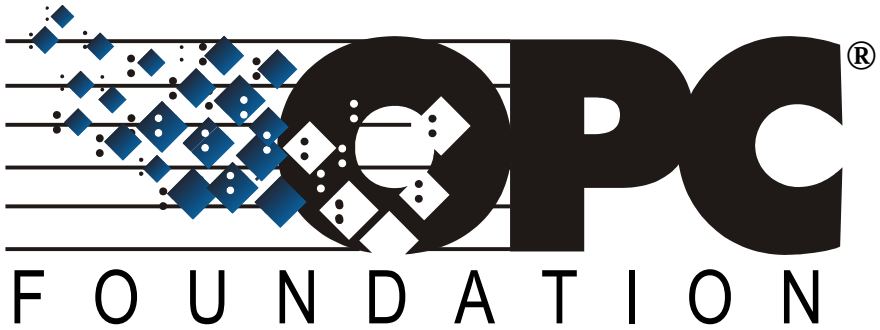

OPC 10000-7

OPC Unified Architecture

Part 7: Profiles

Release 1.04

2017-11-01

OPC UA Specification

Specification Type:	Industry Standard Specification	Comments:	Report or view errata: http://www.opcfoundation.org/errata
Document Number	**OPC 10000-7**		
Title:	OPC Unified Architecture Part 7 :Profiles	Date:	2017-11-01
Version:	Release 1.04	Software:	MS-Word
		Source:	OPC 10000-7 - UA Specification Part 7 - Profiles 1.04.docx
Author:	OPC Foundation	Status:	Release

CONTENTS

FIGURES

TABLES

OPC FOUNDATION

UNIFIED ARCHITECTURE –

FOREWORD

This specification is the specification for developers of OPC UA applications. The specification is a result of an analysis and design process to develop a standard interface to facilitate the development of applications by multiple vendors that shall inter-operate seamlessly together.

AGREEMENT OF USE

GENERAL PROVISIONS

Should any provision of this Agreement be held to be void, invalid, unenforceable or illegal by a court, the validity and enforceability of the other provisions shall not be affected thereby.

This Agreement shall be governed by and construed under the laws of the State of Minnesota, excluding its choice or law rules.

This Agreement embodies the entire understanding between the parties with respect to, and supersedes any prior understanding or agreement (oral or written) relating to, this specification.

ISSUE REPORTING

The OPC Foundation strives to maintain the highest quality standards for its published specifications; hence they undergo constant review and refinement. Readers are encouraged to report any issues and view any existing errata here: http://www.opcfoundation.org/errata

Revision 1.04 Highlights

The following table includes the Mantis issues resolved with this revision.

Mantis ID	Summary	Resolution
3628	New Transport *Profiles*:WSS and HTTPS/JSON	Created encoding units for JSON, UA Binary and XML. Created transport profiles for https/json, wss/binary and wss/json.
3603	Under specified Security Policy Basic256Sha256	Added following explanatory text: “uses PKCS#1 v1.5 padding” to AsymmetricSignatureAlgorithm “uses Sha1 for padding” to AsymmetricEncryptionAlgorithm
3442	No Push model in client profile	Changed global certification facets so that client only includes “pull” and server only includes “push”.
3634	Add profile for JSON web token	JSON web token created for server and client.
3606	Add CU for MultiStateValueDiscreteType	Added CU for MultiStateValueDiscrete type to DataAccess server and client facet.
3644	Require handling of repeated invalid username/pwd	Created new CU and added it to all user tokens
3369	Durable Subscriptions: Determining reasonable queue sizes / timeouts	Updated the facet for durable subscriptions to include multiple storage levels where support of one of them is required.
3233	New facets needed for structures with the new DataTypeDefinition attribute	Created 2017 version for ComplexType *Server* Facet which requires the DataTypeDefinition *Attribute* Created ComplexType read and write facet for client (was an optional CU in read/write)
3347	Need CU/*Profile* for ResendData	Created 2017 version for the “Standard DataChange *Subscription Server* Facet” where GetMonitoredItems and ResendData are mandatory. This required a 2017 version of "Enhanced DataChange *Subscription Server* Facet". Also added to "DataChange Subscriber *Client* Facet**"** as optional CUs.
3650	Estimated return time needs CU	EstimatedReturnTime CUs created and added (optional) to • Core 2017 *Server* Facet • Core *Client* Facet
3673	Need CUs for atomicity	Atomicity CUs created and added to • Core 2017 *Server* Facet (mandatory) • Read/Write *Attribute Client* Facet (optional)
3674	Need CUs for “full array only”	FullArrayOnly CUs created and added to • Core 2017 *Server* Facet (mandatory) • Read/Write *Attribute* + DataChange Subscriber *Client* Facet (optional)
2382	CUs and Facets for state machine	Created *Server* and *Client* facets for state machines.
3640	*Profiles* for user authorization	Created “User Role Management” *Client* and *Server* profiles.
3646	Profiles for sessionless invoke	Created server and client facets. The server facet requires support of a GetEndpoint filter to request only endpoints that support sessionless invocation.
3645	Profiles for server-initiated connections	Added two facets: • Reverse Connect Server Facet • Reverse Connect Client Facet
3748	Add CUs for Available States and Available Transistions.	Created CUs for Client and Server and added them to StateMachine facets.
3759	Alarming: Silencing	Added conformance units for silencing to the Alarm facets.
3763	Alarming: Out Of Service	Added conformance units for out of service state to the Alarm facets.
3771	Alarming: Suppressed State	Added conformance units for suppressed state to the Alarm facets.

Mantis ID	Summary	Resolution
		Separate CU for the Suppress and Unsuppress Methods.
3761	Add Discrepancy Alarm type	Added conformance units for discrepancy Alarm to the Alarm facets.
3760	Alarm Metrics Profiles	Added new facets for Client and Server.
3764	Alarm properties for IEC 62682	Added CUs for: OnDelay and OffDelay ReAlarmTime, ReAlarmRepeatCount FirstInGroup AudibleSound ConditionSubClass
3817	SelectionListType missing	Created CUs for this new Variable Type and inserted them as optional to the Core Facets
3791	SHA1 broken	Deprecated Base128Rsa15 and Base256. Created new security policies.
3769	Remove specific security policies.	Base128Rsa15 and Base256 are now deprecated. In addition, all profiles that explicitely referenced security policies have been updated. They do not reference a security policy but rather require SecurityPolicy [A] and [B].
3756	GDS QueryApplications	Added new optional CUs for Global Client Discovery Facet and for the GDS.
3757	GDS Credential Management	Added new facets for Client and Server as well as for the GDS.
3758	GDS Authorization Service	Added new facets for Client and Server as well as for the GDS.

OPC Unified Architecture Specification

Part 7: Profiles

1 Scope

This part describes the OPC Unified Architecture (OPC UA) *Profiles*. The *Profiles* in this document are used to segregate features with regard to testing of OPC UA products and the nature of the testing (tool based or lab based). This includes the testing performed by the OPC Foundation provided OPC UA CTT (a self-test tool) and by the OPC Foundation provided Independent certification test labs. This could equally as well refer to test tools provided by another organization or a test lab provided by another organization. What is important is the concept of automated tool based testing versus lab based testing. The scope of this standard includes defining functionality that can only be tested in an a lab and defining the grouping of functionality that is to be used when testing OPC UA products either in a lab or using automated tools. The definition of actual *TestCases* is not within the scope of this document, but the general categories of TestCases are within the scope of this document.

Most OPC UA applications will conform to several, but not all of the *Profiles*.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments and errata) applies.

OPC 10000-1*, OPC Unified Architecture - Part 1: Overview and Concepts*
http://www.opcfoundation.org/UA/Part1/

OPC 10000-2, *OPC Unified Architecture - Part 2: Security Model*
http://www.opcfoundation.org/UA/Part2/

OPC 10000-3, *OPC Unified Architecture - Part 3: Address Space Model*
http://www.opcfoundation.org/UA/Part3/

OPC 10000-4, *OPC Unified Architecture - Part 4: Services*
http://www.opcfoundation.org/UA/Part4/

OPC 10000-5, *OPC Unified Architecture - Part 5: Information Model*
http://www.opcfoundation.org/UA/Part5/

OPC 10000-6, *OPC Unified Architecture - Part 6: Mappings*
http://www.opcfoundation.org/UA/Part6/

OPC 10000-8, *OPC Unified Architecture - Part 8: Data Access*
http://www.opcfoundation.org/UA/Part8/

OPC 10000-9, *OPC Unified Architecture - Part 9: Alarms and Conditions*
http://www.opcfoundation.org/UA/Part9/

OPC 10000-10, *OPC Unified Architecture - Part 10: Programs*
http://www.opcfoundation.org/UA/Part10/

OPC 10000-11, *OPC Unified Architecture - Part 11: Historical Access*
http://www.opcfoundation.org/UA/Part11/

OPC 10000-12, *OPC Unified Architecture - Part 12: Discovery and Global Services*

http://www.opcfoundation.org/UA/Part12/

OPC 10000-13, *OPC Unified Architecture - Part 13: Aggregates*

http://www.opcfoundation.org/UA/Part13/

Test Specifications

Compliance Part 8 UA Server, *OPC Test Lab Specification: Part 8 – UA Server*

http://www.opcfoundation.org/Test/Part8/

Compliance Part 9 UA Client, *OPC Test Lab Specification: Part 9 – UA Client*

http://www.opcfoundation.org/Test/Part9/

3 Terms, definitions, and abbreviations

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments and errata) applies.

OPC 10000-1, OPC 10000-2, OPC 10000-3, OPC 10000-4, OPC 10000-6, and OPC 10000-8 as well as the following apply. An overview of the terms defined in this standard and their interaction can be viewed in Figure 1.

3.1.1
application

a software program that executes or implements some aspect of OPC UA

Note 1 to entry: The application could run on any machine and perform any function. The application could be software or it could be a hardware application, the only requirement is that it implements OPC UA.

3.1.2
ConformanceUnit

a specific set of OPC UA features that can be tested as a single entity

Note 1 to entry: A ConformanceUnit can cover a group of services, portions of services or information models.

3.1.3
ConformanceGroup

a group of ConformanceUnits that is given a name

Note 1 to entry: This grouping is only to assist in organizing ConformanceUnits. Typical ConformanceGroups include groups for each of the service sets in OPC UA and each of the Information Model standards.

3.1.4
Facet

a Profile dedicated to a specific feature that a Server or Client may require

Note 1 to entry: Facets are typically combined to form higher-level Profiles. The use of the term Facet in the title of a Profile indicates that the given Profile is not a standalone Profile.

3.1.5
FullFeatured Profile

a Profile that defines all features necessary to build a functional OPC UA Application

Note 1 to entry: A FullFeatured Profile in particular adds definitions of the transport and security requirements.

3.1.6
ProfileCategory

arranges Profiles into application classes, such as Server or Client

Note 1 to entry: These categories help determine the type of Application that a given Profile would be used for. For additional details see 4.4.

3.1.7
TestCase
a technical description of a set of steps required to test a particular function or information model

Note 1 to entry: TestCases provide sufficient details to allow a developer to implement them in code. TestCases also provide a detailed summary of the expected result(s) from the execution of the implemented code and any precondition(s) that must be established before the TestCase can be executed.

3.1.8
TestLab
a facility that is designated to provide testing services

Note 1 to entry: These services include but are not limited to personal that directly perform testing, automated testing and a formal repeatable process. The OPC Foundation has provided detailed standard describing OPC UA TestLabs and the testing they are to provided (see Compliance Part 8 UA Server, Compliance Part 9 UA Client).

3.2 Abbreviations

DA	Data Access
HA	Historical Access
HMI	Human Machine Interface
NIST	National Institute of Standard and Technology
PKI	Public Key Infrastructure
RSA	Rivest-Shamir-Adleman
UA	Unified Architecture

4 Overview

4.1 General

The OPC Unified architecture multipart standard describes a number of *Services* and a variety of information models. These *Services* and information models can be referred to as features of a *Server* or *Client*. *Servers* and *Clients* need to be able to describe which features they support and wish to have certified. This document provides a grouping of these features. The individual features are grouped into *ConformanceUnits* which are further grouped into *Profiles*. Figure 1 provides an overview of the interactions between *Profiles*, *ConformanceUnits* and *TestCases*. The large arrows indicate the components that are used to construct the parent. For example a *Profile* is constructed from *Profiles* and *ConformanceUnits*. The figure also illustrates a feature of the OPC UA Compliance Test Tool (CTT), in that it will test if a requested *Profile* passes all *ConformanceUnits*. It will also test all other *ConformanceUnits* and report any other *Profiles* that pass conformance testing. The individual *TestCases* are defined in separate documents see Compliance Part 8 UA Server and Compliance Part 9 UA Client. The *TestCases* are related back to the appropriate *ConformanceUnits* defined in this standard. This relationship is also displayed by the OPC UA Compliance Test Tool.

