INTERNATIONAL STANDARD

ISO/IEC 20944-4

First edition
2013-01-15

Information technology — Metadata Registries Interoperability and Bindings (MDR-IB) —

Part 4: Protocol bindings

Technologies de l'information — Interopérabilité et liaisons des registres de métadonnées (MDR-IB) —
Partie 4: Liaisons de protocoles

Reference number
ISO/IEC 20944-4:2013(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. 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.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 20944-4 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 32, *Data management and interchange*.

ISO/IEC 20944 consists of the following parts, under the general title *Information technology — Metadata Registries Interoperability and Bindings (MDR-IB)*:

— *Part 1: Framework, common vocabulary, and common provisions for conformance*

— *Part 2: Coding bindings*

— *Part 3: API bindings*

— *Part 4: Protocol bindings*

— *Part 5: Profiles*

Introduction

The ISO/IEC 20944 series of International Standards provides the bindings and their interoperability for metadata registries, such as those specified in the ISO/IEC 11179 series of International Standards.

This part of ISO/IEC 20944 contains provisions that are common to protocol bindings (Clauses 4-10) and the protocol bindings themselves (Clause 11 onward). The protocol bindings have commonality in their conceptualization of the services provided. For example, common features include:

— common data transfer semantics;

— harmonized session services for connection-oriented and connection-less protocols.

Clause 11 and onward are the actual protocol bindings themselves. The clauses of this part of ISO/IEC 20944 are organized such that future amendments are possible, which would include additional protocol bindings.

Information technology — Metadata Registries Interoperability and Bindings (MDR-IB) —

Part 4: Protocol bindings

1 Scope

The ISO/IEC 20944 series of International Standards describe codings, application programming interfaces (APIs), and protocols for interacting with an ISO/IEC 11179 metadata registry (MDR).

This part of ISO/IEC 20944 specifies provisions that are common across protocol bindings for the ISO/IEC 20944 series of International Standards, and provides the individual protocol bindings themselves.

2 Normative references

The following referenced documents are indispensable for the application 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/IEC 11404:2007, *Information technology — General-Purpose Datatypes (GPD)*

ISO/IEC 13886:1996, *Information technology — Language-Independent Procedure Calling (LIPC)*

ISO/IEC 20944-1:2013, *Information technology — Metadata Registries Interoperability and Bindings (MDR-IB) — Part 1: Framework, common vocabulary, and common provisions for conformance*

ISO/IEC 20944-2:2013, *Information technology — Metadata Registries Interoperability and Bindings (MDR-IB) — Part 2: Coding bindings*

IETF RFC 2616, *Hypertext Transfer Protocol — HTTP/1.1*, June 1999

IETF RFC 4918, *HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)*, June 2007

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 20944-1 apply.

4 Intended use of this part of ISO/IEC 20944

The purpose of this part of ISO/IEC 20944 is to provide a common set of services (common protocol provisions) and standardized protocol bindings such that interoperable systems can be created to access the MDR repositories. These systems are interoperable in that they should be able to access MDR repositories that conform to this International Standard.

5 Abstract model

5.1 General

The protocol bindings have commonality in their conceptualization of data access services. For example, common features include:

— common data transfer semantics

— harmonized session services for connection-oriented and connection-less protocols

The conceptual model is divided into two parts: the data model and the control model. The data model describes the logical structure of information as it is transferred. The control model describes the structure of the transactions, events, and protocol state.

5.2 Participant model

The senders and receivers of messages are called participants. The protocol binding permits client-server, peer-to-peer, publish-subscribe, multicast, and broadcast modes of participation. The underlying communication system determines which modes of participation are supported.

5.3 Data model

The data structure of ISO/IEC 20944-2 is incorporated via normative reference.

5.4 Control Model

The control model refers to the methods of causing actions among the parties communicating. The following concepts characterize the control model.

5.4.1 Event

Assuming prior arrangement and appropriate authorization, each participant may send events. For the sender, an event is a one-way message with no expected response. In programming languages, an event is analogous to a "go to". For the receiver, an incoming event is processed by an event handler. The semantics of event handling is outside the scope of this part of ISO/IEC 20944.

5.4.2 Wait

Assuming prior arrangement and appropriate authorization, a participant can ask another participant to wait for an event, e.g., node A asks node B to wait for event E. In operating system kernel technology, this is equivalent to "sleeping on an event".

