IEC 61970-407

Edition 1.0 2007-08

INTERNATIONAL STANDARD

Energy management system application program interface (EMS-API) – Part 407: Time Series Data Access (TSDA)

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

IEC 61970-407:2007(E)

THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2007 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester.

If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch
Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

IEC 61970-407

Edition 1.0 2007-08

INTERNATIONAL STANDARD

Energy management system application program interface (EMS-API) – Part 407: Time Series Data Access (TSDA)

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE S

ICS 33.200

ISBN 2-8318-9253-8

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

CONTENTS

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

INTERNATIONAL ELECTROTECHNICAL COMMISSION

ENERGY MANAGEMENT SYSTEM APPLICATION PROGRAM INTERFACE (EMS-API) –

Part 407: Time Series Data Access (TSDA)

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61970-407 has been prepared by IEC Technical Committee 57: Power systems management and associated information exchange.

The text of this standard is based on the following documents:

FDIS	Report on voting
57/889/FDIS	57/908/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

A list of all parts of the IEC 61970 series, under the general title *Energy Management System Application Program Interface (EMS-API)*, can be found on the IEC website.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

The committee has decided that the contents of this publication will remain unchanged until the maintenance result date indicated on the IEC web site under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed;
- withdrawn;
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

INTRODUCTION

This part of IEC 61970 is part of the IEC 61970 series that defines Application Program Interfaces (APIs) for an Energy Management System (EMS). The IEC 61970-4XX and IEC 61970-5XX series documents comprise Component Interface Specifications (CISs). The IEC 61970-4XX series CIS are specified as Platform Independent Models (PIMs), which means they are independent of the underlying technology used to implement them. PIM specifications are also referred to as Level 1 specifications. The IEC 61970-5XX series CIS, on the other hand, are specified as Platform Specific Models (PSMs). PSM specifications are also referred to as Level 2 specifications.

IEC 61970-4XX CISs specify the functional requirements for interfaces that a component (or application) should implement to exchange information with other components (or applications) and/or to access publicly available data in a standard way. The component interfaces describe the specific event types and message contents that can be used by applications for this purpose.

IEC 61970-407 specifies an interface for the efficient transfer of time series data in a distributed environment. Small amounts of data are transferred with short delay but also large amounts of data are transferred in short time but with possibly longer delay. Replay of time series data is also supported. This is a typical requirement for a SCADA system that acts as a real time data provider to other sub-systems. Other systems than SCADA may also benefit from the characteristics of TSDA. When short delay times as well as bulk data transfer is required TSDA is a good fit.

These component interface specifications refer to entity objects for the power system domain that is defined in the IEC 61970-3XX series, including IEC 61970-301.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

ENERGY MANAGEMENT SYSTEM APPLICATION PROGRAM INTERFACE (EMS-API) –

Part 407: Time Series Data Access (TSDA)

1 Scope

The IEC 61970-407 Time Series Data Access (TSDA) specification specifies a generalized interface for efficient exchange of data. The specification takes into account the latencies caused by a Local Area Network (LAN) providing efficient data exchange also over Local Area Networks.

IEC 61970-407 is derived from the Object Management Group (OMG) Historical Data Access from Industrial Systems (HDAIS) specification. OMG HDAIS relies on the OMG Data Access Facility (DAF) and OPC Historical Data Access (HDA) specifications. OMG HDAIS is a Platform Specific Model (PSM) with CORBA as the platform and OPC HDA is a PSM with Microsoft COM as the platform. The IEC 61970-407 specification describes the functionality of these PSMs in a technology independent way (i.e., as a Platform Independent Model (PIM)). Hence, it explains the functionality to a level that can be used to create additional PSMs or be an introduction to existing PSMs, i.e. HDAIS and OPC HDA. Implementers wanting an introduction to OMG HDAIS and OPC HDA should read these documents.

The TSDA interface is intended to interoperate with other IEC 61970 based interfaces. Hence, it is possible to use information retrieved from other interface to access the same information using this interface, for example:

- object identifiers,
- attribute names or identifiers,
- class names or identifiers.

Subclause 4.6 provides a generic mapping for the CIM classes and attributes.

The way data is organized in a server implementing the TSDA interface can be seen by using the browse interfaces for data and meta data. It is also possible to use the data access interface directly without using the browse interfaces if the client has an *a priori* knowledge of object, class and attribute identifiers. Object identifiers may be retrieved using data from other interfaces, for example a CIMXML file or the IEC 61970-404 interface. Information on what classes and attributes are available will be described in IEC 61970-45X documents, for example historical SCADA data, historical State Estimator results etc.

IEC 61970-1 provides the EMS-API reference model upon which this standard is based. In that reference model, the terminology used in this part of IEC 61970 is introduced and the role of the CIS is explained.

IEC 61970-401 provides an overview and framework for the CIS (IEC 61970-4XX) standards.

The mapping of IEC 61970-407 to implementation specific technologies or PSMs is further described in a separate series of documents, i.e. the future IEC 61970-5XX. For actual implementations the future IEC 61970-5XX, OMG HDAIS, OMG DAF or OPC HDA are used.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

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.

