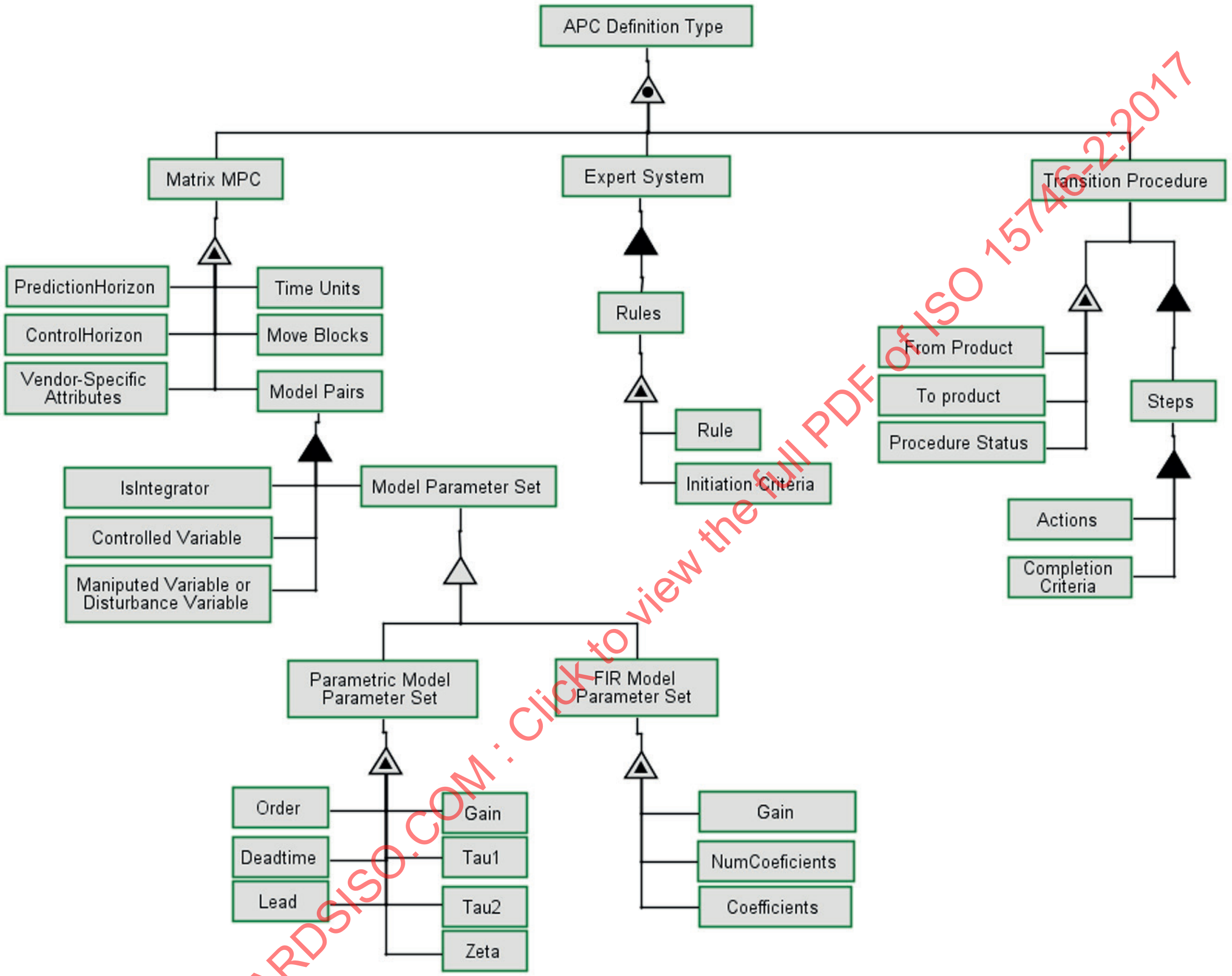


Figure 19 — Example APCDefinition Types

6.4.3.3 Soft sensor definition type

There is no generic SoftSensorDefinition Type, every commercial package shall provide its own unique structure. Figure 20 contains two examples of SoftSensorDefinition Types. These examples do not represent specific technologies in their entirety but rather illustrate how an integration object might look. The attributes shown are those that might be of interest external to the specific instantiation of the SoftSensorModule.

An Equation object represents a type of Soft Sensor based on one or more equations, typically based on physical relationships such as material and energy balances.

A NeuralNetwork object represents a type of Soft Sensor based on an empirical modelling technique inspired by biological neural networks making up the central nervous system of animals. Neural networks typically pass a set of input values through a set of “neurons” where sets of weights and activation functions are used to transform the inputs to the desired output.