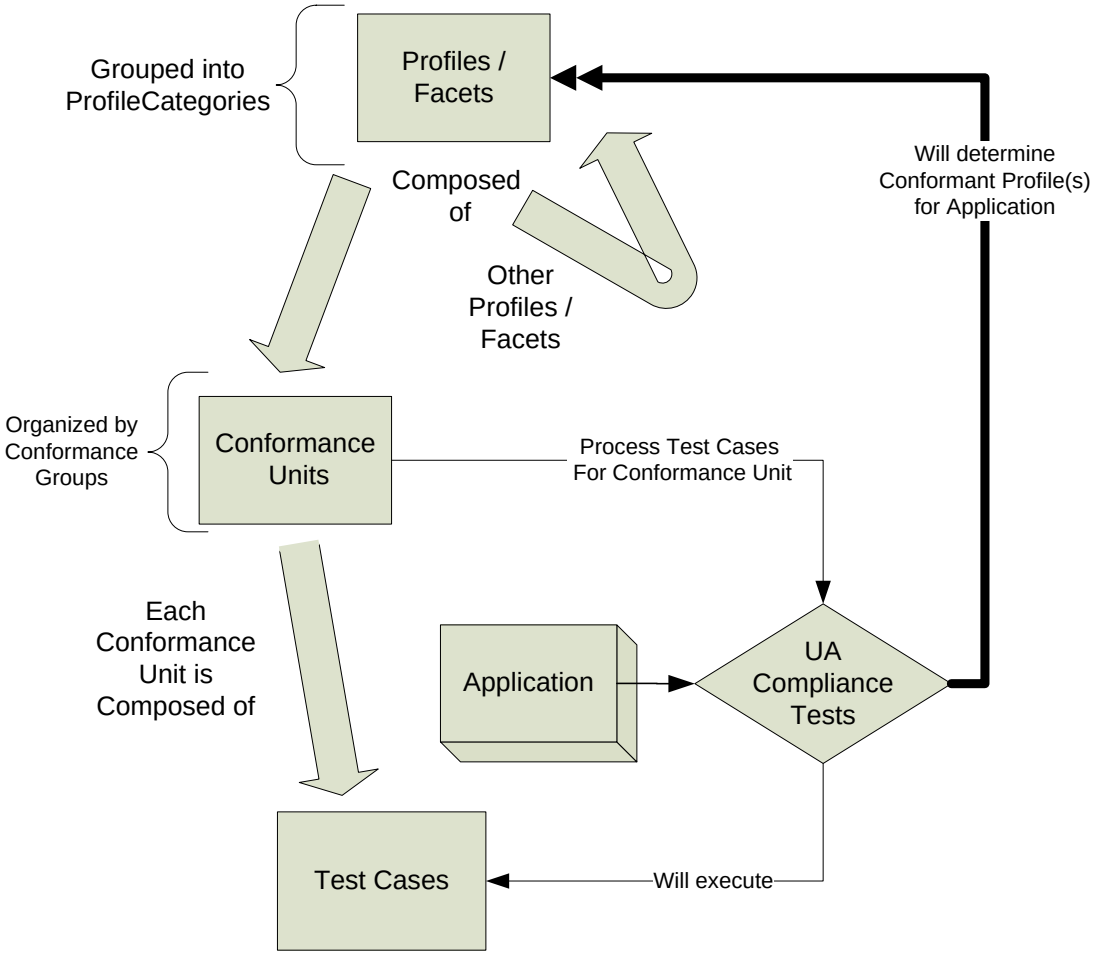

Figure 1 – Profile – ConformanceUnit – TestCases

4.2 ConformanceUnit

Each *ConformanceUnit* represents a specific set of features (e.g. a group of services, portions of services or information models) that can be tested as a single entity. *ConformanceUnits* are the building blocks of a *Profile*. Each *ConformanceUnit* can also be used as a test category. For each *ConformanceUnit,* there would be a number of TestCases that test the functionality described by the *ConformanceUnit.* The description of a *ConformanceUnit* is intended to provide enough information to illustrate the required functionality, but in many cases to obtain a complete understanding of the *ConformanceUnit* the reader may be required to also examine the appropriate part of OPC UA. Additional Information regarding testing of a *ConformanceUnit* are provided in the Compliance Part 8 UA Server or Compliance Part 9 UA Client test standards.

The same features do not appear in more than one *ConformanceUnit*.

4.3 Profiles

A *Profile* is a named aggregation of *ConformanceUnits* and other *Profiles*. To support a *Profile*, an application has to support the *ConformanceUnits* and all aggregated *Profiles*. The definition of *Profiles* is an ongoing activity, in that it is expected that new *Profiles* will be added in the future.

An OPC UA Application will typically support multiple *Profiles*.

Multiple *Profiles* may include the same *ConformanceUnit*.

Testing of a *Profile* consists of testing the individual *ConformanceUnits* that comprise the *Profile*.

Profiles are named based on naming conventions (see 6.3 for details).

4.4 Profile Categories

Profiles are grouped into categories to help vendors and end users understand the applicability of a *Profile*. A *Profile* can be assigned to more than one category.

Table 1 – Profile Categories contains the list of currently defined *ProfileCategories*.

Table 1 – Profile Categories

Category	Description
Client	*Profiles* of this category specify functions of an OPC UA *Client*.
Global Directory *Service*	*Profiles* of this category specify functions for global discovery and certificate management.
Security	*Profiles* of this category specify security related functions. Security policies are part of this category. The URI of security policies has to be part of an Endpoint Description returned from the GetEndpoints service. *Profiles* of this category apply to *Clients* and *Servers*.
Server	*Profiles* of this category specify functions of an OPC UA *Server*. The URI of such *Profiles* can be exposed in the *Server* capabilities.
Transport	*Profiles* of this category specify specific protocol mappings. The URI of such *Profiles* has to be part of an Endpoint Description. These *Profiles* apply to *Clients* and *Servers*.

5 Conformance Units

5.1 Overview

A *ConformanceUnit* represents an individually testable entity. For improved clarity, the large list of *ConformanceUnits* is arranged into named *ConformanceGroups*. These groups reflect the *Service Sets* in OPC 10000-4 and the OPC UA information models. Table 2 lists the *ConformanceGroups*. These groups and the *ConformanceUnits* that they describe are detailed in the Subclauses of chapter 5 starting with clause 5.2. *ConformanceGroups* have no impact on testing; they are used only for organizational reasons, i.e. to simplify the readability of this document.

Table 2 – Conformance Groups

Group	Description
Address Space Model	Defines *ConformanceUnits* for various features of the OPC UA *AddressSpace*.
Aggregates	All *ConformanceUnits* that are related to Aggregates, including individual *ConformanceUnits* for each supported Aggregate as described in Part 13.
Alarms and *Conditions*	All *ConformanceUnits* that are associated with the OPC UA Information Model for *Conditions*, acknowledgeable *Conditions*, confirmations and *Alarms* as specified in Part 9.
Attribute Services	Includes *ConformanceUnits* to read or write current or historical *Attribute* values.
Auditing	User level security includes support for security audit trails, with traceability between *Client* and *Server* audit logs.
Base Information	All information elements as defined in Part 5.
Data Access	*ConformanceUnits* specific to *Clients* and *Servers* that deal with the representation and use of automation data as specified in Part 8.
Discovery Services	*ConformanceUnits* which focus on *Server* Endpoint *Discovery*.
GDS	Conformance Units for a GDS. Includes units for global discovery and global certificate management.
Historical Access	Access to archived data of *Node Attribute* values or Events.

Group	Description
Method Services	Methods represent the function calls of *Objects*. Methods are invoked and return only after completion (successful or unsuccessful).
Miscellaneous	This group contains *ConformanceUnits* that cover miscellaneous subjects, such as recommended behaviours, documentation etc. These *ConformanceUnits* typically do not fit into any of the other groups.
Monitored Item *Services*	*Clients* define *MonitoredItems* to subscribe to data and Events. Each *MonitoredItem* identifies the item to be monitored and the *Subscription* to use to send *Notifications*.
Node Management *Services*	Bundles *ConformanceUnits* for all *Services* to add and delete OPC UA *AddressSpace Nodes* and *References*.
Protocol and Encoding	Covers all transport and encoding combinations that are specified in Part 6.
Redundancy	The design of OPC UA ensures that vendors can create redundant *Clients* and redundant *Servers* in a consistent manner. Redundancy may be used for high availability, fault tolerance and load balancing.
Security	Security related *ConformanceUnits* that can be profiled this includes all aspects of security.
Session Services	An (OPC UA) *Session* is an application layer connection.
Subscription Services	Subscriptions are used to report *Notifications* to the *Client*.
View *Services*	*Clients* use the View *Service* Set to navigate through the OPC UA *AddressSpace* or through a View (a subset) of the OPC UA *AddressSpace*.

5.2 Services

The following tables describe *ConformanceUnits* for the *Services* specified in OPC 10000-4. The tables correlate with the *Service Sets.*

A single *ConformanceUnit* can reference several *Services* (e.g. CreateSession, ActivateSession and CloseSession) but can also refer to individual aspects of *Services* (e.g. the use of ActivateSession to impersonate a new user).

Each table includes a listing of the *Profile Category* to which a *ConformanceUnit* belongs, the title and description of the *ConformanceUnit*. In some cases, a *ConformanceUnit* will be derived from another *ConformanceUnit*. This parent unit will then be specified in the description of each derived unit. In such cases the derived units inherit all of the tests of its parent plus one or more additional TestCases. These TestCases can only further restrict the existing TestCases. An example would be one in which the number of connections is tested, where the TestCase of the parent required at least one connection and the derived *ConformanceUnit* would require a *TestCase* for at least five connections.

The *Discovery Service* Set is composed of multiple *ConformanceUnits* (see Table 3). All *Servers* provide some aspects of this functionality; see *Profiles* categorized as *Server Profiles* for details. *Clients* may support some aspects of this functionality; see *Profiles* categorized as *Client Profiles* for details.

Table 3 – Discovery Services

Category	Title	Description
Server	*Discovery* Get Endpoints	Support the GetEndpoints *Service* to obtain all Endpoints of the *Server*. This includes filtering based on *Profiles*.
Server	*Discovery* Get Endpoints SessionLess	Support at least one endpoint for issueing SessionLess Services. Support obtaining such endpoints by accepting the Transport URI as a filter to the GetEndpoints Service with the query string "SL" appended to the Transport URI. E.g. "http://opcfoundation.org/UA-Profile/Transport/https-uajson?SL"
Server	*Discovery* Find *Servers* Self	Support the FindServers *Service* only for itself.

Category	Title	Description
Server	*Discovery* Register	Call the RegisterServer *Service* to register itself (OPC UA *Server*) with an external *Discovery Service* via a secure channel with a SecurityMode other than NONE.
Server	*Discovery* Register2	Call the RegisterServer2 *Service* to register with an external *Discovery Service* via a Secure Channel with a SecurityMode other than "None". This includes passing a list of short capability identifiers. The identifiers and their use are specified in Part 12.
Server	*Discovery Server* Announcement using mDNS	Provide mDNS functionality to announce a *Server* with its capabilities. The capability identifiers and the use of mDNS records for the purpose of OPC UA *Discovery* is specified in Part 12. Note that this functionality is only required for *Servers* that do not register with an LDS. The capability identifiers and their use in mDNS records are specified in Part 12.
Server	*Discovery* Configuration	Allow configuration of the *Discovery Server* URL where the *Server* will register itself. Allow complete disabling of registration with a *Discovery Server*.
Client	*Discovery Client* Find *Servers* Basic	Uses the FindServers *Service* to obtain all *Servers* installed on a given platform.
Client	*Discovery Client* Find *Servers* with URI	Use FindServers *Service* to obtain URLs for specific *Server* URIs.
Client	*Discovery Client* Find *Servers* Dynamic	Detect new *Servers* after an initial FindServers *Service* call.
Client	*Discovery Client* Find *Servers* on Network	Support one of the options to locate *Servers* on the network.
Client	*Discovery Client* Find *Servers* on Network using LDS-ME	Use FindServersOnNetwork *Service* to obtain URLs for specific *Server* URIs. Note that this *Service* is available via the Local *Discovery Server* with multicast extension (LDS-ME).
Client	*Discovery Client* Find *Servers* on Network using mDNS	Use mDNS based *Service Discovery* to locate *Servers* on the same multicast network. The contents of mDNS records for OPC UA *Discovery* are described in Part 12. Note that this functionality is only required for *Clients* when there is no Local *Discovery Server* with multicast extension (LDS-ME). The capability identifiers and their use in mDNS records are specified in Part 12.
Client	*Discovery Client* Find *Servers* in GDS	Use the QueryServers *Method* on the GDS Directory *Object* to locate *Servers* that meet filter criteria specified in the request. This *Method* is specified in Part 12.
Client	*Discovery Client* Find Applications in GDS	Use the QueryApplications *Method* on the GDS Directory *Object* to locate Applications that meet filter criteria specified in the request. This *Method* is specified in Part 12.
Client	*Discovery Client* Get Endpoints Basic	Uses the GetEndpoints *Service* to obtain all Endpoints for a given *Server* URI.

Category	Title	Description
Client	*Discovery Client* Get Endpoints SessionLess	Uses the GetEndpoints *Service* with a filter to obtain Endpoints that can be used for SessionLess *Service* invocation. The filter is the Transport URI extended with the query string "SL". E.g. http://opcfoundation.org/UA-*Profile*/Transport/https-uajson?SL.
Client	*Discovery Client* Get Endpoints Dynamic	Detect changes to the Endpoints after an initial GetEndpoints *Service* call.
Client	*Discovery Client* Configure Endpoint	Allow specification of an Endpoint without going through the *Discovery Service* Set.

The *Session Service* Set is composed of multiple *ConformanceUnits* (see Table 4). The CreateSession, ActivateSession, and CloseSession services are supported as a single unit. All *Servers* and *Clients* provide this functionality.