IEC 61970-1, *Energy management system application program interface (EMS-API) – Part 1: Guidelines and general requirements*

IEC/TS 61970-2, *Energy management system application program interface (EMS-API) – Part 2: Glossary*

IEC 61970-301:2005, *Energy management system application program interface (EMS-API) – Part 301: Common Information Model (CIM) base*

IEC 61970-401, *Energy management system application program interface (EMS-API) – Part 401: Component Interface Specification (CIS) Framework*

IEC 61970-402, *Energy management system application program interface (EMS-API) – Part 402: Component Interface Specification (CIS) - Common Services*

Historical Data Access from Industrial Systems (HDAIS), OMG Adopted Specification Version 1.0, dtc/2003-02-01 November 2003 (Referred herein as 'OMG HDAIS')

Utility Management System (UMS) Data Access Facility (DAF), OMG Adopted Specification, Version 2.0, formal/02-11-08, November 2002 (Referred to herein as ‘OMG DAF’)

Data Acquisition from Industrial Systems (DAIS), OMG Adopted Specification Version 1.0, formal/2002-11-07 November 2002 (Referred herein as 'OMG DAIS')

OPC Historical Data Access Custom Interface Specification, Version 1.20, OPC Foundation, December 2003 (Referred to herein as ‘OPC HDA’)

3 Terms, definitions and identification conventions

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 61970-2 apply.

3.2 Conventions

A convention used in this document to uniquely identify a UML attribute is to concatenate the class name and the attribute name with a period in between, for example, the attribute “id” in the class “Node” will then be named “Node.id”. For attributes in sub structures multiple attribute names may be used, for example “Item.id.node_id” where the “node_id” is a part in the structure “Item.id”.

4 CIS Specification

4.1 Background (informative)

For historical reasons, control systems for different industrial processes have evolved along different lines. Control systems for power systems have evolved on a UNIX base and control systems for most other industrial processes have evolved on a Windows base. For Windows based control systems, OPC has become the dominating standard. For UNIX based systems, the DAIS/HDAIS API defined in Common Object Request Broker Architecture (CORBA)

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

Interface Definition Language (IDL) has been developed. DAIS/HDAIS is based on OPC to benefit from the success of OPC and enable easy bridging to OPC. With this intent Object Management Group (OMG) started in 1997 to develop a CORBA based interface with the same functionality as OPC. TSDA has the functionality from OMG HDAIS and OPC HDA described in a technology neutral way, hence TSDA is intended to be a Platform Independent Model (PIM).

4.2 Historian use case (informative)

An important component in a utility operational system is a Historian or Utility Data Warehouse (UDW) that acts as a repository for past or future time series data. A UDW acts as a server to client components. Such a system typically has the following components:

- Process instrumentation making sensor data and actuation capabilities available.
- Remote Terminal Units (RTUs) or substation control systems reading sensor data and controlling actuators.
- Process communication units connecting to RTUs or substation control systems. Remote communication is typically solved by specialized RTU or field bus protocols (e.g. IEC 60870-5). The IEC 61850 is a new standard for communication with and within substations. The interfaces defined in this specification can be used as a standard API to encapsulate such communication solutions at the client side.
- SCADA subsystem making processed sensor data and control capabilities available to operators, applications or other systems.
- A Model based management system, for example Energy Management System (EMS), using the SCADA subsystem for extended processing and control.
- A Graphical User Interface (GUI).
- UDW that stores time series data.

The architecture is shown in Figure 1 below.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