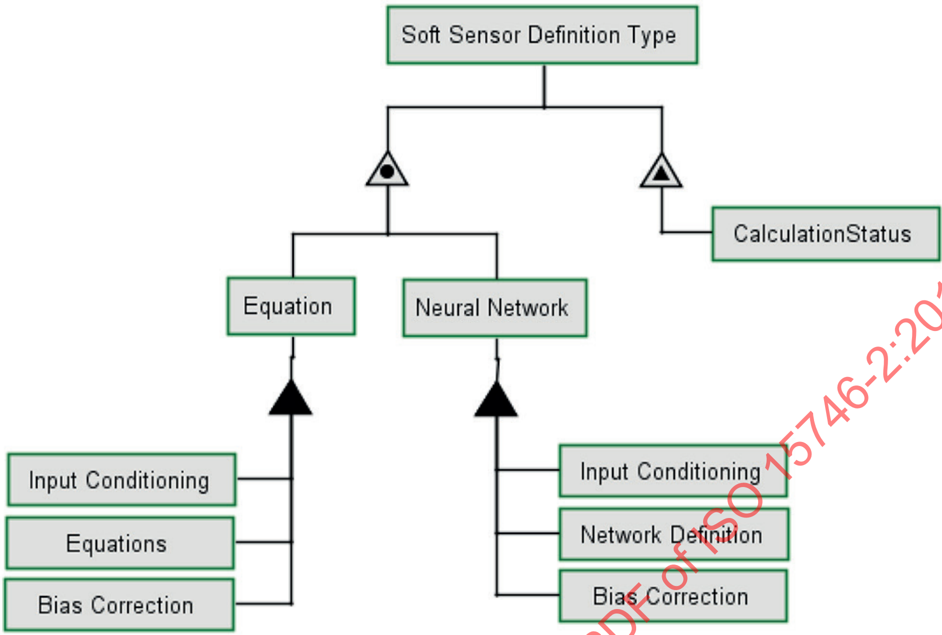


Figure 20 — Example SoftSensorDefinition Types

6.4.3.4 Optimization definition type

There is no generic OptimizationDefinition Type, every commercial package shall provide its own unique structure. Figure 21 contains two examples of OptimizationDefinition Types. These examples do not represent specific technologies in their entirety but rather illustrate how an integration object might look. The attributes shown are those that might be of interest external to the specific instantiation of the OptimizationModule.

A SteadyStateOpt object represents a type of Optimization where an objective function is minimized using steady-state process models. The path the process takes to achieve the optimum steady-state conditions is not considered.

A DynamicOpt object represents a type of Optimization where an objective function is minimized over a fixed horizon using dynamic process models. Both the path and the final steady-state conditions are considered in the solution.

An ExpertSystemOpt object represents a type of Optimization where optimum conditions are determined by a set of rules similar to if-then-else logic. Process models may be embedded in and used by the rules, but these models are not the fundamental basis for determining optimum conditions.