5.4.3 Call

Assuming prior arrangement and appropriate authorization, a participant can send a "call" message, similar to a remote procedure call. A "call" is equivalent to (1) sending data (input parameters) from caller to callee, (2) sending a "start" event to the "callee", (3) posting a "wait" on the callee's "finish" event, (4) receiving data (output parameters) from callee to caller. The advantage of "call" over combined "event" and "wait" is that "call" handles the storage of temporary information in the "event" and "wait" services.

5.4.4 Parameters

The "event", "wait", and "call" messages may pass parameters. Parameters are passed in a list along with the target event.

EXAMPLE A hypothetical function named ls is called with two function properties: (1) the options to the command, and (2) three parameters to the command (the third parameter has properties, too). The following is illustration shows the syntax:

```
{ call ls .listing-type: long .dir-expand: no a b c .listing-encoding: iso-646 }
```

5.5 Connections

A connection is created when a client successfully completes a connection command to a server. A participant is one end of communication regardless of role (e.g., client or server). Some implementations may support multiple roles. Once a connection is established, parameters may be negotiated and sessions may be created for data and control transfer. A session is a connection established between one or more participants.

5.6 State model

A participant may act in the role of client or server.

5.6.1 Server state model

A server behaves according to the following state model.

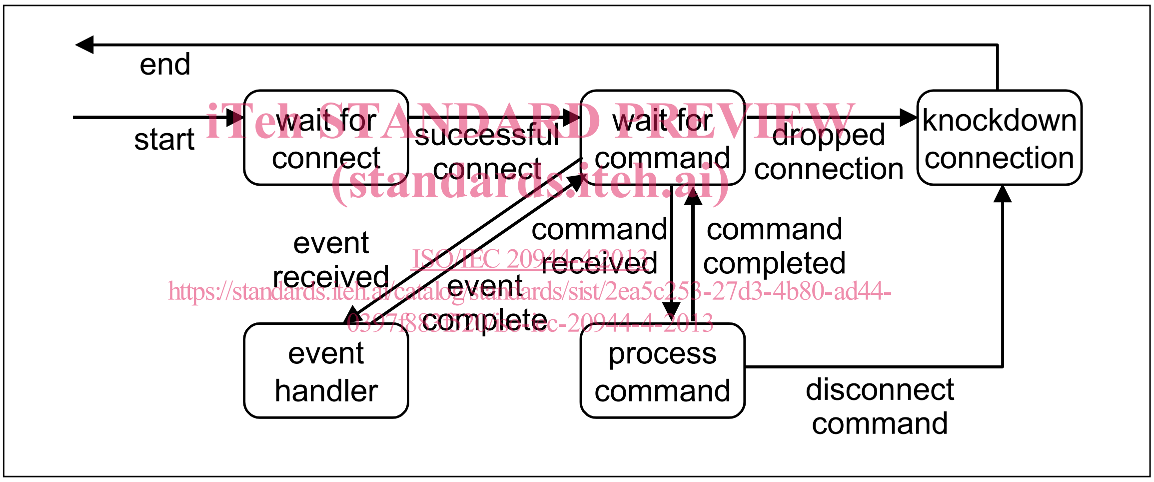

Figure 1 — State model for server

The following are the states for the server.

- Start: The state model begins in this state.
- Wait For Connect (state): The server is waiting for a connect.
- Successful Connect (event): Starts up the connection. Changes to Wait For Command (state).
- Wait For Command (state): The server is waiting for a command.
- Command Received (event): Begins the processing of a command. Changes to Process Command (state).
- Event Received (event): Begins the event handling. Changes to Event Handler (state).
- Dropped Connection (event): The connection was dropped. Changes to Knockdown Connection (state).
- Process Command (state): Processes a command, consumes and produces data, returns command status.

— Command Completed (event): The command has completed. Changes to Wait For Command (state).

— Disconnect Command (event): The Disconnect command was executed. Changes to Knockdown Connection (state).

— Event Handler (state): Processes an unsolicited event (e.g., security requests) and returns event status.

— Event Complete (event): The Event completed. Changes to Wait For Command (state).

— Knockdown Connection (state): Closes the connection. Changes to End (state)

— End: The state model exits.