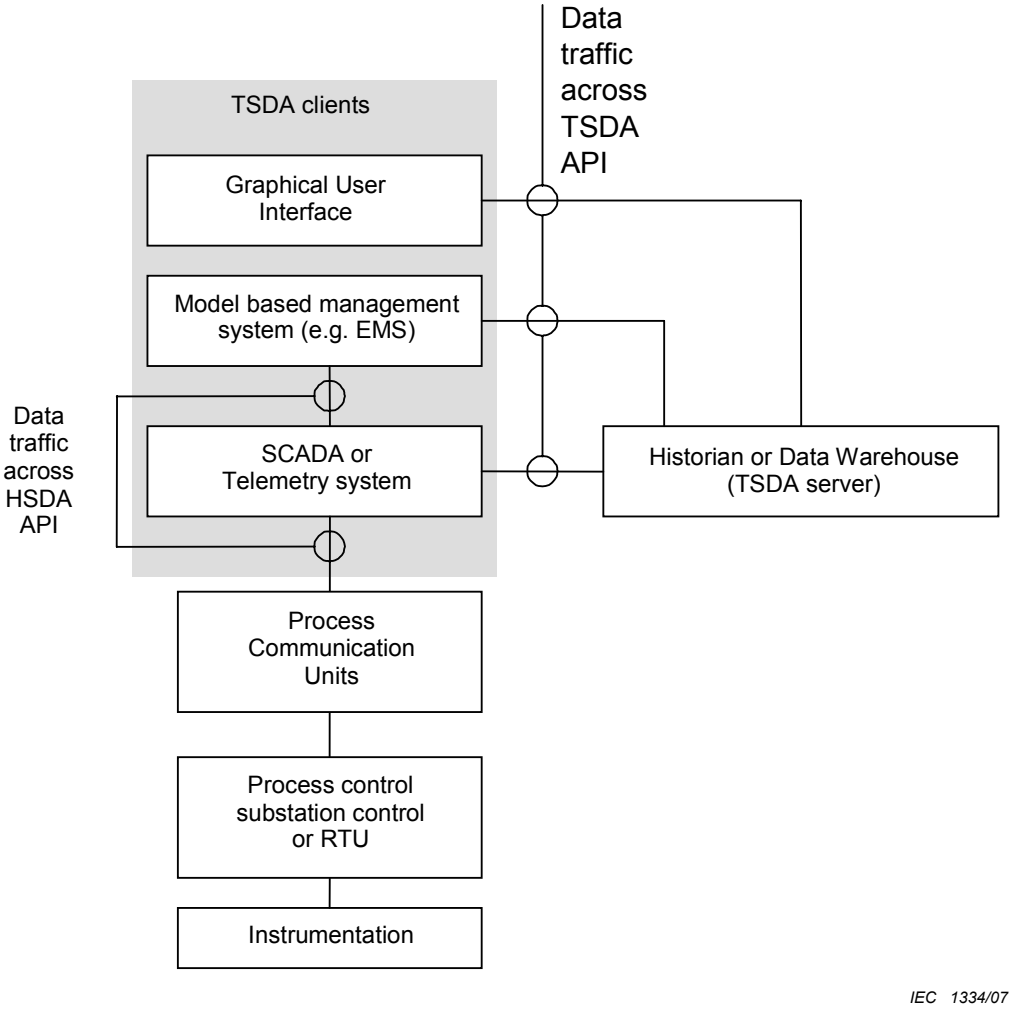

Figure 1 – Control system structure

A UDW and possible clients using it are shown in Figure 2:

- Data sources recording data as the SCADA recording real time data, EMS recording calculation results or GUI recording manually entered data.
- A Process Communication Unit providing time series data from remote sources.
- Clients, for example a GUI, retrieving data for discovery, presentation and updates/corrections.
- Clients, for example analysis programs, retrieving data for analysis and or input to calculations
- A UDW typically also has calculations that create new results from already stored time series data.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

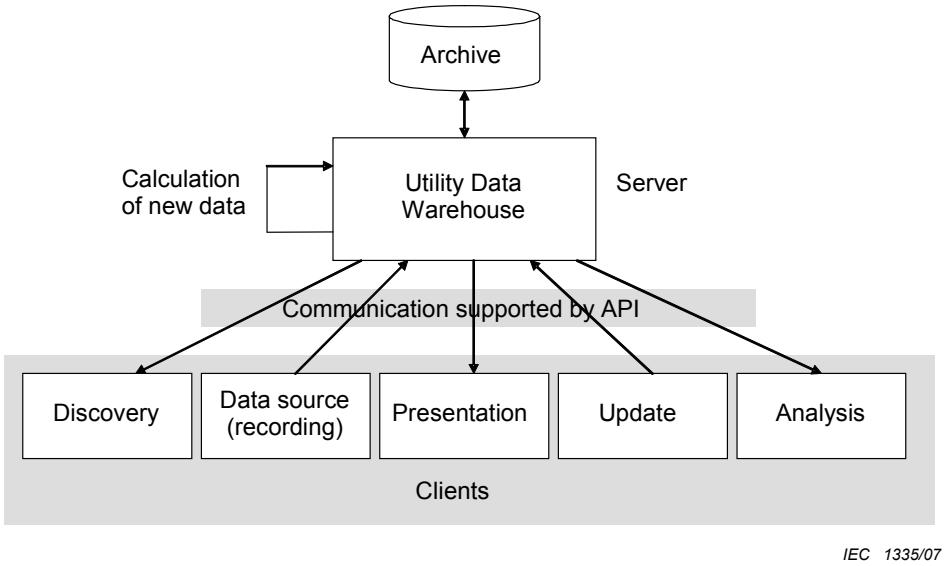

Figure 2 – TSDA server and clients

TSDA supports both subscription and read/write operations. The concept of subscription is described in Figure 3.

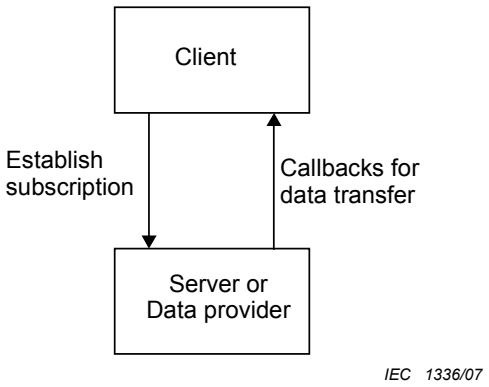

Figure 3 – Data subscription

A subscription involves a server that publishes data and clients that subscribe to receive the data. The server has no *a priori* knowledge of its clients; they become known to the server when a client creates a subscription. Once a subscription is established, the server calls the client back when data becomes available or is updated.

4.3 Data model

The TSDA data model describes how data seen through the TSDA interface appears to be organized within a server. A server implementation may organize data differently, but a client using TSDA will see a data model as shown in Figure 4.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