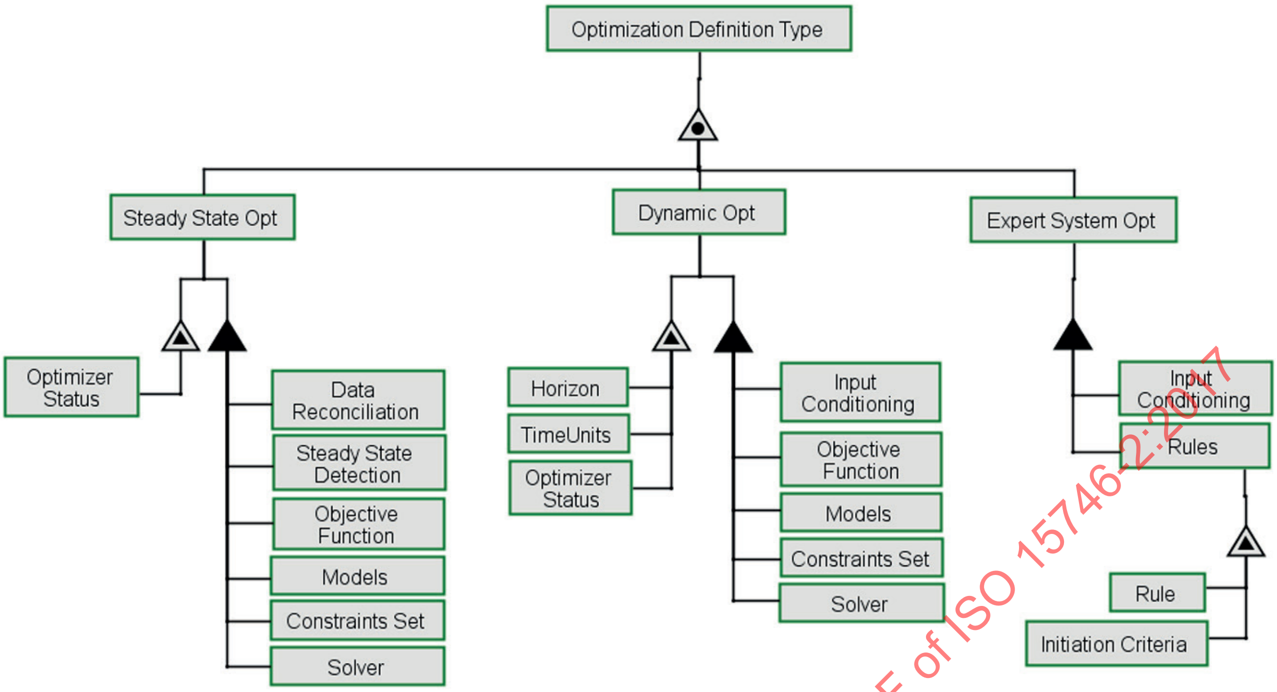


Figure 21 — Example OptimizationDefinition Types

6.4.3.5 Performance assessment module

The complete definition of a PerformanceAssessmentModule is given in 6.2 and shown in Figure 6. There are no additional behaviour specific attributes. A PerformanceAssessment Module shall be made up of sets of input and output variables and sets of KPIs. KPIs are defined based on ISO 22400-1. Examples include the following:

Table 4 — APC Utilization

KPI definition	
Content	
Name	APC Utilization
ID	
Description	APC Utilization is the percentage of actual production time where at least one MV is active in the APC application.
Scope	Work unit controlled by the APC application
Formula	(Actual time that at least one MV is active)/(Actual time during which the work unit is producing)
Unit of measure	%
Range	0 – 100 %
Trend	The higher the better
Context	
Timing	Periodically
Audience	Supervisor, management

Table 4 *(continued)*

Production methodology	Continuous
Effect model diagram	
Notes	APC utilization is often an indicator of the level of acceptance by operators. Applications that have a high level of acceptance will exhibit high utilization.

Table 5 — APC Utilization Ratio

KPI definition	
Content	
Name	APC Utilization Ratio
ID	
Description	APC Utilization Ratio is the percentage MVs active in the APC application.
Scope	Work unit controlled by the APC application
Formula	(Number of MVs active in the APC application)/(Total number of MVs in the APC application)
Unit of measure	%
Range	0 – 100 %
Trend	The higher the better
Context	
Timing	Periodically
Audience	Supervisor, management
Production methodology	Continuous
Effect model diagram	
Notes	APC utilization ratio is often an indicator of the effectiveness of the APC application. Applications that perform well will exhibit a high utilization ratio.

Table 6 — APC Utilization Effectiveness

KPI definition	
Content	
Name	APC Utilization Effectiveness
ID	
Description	APC Utilization Effectiveness combines APC Utilization and APC Utilization Ratio
Scope	Work unit controlled by the APC application
Formula	(APC Utilization)*(APC Utilization Ratio)
Unit of measure	%
Range	0 – 100 %
Trend	The higher the better
Context	
Timing	Periodically
Audience	Supervisor, management

Table 6 *(continued)*

Production methodology	Continuous
Effect model diagram	
Notes	APC utilization effectiveness is an indicator of both the effectiveness of the APC application and the level of acceptance by operators. Applications that perform well and have a high level of acceptance will exhibit high utilization and will manage the process effectively.

Table 7 — Time at MV Constraints

KPI definition	
Content	
Name	Time at MV Constraints
ID	
Description	Time at MV Constraints is the percentage of time a Manipulated Variable is at a constraint.
Scope	Work unit controlled by the APC application
Formula	(Actual time the MV is at one or more constraints)(Actual time that the MV is active)
Unit of measure	%
Range	0 – 100 %
Trend	Typically the lower the better, but there are cases where operating at a constraint is desirable
Context	
Timing	Periodically
Audience	Supervisor, management
Production methodology	Continuous
Effect model diagram	
Notes	APC and Optimization applications generally perform best when all degrees of freedom are available to achieve the goals of the application. However, there are cases where the MV will be at a minimum or maximum safe operating limit under optimum conditions and in those cases a high percentage of time at the constraint is desirable.