5.6.2 Client state model

A client behaves according to the following state model.

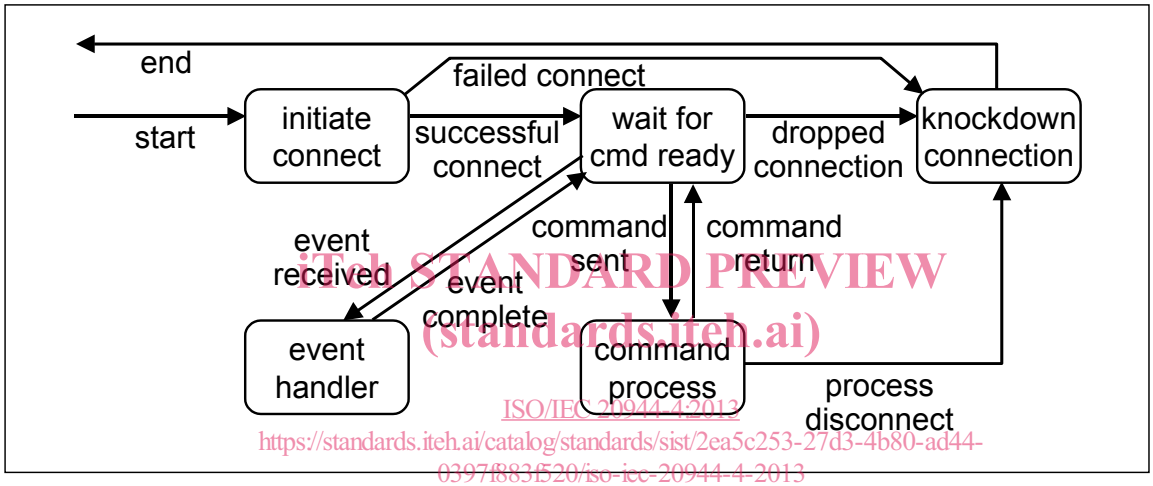

Figure 2 — State model for client

The following are the states for the client.

— Start: The state model begins in this state.

— Initiate Connect (state): The client initiates a connection the server.

— Successful Connect (event): Starts up the connection. Changes to Wait For Cmd Ready (state).

— Failed Connect (event): Connection failure. Changes to Knockdown Connection (state).

— Wait For Cmd Ready (state): The client is waiting for the server to indicate that it is ready for a command.

— Command Sent (event): Sends the command to process. Changes to Command Process (state).

— Event Received (event): Begins the event handling. Changes to Event Handler (state).

— Dropped Connection (event): The connection was dropped. Changes to Knockdown Connection (state).

— Command Process (state): Requests the processing of a command, produces and consumes data, receives the command status upon command completion.

— Command Return (event): The command has completed. Changes to Wait For Cmd Ready (state).

— Process Disconnect (event): Sends the Disconnect command. Changes to Knockdown Connection (state).

— Event Handler (state): Processes an unsolicited event (e.g., security requests) and returns event status.

— Event Complete (event): The Event completed. Changes to Wait For Cmd Ready (state).

— Knockdown Connection (state): Closes the connection. Changes to End (state)

— End: The state model exits.

6 Services

6.1 Use of ISO/IEC 13886

The notation of ISO/IEC 13886 Language-Independent Procedure Calls is used to describe service interfaces.

NOTE The use of ISO/IEC 13886 notation is intended to provide a common description of services, but the protocol bindings themselves are not based upon APIs.

6.2 Session establishment services

The following messages start up and shut down sessions for protocols that use session-based access to metadata registries.

6.2.1 Connect

Synopsis

```
mdrib_connect:
procedure
(
    in target: string, // repository to connect to
    in options: string, // connect options
)
returns mdrib_handle,
```

Description

Creates a new session to a data repository as named by target. The options parameter is an implementation-defined set of connection options. If successful, returns a session handle to the repository, but does not request access (see mdrib_open). If not successful, returns a null session handle.

6.2.2 Disconnect

Synopsis

```
mdrib_disconnect:
procedure
(
    in session: mdrib_handle // session handle
)
returns (state(success,failure)),
```

Description

Closes and disconnects a session to a data repository associated with the handle session and all of its child sessions. Returns success if successful, failure if unsuccessful.