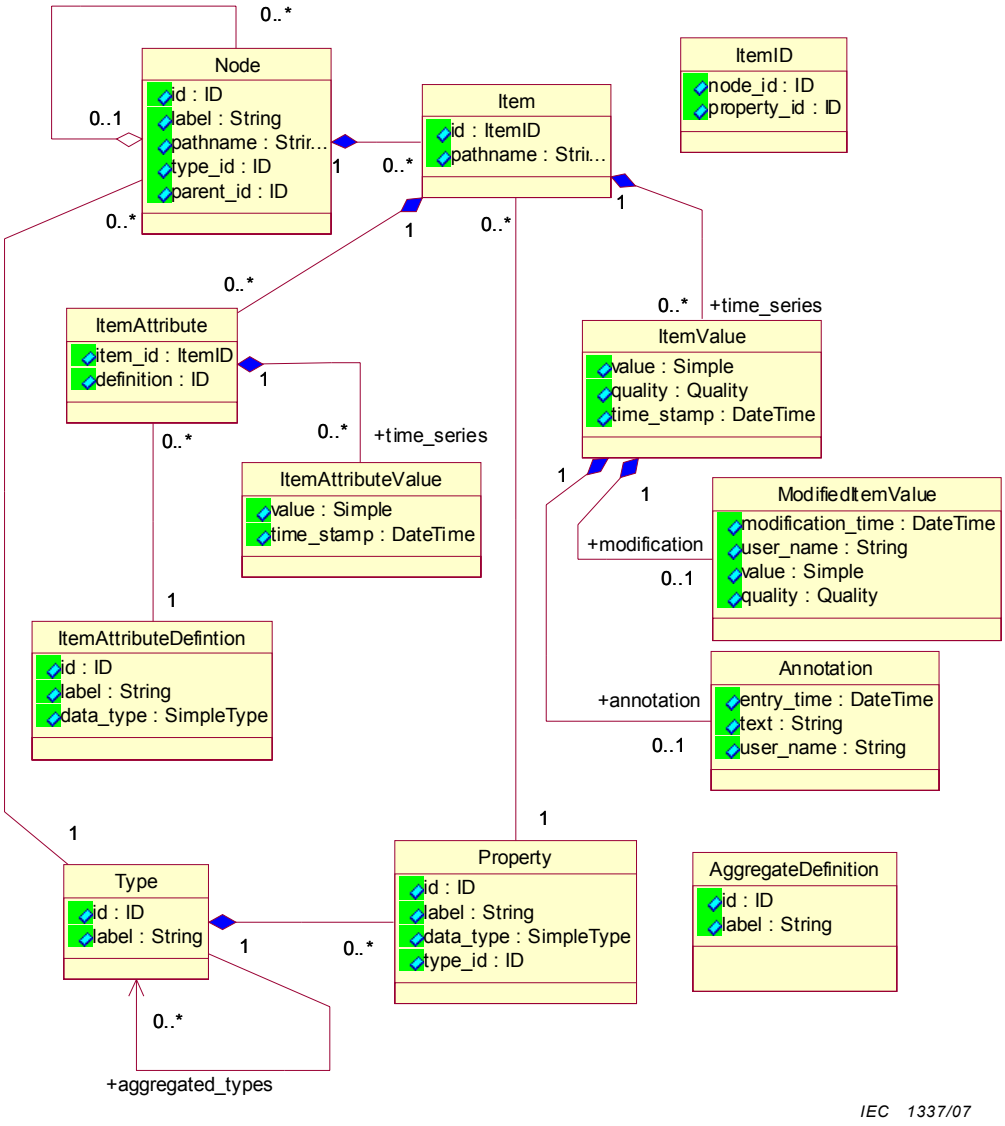

Figure 4 – TSDA data model

The following objects are defined in the model:

- Type, which describes objects with the same time series characteristics, for example the time interval between values, calculations of values etc. Hence, the TSDA Type is different from the HSDA Type. The HSDA Types typically describe the types defined in IEC 61970-301 while the TSDA Type has no corresponding description in IEC 61970-301. However, a TSDA Type usually refers to a HSDA Type.
- Property, which describes the Items recorded as time series. A TSDA Property usually refers to a HSDA Property. A Type can have any number of Properties.
- ItemAttributeDefinition, which describes ItemAttributes specific to time series data. ItemAttributeDefinitions are similar to the Properties in that it describe time series data but different as the data is specific to the time series. Several ItemAttributeDefinitions usually exist for a Type.
- ItemAttribute, which describes how Items are treated within a TSDA server, i.e. it is meta data for Item time series. Hence ItemAttributes cannot be accessed using HSDA. As an

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

ItemAttribute may change over time it is a time series by itself, i.e. the ItemAttributeValues.

- Node, which is an object that has one or more Items recorded as time series. A TSDA Node typically has a corresponding HSDA Node.
- Item, which is an object that has one time series, i.e. the ItemValues. An Item has a number of ItemAttributes describing the time series.
- ItemValue, which is a time stamped and quality coded value. An Item has a time series consisting of ItemValues.
- ModifiedItemValue, which is a modification made to an ItemValue. As it is not allowed to alter an ItemValue changes are instead recorded by the ModifiedItemValue.
- Annotation is an annotation text that can be added to an ItemValue.
- AggregateDefinition, which describes calculations that can be used on time series data, for example mean value, max value etc.

The following UML attributes and references related to identification:

- Type.id, Property.id, Node.id, ItemAttributeDefinition.id and AggregateDefinition.id are system unique identification codes of type ID. The ID type is a string or number with a range sufficient to enable globally unique identifiers. However, the id is not required to be globally unique. The id is intended for use by machines.
- Item.id of type ItemID is a system-unique Item identification that consists of the ItemID.node_id that refer to a Node and the ItemID.property_id that refers to a Property. The Item.id is intended for use by machines.
- Type.label, Property.label, Node.label, ItemAttributeDefinition.label and Aggregate Definition.label are human readable names. The Type.label or Property.label uniquely identifies a Type or Property. A Node.label is unique only among the Nodes that are children of the same parent Node. The label is intended for use by humans.
- Node.pathname is a name unique within a system. It consists of all labels from the Node in question to the root. This is the same idea as for a file path. The exact layout of the Node.pathname is platform and implementation specific. However, it is suggested that whenever possible an implementation shall use existing standards, for example XPath from W3C. The pathname is intended for use by humans.
- Item.pathname is a name unique within a system. It consists of all names from the Node in question to the root and ends with the Property.label for the Property describing the Item. The exact layout of the Item.pathname is platform and implementation specific. However, it is suggested that whenever possible, an implementation shall use existing standards, for example XPath from W3C. The pathname is intended for use by humans.