Table 4 – Session Services

Category	Title	Description
Server	*Session* General *Service* Behaviour	Implement basic *Service* behaviour. This includes in particular: – checking the authentication token – returning the requestHandle in responses – returning available diagnostic information as requested with the 'returnDiagnostics' parameter – respecting a timeoutHint
Server	*Session* Base	Support the *Session Service* Set (CreateSession, ActivateSession, CloseSession) except the use of ActivateSession to change the *Session* user. This includes correct handling of all parameters that are provided. Note that for the CreateSession and ActivateSession services, if the SecurityMode = None then: 1) The Application *Certificate* and Nonce are optional. 2) The signatures are null/empty. The details of this are described in Part 4.
Server	*Session* Change User	Support the use of ActivateSession to change the *Session* user.
Server	*Session* Cancel	Support the Cancel *Service* to cancel outstanding requests.
Server	*Session* Minimum 1	Support minimum 1 *Session* (total).
Server	*Session* Minimum 2 Parallel	Support minimum 2 parallel Sessions (total for all *Clients*).
Server	*Session* Minimum 50 Parallel	Support minimum 50 parallel Sessions (total for all *Clients*).
Server	Session Sessionless Invocation	Defines the support of the SessionlessInvoke Service defined in UA Part 4 to process any of the Services (like Read/Write, Browse, or Call) that are designated for Session-less invocation.
Client	*Session Client* General *Service* Behaviour	Implement basic *Service* behaviour. This includes in particular: – including the proper authentication token of the *Session* – creating a requestHandle if needed – requesting diagnostic information with the 'returnDiagnostics' parameter – evaluate the serviceResult and operational results
Client	*Session Client* Base	Use the *Session Service* Set (CreateSession, ActivateSession, and CloseSession) except the use of ActivateSession to change the *Session* user. This includes correct handling of all parameters that are provided Note that for the CreateSession and ActivateSession services, if the SecurityMode = None then: 1) The Application *Certificate* and Nonce are optional. 2) The signatures are null/empty.

Category	Title	Description
Client	*Session Client* Multiple Connections	Support unlimited connections (client side) with multiple *Servers*. Any limit on numbers of connections is from server side. May have a memory based limit, but not a software constraint limit.
Client	*Session Client* Renew NodeIds	This *ConformanceUnit* applies to *Clients* that allow persisting NodeIds. Verify that the Namespace Table has not changed for NodeIds that the *Client* has persisted and is going to re-use beyond a *Session* lifetime. If changes occurred the *Client* has to recalculate the Namespace Indices of the respective NodeIds.
Client	*Session Client* Impersonate	Uses ActivateSession to change the Session user (impersonation).
Client	*Session Client* KeepAlive	Make periodic requests to keep the *Session* alive.
Client	*Session Client* Detect Shutdown	Read or monitor the ServerStatus/State Variable to recognize a potential shutdown of the Server and clean up resources.
Client	*Session Client* Cancel	Use the Cancel *Service* to cancel outstanding requests.
Client	*Session Client* Auto Reconnect	Automatic *Client* reconnect including: – ActivateSession with new SecureChannel if SecureChannel is no longer valid but *Session* is still valid – Creation of a new *Session* only if *Session* is no longer valid
Client	*Session Client* Single *Session*	The *Client* shall interoperate with *Servers* that only support one *Session*.
Client	*Session Client* SessionLess *Service* Calls	Defines the use of the SessionlessInvoke *Service* defined in UA Part 4 to request one of the *Services* (like Read or Browse) that are allowed for sessionless invocation. UA Part 6 specifies which transports may be used and how.

The *Node* Management *Service* Set is composed of multiple *ConformanceUnits* (see Table 5). *Servers* may provide some aspects of this functionality; see *Profiles* categorized as *Server Profiles* for details. *Clients* may support some aspects of this functionality; see *Profiles* categorized as *Client Profiles* for details.

Table 5 – Node Management Services

Category	Title	Description
Server	*Node* Management Add *Node*	Support the AddNodes *Service* to add one or more *Nodes* into the OPC UA *AddressSpace*.
Server	*Node* Management Delete *Node*	Support the DeleteNodes *Service* to delete one or more *Nodes* from the OPC UA *AddressSpace*.
Server	*Node* Management Add Ref	Support the AddReferences *Service* to add one or more *References* to one or more *Nodes* in the OPC UA *AddressSpace*.
Server	*Node* Management Delete Ref	Support the DeleteReferences *Service* to delete one or more *References* of a *Node* in the OPC UA *AddressSpace*.
Client	*Node* Management *Client*	Uses *Node* Management *Services* to add or delete *Nodes* and to add or delete *References* in *Server*'s OPC UA *AddressSpace*.

The View *Service* Set is composed of a multiple *ConformanceUnits* (see Table 6). All *Servers* support some aspects of this conformance group. *Clients* may support some aspects of this functionality; see *Profiles* categorized as *Client Profiles* for details.

Table 6 – View Services

Category	Title	Description
Server	View Basic	Support the View *Service* Set (Browse, BrowseNext).
Server	View TranslateBrowsePath	Support TranslateBrowsePathsToNodeIds *Service*.
Server	View RegisterNodes	Support the RegisterNodes and UnregisterNodes *Services* as a way to optimize access to repeatedly used *Nodes* in the *Server*'s OPC UA *AddressSpace*.

Category	Title	Description
Server	View Minimum Continuation Point 01	Support minimum 1 continuation point per *Session*.
Server	View Minimum Continuation Point 05	Support minimum 5 continuation points per *Session*. This number has to be supported for at least half of the minimum required sessions.
Client	View *Client* Basic Browse	Uses Browse and BrowseNext *Services* to navigate through the *Server*'s OPC UA *AddressSpace*. Make use of the referenceTypeId and the nodeClassMask to specify the needed *References*.
Client	View *Client* Remote *Nodes* Browse	The *Client* can browse to nodes that have an extended NodeID that reference a *Server* different than the originating *Server*. This includes automatic connection to the remote *Server*. It is acceptable that the *Server* configuration information be pre-configured on the *Client* and / or that the user is prompted to connect.
Client	View *Client* Basic ResultSet Filtering	Makes use of the resultMask parameter to optimize the result set to be returned by the *Server*.
Client	View *Client* TranslateBrowsePath	Uses the TranslateBrowsePathsToNodeIds *Service* to identify the NodeIds for *Nodes* where a starting *Node* and a BrowsePath is known. Makes use of bulk operations rather than multiple calls whenever possible.
Client	View *Client* Remote *Nodes* Translate Browse	The *Client* can translate browse paths that include nodes with extended NodeID that reference a *Server* different than the originating *Server* and return them as part of the TranslateBrowsePathsToNodeIds *Service*. It is acceptable that the *Server* configuration information be pre-configured on the *Client*.
Client	View *Client* RegisterNodes	Uses the RegisterNodes *Service* to optimize access for *Nodes* that are used repeatedly. Use UnregisterNodes when *Nodes* are not used anymore.

The *Attribute Service* Set is composed of multiple *ConformanceUnits* (see Table 7). The majority of the *Attribute* service set is a core functionality of OPC UA and as such is supported by most *Servers*. Most *Clients* will also support some aspects of the *Attribute Service* Set.

Table 7 – Attribute Services

Category	Title	Description
Server	*Attribute* Read	Supports the Read *Service* to read one or more *Attributes* of one or more *Nodes*. This includes support of the IndexRange parameter to read a single element or a range of elements when the *Attribute* value is an array.
Server	*Attribute* Read Complex	Supports reading and encoding Values with structured DataTypes.
Server	*Attribute* Write Values	Supports writing to values to one or more *Attributes* of one or more *Nodes*.
Server	*Attribute* Write Complex	Supports writing and decoding Values with structured DataTypes.
Server	*Attribute* Write StatusCode & Timestamp	Supports writing of StatusCode and Timestamps along with the Value.
Server	*Attribute* Write Index	Supports the IndexRange to write a single element or a range of elements when the *Attribute* value is an array and partial updates is allowed for this array.
Server	*Attribute* Alternate Encoding	Supports alternate Data Encoding when reading value *Attributes*. By default, every *Server* has to support the Data Encoding of the currently used Stack *Profile* (i.e. binary with UA Binary Encoding and XML with XML Encoding). This *ConformanceUnit* - when supported - specifies that the other Data Encoding is supported in addition.
Server	*Attribute* Historical Read	Supports the HistoryRead *Service*. The details of what aspects of this service are used are listed in additional *ConformanceUnits*, but at least one of ReadRaw, ReadProcessed, ReadModified, ReadAtTime or ReadEvents must be supported.

Category	Title	Description
Server	*Attribute* Historical Update	Supports the HistoryUpdate service. The details of the supported features of this service are described by additional *ConformanceUnits*, but at least one of the following must be supported: InsertData, InsertEvents, ReplaceData, ReplaceEvents, UpdateData, UpdateEvents, DeleteData, DeleteEvents or DeleteAtTime.
Client	*Attribute Client* Read Base	Use the Read *Service* to read one or more *Attributes* of one or more *Nodes*. This includes use of an IndexRange to select a single element or a range of elements when the *Attribute* value is an array. *Clients* shall use bulk operations whenever possible to reduce the number of *Service* invocations.
Client	*Attribute Client* Remote *Nodes* *Attribute* Access	The *Client* can retrieve attributes of nodes that have an extended NodeID that reference a *Server* different than the originating *Server*. This requires a connection to the remote *Server* for access (not necessarily displayed as a connection). It is acceptable that the *Server* configuration information be pre-configured on the *Client*.
Client	*Attribute Client* Read with proper Encoding	This *ConformanceUnit* refers to the ability of a *Client* to discover the available encodings and choose a specific one when calling the Read *Service*.
Client	*Attribute Client* Read Complex	Read and decode Values with structured DataTypes.
Client	*Attribute Client* Write Base	Use the Write *Service* to write values to one or more *Attributes* of one or more *Nodes*. This includes use of an IndexRange to select a single element or a range of elements when the *Attribute* value is an array. *Clients* shall use bulk operations whenever possible to reduce the number of *Service* invocations.
Client	*Attribute Client* Write Complex	Write and Encode Values with structured DataTypes.
Client	*Attribute Client* Write Quality & Timestamp	Use the Write *Service* to also write StatusCode and/or Timestamps along with a Value.
Client	*Attribute Client* Historical Read	The *Client* makes use of the HistoryRead service. The details of which aspect of this service are used are provided by additional *ConformanceUnits*, but at least one or more of the following is used ReadRaw, ReadAtTime, ReadProcessed, ReadModified or ReadEvents.
Client	*Attribute Client* Historical Updates	The *Client* makes use of the HistoryUpdate service. The details of this usage are provided by additional *ConformanceUnits*, but at least one or more of the following must be provided InsertData, InsertEvents, ReplaceData, ReplaceEvents, UpdateData, UpdateEvents, DeleteData or DeleteEvents or DeleteAtTime.

The *Method Service* Set is composed of *ConformanceUnits* (see Table 8). The primary *ConformanceUnits* provide support for the call functionality. *Servers* may provide some aspects of this functionality; see *Profiles* categorized as *Server Profiles* for details. *Clients* may support some aspects of this functionality; see *Profiles* categorized as *Client Profiles* for details.

Table 8 – Method Services

Category	Title	Description
Server	*Method* Call	Support the Call *Service* to call (invoke) a *Method* which includes support for *Method* Parameters.
Client	*Method Client* Call	Use the Call *Service* to call one or several Methods.

The *MonitoredItem Service* Set is composed of multiple *ConformanceUnits* (see Table 9). *Servers* may provide some aspects of this functionality; see *Profiles* categorized as *Server Profiles* for details. *Clients* may support some aspects of this functionality; see *Profiles* categorized as *Client Profiles* for details.

Table 9 – Monitored Item Services

Category	Title	Description
Server	Monitor Basic	Support the following *MonitoredItem Services*: CreateMonitoredItems, ModifyMonitoredItems, DeleteMonitoredItems and SetMonitoringMode.
Server	Monitor Value Change	Support creation of *MonitoredItems* for *Attribute* value changes. This includes support of the IndexRange to select a single element or a range of elements when the *Attribute* value is an array.
Server	Monitor Complex Value	Supports monitoring and encoding Values with structured DataTypes.
Server	Monitored Items Deadband Filter	Supports an absolute Deadband filter as a DataChangeFilter for numeric data types.
Server	Monitor Aggregate Filter	Support for Aggregate filters for *MonitoredItems*. The result of this *ConformanceUnit* includes a list of Aggregates that are supported as part of the *Profile Certificate*.
Server	Monitor Alternate Encoding	Support alternate encoding when monitoring value *Attributes*. By default, every *Server* has to support the encoding of the currently used Stack *Profile* (i.e. binary with UA Binary Encoding and XML with XML Encoding). This *ConformanceUnit* - when supported - specifies that the other encoding is supported in addition.
Server	Monitor Items 2	Support at least 2 *MonitoredItems* per *Subscription* where the size of each *MonitoredItem* is at least equal to size of Double.
Server	Monitor Items 10	Support at least 10 *MonitoredItems* per *Subscription* where the size of each *MonitoredItem* is at least equal to size of Double.
Server	Monitor Items 100	Support at least 100 *MonitoredItems* per *Subscription*. This number has to be supported for at least half of the required Subscriptions for half of the required Sessions.
Server	Monitor Items 500	Support at least 500 *MonitoredItems* per *Subscription*. This number has to be supported for at least half of the required Subscriptions for half of the required Sessions.
Server	Monitor QueueSize_1	This *ConformanceUnit* does not require queuing when multiple value changes occur during a "publish period". I.e. the latest change will be sent in the *Notification*.
Server	Monitor MinQueueSize_02	Support at least 2 queue entries for *MonitoredItems*. *Servers* often will adapt the queue size to the number of currently *MonitoredItems*. However, it is expected that *Servers* support this minimum queue size for at least one third of the supported *MonitoredItems*.
Server	Monitor MinQueueSize_05	Support at least 5 queue entries for *MonitoredItems*. *Servers* often will adapt the queue size to the number of currently *MonitoredItems*. However, it is expected that *Servers* support this minimum queue size for at least one third of the supported *MonitoredItems*.
Server	Monitor QueueSize_ServerMax	This *ConformanceUnit* is for events. When the *Client* requests queuesize=MAXUInt32 the *Server* is to return the maximum queue size that it can support for event notifications as the revisedQueueSize.
Server	Monitor Triggering	Support the SetTriggering *Service* to create and/or delete triggering links for a triggering item.
Server	Monitor Events	Support creation of *MonitoredItems* for an "*EventNotifier Attribute*" for the purpose of *Event Notification*. The subscription includes supporting a filter that includes SimpleAttribute Operands and a select list of Operators. The list of Operators includes: Equals, IsNull, GreaterThan, LessThan, GreaterThanorEqual, LessThatorEqual, Like, Not, Between, InList, And, Or, Cast, BitwiseAnd, BitwiseOr.
Server	Monitor Complex *Event* Filter	Support for the 'TypeOf' complex *Event* filter operator.