Table 8 — CV Error from Setpoint

KPI definition	
Content	
Name	CV Error from Setpoint
ID	
Description	CV Error from Setpoint is the mean deviation from the CV setpoint
Scope	Work unit controlled by the APC application
Formula	Mean of |(Actual CV ProcessValue) – (CV Setpoint)|
Unit of measure	Same unit of measure as the CV
Range	Will depend on the range of the CV
Trend	The lower the better
Context	
Timing	Periodically
Audience	Supervisor, management

Table 8 *(continued)*

Production methodology	Continuous
Effect model diagram	
Notes	The mean error from setpoint is a measure of the effectiveness of control of the CV.

6.4.4 Variable source type

Two specific VariableSource Types originate within an APC-OModule. These are the outputs of a SoftSensorModule and shall be referenceable as APC-OVariable Types included in the OutputVariableSet as shown in Figure 6.

Annex A
(informative)

APC-O Event Set Example

While in the Execution phase, an APC-O system may provide status signals in the form of events to non-APC-O systems. These events may be used by the underlying control system to determine if the APC-O system is in a failed state and then take actions to protect the process.

Figure A.1 is the same Execution phase workflow shown in Figure 3 except additional states have been added to the "result" object and the "output results, alarms, and events" has been split into two separate processes. These modifications are allowed by ISO 15746 and would be done at a software vendor's discretion. Transitions into each of these states trigger events that would be of interest to a level 2 or level 3 system.

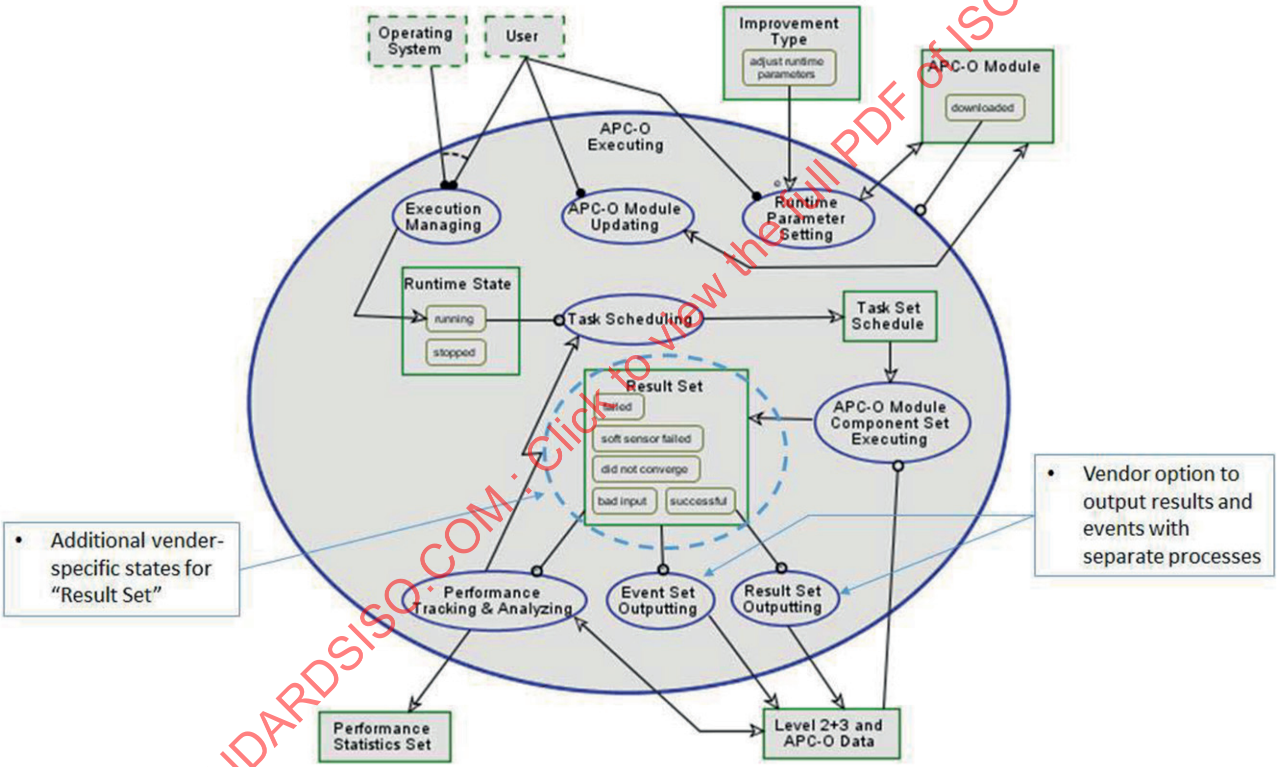


Figure A.1 — APC-O Execution phase with example event set

In this example, the following APC-O events are defined to support each state defined for the "result" object. These definitions follow the information model presented in Figure 7.