The following UML references related to associations:

- Type.aggregated_types enumerates the IDs of the Type of Nodes that may be children to a Node of this Type. This is used to restrict what Type of Nodes may appear as children to other Nodes in the Node hierarchy, for example, if a Node of Type Substation may contain Nodes of Type Bay or Measurement, Type.aggregated_types enumerates the Types Substation and Measurement.
- Property.type_id refers to the Type a Property belongs to. It implements the Property.type reference.
- Node.type_id is the Type.id of the Type that a Node has.
- Node.parent_id refers to a parent Node, if it exists. It creates a hierarchical structure of Nodes that is the basis for creation of the pathnames from the labels in the path to the root Node.
- ItemID.node_id refers to the Node that the Item belongs to.
- ItemID.property_id refers to the Property that the Item represents.
- Item.time_series is a sequence of ItemValues representing the actual time series.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

- ItemValue.modification holds a possible modification the ItemValue.
- ItemValue.annotation holds a possible annotation to the ItemValue.
- ItemAttribute.item_id and ItemAttribute.definition refer to an Item and an Item AttributeDefinition.
- ItemAttribute.time_series is a sequence of ItemAttributeValues representing the actual time series.

The following UML attributes holding data:

- Property.data_type describes the Item.value data type, for example, any string, number, boolean, etc.
- ItemAttributeDefinition.data_type describes the ItemAttribute.value data type, for example, any string, number, boolean, etc.
- ItemValue.value is a value of the data type Property.data_type. The data type Simple is any type not divisible into sub types, for example any type of number, string or boolean.
- ItemValue.quality is the Quality code of the ItemValue.value. The Quality code tells if the ItemValue.value is valid and if not valid it supplies the reason why the ItemValue.value is invalid. The default Quality code is "Good", i.e. a ItemValue.value that has no particular Quality code associated with it will always appear as "Good" seen through the HSDA interface. A large number specifications defining Quality codes exists, for example RTU protocols, ICCP, ELCOM, OPC DA, OMG DAIS DA etc. This specification does not specify a unified Quality code system, but relies on the Quality codes defined in OMG DAIS DA.
- ItemValue.time_stamp is the time when the ItemValue.value was last updated. If no time has been assigned to an ItemValue.value, the Item.time_stamp is left unspecified. For configuration parameters, if available, the time when the ItemValue.value was entered or updated may be used as ItemValue.time_stamp.
- ItemAttributeValue.value is a value of the data type ItemAttributeDefinition.data_type. The data type Simple is any type not divisible into sub types, for example any type of number, string or boolean.
- ItemAttributeValue.time_stamp is the same as ItemValue.time_stamp.

4.4 Messages (normative)

The payload in the time series messages between the server and the clients (both directions) consists of:

- An Item identification that uniquely identifies the Item (the Item.id in Figure 4) within the server.
- Time stamped values where each value consists of
  - A data value (the ItemValue.value in Figure 4).
  - The quality (the ItemValue.quality in Figure 4) of the data value, i.e. if the value is reliable. If the value is bad, the quality also indicates the reason why the value is bad. The default quality code is "Good", i.e. the value is valid.
  - A time stamp (the Item.time_stamp in Figure 4) that indicates when the item value was recorded for past values or the planned time for a future value.

Messages from clients to the server may also contain additional information if a time stamped value shall replace an already existing value, i.e. a ModifiedItemValue shall be created.

4.5 Interface (normative)

4.5.1 Objects and interfaces

The TSDA interfaces, objects and their relations are shown in Figure 5.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