Category	Title	Description
Client	Monitor *Client* Value Change	Use the *MonitoredItem Service* Set to register items for changes in *Attribute* value. Use CreateMonitoredItems to register the *Node*/*Attribute* tuple. Set proper sampling interval, Deadband filter and queuing mode. Use disabling / enabling instead of deleting and re-creating a *MonitoredItem*. Use bulk operations rather than individual service requests to reduce communication overhead.
Client	Monitor *Client* Complex Value	Monitor and decode Values with structured DataTypes.
Client	Monitor *Client* Deadband Filter	Uses Absolute Deadband filters for subscriptions.
Client	Monitor *Client* by Index	Use the IndexRange to select a single element or a range of elements when the *Attribute* value is an array.
Client	Monitor *Client* Aggregate Filter	Uses Aggregate filters for Subscriptions.
Client	Monitor *Client* Events	Use the *MonitoredItem Service* Set to create *MonitoredItems* for *Event* notifications.
Client	Monitor *Client Event* Filter	Use the *Event* filter when calling CreateMonitoredItems to filter the desired Events and to select the columns to be provided for each *Event Notification*.
Client	Monitor *Client* Complex *Event* Filter	Use of the 'TypeOf' complex *Event* filter operator.
Client	Monitor *Client* Modify	Use ModifyMonitoredItems *Service* to change the configuration setting. Use SetMonitoringMode *Service* to disable / enable sampling and / or publishing.
Client	Monitor *Client* Trigger	Use the Triggering Model if certain items are to be reported only if some other item triggers. Use proper monitoring mode for these items. Use SetTriggering *Service* to link these items to the trigger item.

The *Subscription Service* Set is composed of multiple *ConformanceUnits* (see Table 10). *Servers* may provide some aspects of this functionality; see *Profiles* categorized as *Server Profiles* for details. *Clients* may support some aspects of this functionality; see *Profiles* categorized as *Client Profiles* for details.

Table 10 – Subscription Services

Category	Title	Description
Server	*Subscription* Basic	Support the following *Subscription Services*: CreateSubscription, ModifySubscription, DeleteSubscriptions, Publish, Republish and SetPublishingMode.
Server	*Subscription* Minimum 1	Support at least 1 *Subscription* per *Session*. This number has to be supported for all of the minimum required sessions.
Server	*Subscription* Minimum 02	Support at least 2 Subscriptions per *Session*. This number has to be supported for at least half of the minimum required sessions.
Server	*Subscription* Minimum 05	Support at least 5 Subscriptions per *Session*. This number has to be supported for at least half of the minimum required sessions.
Server	*Subscription* Publish Min 02	Support at least 2 Publish *Service* requests per *Session*. This number has to be supported for all of the minimum required sessions. Support of a NotificationMessage retransmission queue is not required; if not available the Republish *Service* returns Bad_MessageNotAvailable.

Category	Title	Description
Server	*Subscription* Publish Min 05	Support at least 5 Publish *Service* requests per *Session*. This number has to be supported for at least half of the minimum required sessions. Support, as a minimum, the number of Publish requests per session as the size of the NotificationMessage retransmission queue for Republish.
Server	*Subscription* Publish Min 10	Support at least 10 Publish *Service* requests per *Session*. This number has to be supported for at least half of the minimum required sessions. Support, as a minimum, the number of Publish requests per session as the size of the NotificationMessage retransmission queue for Republish.
Server	*Subscription* Publish Discard Policy	Respect the specified policy for discarding Publish *Service* requests. If the maximum number of Publish *Service* requests has been queued and a new Publish *Service* request arrives, the "oldest" Publish request has to be discarded by returning the proper error.
Server	*Subscription* Transfer	Support TransferSubscriptions *Service* to transfer a *Subscription* from one *Session* to another.
Server	*Subscription* Durable	Support setting Subscriptions in durable mode. This mode requires that collected data and events are stored and delivered even if a *Client* was disconnected for a longer time or the *Server* was restarted. Support one of the "*Subscription* Durable StorageLevel nnn" *ConformanceUnits*.
Server	*Subscription* Durable StorageLevel Small	Support at least 20 monitored items with a queue size of 10000 for each item and where the size of each *MonitoredItem* is at least equal to size of Double. This requires storage capacity for 200 thousand values of DataType Double.
Server	*Subscription* Durable StorageLevel Medium	Support at least 100 monitored items with a queue size of 50000 for each item and where the size of each *MonitoredItem* is at least equal to size of Double. This requires storage capacity for 5 million values of DataType Double.
Server	*Subscription* Durable StorageLevel High	Support at least 2000 monitored items with a queue size of 200000 for each item and where the size of each *MonitoredItem* is at least equal to size of Double. This requires storage capacity for 400 million values of DataType Double.
Client	*Subscription Client* Basic	Use the *Subscription* and *MonitoredItem Service* Set as an efficient means to detect changes of *Attribute* values and / or to receive *Event* occurrences. Set appropriate intervals for publishing, keep alive notifications and total *Subscription* lifetime. Supply a sufficient number of Publish requests to the *Server* so that *Notifications* can be sent whenever a publish timer expires. Acknowledge received *Notifications* with subsequent Publish requests.
Client	*Subscription Client* Fallback	The *Client* shall interoperate with *Servers* that do not support Subscriptions, or have exhausted *Subscription* limits, for Monitoring by using Read *Service*.
Client	*Subscription Client* Republish	Evaluate the sequence number in *Notifications* to detect lost *Notifications*. Use Republish to request missing *Notifications*.
Client	*Subscription Client* Modify	Allow modification of the *Subscription* configuration using the ModifySubscription *Service*.
Client	*Subscription Client* TransferSubscriptions	The *Client* supports transferring *Subscription* from other *Clients*. This *ConformanceUnit* is used as part of redundant *Clients*.
Client	*Subscription Client* Multiple	Use multiple Subscriptions to reduce the payload of individual *Notifications*.
Client	*Subscription Client* Publish Configurable	Send multiple Publish *Service* requests to assure that the *Server* is always able to send *Notifications*. The number of parallel Publish *Service* requests per *Session* shall be configurable.
Client	*Subscription Client* Durable	Use durable Subscriptions.

5.3 Transport and communication related features

Table 11 describes security related *ConformanceUnits*. All of these *ConformanceUnits* apply equally to both *Clients* and *Servers*, where a *Client* uses the related security unit and a *Server* supports the use of it. These items are defined in detail in OPC 10000-6. It is recommended that a *Server* and *Client* support as many of these options as possible in order to achieve increased levels of interoperability. It is the task of an administrator to determine which of these *ConformanceUnits* are exposed in a given deployed *Server* or *Client* application.

Table 11 – Security

Category	Title	Description
Security	Security User Name Password	The *Server* supports User Name/Password combination(s). The token will be encrypted if required by the security policy of the User Token Policy or by the security policy of the endpoint. An unencrypted token either requires message encryption or means outside the scope of OPC UA to secure the identity token so that it cannot be retrieved by sniffing the communication. One option would be a secure transport like a VPN.
Security	Security User X509	The *Server* supports a public/private key pair for user identity. The use of this feature must be able to be enabled or disabled by an administrator.
Security	Security User IssuedToken Kerberos	The *Server* supports a Kerberos *Server* token for User Identity. The use of this feature must be able to be enabled or disabled by an Administrator. The use of this token is defined in Kerberos Token Documentation. The token will be encrypted if required by the security policy of the User Token Policy or by the security policy of the endpoint. An unencrypted token either requires message encryption or means outside the scope of OPC UA to secure the identity token so that it cannot be retrieved by sniffing the communication. One option would be a secure transport like a VPN.
Security	Security User IssuedToken Kerberos Windows	The *Server* supports the Windows implementation of Kerberos Tokens. This *ConformanceUnit* only applies if the "Security User IssuedToken Kerberos" is supported. The token will be encrypted if required by the security policy of the User Token Policy or by the security policy of the endpoint. An unencrypted token either requires message encryption or means outside the scope of OPC UA to secure the identity token so that it cannot be retrieved by sniffing the communication. One option would be a secure transport like a VPN.
Security	Security User JWT IssuedToken	The *Server* supports a JSON Web Token (JWT) for user identity. Part 6 describes OAuth2 and JWTs in more detail. The use of this feature must be able to be enabled or disabled by an Administrator. The token will be encrypted if required by the security policy of the User Token Policy or by the security policy of the endpoint. An unencrypted token either requires message encryption or means outside the scope of OPC UA to secure the identity token so that it cannot be retrieved by sniffing the communication. One option would be a secure transport like a VPN.
Security	Security User Anonymous	The *Server* provides support for Anonymous access. The use of this feature must be able to be enabled or disabled by an Administrator. By default Anonymous access shall be disabled.
Security	Security User Name Password *Client*	A *Client* uses a User Name/Password combination. The token will be encrypted if required by the security policy of the User Token Policy or by the security policy of the endpoint. An unencrypted token either requires message encryption or means outside the scope of OPC UA to secure the identity token so that it cannot be retrieved by sniffing the communication. One option would be a secure transport like a VPN.

Category	Title	Description
Security	Security User X509 *Client*	A *Client* uses a public/private key pair for user identity. This includes all validation and trust issues associated with a certificate.
Security	Security User IssuedToken Kerberos *Client*	A *Client* uses a Kerberos *Server* token. The use of this token is defined by the Kerberos documentation. The token will be encrypted if required by the security policy of the User Token Policy or by the security policy of the endpoint. An unencrypted token either requires message encryption or means outside the scope of OPC UA to secure the identity token so that it cannot be retrieved by sniffing the communication. One option would be a secure transport like a VPN.
Security	Security User IssuedToken Kerberos Windows *Client*	A *Client* uses the Windows implementation of Kerberos tokens. This *ConformanceUnit* only applies if the "Security User IssuedToken Kerberos *Client*" is supported. The token will be encrypted if required by the security policy of the User Token Policy or by the security policy of the endpoint. An unencrypted token either requires message encryption or means outside the scope of OPC UA to secure the identity token so that it cannot be retrieved by sniffing the communication. One option would be a secure transport like a VPN.
Security	Security User JWT IssuedToken *Client*	A *Client* uses a JSON Web Token (JWT) for user identity. Part 6 describes OAuth2 and JWTs in more detail. The token will be encrypted if required by the security policy of the User Token Policy or by the security policy of the endpoint. An unencrypted token either requires message encryption or means outside the scope of OPC UA to secure the identity token so that it cannot be retrieved by sniffing the communication. One option would be a secure transport like a VPN.
Security	Security Invalid user token	*Servers* shall take proper measures to protect against attacks on user identity tokens. Such an attack is assumed if repeated connection attempts with invalid user identity tokens happen. See ActivateSession *Service* in UA Part 4.
Security	Security User JWT Token Policy	The Server supports one or more Endpoints with a UserTokenPolicy that includes a JWT IssuerEndpointUrl as defined in UA Part 6. For JWT the issuerEndpointUrl is a JSON object that includes all parameters that define the AuthorizationService. As part of the JWT Token Policy, the Server shall support at least one of the following Authority Profile Conformance Units. The URIs defined in the ConformanceUnit shall be exposed in the authorityProfileURI field of the JWT Token Policy.
Security	Security User JWT Token Policy *Client*	The Client understands and uses the Authorization Service definition inside the JWT UserTokenPolicy returned with GetEndpoints. It shall support at least one of the following Authority Profile Conformance Units. The URIs defined in the ConformanceUnit are in the authorityProfileURI field of the JWT Token Policy exposed in Server Endpoints.
Security	OAuth2 Authority *Profile*	This unit indicates support of OAuth2 over HTTPS to request access tokens. The URI for the interactions with this authority is "http://opcfoundation.org/UA/Authorization#OAuth2"
Security	OPC UA Authority *Profile*	This unit indicates support of the OPC UA *Methods* defined in UA Part 12 to request access tokens. The URI for the interactions with this authority is "http://opcfoundation.org/UA/Authorization#OPCUA"
Security	Azure Identity Provider Authority *Profile*	This unit indicates support of the Azure identity provider to request access tokens. The URI for the interactions with this authority is "http://opcfoundation.org/UA/Authorization#Azure"

