INTERNATIONAL STANDARD

ISO/IEC 10164-21

First edition
1998-12-15

Information technology — Open Systems Interconnection — Systems Management: Command sequencer for Systems Management

Technologies de l'information — Interconnexion de systèmes ouverts (OSI) — Gestion-systèmes: Séquenceur de commande pour la gestion-systèmes

Contents

Page

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 10164-21 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 33, *Distributed application services*, in collaboration with ITU-T. The identical text is published as ITU-T Recommendation X.753.

ISO/IEC 10164 consists of the following parts, under the general title *Information technology — Open Systems Interconnection — Systems Management*:

— *Part 1: Object management function*

— *Part 2: State management function*

— *Part 3: Attributes for representing relationships*

— *Part 4: Alarm reporting function*

— *Part 5: Event report management function*

— *Part 6: Log control function*

— *Part 7: Security alarm reporting function*

— *Part 8: Security audit trail function*

— *Part 9: Objects and attributes for access control*

— *Part 10: Usage metering function for accounting purposes*

— *Part 11: Metric objects and attributes*

— *Part 12: Test management function*

— *Part 13: Summarization function*

— *Part 14: Confidence and diagnostic test categories*

— *Part 15: Scheduling function*

— *Part 16: Management knowledge management function*

— *Part 17: Change over function*

— *Part 18: Software management function*

— *Part 19: Management domain and management policy management functions*

— *Part 20: Time management function*

— *Part 21: Command sequencer for Systems Management*

— *Part 22: Response time monitoring function*

Annexes A, B, D and F to H form an integral part of this part of ISO/IEC 10164. Annexes C and E are for information only.

INTERNATIONAL STANDARD

ITU-T RECOMMENDATION

INFORMATION TECHNOLOGY – OPEN SYSTEMS INTERCONNECTION – SYSTEMS MANAGEMENT: COMMAND SEQUENCER FOR SYSTEMS MANAGEMENT

1 Scope

This Recommendation | International Standard defines a Systems Management Function which may be used by an application process in a centralized or decentralized management environment to interact for the purpose of systems management, as defined by CCITT Rec. X.700 | ISO/IEC 7498-4. This Recommendation | International Standard defines the Command Sequencer which consists of generic definitions, services and functional units. This function is positioned in the application layer of ITU-T Rec. X.200 | ISO/IEC 7498-1 and is defined according to the model provided by ISO 9545. The role of systems management functions is described by ITU Rec. X.701 | ISO/IEC 10040.

This Recommendation | International Standard:

- establishes user requirements for the Command Sequencer;
- establishes models that relate the services provided by the function to user requirements;
- defines the services provided by the function;
- specifies the protocol that is necessary in order to provide the services;
- defines the relationship between the services and SMI operations and notifications;
- defines relationships with other systems management functions;
- specifies conformance requirements;
- defines a scripting language for use in the command sequencer environment.

This Recommendation | International Standard does not:

- define the nature of any implementation intended to provide the Command Sequencer;
- specify the manner in which management is accomplished by the use of the Command Sequencer;
- define the nature of any instructions which result in the use of the Command Sequencer;
- specify the services necessary for the establishment, normal, abnormal release of management associations.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

2.1 Identical Recommendations | International Standards

- ITU-T Recommendation X.200 (1994) | ISO/IEC 7498-1:1994, *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model.*
- ITU-T Recommendation X.210 (1993) | ISO/IEC 10731:1994, *Information technology – Open Systems Interconnection – Basic Reference Model: Conventions for the definition of OSI services*.