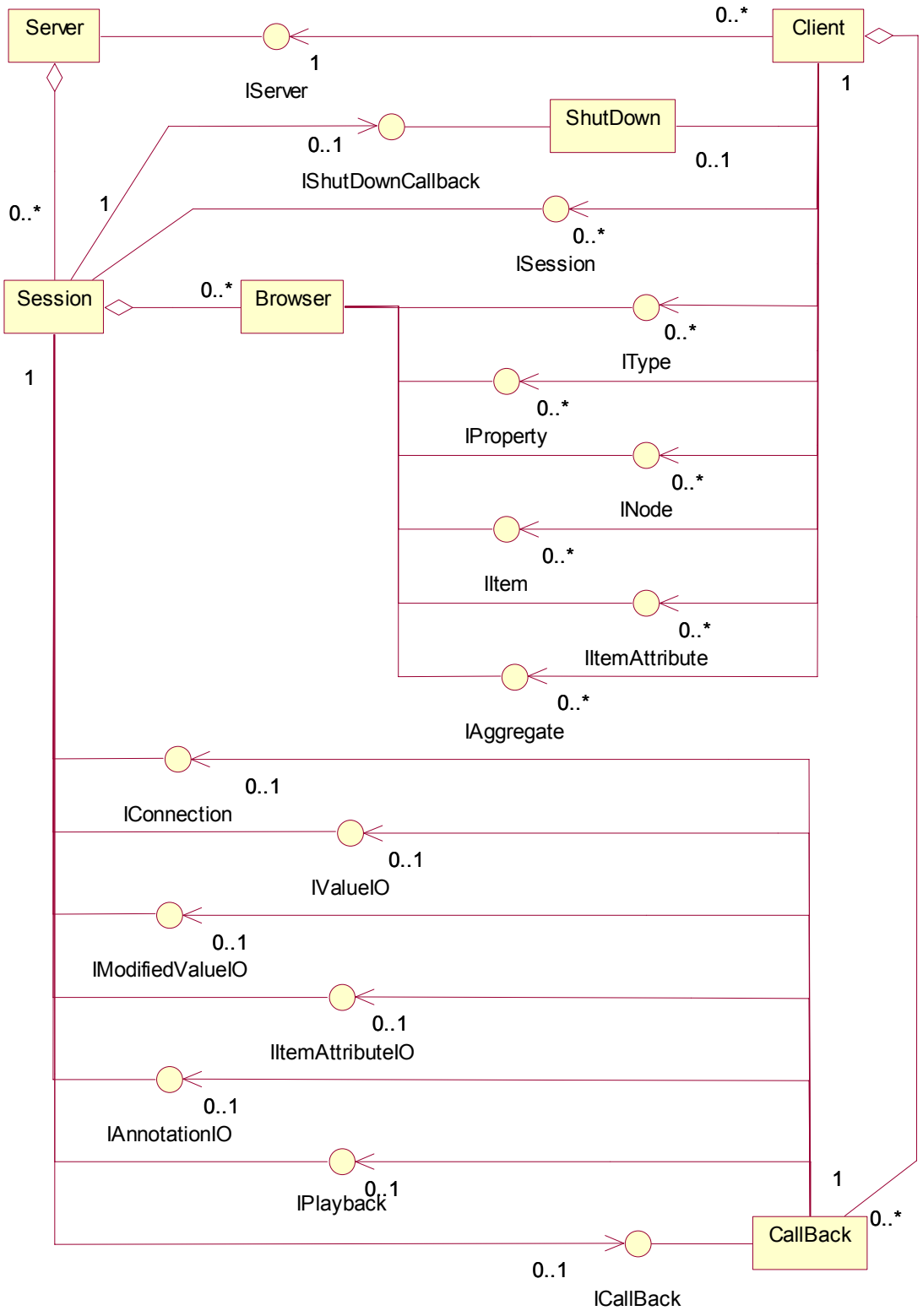

Figure 5 – TSDA objects and interfaces

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

Figure 5 indicates the objects and the interfaces a HSDA server and client implements. The diagram uses the UML 1 notation where interfaces are shown as small circles attached with a line to the class implementing the interface. Interface names also have an initial upper case "I".

The TSDA interfaces are divided in the groups:

- Browse, which is used for discovery of data. The Browse interfaces are located at the Browser object.
- Data access, which is used to actually access the data. The Data access interfaces are located at the Session object.
- Management, which is used to manage the connection between the Server and a Client.
- Call back, which is used by the Server to deliver data to the Clients.

Configuration of the data objects within the server hosting the data is outside the scope of this part of IEC 61970.

The Server is an object that implements the IServer interface and may have any number of Clients using it. The Server object has a number of Session objects. Also note that the Server and Session objects may be combined into one object as in OPC HDA.

The Session object has a number of interfaces:

- ISession, which is used to manage the session.
- IConnection, which is used to manage the connection between a Client and the Server.
- IValueIO, which is used to access Item time series data. In this interface modifications, if present, override the original recorded values.
- IModifiedValueIO, which is used to read old ItemValues as they were before a modification.
- IItemAttributeIO, which is used to access the ItemAttribute time series data.
- IAnnotationIO, which is used to access ItemValue Annotations.
- IPlayBack, which is used to replay recorded data in simulated time.

The Session object may also have Browser objects having the following interfaces:

- IType is a browser that is used to find meta-data about the Node data objects implemented by the TSDA server.
- IProperty is a browser that is used to find meta-data about the Item data implemented by the TSDA server.
- INode browser, which is used to find Node data objects that have been instantiated within the TSDA server.
- IItem browser, which is used to find Items that exist for an Node object.
- IItemAttribute browser, which is used to find the ItemAttributeDefinitions describing the ItemAttributeValues time series data.
- IAggregate browser that is used to get the AggregateDefintions that exists.

The CallBack object implements the ICallBack interface and is implemented by the Client.. Each Session object may have an associated Callback object. A Client may create any number of Session and associated Callback objects.

A typical interaction sequence between a DA Server and a Client is shown in Figure 6.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