Category	Title	Description
Security	Security *Certificate* Validation	A certificate will be validated as specified in Part 4. This includes among others structure and signature examination. Allowing for some validation errors to be suppressed by administration directive.
Security	Security Default ApplicationInstance Certificate	An application, when installed, has a default ApplicationInstanceCertificate that is valid. The default ApplicationInstanceCertificate shall either be created as part of the installation or installation instructions explicitly describe the process to create and apply a default ApplicationInstanceCertificate to the application.
Security	Security – No Application Authentication	The *Server* supports being able to be configured for no application authentication, just User authentication and normal encryption/signing: – Configure *Server* to accept all certificates – *Certificates* are just used for message security (signing and encryption) – Users level is used for authentication
Security	Security Policy Required	Support at least Security Policy [A] and Security Policy [B]. Support of multiple Security Policies - even obsolete ones - is recommended. This will provide best interoperability and allows the end user to choose the required level of security. Obsolete Security Policies shall not be enabled / usable without administrative intervention.
Security	Security None CreateSession ActivateSession	When SecurityPolicy=None, the CreateSession and ActivateSession service allow for a NULL/empty signature and do not require Application *Certificates* or a Nonce.
Security	Security None CreateSession ActivateSession 1.0	The *Client* can connect to *Servers* that require a certificate being passed on *Session* establishment. The *Client* in this case will first try without a certificate and if this fails present a certificate.
Security	Security TLS General	This *ConformanceUnit* indicates that at least one of the transport security *Profiles* for TLS is supported by this application. It is used in TLS transport *Profiles*, but the choice of transport security profile is optional. The actual used security profile will default to the most secure one.
Security	Security TLS_RSA with AES_256_CBC_SHA 256	The connection is established using TLS_RSA_WITH_AES_256_CBC_SHA256. That has a MinAsymmetricKeyLength – 2048, MaxAsymmetricKeyLength – 4096, AsymmetricSignatureAlgorithm – RSA_SHA256. (TLS 1.2)
Security	Security TLS_DHE_RSA with AES_nnn_CBC_SHA 256	The connection is established using TLS_DHE_RSA with AES_128_CBC_SHA256 or TLS_DHE_RSA with AES_256_CBC_SHA256. That has a MinAsymmetricKeyLength – 2048, MaxAsymmetricKeyLength – 4096, CertificateSignatureAlgorithm – RSA_SHA256. (TLS 1.2). *Clients* and *Servers* have to support both algorithms.
Security	Security Encryption Required	Encryption is required using the algorithms provided in the security algorithm suite.
Security	Security Signing Required	Signing is required using the algorithms provided in the security algorithm suite.
Security	Security Time Synch – Configuration	Application supports configuring acceptable clock skew.
Security	Security Time Synch – NTP / OS Based support	Application supports time synchronization, either via an implementation of Network Time Protocol (NTP), or via features of a standard operating system.
Security	Security Time Synch – UA based support	An application makes use of the responses header timestamp provided by a configured well know source, such as a *Discovery Server* to synchronize the time on the application and that this time synchronization occurs periodically. Use of this TimeSyncing can be configured.

Category	Title	Description
Security	Security Administration	Allow configuration of the following Security related items (when they apply). * select the allowed User identification policy or policies (e.g. User Name/Password or X509). * enable/disable the security policy "None" or other security policies. * enable/disable endpoints with MessageSecurityMode SIGN or SIGNANDENCRYPT. * set the permitted certification authorities. * define how to react to unknown Certificates. * allow accepting any valid Certificate
Security	Security Administration – XML Schema	Support the OPC UA defined XML schema for importing and exporting security configuration information. This schema is defined in Part 6.
Security	Security *Certificate* Administration	Allow a site administrator to be able to assign a site specific ApplicationInstanceCertificate and if desired to configure a site specific *Certificate* Authority (CA).
Security	Security Role *Server* Base	Support the User Authorization Information Model defined in UA Parts 3 and 5 - like Roles - and the RolePermissions and UserRolePermissions *Attributes*.
Security	Security Role Well Known	Support the well-known Roles "ConfigureAdmin" and "SecurityAdmin" with suggested permissions defined in UA Part 3.
Security	Security Role *Server* IdentityManagement	Allow authorized users to add and/or remove Identities from Roles with the appropriate *Methods*.
Security	Security Role *Server* Management	Allow authorized users to create new Roles and/or remove Roles with the appropriate Methods.
Security	Security Role *Server* Restrict Applications	Support adding applications to a Role with the appropriate *Methods* so that only these applications can use this Role.
Security	Security Role *Server* Restrict Endpoints	Support adding Endpoints to a Role with the appropriate *Methods*. With this restriction a Role is only applied when a *Client* connects via one of these Endpoints.
Security	Security Role *Server* DefaultRolePermissions	Allow authorized users to set the DefaultRolePermissions *Property* for certain NameSpaces. DefaultRolePermissions are applied if no RolePermissions are associated with a *Node*.
Security	Security Role *Server* RolePermissions	Allow authorized users to set the RolePermissions *Attribute* on *Nodes*.
Security	Security Role *Server* Authorization	Restrict access based on the configured Roles and permissions.
Security	Security Role *Client* Base	Understand and use the User Authorization Information Model defined in UA Part 5 and the RolePermissions *Attribute*.
Security	Security Role *Client* Management	Support creating new Roles and adding Identities as well as remove Roles or Identities using the appropriate *Methods*.
Security	Security Role *Client* Restrict Applications	Use the appropriate *Methods* to add applications to a Role so that only these applications can use this Role.
Security	Security Role *Client* Restrict Endpoints	Use the appropriate *Methods* to add Endpoints to a Role. With this restriction a Role is only applied when a *Client* connects via one of these Endpoints.
Security	Security Role *Client* DefaultRolePermissions	Ability to set the DefaultRolePermissions *Property* for certain NameSpaces. DefaultRolePermissions are applied if no RolePermissions are associated with a *Node*.
Security	Security Role *Client* RolePermissions	Support setting the RolePermissions *Attribute* on *Nodes*.
Security	Pull Model for Global *Certificate* and TrustList Management	Use the *Certificate* Management *Services* of UA Part 12 for the Pull model to manage Application Instance *Certificates* and Trust Lists including Revocation Lists.
Security	Push Model for Global *Certificate* and TrustList Management	Support the *Certificate* Management *Services* of UA Part 12 for the Push model to manage Application Instance *Certificates* and Trust Lists including Revocation Lists.

Category	Title	Description
Security	Pull Model for KeyCredential *Service*	Use the *Methods* on an instance of the KeyCredentialServiceType (Pull model) to obtain KeyCredentials as specified in UA Part 12.
Security	Push Model for KeyCredential *Service*	Support the KeyCredential *Services* Push model of UA Part 12 to obtain KeyCredentials. This includes support of one or more instances of the KeyCredentialConfigurationType and the *Methods* to update or delete credentials.
Security	Authorization Service Configuration Server	Support the Object Types defined in Part 12 to allow configuration of information needed to accept Access Tokens when presented by the Client during session establishment. Access Tokens are issued by Authorization Services.
Security	Authorization Service Client	Use the RequestAccessToken Method defined in UA Part 12.
Security	SymmetricSignatureAlgorithm_None	This algorithm does not apply.
Security	SymmetricSignatureAlgorithm_HMAC-SHA1	A keyed hash which is defined in https://tools.ietf.org/html/rfc2104. The hash algorithm is SHA1 and is described in https://tools.ietf.org/html/rfc3174. The URI is http://www.w3.org/2000/09/xmldsig#hmac-sha1. No known exploits exist when using SHA1 with a keyed hash, however, SHA1 was broken in 2017 so use of this algorithm is not recommended.
Security	SymmetricSignatureAlgorithm_HMAC-SHA2-256	A keyed hash used for message authentication which is defined in https://tools.ietf.org/html/rfc2104. The hash algorithm is SHA2 with 256 bits and described in https://tools.ietf.org/html/rfc4634
Security	SymmetricEncryptionAlgorithm_None	This algorithm does not apply.
Security	SymmetricEncryptionAlgorithm_AES128-CBC	The AES encryption algorithm which is defined in http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf. Multiple blocks encrypted using the CBC mode described in http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf. The key size is 128 bits. The block size is 16 bytes. The URI is http://www.w3.org/2001/04/xmlenc#aes128-cbc.
Security	SymmetricEncryptionAlgorithm_AES256-CBC	The AES encryption algorithm which is defined in http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf. Multiple blocks encrypted using the CBC mode described in http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf. The key size is 256 bits. The block size is 16 bytes. The URI is http://www.w3.org/2001/04/xmlenc#aes256-cbc.
Security	SymmetricEncryptionAlgorithm_AES128-CTR	The AES encryption algorithm which is defined in http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf. Multiple blocks encrypted using the CTR mode described in http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf. The counter block format is defined in https://tools.ietf.org/html/rfc3686. The key size is 128 bits. The block size is 16 bytes. The input nonce length is 4 bytes. The URI is http://opcfoundation.org/UA/security/aes128-ctr.

Category	Title	Description
Security	SymmetricEncryptionAlgorithm_AES256-CTR	The AES encryption algorithm which is defined in http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf. The key size is 256 bits. The block size is 16 bytes. Multiple blocks encrypted using the CTR mode described in http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf. The counter block format is defined in https://tools.ietf.org/html/rfc3686. The key size is 128 bits. The block size is 16 bytes. The input nonce length is 4 bytes. The URI is http://opcfoundation.org/UA/security/aes256-ctr.
Security	AsymmetricSignatureAlgorithm_None	This algorithm does not apply.
Security	AsymmetricSignatureAlgorithm_RSA-PKCS15-SHA1	The RSA signature algorithm which is defined in https://tools.ietf.org/html/rfc3447. The RSASSA-PKCS1-v1_5 scheme is used. The hash algorithm is SHA1 and is described in https://tools.ietf.org/html/rfc3174. The URI is http://www.w3.org/2000/09/xmldsig#rsa-sha1. SHA1 was broken in 2017 so this algorithm should not be used.
Security	AsymmetricSignatureAlgorithm_RSA-PKCS15-SHA2-256	The RSA signature algorithm which is defined in https://tools.ietf.org/html/rfc3447. The RSASSA-PKCS1-v1_5 scheme is used. The hash algorithm is SHA2 with 256bits and is described in https://tools.ietf.org/html/rfc6234. The URI is http://www.w3.org/2001/04/xmldsig-more#rsa-sha256.
Security	AsymmetricSignatureAlgorithm_RSA-PSS -SHA2-256	The RSA signature algorithm which is defined in https://tools.ietf.org/html/rfc3447. The RSASSA-PSS scheme is used. The hash algorithm is SHA2 with 256bits and is described in https://tools.ietf.org/html/rfc6234. The mask generation algorithm also uses SHA2 with 256 bits. The salt length is 32 bytes. The URI is http://opcfoundation.org/UA/security/rsa-pss -sha2-256.
Security	AsymmetricEncryptionAlgorithm_None	This algorithm does not apply.
Security	AsymmetricEncryptionAlgorithm_RSA-PKCS15	The RSA encryption algorithm which is defined in https://tools.ietf.org/html/rfc3447. The RSAES-PKCS1-v1_5 scheme is used. The URI is http://www.w3.org/2001/04/xmlenc#rsa-1_5. The RSAES-PKCS1-v1_5 scheme has known weaknesses and is not recommended.
Security	AsymmetricEncryptionAlgorithm_RSA-OAEP-SHA1	The RSA encryption algorithm which is defined in https://tools.ietf.org/html/rfc3447. The RSAES-OAEP scheme is used. The hash algorithm is SHA1 and is described in https://tools.ietf.org/html/rfc6234. The mask generation algorithm also uses SHA1. The URI is http://www.w3.org/2001/04/xmlenc#rsa-oaep. No known exploits exist when using SHA1 with RSAES-OAEP, however, SHA1 was broken in 2017 so use of this algorithm is not recommended.

Category	Title	Description
Security	AsymmetricEncryptionAlgorithm_RSA-OAEP-SHA2-256	The RSA encryption algorithm which is defined in https://tools.ietf.org/html/rfc3447. The RSAES-OAEP scheme is used. The hash algorithm is SHA2 with 256 bits and is described in https://tools.ietf.org/html/rfc6234. The mask generation algorithm also uses SHA2 with 256 bits. The URI is http://opcfoundation.org/UA/security/rsa-oaep-sha2-256.
Security	KeyDerivationAlgorithm_None	This algorithm does not apply.
Security	KeyDerivationAlgorithm_P-SHA1	The P_SHA-1 pseudo-random function defined in https://tools.ietf.org/html/rfc4346. The URI is http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/dk/p_sha1. No known exploits exist when using SHA1 with P-SHA-1, however, SHA1 was broken in 2017 so use of this algorithm is not recommended.
Security	KeyDerivationAlgorithm_P-SHA2-256	The P_SHA256 pseudo-random function defined in https://tools.ietf.org/html/rfc5246. The URI is http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/dk/p_sha256.
Security	CertificateSignatureAlgorithm_None	This algorithm does not apply.
Security	CertificateSignatureAlgorithm_RSA-PKCS15-SHA2-256	The RSA signature algorithm which is defined in https://tools.ietf.org/html/rfc3447. The RSASSA-PKCS1-v1_5 scheme is used. The hash algorithm is SHA2 with 256bits and is described in https://tools.ietf.org/html/rfc6234. The SHA2 algorithm with 384 or 512 bits may be used instead of SHA2 with 256 bits. The URI is http://www.w3.org/2001/04/xmldsig-more#rsa-sha256.
Security	CertificateSignatureAlgorithm_RSA-PKCS15-SHA1	The RSA signature algorithm which is defined in https://tools.ietf.org/html/rfc3447. The RSASSA-PKCS1-v1_5 scheme is used. The hash algorithm is SHA1 and is described in https://tools.ietf.org/html/rfc3174. The URI is http://www.w3.org/2000/09/xmldsig#rsa-sha1. SHA1 was broken in 2017 so this algorithm should not be used. The SHA2 algorithm with 244, 256, 384 or 512 bits may be used instead of SHA1. The SHA2 algorithm is described in https://tools.ietf.org/html/rfc6234.
Security	SecurtyPolicy_None_Limits	DerivedSignatureKeyLength: 0
Security	Aes128-Sha256-RsaOaep_Limits	-> DerivedSignatureKeyLength: 256 bits -> MinAsymmetricKeyLength: 2048 bits -> MaxAsymmetricKeyLength: 4096 bits -> SecureChannelNonceLength: 32 bytes
Security	Basic256Sha256_Limits	-> DerivedSignatureKeyLength: 256 bits -> MinAsymmetricKeyLength: 2048 bits -> MaxAsymmetricKeyLength: 4096 bits -> SecureChannelNonceLength: 32 bytes
Security	Aes256-Sha256-RsaPss_Limits	-> DerivedSignatureKeyLength: 256 bits -> MinAsymmetricKeyLength: 2048 bits -> MaxAsymmetricKeyLength: 4096 bits -> SecureChannelNonceLength: 32 bytes

Category	Title	Description
Security	Basic128Rsa15_Limits	-> DerivedSignatureKeyLength: 128 bits -> MinAsymmetricKeyLength: 1024 bits -> MaxAsymmetricKeyLength: 2048 bits -> SecureChannelNonceLength: 16 bytes
Security	Basic256_Limits	-> DerivedSignatureKeyLength: 192 bits -> MinAsymmetricKeyLength: 1024 bits -> MaxAsymmetricKeyLength: 2048 bits -> SecureChannelNonceLength: 32 bytes

Table 12 describes protocol and encoding related features that can be profiled. These features are defined in detail in OPC 10000-6. It is recommended that *Servers* and *Clients* support as many of these options as possible for greatest interoperability.