- CCITT Recommendation X.701 (1992) | ISO/IEC 10040:1992, *Information technology – Open Systems Interconnection – Systems management overview*.
- ITU-T Recommendation X.710 (1997) | ISO/IEC 9595:1998, *Information technology – Open Systems Interconnection – Common management information service*.
- ITU-T Recommendation X.711 (1997) | ISO/IEC 9596-1:1998, *Information technology – Open Systems Interconnection – Common management information protocol: Specification*.
- CCITT Recommendation X.721 (1992) | ISO/IEC 10165-2:1992, *Information technology – Open Systems Interconnection – Structure of management information: Definition of management information*.
- CCITT Recommendation X.722 (1992) | ISO/IEC 10165-4:1992, *Information technology – Open Systems Interconnection – Structure of management information: Guidelines for the definition of managed objects*.
- ITU-T Recommendation X.724 (1996) | ISO/IEC 10165-6:1997, *Information technology – Open Systems Interconnection – Structure of management information: Requirements and guidelines for implementation conformance statement proformas associated with OSI management*.
- ITU-T Recommendation X.725 (1995) | ISO/IEC 10165-7:1996, *Information technology – Open Systems Interconnection – Structure of management information: General relationship model*.
- CCITT Recommendation X.730 (1992) | ISO/IEC 10164-1:1993, *Information technology – Open Systems Interconnection – Systems management: Object management function*.
- CCITT Recommendation X.731 (1992) | ISO/IEC 10164-2:1992, *Information technology – Open Systems Interconnection – Systems management: State management function*.
- CCITT Recommendation X.733 (1992) | ISO/IEC 10164-4:1992, *Information technology – Open Systems Interconnection – Systems management: Alarm reporting function*.
- CCITT Recommendation X.734 (1992) | ISO/IEC 10164-5:1993, *Information technology – Open Systems Interconnection – Systems management: Event report management function*.
- CCITT Recommendation X.735 (1992) | ISO/IEC 10164-6:1993, *Information technology – Open Systems Interconnection – Systems management: Log control function*.
- ITU-T Recommendation X.739 (1993) | ISO/IEC 10164-11:1994, *Information technology – Open Systems Interconnection – Systems management: Metric objects and attributes*.
- ITU-T Recommendation X.741 (1995) | ISO/IEC 10164-9:1995, *Information technology – Open Systems Interconnection – Systems management: Objects and attributes for access control*.
- ITU-T Recommendation X.746 (1995) | ISO/IEC 10164-15:1995, *Information technology – Open Systems Interconnection – Systems management: Scheduling function*.

2.2 Paired Recommendations | International Standards equivalent in technical content

- CCITT Recommendation X.209 (1988), *Specification of basic encoding rules for Abstract Syntax Notation One (ASN.1)*.

  ISO/IEC 8825:1990, *Information technology – Open Systems Interconnection – Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)*.
- ITU-T Recommendation X.291 (1995), *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications – Abstract test suite specification*.

  ISO/IEC 9646-2:1994, *Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 2: Abstract Test Suite specification*.
- ITU-T Recommendation X.296 (1995), *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications – Implementation conformance statement*.

  ISO/IEC 9646-7:1995, *Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 7: Implementation Conformance Statements.*.
- CCITT Recommendation X.700 (1992), *Management framework for Open Systems Interconnection (OSI) for CCITT Applications.*

  ISO/IEC 7498-4:1989, *Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 4: Management framework*.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

3.1 Basic Reference Model definitions

This Recommendation | International Standard makes use of the following terms defined in ITU-T Rec. X.200 | ISO/IEC 7498-1.

a) open system;

b) systems management.

3.2 Service convention definitions

This Recommendation | International Standard makes use of the following term defined in ITU-T Rec. X.210 | ISO/IEC 10731.

– primitive.

3.3 Management framework definitions

This Recommendation | International Standard makes use of the following terms defined in CCITT Rec. X.700 | ISO/IEC 7498-4.

a) management information;

b) managed object.

3.4 Systems management overview definitions

This Recommendation | International Standard makes use of the following terms defined in CCITT Rec. X.701 | ISO/IEC 10040.

a) agent role;

b) management support object;

c) managed object class;

d) manager role;

e) notification;

f) systems management functional unit;

g) system management operation.

3.5 Common management information service definitions

This Recommendation | International Standard makes use of the following terms defined in ITU-T Rec. X.710 | ISO/IEC 9595.

a) attribute;

b) common management information services.

3.6 Additional definitions

The following terms are defined in this Recommendation | International Standard.

3.6.1 command sequencer: A management support object representing a resource which functions in a manager role as a notification destination and as an initiator of operations determined by its launch scripts, with the ability to delegate management activities.

3.6.2 launch script: A managed object representing the instructions to be performed by a command sequencer.

3.6.3 thread: A managed object representing the execution of a launch script. The execution results or errors from launch script executions are returned by the thread.

3.6.4 suspendable thread: The suspendable thread is derived from the thread managed object class. These threads are spawned by asynchronous launch pads. They can be suspended by means of suspend action directed at them and resumed by means of a resume action directed at them.

3.6.5 launch pad: A management support object to which a trigger may be directed to initiate the execution of a launch script. A launch pad serves as an Initial Value Managed Object (IVMO) for a thread.

3.6.6 asynchronous launch pad: An asynchronous launch pad is derived from launch pad. It returns a trigger result notification without waiting for results of execution of the launch scripts. Execution results or errors from launch script executions are notified directly from the thread.