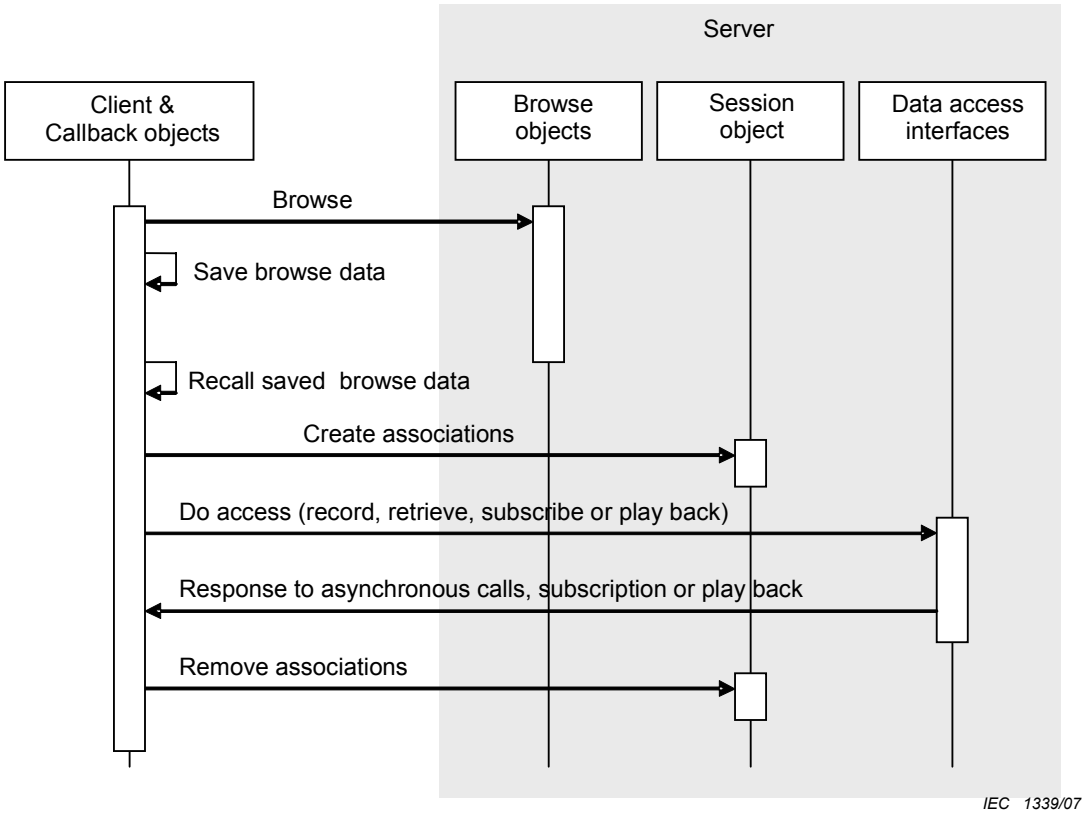

Figure 6 – Typical Interaction between the TSDA objects

A Client will typically start to browse the server to find out what time series data is available. The browe interfaces (IType, IProperty, INode, IItem, IItemAttribute and IAggregate) are used for this. The Client selects a subset of the Items found when browsing and saves them for later use. The Client that does the browse can be an editor for display building, dialog building or data base generation.

At a later time, for example, at picture call up, the saved Items will be recalled and used by the Client to populate the Session object. When a Session object has been created, it can be used to read, write or subscribe for data. A subscription or asynchronous read or write call results in a callback. For a subscription, the call backs will continue until the Client terminates them.

4.5.2 Server and Session interfaces

The IServer interface has the following attributes and methods:

- A read only attribute describing the Server status, for example, server health, start-time, current time, vendor information etc.
- create_historical_data_access_session(), which is used to create a Session object.
- create_historical_data_access_session_for_view(), which is used to create a Session object. This method can be used if a server supports multiple hierarchies. Each hierarchy then corresponds to a view.
- find_views(), which returns the views that are supported by the server.
- A read only attribute, which gives the functions supported by the interface.
- create_historical_data_access_session()
- A read only attribute, which gives the maximum number of time series values (ItemValues) that the server may return.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

- A read only attribute Server status, which gives the status of the Server, for example, vendor info, time when started, current time etc.
- A read only attribute, which gives the functions supported by the interface.

The Session object implements the ITSDASession interface that has the following attributes and methods:

- A read only status attribute that gives the status of the Session, for example, name, time when started, current time and number of groups.
- An attribute holding the optional ShutDown object.
- create_browser(), which is used to create a Browse object.

4.5.3 Management interfaces

The Session object also implements the IConnection interface that has the following attributes and methods:

- An attribute having the Callback object created by the Client.
- create(), which is a method used to populate the Session with Items.
- remove(), which is a method used to remove Items from the Session.
- validate(), which is a method used to validate if Items exists.
- cancel(), which is a method to cancel ongoing asynchronous calls.

4.5.4 Browse interfaces

The Browse object implements the following attributes and interfaces:

- An attribute browse_base_time, which is used by the client to set the base time for the browse. The browser use this time to locate objects that are was defined at this time. This is useful to retrieve objects deleted objects that existed in past times. This also implies that a server shall also keep the history for deleted objects. If no base time is provided, current time is used.
- IType browser.
- IProperty browser.
- INode browser.
- IItem browser.
- IItemAttribute browser.
- IAggregate browser.

The INode and IItem interfaces have the following methods:

- find(), which returns more information about one Node or Item specified by it's id's.
- find_each(), which returns more information about a number of Nodes or Items specified by their id's.
- find_by_parent(), which returns all children for a parent with a specified id.
- find_by_type() which, recursively returns all children with a given Type.id for a parent with a specified id.
- get_pathnames(), which translates a number of pathnames to the corresponding id's.
- get_ids(), which translates a number of id's to pathnames.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

The IType interface has the following methods:

- find(), which returns more information about a Type specified by its id.
- find_by_schema(), which returns the Type.id's for all Types that belong to a schema specified by a given id.

The IProperty interface has the following methods:

- find(), which returns more information about a number of Properties specified by their id's.
- find_by_node(), which returns all Properties for a Node with a specified id.
- find_by_type(), which returns all Properties for a Type with a specified id.

The IItemAttribute and IAggregate interfaces has the following methods and data:

- find(), which returns more information about an object specified by its id.
- find_all(), which returns information about all objects.
- A definition of all object identifications (ItemAttribute and Aggregate IDs) that exist.

4.5.5 IO interfaces

The IO interfaces supports the following functions:

- Synchronous read and write.
- Asynchronous read and write.
- Subscription.
- Play back.

The IO interfaces are:

- IValueIO.
- IModifiedValueIO.
- IItemAttributeIO.
- IAnnotationIO.
- IPlayback.