Table 12 – Protocol and Encoding

Category	Title	Description
Server	Protocol Reverse Connect *Server*	Support reverse connectivity by sending a ReverseHello message to a Client. The reverse connect procedure can be applied to several transports as specified in UA Part 6 and shall be supported for all of these that are available in a Server.
Server	Protocol Configuration	Allow administration of the Endpoints and the port number used by the Endpoints.
Client	Protocol Reverse Connect *Client*	Support reverse connectivity by accepting Reverse Hello messages from Servers and establish a Secure Channel if the URI of the Server is accepted by Client or Client user. The reverse connect procedure can be applied to several transports as specified in UA Part 6 and shall be supported for all of these transports that are supported by the Client.
Transport	Protocol UA TCP	Support the UA TCP transport protocol as defined in UA Part 6.
Transport	Protocol HTTPS	Support the HTTPS protocol as defined in UA Part 6.
Transport	Protocol Web Sockets	Support the WebSocket protocol (WSS) with at least one of the sub-protocols defined in UA Part 6.
Transport	UA Secure Conversation	Support UA Secure Conversation specified in UA Part 6.
Transport	UA Binary Encoding	Support UA Binary Encoding. Values of these data types are encoded in compact binary formats, contiguously and without tagging. I.e. the receiver is assumed to understand the structure it is decoding.
Transport	UA SOAP-XML Encoding	Support Soap V1.2 based Xml Encoding as defined in UA Part 6. The XML elements include all information necessary to convert it back into OPC UA structures of any language.
Transport	JSON Reversible Encoding	Support reversible JSON Encoding as defined in UA Part 6. The JSON object includes all information necessary to convert it back into OPC UA structures of any language.

5.4 Information Model and AddressSpace related features

Table 13 describes base features related items that can be profiled. For additional information about these items, please refer to OPC 10000-3, OPC 10000-5 and OPC 10000-6. *Servers* with a larger resource capacity would support most of this functionality, but smaller resource constraint *Server* may omit some of this functionality. Many *Clients* would utilize some of this functionality and more robust *Clients* would utilize most of this functionality.

Table 13 – Base Information

Category	Title	Description
Server	Base Info Core Structure	The *Server* supports the *Server Object*, ServerCapabilities and supports the OPC UA *AddressSpace* structure.

Category	Title	Description
Server	Base Info *Server* Capabilities	The *Server* supports publishing of the *Server* limitation in the ServerCapabilities, including MaxArrayLength, MaxStringLength, MaxNodePerRead, MaxNodesPerWrite, MaxNodesPerSubscription and MaxNodesPerBrowse.
Server	Base Info Progress Events	The *Server* exposes if generation of Progress events for long running service calls such as HistoryRead or Query is supported. If it is listed as supported in ServerCapabilities, than the actual events are verified.
Server	Base Info Diagnostics	The *Server* supports the collection of diagnostic information. The EnabledFlag in the ServerDiagnostics *Object* can be set TRUE and in that case all static and dynamic *Objects* and *Variables* for diagnostic data as defined in UA Part 5 are supported.
Server	Base Info System Status	The *Server* supports generating SystemStatusChangeEventType indicating shutdown of the *Server* (SourceNode=*Server*).
Server	Base Info Estimated Return Time	*Server* supports the EstimatedReturnTime *Property*. It indicates the time at which the *Server* is expected to have a ServerStatus.State of RUNNING_0. *Clients* can use this information to govern the reconnect logic.
Server	Base Info System Status Underlying System	The *Server* supports generating SystemStatusChangeEventType indicating changes to an Underlying System (SourceNode = *Server*). This event can also be used to indicate that the OPC UA *Server* has underlying systems.
Server	Base Info Device Failure	The *Server* supports generating DeviceFailureEventType indicating changes to individual devices in an underlying system.
Server	Base Info GetMonitoredItems *Method*	The *Server* supports obtaining subscription information via GetMonitoredItems *Method* on the *Server* object.
Server	Base Info ResendData *Method*	Support the standard *Method* ResendData (defined in UA Part 5) to get the latest value of the monitored items of a *Subscription*.
Server	Base Info Type System	The *Server* exposes a Type System with DataTypes, ReferenceTypes, *ObjectTypes* and VariableTypes including all of the OPC UA (namespace 0) types that are used by the *Server*, as defined in Part 5. Items that are defined in Namespace 0 but are defined in other specification parts are tested as part of the other information models.
Server	Base Info Custom Type System	The *Server* supports custom types (i.e. types that are derived from well-known *ObjectTypes*, VariableTypes, ReferenceTypes or DataTypes). Supporting this conformance unit requires that the custom types with their full inheritance tree are exposed in the *AddressSpace*.
Server	Base Info Model Change	The *Server* supports ModelChange *Event* and NodeVersion *Property* for all *Nodes* that the server allows Model changes for.
Server	Base Info Placeholder Modelling Rules	The *Server* supports defining custom *Object* or *Variables* that include the use of OptionalPlaceholder or MandatoryPlaceholder modelling rules.
Server	Base Info SemanticChange	The *Server* supports SemanticChangeEvent for some Properties. This includes setting the SemanticChange Bit in the status when a semantic change occurs, such as a change in the engineering unit associated with a value.
Server	Base Info EventQueueOverflow EventType	The *Server* supports the EventQueueOverflowEventType as defined in Part 4.
Server	Base Info OptionSet	The *Server* supports the *VariableType* OptionSet.
Server	Base Info ValueAsText	The *Server* supports the *Property* ValueAsText for enumerated DataTypes.

Category	Title	Description
Server	Base Info Engineering Units	The *Server* supports defining *Variables* that include the Engineering Units *Property*. This property makes use of the EUInformation data structure. This structure by default represents the UN/CEFACT "Codes for Units of Measurement". If a different EU representation is required then the EUInformation.namespaceUri will indicate the alternate namespace.
Server	Base Info Selection List	The *Server* supports Variables of the SelectionListType VariableType.
Server	Base Info FileType Base	The *Server* supports the FileType *Object* (see Part 5). File writing may be restricted.
Server	Base Info FileType Write	The *Server* supports the FileType *Object*, including writing of files. Also included is the support of user access control on FileType *Object*.
Server	Base Info RequestServerStateChange *Method*	The *Server* supports the RequestServerStateChange *Method*.
Server	Base Info State Machine Instance	Support instances of the StateMachineType or a sub-type in the *AddressSpace*. Generate Events when significant state changes occur. At least one GeneratesEvent *Reference* exists to define the *Event*(s) triggered on state changes.
Server	Base Info Finite State Machine Instance	Support instances of the FiniteStateMachineType or a sub-type in the *AddressSpace*.
Server	Base Info Available States and Transitions	Support the Properties AvailableStates and AvailableTransitions defined for the FiniteStateMachineType.
Client	Base Info *Client* Basic	The *Client* uses the defined OPC UA *AddressSpace*. Access or provide access to *Server* information like the *Server*'s state, BuildInfo, capabilities, Namespace Table and Type Model.
Client	Base Info *Client* Honour Operation Limits	The *Client* shall honour *Server* limits described in ServerCapabilites *Object* of *Server*.
Client	Base Info *Event* Processing	The *Client* is able to subscribe for and process base OPC UA Events.
Client	Base Info *Client* System Status	The *Client* makes use of SystemStatusChangeEventType to detect server shutdowns.
Client	Base Info *Client* Estimated Return Time	*Client* uses the EstimatedReturnTime *Property* to govern the reconnect logic.
Client	Base Info *Client* System Status Underlying System	The *Client* makes use of SystemStatusChangeEventType to detect changes to an Underlying System (SourceNode = *Server*).
Client	Base Info *Client* Device Failure	The *Client* makes use of DeviceFailureEventType to detect failed devices in underlying systems
Client	Base Info *Client* Progress Events	The *Client* makes use of ProgressEvents, including checking for their support.
Client	Base Info *Client* Diagnostics	The *Client* provides interactive or programmatic access to the *Server*'s diagnostic information.
Client	Base Info *Client* Type Programming	The *Client* programmatically process instances of *Objects* or *Variables* by using their type definitions. This includes custom DataTypes, *ObjectTypes* and VariableTypes.
Client	Base Info *Client* Type Pre-Knowledge	The *Client* shall interoperate with *Servers* that do not expose OPC UA Types in *AddressSpace*.
Client	Base Info *Client* Remote *Nodes*	The *Client* can access Nodes that have an extended NodeID that reference a *Server* different then the orginating *Server*. It is acceptable that the *Server* configuration information be pre-configured on the *Client*.
Client	Base Info *Client* Change Events	The *Client* processes ModelChangeEvents to detect changes in the *Server*'s OPC UA *AddressSpace* and take appropriate action for a given change.

Category	Title	Description
Client	Base Info *Client* GetMonitoredItems *Method*	The *Client* makes use of GetMonitoredItems *Method* to recover for communication interruptions and/or to recover subscription information.
Client	Base Data *Client* ResendData *Method*	The *Client* makes use of ResendData *Method* to fetch the last value of the data monitored items.
Client	Base Info *Client* Selection List	The *Client* uses and understands Variables of the SelectionListType VariableType.
Client	Base Info *Client* FileType Base	The *Client* can access a FileType *Object* to transfer a file from the *Server* to the *Client*. This includes large files.
Client	Base Info *Client* FileType Write	The *Client* can access a FileType *Object* to transfer a file from the *Client* to the *Server*. This includes large files.
Client	Base Info *Client* RequestServerStateChange	The *Client* can invoke the RequestServerStateChange *Method*.
Client	Base Info *Client* State Machine Instance	Use instances of the StateMachineType or a sub-type. Monitor either the CurrentState component *Variable* of the instance or the Events triggered as effect of state changes. Use Methods when defined for the StateMachineType to affect the state.
Client	Base Info *Client* Finite State Machine Instance	Use instances of the FinitStateMachineType or a sub-type. Monitor either the CurrentState component *Variable* of the instance or the *Events* triggered as effect of state changes.
Client	Base Info Client Available States and Transitions	Use the Properties AvailableStates and AvailableTransitions when exposed by a *Server*.

Table 14 describes Address Space Model information related items that can be profiled. The details of these model items are defined in OPC 10000-3 and OPC 10000-5. This includes *Server Facets* that describe what a *Server* exposes and *Client Facets* that describe what a *Client* consumes.

Table 14 – Address Space Model

Category	Title	Description
Server	Address Space Base	Support the *NodeClasses* with their *Attributes* and *References* as defined in Part 3. This includes for instance: *Object*, *ObjectType*, *Variable*, *VariableType*, *References* and DataType.
Server	Address Space Atomicity	Support setting the NonatomicRead and NonatomicWrite flags in the AccessLevelEx *Attribute* for *Variable Nodes* to indicate whether Read or Write operations can be performed in atomic manner. If the flags are set to '1', atomicity cannot be assured.
Server	Address Space Full Array Only	Support setting the WriteFullArrayOnly flag in the AccessLevelEx *Attribute* for *Variable Nodes* of non-scalar data types to indicate whether write operations for an array can be performed with an IndexRange.
Server	Address Space Events	Support OPC UA *AddressSpace* elements for generating *Event* notifications. This includes at least one *Node* with an *EventNotifier Attribute* set to True (*Server Node*).
Server	Address Space Complex Data Dictionary	Support structured DataTypes with a Data Dictionary. Note that V1.04 of OPC UA Part 3 specifies a simplified approach using the new DataTypeDefinition Attribute. The "Address Space DataTypeDefinition Attribute" Conformance Unit requires support of the DataTypeDefinition Attribute. Support of a DataDictionary will be deprecated in one of the next OPC UA versions.
Server	Address Space DataTypeDefinition Attribute	Support structured DataTypes and expose the meta data and encoding information with a StructureDefinitionType via the DataTypeDefinition *Attribute*.
Server	Address Space *Method*	Support *Method Nodes*.
Server	Address Space Notifier Hierarchy	Supports using the HasNotifier reference to build a hierarchy of *Object Nodes* that are notifiers with other notifier *Object Nodes*.