3.6.7 synchronous launch pad: A synchronous launch pad is derived from the launch pad. It returns trigger result notification or processing error alarm after it gets all the execution results and errors from threads after the threads complete their execution.

3.6.8 trigger activator: An initiator of script execution by causing a launch pad to spawn one or more threads. It directs a command to a launch pad, in the form of scheduler, operations, notifications or local action.

3.6.9 command: An instruction for a management activity that is performed in the agent system in accordance with contents of a launch script. A command is described with a scripting language. Currently, the system management scripting language is defined in Annex F.

4 Abbreviations

For the purposes of this Recommendation | International Standard, the following abbreviations apply:

ASN.1	Abstract Syntax Notation One
CMIS	Common Management Information Service
CS	Command Sequencer
IVMO	Initial Value Managed Object
OSI	Open Systems Interconnection
LP	Launch Pad
SMSL	Systems Management Scripting Language

5 Conventions

This Specification defines services for the command sequencer following the descriptive conventions defined in ITU-T Rec. X.210 | ISO/IEC 10731. In clause 9, the definition of each service includes a table that lists the service parameters. For a given service primitive, the presence of each parameter is described by one of the following values:

M	The parameter is mandatory.
(=)	The value of the parameter is equal to the value of the parameter in the column to the left.
U	The use of the parameter is a Service-user option.
–	The parameter is not present in the interaction described by the primitive concerned.
C	The parameter is conditional. The conditions are defined by the test which describes this parameter.
P	The parameter is subject to the constraints imposed by ITU-T Rec. X.710 | ISO/IEC 9595.

NOTE – The parameters which are marked "P" in service tables of this Specification are mapped directly onto the corresponding parameters of the CMIS service primitive, without changing the semantics or syntax of the parameters.

The font used for GDMO, ASN.1 and GRM in this Recommendation | International Standard is Courier. The BNF for SMSL in F.14.11 is in Courier New. In Annexes F and G, SMSL function parameters have been italicized.

6 Requirements

The requirements to be satisfied are:

- User requirements:
  - Allow the delegation of management activities.
  - Reduce the amount of communication that must occur between manager and agents.

- Allow delegated manager systems to operate on agent systems even when communications between a manager and the agent systems have been disrupted or are not possible.
- Provide flexible control of management activities.
- Provide a scripting language which can describe procedures to perform management operations.
- Allow delegated systems to execute CMIS operations in sequence.

- Operational requirements:
  - Pre-scheduled or delayed execution of a systems management operation.
  - Capabilities for modifying the request for pre-scheduled or delayed execution.
  - Capabilities for initiating, suspending, resuming and terminating systems management operations based on time management actions or the occurrence of events.
  - Capabilities for reporting and recording the outcome of pre-scheduled or delayed execution.
  - The ability to send notifications when state changes occur.

7 Model

7.1 Model description

The model describes how triggered, pre-scheduled or delayed execution of system management operations can be performed by the command sequencer. It describes the conceptual components, the relationship between these components, a description of the states and possible state transitions.