The IValueIO synchronous interface has the following methods:

- sync_read_raw(), which is used to synchronously read data as recorded or updated in a specified time interval.
- sync_read_processed(), which is used to synchronously read data using an Aggregate calculation in a specified time interval.
- sync_read_at_time(), which is used to synchronously read data at a number of specified times.
- sync_insert(), which is used to synchronously insert new time series data values (i.e. ItemValues).
- sync_replace(), which is used to synchronously replace existing time series data values (i.e. ItemValues).
- sync_insert_replace(), which is used to synchronously replace existing time series data values (i.e. ItemValues) or if a value does not exist insert it as new.
- sync_delete_raw(), which is used to synchronously delete recorded data in a specified time interval.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

- sync_delete_at_time(), which is used to synchronously delete recorded data at specified times.

The IValueIO asynchronous interface has the following methods:

- async_read_raw(), which is used to asynchronously read data as recorded in a specified time interval. The callback interface used with this method is on_read_complete().
- async_read_processed(), which is used to asynchronously read data using an Aggregate calculation in a specified time interval.
- async_read_at_time(), which is used to asynchronously read data at a number of specified times. The callback interface used with this method is on_read_complete().
- async_insert(), which is used to asynchronously insert new time series data values (i.e. ItemValues). The callback interface used with this method is on_update_complete().
- async_replace(), which is used to asynchronously replace existing time series data values (i.e. ItemValues). The callback interface used with this method is on_update_complete().
- async_insert_replace(), which is used to asynchronously replace existing time series data values (i.e. ItemValues) or if a value does not exist insert it as new. The callback interface used with this method is on_update_complete().
- async_delete_raw(), which is used to asynchronously delete recorded data in a specified time interval. The callback interface used with this method is on_data_change().
- async_delete_at_time(), which is used to asynchronously delete recorded data at specified times. The callback interface used with this method is on_update_complete().

The IValueIO subscription interface has the following methods:

- subscribe_raw(), which is used to read available recorded raw data and future data that has not yet been recorded. Available data is delivered immediately, while new data is delivered when recorded. The callback interface used with this method is on_data_change().
- subscribe_processed(), which is used to read available recorded data using an aggregate calculation and future data that has not yet been recorded. Available data is delivered immediately, while new data is delivered when recorded. The callback interface used with this method is on_data_change().

The IModifiedValueIO synchronous interface has the following method:

- sync_read_modified(), which is used to synchronously read data as recorded in a specified time interval. Existing updates (i.e. ModifiedItemValues) are ignored.

The IModifiedValueIO asynchronous interface has the following method:

- async_read_modified(), which is used to asynchronously read data as recorded in a specified time interval. Existing updates (i.e. ModifiedItemValues) are ignored. The callback interface used with this method is on_read_modified_complete().

The ItemAttributeIO synchronous interface has the following method:

- sync_read_attribute(), which is used to synchronously read ItemAttributeValues in a specified time interval.

The ItemAttributeIO asynchronous interface has the following method:

- async_read_attribute(), which is used to asynchronously read ItemAttributeValues in a specified time interval.

The IAnnotationIO synchronous interface has the following methods:

- sync_read(), which is used to synchronously read annotations in a specified time interval.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

- sync_insert(), which is used to synchronously insert annotations in at specified ItemValues.

The IAnnotationIO asynchronous interface has the following methods:

- async_read(), which is used to asynchronously read annotations in a specified time interval. The callback interface used with this method is on_read_annotation_complete().
- async_insert(), which is used to asynchronously insert annotations in at specified ItemValues. The callback interface used with this method is on_insert_ annotation_complete().

The IPlayback asynchronous interface has the following methods:

- playback_raw_with_update(), which is used to asynchronously play back data as recorded from the specified time interval. The callback interface used with this method is on_playback().
- playback_processed_with_update(), which is used to asynchronously play back data using an Aggregate calculation from the specified time interval. The callback interface used with this method is on_playback().

4.5.6 Client interfaces

To support the asynchronous calls and subscriptions, the client shall implement the Callback object that has the following methods:

- on_data_change(), which is used to receive subscription callbacks by the Client.
- on_read_complete(), which is used to receive asynchronous read responses.
- on_update_complete(), which is used to receive asynchronous insert, update or delete responses.
- on_read_modified_complete(), which is used to receive asynchronous read responses for non modified values.
- on_read_annotation_complete(), which is used to receive asynchronous read responses for annotations.
- on_insert_annotation_complete(), which is used to receive asynchronous responses for inserted annotations.
- on_playback(), which is used to receive play back data.

4.6 Mapping of TSDA

The TSDA interfaces may convey data originating from an IEC 61970-301 compliant data source. For an IEC 61970-301 compliant server supporting TSDA, the interfaces shall be used as follows:

- The INode brows interface exposes objects that have time series information. The Measurement classes in IEC 61970-301 typically define such objects, for example Analog, AnalogValue etc. The objects may be hierarchically organized in a containment structure. If so, this is the PhysicalView as defined in IEC 61970-402. The actual hierarchy is transparent to the INode browse interface and other hierarchical structures can be exposed.
- The IItem brows interface exposes property time series values that are present within the TSDA server. Such properties may be defined by the classes in IEC 61970-301, for example, Analog.normalValue=”500”, AnalogValue.value="450" etc.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

- The IType brows interface expose the types of time series data that a TSDA server supports, for example planned production values, minute values, hourly values etc. IEC 61970-301 does not currently contain any definitions for this kind of data.
- The IProperty brows interface exposes the class attributes (meta data) defined for a Type, i.e. the properties that exist in the TSDA server for a particular Type. Such properties may be defined by the classes in IEC 61970-301, for example, Analog.normalValue, AnalogValue.value etc.

_____

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
P.O. Box 131
CH-1211 Geneva 20
Switzerland

Tel: + 41 22 919 02 11
Fax: + 41 22 919 03 00
info@iec.ch
www.iec.ch

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.