Category	Title	Description
Server	Address Space Source Hierarchy	Supports hierarchies of event sources where each hierarchy roots in an *Object Node* that is a notifier. The HasEventSource *Reference* is used to relate the *Nodes* within a hierarchy. If *Conditions* are supported, the hierarchy shall include HasCondition *References*.
Server	Address Space WriteMask	Supports WriteMask indicating the write access availability for all attributes, including not supported attributes.
Server	Address Space UserWriteMask	Supports UserWriteMask indicating the write access availability for all attributes for the given user, including not supported attributes. Support includes at least two levels of users.
Server	Address Space UserWriteMask Multilevel	Supports UserWriteMask indicating the write access availability for all attributes for the given user, including not supported attributes. This includes supporting multiple levels of access control for all nodes in the system.
Server	Address Space User Access Level Full	Implements User Access Level security, this includes supporting multiple levels of access control for *Variable* nodes in the system. This includes an indication of read, write, Historical read and Historical write access to the Value *Attribute*.
Server	Address Space User Access Level Base	Implements User Access Level Security for *Variable* nodes, this includes at least two users in the system. This includes an indication of read, write, historical read and Historical write access to the value attribute
Client	Address Space *Client* Base	Uses and understands the *NodeClasses* with their *Attributes* and behaviour as defined in Part 3. This includes for instance: *Object*, *ObjectType*, *Variable*, *VariableType*, *References* and DataType. This includes treating BrowseNames and String NodeIds as case sensitive.
Client	Address Space *Client* Atomicity	Access the NonatomicRead or NonatomicWrite flags in the AccessLevelEx *Attribute* of *Variable Nodes* to determine whether Read or Write operations can be performed in atomic manner. This information will typically be shown to a user for further action.
Client	Address Space *Client* Full Array Only	Access the WriteFullArrayOnly flag in the AccessLevelEx *Attribute* of *Variable Nodes* with non-scalar data types to determine whether writing to an array with an IndexRange is allowed.
Client	Address Space *Client* Complex Data Dictionary	Uses and understands arbitrary structured DataTypes via Data Dictionary. Note that V1.04 of OPC UA Part 3 specifies a simplified approach using the new DataTypeDefinition Attribute. The "Address Space Client DataTypeDefinition Attribute" Conformance Unit requires support of the DataTypeDefinition Attribute.
Client	Address Space *Client* DataTypeDefinition Attribute	Uses and understands arbitrary structured DataTypes where the meta data and encoding information are exposed with the StructureDefinitionType via the DataTypeDefinition *Attribute*.
Client	Address Space *Client* Notifier Hierarchy	Uses hierarchy of *Object Nodes* that are notifiers to detect specific areas where the *Client* can subscribe for Events.
Client	Address Space *Client* Source Hierarchy	Detect and use the hierarchy of event sources exposed for specific *Object Nodes* that are event notifiers.

Table 15 describes Data Access information model related items that can be profiled. The details of this model are defined in OPC 10000-8. *Servers* could expose this information model and *Clients* could utilize this information model.

Table 15 – Data Access

Category	Title	Description
Server	Data Access DataItems	Provide *Variables* of DataItemType or one of its subtypes. Support the StatusCodes specified in Part 8. Support of optional Properties (e.g. "InstrumentRange") shall be verified during certification testing and will be shown in the *Certificate*.
Server	Data Access AnalogItems	Support AnalogItemType *Variables* with corresponding Properties. The support of optional properties will be listed.

Category	Title	Description
Server	Data Access PercentDeadband	Support PercentDeadband filter when monitoring AnalogItemType *Variables*.
Server	Data Access Semantic Changes	Support semantic changes of AnalogItemType items (EURange *Property* and/or EngineeringUnits *Property*). Support semantic change StatusCode bits where appropriate.
Server	Data Access TwoState	Support TwoStateDiscreteType *Variables* with corresponding Properties.
Server	Data Access MultiState	Support MultiStateDiscreteType *Variables* with corresponding Properties.
Server	Data Access MultiStateValueDiscrete	Support MultiStateValueDiscreteType *Variables* with corresponding Properties.
Server	Data Access ArrayItemType	Provide *Variables* of ArrayItemType or one of its subtypes (YArrayItemType, XYArrayItemType, ImageArrayType, CubeArrayType and NDimensionArrayType). The supported subtypes will be listed. Support for this type includes supporting all of the mandatory properties including AxisInformation.
Server	Data Access Complex Number	Supports the Complex Number data type. This data type is available for any variable types that do not have other explicit restrictions.
Server	Data Access DoubleComplex Number	Supports the DoubleComplex Number data type. This data type is available for any variable types that do not have other explicit restrictions.
Client	Data Access *Client* Basic	Understand the DataAccess *Variable* Types. Make use of the standard Properties if applicable.
Client	Data Access *Client* AnalogItems	Understand AnalogItemType *Variables* with corresponding Properties.
Client	Data Access *Client* TwoState	Understand TwoStateDiscreteType *Variables* with corresponding Properties.
Client	Data Access *Client* MultiState	Understand MultiStateDiscreteType *Variables* with corresponding Properties.
Client	Data Access *Client* MultiStateValueDiscrete	Understand MultiStateValueDiscreteType *Variables* with corresponding Properties.
Client	Data Access *Client* Deadband	Use PercentDeadband to filter value changes of AnalogItemType *Variables*.
Client	Data Access *Client* SemanticChange	Recognize the semantic change bit in the StatusCode while monitoring items and take proper action. Typically, the *Client* has to re-read Properties that define type-specific semantic like the EURange and EngineeringUnits Properties.

Table 16 describes *Alarm* and *Conditions* information model related items that can be profiled. The details of this model are defined in OPC 10000-9. *Servers* that deal with *Alarm* and *Conditions* would expose this information model and *Clients* that process *Alarms* and *Conditions* would utilize this information model.

Table 16 – Alarms and Conditions

Category	Title	Description
Server	A & C Basic	Supports *Alarm & Condition* model ConditionType.
Server	A & C Enable	Supports Enable and Disable Methods.
Server	A & C Refresh	Supports ConditionRefresh *Method* and the concept of a refresh.
Server	A & C Refresh2	Supports ConditionRefresh2 *Method* and the concept of a monitored item based refresh.
Server	A & C Instances	Support exposing of A&C *Condition* instances in the *AddressSpace*.
Server	A & C ConditionClasses	Supports multiple *Condition* classes for grouping and filtering of *Alarms*.
Server	A & C *Condition* Sub-Classes	Support assigning multiple *Condition* sub-classes for grouping and filtering of *Alarms*.
Server	A & C Acknowledge	Supports the Acknowledge concept, Acknowledge *Method*, and AcknowledgeableCondition Type.

Category	Title	Description
Server	A & C Confirm	Supports the concept of Confirm and the Confirm *Method*.
Server	A & C Comment	Supports the concept of Comments and the AddComment *Method*.
Server	A & C *Alarm*	Supports the mandatory features of the AlarmCondition Type.
Server	A & C *Alarm* Metrics	Support the collection of alarm metrics data as defined in UA Part 9. This implies one of more instances of the AlarmMetricsType.
Server	A & C Branch	Support for branching of *Condition* Types and any subtypes, such as AcknowledgeableConditionType and AlarmConditionType etc.
Server	A & C Shelving	Support the concept of shelving and the TimedShelve, OneShotShelve and Unshelve Methods.
Server	A & C Suppression	Support the SuppressedState.
Server	A & C Suppression by Operator	Support the Suppress and UnSuppress *Methods* to allow an operator control over the SuppressedState.
Server	A & C Silencing	Support the concept of silencing and the Silence Method.
Server	A & C Out Of Service	Support the OutOfService state and the OutOfService Method.
Server	A & C On-Off Delay	Support the OnDelay and OffDelay Properties to eliminate nuisance *Alarms*.
Server	A & C Re-Alarming	Support the ReAlarmTime and ReAlarmRepeatCount Properties that define automatic re-annunciation of *Alarms* in certain conditions.
Server	A & C First in Group *Alarm*	Support the "FirstInGroup" elements for an *Alarm*, indicating which *Alarm* of a group was the trigger.
Server	A & C Audible Sound	Support the AudibleSound *Property*. This *Property* contains the sound file that is to be played if an audible *Alarm* is to be generated.
Server	A & C Exclusive Level	Supports Exclusive Level *Alarm* type.
Server	A & C Exclusive Limit	Supports Exclusive Limit *Alarms*. A *Server* that supports this must support at least one of the sub-types: Level, Deviation or RateofChange.
Server	A & C Exclusive Deviation	Supports Exclusive Deviation *Alarm* type.
Server	A & C Exclusive RateOfChange	Supports Exclusive RateOfChange *Alarm* type.
Server	A & C Non-Exclusive Limit	Supports Non-Exclusive Limit *Alarms*. A *Server* that supports this must support at least one of the sub-types: Level, Deviation or RateofChange.
Server	A & C Non-Exclusive Level	Supports Non-Exclusive Level *Alarm* type.
Server	A & C Non-Exclusive Deviation	Supports Non-Exclusive Deviation *Alarm* type.
Server	A & C Non-Exclusive RateOfChange	Supports Non-Exclusive RateOfChange *Alarm* type.
Server	A & C Discrete	Supports Discrete *Alarm* types.
Server	A & C OffNormal	Supports OffNormalAlarmType.
Server	A & C SystemOffNormal	Supports SystemOffNormalAlarmType.
Server	A & C Trip	Supports Trip *Alarm* type.
Server	A & C Discrepancy	Supports Discrepancy *Alarm* type.
Server	A & C Dialog	Supports DialogConditionType including Respond *Method*.
Server	A & C CertificateExpiration	Supports CertificateExpirationAlarmType.
Server	A & E Wrapper Mapping	The *Server* uses the COM A&E mapping specified in the annex of Part 9 to map OPC-COM Events to A&C Events. This includes *Condition* Class mapping.
Client	A & C Basic *Client*	Uses the *Alarm* & *Condition* model ConditionType.
Client	A & C Enable *Client*	Uses Enable and Disable Methods.
Client	A & C Refresh *Client*	Uses ConditionRefresh *Method* and the concept of a refresh.
Client	A & C Refresh2 *Client*	Uses ConditionRefresh2 *Method* and the concept of a monitored item based refresh.

Category	Title	Description
Client	A & C Instances *Client*	Uses A&C *Condition* instances when they are exposed in the *AddressSpace*.
Client	A & C ConditionClasses *Client*	Uses *Condition* classes to group *Alarms*.
Client	A & C *Condition* Sub-Classes *Client*	Uses *Condition* sub-classes to group or filter *Alarms*.
Client	A & C Acknowledge *Client*	Understands the Acknowledge concept and the AcknowledgeableCondition Type, and uses the Acknowledge *Method* if requested.
Client	A & C Confirm *Client*	Understands the concept of confirming *Conditions* and uses the Confirm *Method*.
Client	A & C Comment *Client*	Understands the concept of Comments and uses the AddComment *Method*.
Client	A & C *Alarm Client*	Understands the concept of *Alarms* and uses the mandatory features of the AlarmCondition Type,
Client	A & C *Alarm* Metrics *Client*	Understand and use *Alarm* metrics data as defined in UA Part 9. This implies discovery of instances of the AlarmMetricsType that can exist anywhere in the HasNotifier hierarchy.
Client	A & C Branch *Client*	Can make use of and process *Condition* Branches, including all actions associated with previous *Condition* instances.
Client	A & C Shelving *Client*	Understand the shelving model and use the TimedShelve, OneShotShelve and Unshelve Methods.
Client	A & C Suppression Client	Understand the SuppressedState model.
Client	A & C Suppression by Operator *Client*	Use the Suppress and UnSuppress *Methods* to allow an operator control over the SuppressedState.
Client	A & C Silencing Client	Understand the SilencedState model and use the Silence Method.
Client	A & C Out Of Service Client	Understand the OutOfServiceState model and use the OutOfService Method.
Client	A & C On-Off Delay *Client*	Uses the OnDelay and OffDelay Properties to eliminate nuisance *Alarms*.
Client	A & C Re-Alarming *Client*	Understand and use the ReAlarmTime and ReAlarmRepeatCount Properties. Configure the ReAlarmTime *Property* for automatic re-annunciation of an *Alarm*. Note that configuration is only possible for *Servers* that expose *Alarm* instances.
Client	A & C First in Group *Alarm Client*	Use the "FirstInGroup" elements for an *Alarm* to determine which *Alarm* of a group was the trigger.
Client	A & C Audible Sound *Client*	Use the AudibleSound *Property* and - if reported - play the sound file.
Client	A & C Exclusive Level *Client*	Uses Exclusive Level *Alarms*.
Client	A & C Exclusive Limit *Client*	Uses Exclusive Limit *Alarms*. Requires that at least one of the sub-types be used.
Client	A & C Exclusive Deviation *Client*	Uses Exclusive Deviation *Alarms*.
Client	A & C Exclusive RateOfChange *Client*	Uses Exclusive RateOfChange *Alarms*.
Client	A & C Non-Exclusive Level *Client*	Uses Non-Exclusive Level *Alarms*.
Client	A & C Non-Exclusive Limit *Client*	Uses Non-Exclusive Limit *Alarms*. Requires that at least one of the sub-types be used.
Client	A & C Non-Exclusive Deviation *Client*	Uses Non-Exclusive Deviation *Alarms*.

Category	Title	Description
Client	A & C Non-Exclusive RateOfChange *Client*	Uses Non-Exclusive RateOfChange *Alarms*.
Client	A & C Discrete *Client*	Uses Discrete *Alarm* types.
Client	A & C OffNormal *Client*	Uses OffNormalAlarmType.
Client	A & C SystemOffNormal *Client*	Uses SystemOffNormalAlarmType.
Client	A & C Trip *Client*	Uses TripAlarmType.
Client	A & C Discrepancy Client	Uses Discrepancy *Alarm* type.
Client	A & C Dialog *Client*	Uses DialogConditionType including Respond *Method*.
Client	A & C CertificateExpiration *Client*	Uses CertificateExpirationAlarmType.

Table 17 describes Historical Data Access information model related items that can be profiled. The details of this model are defined in OPC 10000-11. *Servers* that support some level of historical data would expose this information model and *Clients* that utilize historical data would utilize this information model.