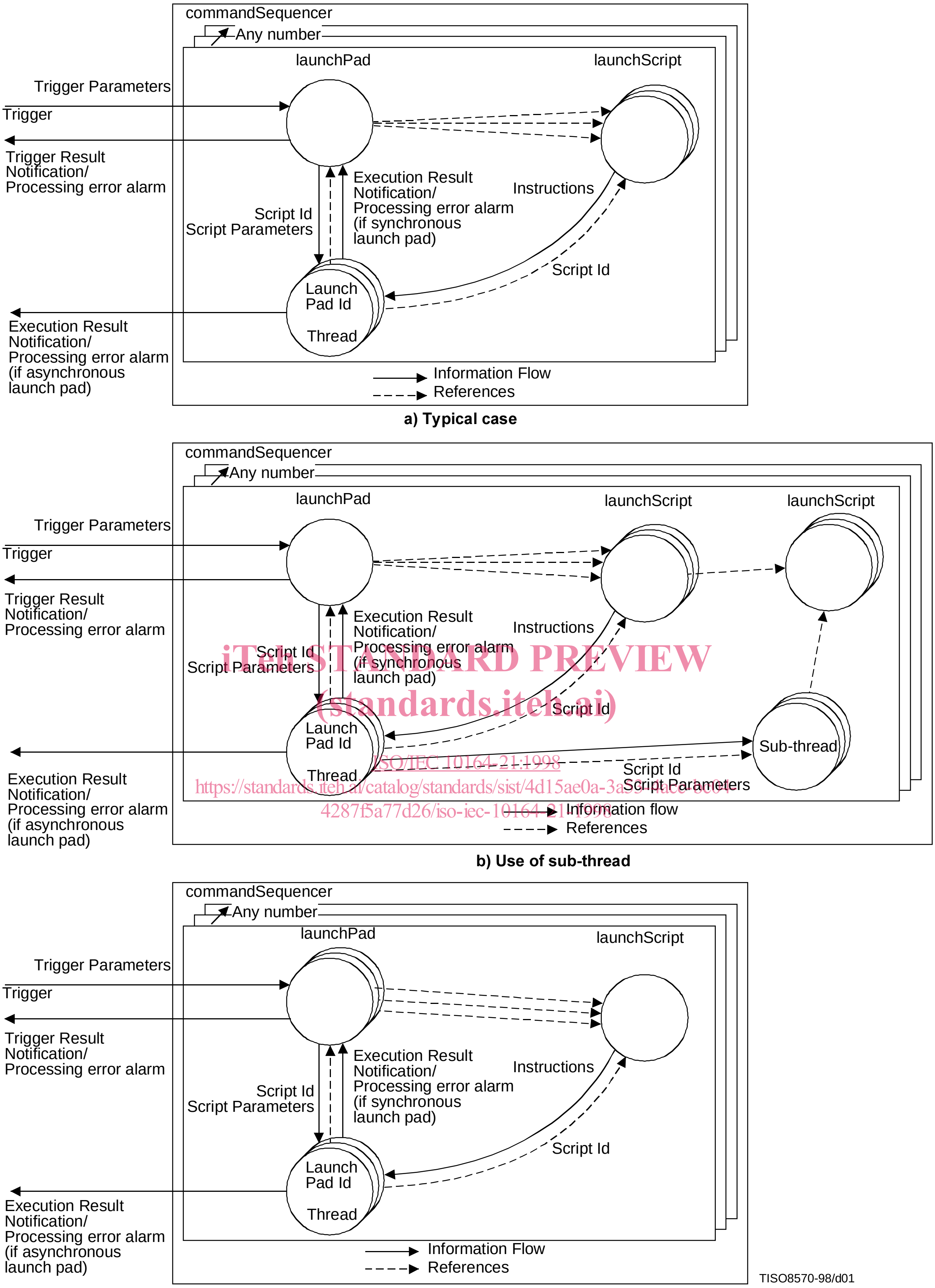
Figure 1 is a schematic description of the command sequencer capability of a system.

The functionality of a command sequencer is modeled by the launch script, thread, and launch pad objects. It is an OSI abstraction of pre-scheduled or delayed operation execution in open systems. A command sequencer may contain any number of launch pads, for which the command sequencer serves as a service provider. Each launch pad may execute one launch script at a time, or may execute multiple launch scripts at a time. On receiving a trigger from a trigger activator, a launch pad initiates the execution of a launch script. In addition to the trigger id component, the trigger may specify a launch script name (script id) and input arguments to the script as parameters within the execution parameter list component.

There are two types of launch pads, asynchronous launch pad and synchronous launch pad. An asynchronous launch pad returns a trigger result notification without waiting for results of execution of the launch scripts. Execution results or errors from launch script executions are notified directly from the thread. A synchronous launch pad, on the other hand, returns trigger result notification or processing error alarm after it gets all the execution results and errors from threads when the threads complete their execution.

A launch script instance may contain any number of individual instructions. The execution parameter list component of the trigger is a list of scripts (identified by their script ids) to be executed and the corresponding input parameters needed to execute those scripts. A default execution parameter list may be specified for a launch pad to execute in case it receives a trigger in which the trigger parameters are not specified. If the launch pad is not configured to execute a default execution parameter list and the execution parameter list component is not supplied by the trigger and if the launch pad receives a trigger attempting to activate it, a no script error code is returned in the error code field of the trigger result notification. The launch pad has an available script list attribute which can be configured to identify scripts that can be executed by it. If a execution parameter list component is present in the trigger, the launch pad verifies whether each script id from the execution parameter component is present in its available script list attribute. Only those script instances indicated by the script id which are present in the available script list attribute are executed. If none of the script ids are present in the available script list, the launch pad returns a script rejected error code in the trigger result, and script execution does not take place.

Specialized scripting language object classes are derived from the launch script object class. Hence these instructions may be specified as specialized script instances. Multiple sets of launch script instructions may be executed sequentially or in parallel by threads, in accordance with the execution parameter data type. Several nested levels of sub-threading may be necessary in order to execute script instances.



a) Typical case

b) Use of sub-thread

c) N:1 mapping between launch pads and a launch script

Figure 1 – Command sequencer model

In order to initiate the execution behaviour of launch script instances, a trigger shall be directed at the launch pad object instance. Unparameterized triggers may activate the launch pad in cases where the launch pad has a default execution parameter list.

The launch pad acts as an IVMO for a thread and supplies the execution parameter list to the executing parameters attribute of the thread. The execution parameter list may be a single execution parameter, a sequence of execution parameters or a set of execution parameters. The execution parameter is a sequence of script ids and script parameters. The script id identifies the managed object instance name of the scripting object instance to be executed and the script parameters supply the parameter values which are needed as inputs to the scripting object instance. If a sequence of execution parameters is specified, the launch pad spawns a thread in order to execute the script instances and supplies the script id and the script parameters (if required) from the execution parameters to the thread. On completion of the thread, this is repeated for the rest of the script ids in the list in sequence. If a set of execution parameters is specified, the launch pad supplies the set of script ids and script parameters (if required) to threads and the corresponding scripts instances are executed in parallel. The semantics of parameter passing between the launch pad and threads depend on the parameter passing mechanism supported by the scripting language in which the script is written.

One thread is assigned the execution of one script instance. This thread may spawn other threads if necessary. This may happen when one script instance invokes another script instances. When this happens, the thread executing a calling script spawns a sub-thread, passing the script id and script parameters (if required) of the called script to the sub-thread. The semantics of parameter passing between threads and sub-threads depend on the parameter passing mechanism supported by the scripting language in which the script is written.

Asynchronous launch pads should spawn suspendable threads. A suspendable thread can be suspended and resumed by means of suspend and resume actions respectively. Individual threads spawned by synchronous launch pads may not be suspended and resumed. All threads related to execution of a script may be suspended or resumed in both cases.

A thread is complete after all its sub-threads have completed successfully or reported an error. Once this happens, execution of a launch script is complete; the corresponding launch pad then returns to an inactive (idle) state. A thread is contained by the object which spawned it. A thread may be contained by a launch pad or another thread.

Multiple launch pads may reference a particular launch script. Multiple threads may reference that same launch script. The existence of a script is independent of any references to it by launch pads or threads. Launch scripts are defined in the specialized script classes derived from the launch script object class. The semantics and syntax of these scripts are specified in the definition of the scripting language in which the scripts are written. The scripting language definition also specifies a set of basic scripting functions which are necessary to provide control and processing ability to the launch scripts.

The general string script managed object class should be used for writing scripts which are represented in the form of a general string. The script language name attribute indicates the name of that scripting language and the script content attribute represents the script written in this scripting language in the form of a general string. Annexes F and G define a specialized scripting language, System Management Scripting Language (SMSL), as the scripting language in which scripts represented in the form of a general string should be written. It is possible to define other classes of scripts. Annex D defines, cmisScript, a scripting language, in the form of managed objects which can be used to write scripts in the CMIS environment.

7.2 Triggering process and reporting results

Triggers activators directed at the launch pad may be in various forms such as schedulers, operations, notifications and local action. When a launch pad receives a trigger, it spawns one or more threads in order to execute a script. After all threads related to a trigger are spawned, an asynchronous launch pad emits a trigger result notification which includes sets of thread id and script id. Results of script execution are propagated by threads as execution result notifications directly to the manager in the case of the asynchronous launch pad. After all threads related to a trigger are completed, a synchronous launch pad synchronizes all the execution results or errors from threads and emits a trigger result notification which includes sets of thread id, script id and execution results or errors to the manager.

The execution result type attribute of the script identifies the type of result expected from execution of the script and should correspond to the execution result type attribute of the execution result. The errorCode field of the execution result is set to the no error code when the execution is successful otherwise it is set to the appropriate error code.

An executing thread may terminate spontaneously either upon the completion of its execution or in abnormal conditions. In the latter case, the thread indicates abnormal termination by issuing a processing error alarm notification.

The execution result and processing error alarm notifications are issued by the thread and forwarded to appropriate notification destination(s). In the case of an asynchronous launch pad, these notifications are forwarded to external notification destinations whereas in the case of a synchronous launch pad, these notifications are propagated to the launch pad.

A manager may voluntarily terminate all launching processes by means of a delete operation to the corresponding launch pad. On receiving a delete operation, if the thread-launchPad name binding includes “DELETE DELETES-CONTAINED-OBJECTS” definition, all its threads which cause the execution of the script are terminated and deleted. The launch pad is then deleted.

A manager may voluntarily terminate all executions related to a thread by means of a delete operation to the corresponding thread. On receiving a delete operation, if the thread-thread name binding includes “DELETE DELETES-CONTAINED-OBJECTS” definition, all its sub-threads related to the execution of the script are terminated and deleted. The thread is then deleted.

In order to cause execution of all scripts being currently executed by a synchronous or asynchronous launch pad to be terminated, a terminate action may be directed at the launch pad. All threads related to the execution of scripts are terminated and deleted when a terminate action is received by the launch pad.

Launching of all threads being currently executed by a synchronous or asynchronous launch pad may be suspended by a suspend action directed at the launch pad and subsequently resumed by a resume action.

Execution of scripts by suspendable threads spawned by an asynchronous launch pad, may be suspended by a suspend action directed at the thread and subsequently resumed by a resume action. The thread id, returned in the trigger result notification should be supplied as a parameter to suspend and resume actions.

The launch pad has attributes to monitor a specific attribute of a specific object instance. If the value of the monitored attribute is changed, a trigger is generated to cause execution of a specified script list.

If the monitored attribute is an Event Discrimination Counter (EDC) counter as defined in Annex C, the notifications filtered by the EDC, trigger the launching of scripts by the launch pad.

7.3 Management of command sequencer

The attribute values of the launch pad, thread, launch script, and specialized scripting managed object instances are retrieved and modified through Get and Set operations, respectively.

Tables 1 to 5 map the status attributes of the command sequencer, launch pad, thread and script managed objects to the states defined in CCITT Rec. X.731 | ISO/IEC 10164-2.

NOTE – "–" means any value.

Table 1 – Status table of command sequencer

Status of command sequencer	Administrative state	Operational state
CS not operational	–	disabled
CS is operational	unlocked	enabled
CS is locked	locked	enabled
CS is shutting down	shuttingDown	enabled

When a command sequencer has a disabled operational state it is in a totally inoperable state and its launch pads are not executing scripts. If it is in an enabled state, an event which consists of an operation being performed at the managed object boundary may cause a transition from a locked administrative state to an unlocked state or vice versa. When the command sequencer goes into a locked state, it causes its launch pads to suspend the execution of launch scripts. Alternatively, when it goes into an unlocked administrative state, the launch pads are available to start or resume the execution of launch scripts.

Table 2 – Status table of launch pad

Status of launch pad	Administrative state	Operational state	Control status	Usage state	Availability status
LP is not operational	–	disabled	–	–	
LP is operational	unlocked	enabled	–	Busy	
LP is operational	unlocked	enabled	–	Idle	–
LP is locked	locked	enabled	–	Idle	–
LP is on duty	–	–	–	–	Not Off duty
LP is off duty	–	–	–	–	Off duty
LP is suspended	–	–	Suspended	–	–
LP is resumed	–	–	Empty	–	–

When a launch pad has a disabled operational state, it is in a totally inoperable state and cannot execute scripts. If it is in an enabled state, an event which consists of an operation being performed at the managed object boundary may cause it to transition from a locked administrative state to an unlocked state or vice versa. When the launch pad goes into a locked state, it suspends the execution of launch scripts. Alternatively, when it goes into an unlocked administrative state, the launch pads are available to start or resume the execution of launch scripts. The launch pad is made inactive by an internal control process according to a predetermined time schedule and its availability status value is off duty. A suspend action causes the control status to change to suspended and a resume action changes it back to its default value, empty.

Table 3 – Status table of thread

Status of thread	Operational state
Thread not operational	Disabled
Thread operational	Enabled

The thread is in an enabled state when it is performing a script execution and in a disabled state when it is not.

Table 4 – Status table of suspendable thread

Status of suspendable thread	Operational state	Control status
Suspendable thread not operational	Disabled	–
Suspendable thread operational	Enabled	–
Suspendable thread suspended	–	Suspended
Suspendable thread resumed	–	Empty

A suspend action causes the control status of the suspendable thread to change to suspended and a resume action changes it back to its default value, empty.

Table 5 – Status table of launch script

Status of launch script	Administrative state
LS execution allowed	Unlocked
LS execution not allowed	Locked