Table 17 – Historical Access

Category	Title	Description
Server	Historical Access Read Raw	General support for basic historical access, reading raw data using the ReadRawModifiedDetails structure. Where the time range is specified using a start time, stop time and number of values (a minimum of two of the three parameters must be provided) and the ReadModified flag is set to False.
Server	Historical Access Data Max *Nodes* Read Continuation Point	Supports enough continuation points to cover the number of supported points indicated in the MaxNodesPerHistoryReadData *Server* OperationLimits *Property* for historical data access.
Server	Historical Access Time Instance	Supports reading historical data at a specified instance in time using the ReadAtTimeDetails structure.
Server	Historical Access Aggregates	Supports reading one or more Aggregates of historical values of *Variables* using the ReadProcessedDetails structure. At least one of the Aggregates described in Part 13 must be supported.
Server	Historical Access Insert Value	Supports inserting historical values of *Variables*.
Server	Historical Access Delete Value	Supports deleting historical values of *Variables*.
Server	Historical Access Update Value	Supports updating historical values of *Variables*.
Server	Historical Access Replace Value	Supports replacing historical values of *Variables*.
Server	Historical Access Modified Values	Supports maintaining old values for historical data that have been updated and the retrieval of these values using the ReadRawModifiedDetails structure (ReadModified flag set to true).
Server	Historical Access Annotations	Supports the entry and retrieval of Annotations for historical data. The retrieval is accomplished using the standard historical read raw functionality (ReadRawModifiedDetails). The entry uses the standard historical update (UpdateStructureDataDetails) functionality.
Server	Historical Access ServerTimestamp	Supports providing a ServerTimestamp (as well as the default SourceTimestamp).

Category	Title	Description
Server	Historical Access Structured Data Read Raw	Supports ReadRawModified historical access for structured data. Supporting the structure for an annotation is not considered supporting generic structured data.
Server	Historical Access Structured Data Time Instance	Supports historical access for structured data. Supporting ReadAtTimeDetails for structured data. Supporting the structure for an annotation is not considered supporting generic structured data.
Server	Historical Access Structured Data Insert	Supports historical access for structured data. Inserting Structured data. Supporting the structure for an annotation is not considered supporting generic structured data.
Server	Historical Access Structured Data Delete	Supports historical access for structured data. Delete of existing data. Supporting the structure for an annotation is not considered supporting generic structured data.
Server	Historical Access Structured Data Update	Supports historical access for structured data. Updates of existing data. Supporting the structure for an annotation is not considered supporting generic structured data.
Server	Historical Access Structured Data Replace	Supports replacing structured historical data. Supporting the structure for an annotation is not considered supporting generic structured data.
Server	Historical Access Structured Data Read Modified	Supports maintaining old values for historical structured data that have been updated and the retrieval of these values. Using the ReadRawModifiedDetails structure (ReadModified flag set to true) for structured data. Supporting the structure for an annotation is not considered supporting generic structured data.
Server	Historical Access Events	Supports the retrieval of historical Events using the ReadEventDetails structure. This includes support for simple filtering of Events. The *Event* fields that are stored are server specific, but at least the mandatory fields of BaseEventType are required.
Server	Historical Access *Event* Max Events Read Continuation Point	Supports enough continuation points to cover the number of supported *Event* reads indicated in the MaxNodesPerHistoryReadEvents *Server* OperationLimits *Property* for Historical *Event* access.
Server	Historical Access Insert *Event*	Supports inserting historical Events.
Server	Historical Access Update *Event*	Supports updating historical Events.
Server	Historical Access Replace *Event*	Supports replacing historical Events.
Server	Historical Access Delete *Event*	Supports deleting of historical Events.
Client	Historical Access *Client* Browse	Uses the View *Service* Set to discover *Nodes* with historical data.
Client	Historical Access *Client* Read Raw	Uses the HistoryRead *Service* to read raw historical data using the ReadRawModifiedDetails Structure (ReadModified Flag set to False).
Client	Historical Access *Client* Read Modified	Uses the HistoryRead *Service* to read modified historical data using the ReadRawModifiedDetails Structure (ReadModified Flag set to True).
Client	Historical Access *Client* Read Aggregates	Uses the HistoryRead *Service* to read Aggregated historical data. This includes using at least one of the Aggregates defined in Part 13.
Client	Historical Access *Client* Structure Data Raw	Uses the HistoryRead *Service* to read raw historical data using the ReadRawModifiedDetails Structure (ReadModified Flag set to False) for structured data.
Client	Historical Access *Client* Structure Data Read Modified	Uses the HistoryRead *Service* to read modified structured historical data using the ReadRawModifiedDetails Structure (ReadModified Flag set to True).
Client	Historical Access *Client* Structure Data Insert	Uses the HistoryUpdate *Service* to insert historical data values for structured data.

Category	Title	Description
Client	Historical Access *Client* Structure Data Delete	Uses the HistoryUpdate *Service* to delete historical data values for structured data.
Client	Historical Access *Client* Structure Data Update	Uses the HistoryUpdate *Service* to update historical data values for structured data.
Client	Historical Access *Client* Structure Data Replace	Uses the HistoryUpdate *Service* to replace historical data values for structured data.
Client	Historical Access *Client* Structure Data Time Instance	Reads historical data at a specified instance in time for structured data. Using the ReadAtTimeDetails structure.
Client	Historical Access *Client* Read Events	Uses the HistoryRead *Service* to read historical *Event* data using the ReadEventDetails Structure.
Client	Historical Access *Client Event* Inserts	Uses the HistoryUpdate *Service* to insert historical Events.
Client	Historical Access *Client Event* Updates	Uses the HistoryUpdate *Service* to update historical Events.
Client	Historical Access *Client Event* Replaces	Uses the HistoryUpdate *Service* to replace historical Events.
Client	Historical Access *Client Event* Deletes	Uses the HistoryUpdate *Service* to delete historical Events.
Client	Historical Access *Client* Data Insert	Uses the HistoryUpdate *Service* to insert historical data values.
Client	Historical Access *Client* Data Delete	Uses the HistoryUpdate *Service* to delete historical data values.
Client	Historical Access *Client* Data Update	Uses the HistoryUpdate *Service* to update historical data values.
Client	Historical Access *Client* Data Replace	Uses the HistoryUpdate *Service* to replace historical data values.
Client	Historical Access *Client* Annotations	Enters and retrieves Annotations of historical data. The retrieval is accomplished using the standard historical read raw functionality (ReadRawModifiedDetails). The entry uses the standard Historical Update (UpdateStructureDataDetails) functionality.
Client	Historical Access *Client* Time Instance	Reads historical data at a specified instance in time using the ReadAtTimeDetails structure.
Client	Historical Access *Client Server* Timestamp	Uses the ServerTimestamp (as well as the default SourceTimestamp), if it is provided by the *Server*.

Table 18 describes Aggregate related items that can be profiled. *Servers* that support the Aggregates would expose this functionality and *Clients* that utilize Aggregates would implement some of this functionality.

Table 18 – Aggregates

Category	Title	Description
Server	Aggregate Master Configuration	Supports an AggregateConfigurationType *Object* as part of the HistoricalServerCapabilities (defined in UA Part 11).
Server	Aggregate Historical Configuration	Supports at least one AggregateConfigurationType *Object*. AggregateConfigurationType *Objects* occur as part of an HistoricalConfiguration *Object*, allowing *Variable* specific configurations.
Server	Aggregate – Interpolative	Supports the Interpolative Aggregate for Historical access.

Category	Title	Description
Server	Aggregate – Average	Supports the Average Aggregate for Historical access.
Server	Aggregate – TimeAverage	Supports the TimeAverage Aggregate for Historical access.
Server	Aggregate – TimeAverage2	Supports the TimeAverage2 Aggregate for Historical access.
Server	Aggregate – Total	Supports the Total Aggregate for Historical access.
Server	Aggregate – Total2	Supports the Total2 Aggregate for Historical access.
Server	Aggregate – Minimum	Supports the Minimum Aggregate for Historical access.
Server	Aggregate – MinimumActualTime	Supports the MinimumActualTime Aggregate for Historical access.
Server	Aggregate – Minimum2	Supports the Minimum2 Aggregate for Historical access.
Server	Aggregate – MinimumActualTime2	Supports the MinimumActualTime2 Aggregate for Historical access.
Server	Aggregate – Maximum	Supports the Maximum Aggregate for Historical access.
Server	Aggregate – MaximumActualTime	Supports the MaximumActualTime Aggregate for Historical access.
Server	Aggregate – Maximum2	Supports the Maximum2 Aggregate for Historical access.
Server	Aggregate – MaximumActualTime2	Supports the MaximumActualTime2 Aggregate for Historical access.
Server	Aggregate – Range	Supports the Range Aggregate for Historical access.
Server	Aggregate – Range2	Supports the Range2 Aggregate for Historical access.
Server	Aggregate – Count	Supports the Count Aggregate for Historical access.
Server	Aggregate – DurationInStateZero	Supports the DurationInStateZero Aggregate for Historical access.
Server	Aggregate – DurationInStateNonZero	Supports the DurationInStateNonZero Aggregate for Historical access.
Server	Aggregate – NumberOfTransitions	Supports the NumberOfTransitions Aggregate for Historical access.
Server	Aggregate – Start	Supports the Start Aggregate for Historical access.
Server	Aggregate – StartBound	Supports the StartBound Aggregate for Historical access.
Server	Aggregate – End	Supports the End Aggregate for Historical access.
Server	Aggregate – EndBound	Supports the EndBound Aggregate for Historical access.
Server	Aggregate – Delta	Supports the Delta Aggregate for Historical access.
Server	Aggregate – DeltaBounds	Supports the DeltaBounds Aggregate for Historical access.
Server	Aggregate – DurationGood	Supports the DurationGood Aggregate for Historical access.
Server	Aggregate – DurationBad	Supports the DurationBad Aggregate for Historical access.
Server	Aggregate – PercentGood	Supports the PercentGood Aggregate for Historical access.
Server	Aggregate – PercentBad	Supports the PercentBad Aggregate for Historical access.
Server	Aggregate – WorstQuality	Supports the WorstQuality Aggregate for Historical access.
Server	Aggregate – WorstQuality2	Supports the WorstQuality2 Aggregate for Historical access.
Server	Aggregate – AnnotationCount	Supports the AnnotationCount Aggregate for Historical access.
Server	Aggregate – StandardDeviationSample	Supports the StandardDeviationSample Aggregate for Historical access.
Server	Aggregate – VarianceSample	Supports the VarianceSample Aggregate for Historical access.
Server	Aggregate – StandardDeviationPopulation	Supports the StandardDeviationPopulation for Historical access.
Server	Aggregate – VariancePopulation	Supports the VariancePopulation for Historical access.
Server	Aggregate – Custom	The *Server* supports custom Aggregates for Historical access that do not have standard tests defined. These Aggregates are list as untested by this *ConformanceUnit*.
Server	Aggregate *Subscription* – Filter	Supports Aggregate subscription filters which requires at least one of the defined Aggregates is supported as defined in Part 13.
Server	Aggregate *Subscription* – Interpolative	Supports subscription filter for the Interpolative Aggregate.
Server	Aggregate *Subscription* – Average	Supports subscription filter for the Average Aggregate.

Category	Title	Description
Server	Aggregate *Subscription* – TimeAverage	Supports subscription filter for the TimeAverage Aggregate.
Server	Aggregate *Subscription* – TimeAverage2	Supports subscription filter for the TimeAverage2 Aggregate.
Server	Aggregate *Subscription* – Total	Supports subscription filter for the Total Aggregate.
Server	Aggregate *Subscription* – Total2	Supports subscription filter for the Total2 Aggregate.
Server	Aggregate *Subscription* – Minimum	Supports subscription filter for the Minimum Aggregate.
Server	Aggregate *Subscription* – MinimumActualTime	Supports subscription filter for the MinimumActualTime Aggregate.
Server	Aggregate *Subscription* – Minimum2	Supports subscription filter for the Minimum2 Aggregate.
Server	Aggregate *Subscription* – MinimumActualTime2	Supports subscription filter for the MinimumActualTime2 Aggregate.
Server	Aggregate *Subscription* – Maximum	Supports subscription filter for the Maximum Aggregate.
Server	Aggregate *Subscription* – MaximumActualTime	Supports subscription filter for the MaximumActualTime Aggregate.
Server	Aggregate *Subscription* – Maximum2	Supports subscription filter for the Maximum2 Aggregate.
Server	Aggregate *Subscription* – MaximumActualTime2	Supports subscription filter for the MaximumActualTime2 Aggregate.
Server	Aggregate *Subscription* – Range	Supports subscription filter for the Range Aggregate.
Server	Aggregate *Subscription* – Range2	Supports subscription filter for the Range2 Aggregate.
Server	Aggregate *Subscription* – Count	Supports subscription filter for the Count Aggregate.
Server	Aggregate *Subscription* – DurationInStateZero	Supports subscription filter for the DurationInStateZero Aggregate.
Server	Aggregate *Subscription* – DurationInStateNonZero	Supports subscription filter for the DurationInStateNonZero Aggregate.
Server	Aggregate *Subscription* – NumberOfTransitions	Supports subscription filter for the NumberOfTransitions Aggregate.
Server	Aggregate *Subscription* – Start	Supports subscription filter for the Start Aggregate.
Server	Aggregate *Subscription* – StartBound	Supports subscription filter for the StartBound Aggregate.
Server	Aggregate *Subscription* – End	Supports subscription filter for the End Aggregate.
Server	Aggregate *Subscription* – EndBound	Supports subscription filter for the EndBound Aggregate.
Server	Aggregate *Subscription* – Delta	Supports subscription filter for the Delta Aggregate.
Server	Aggregate *Subscription* – DeltaBounds	Supports subscription filter for the DeltaBounds Aggregate.
Server	Aggregate *Subscription* – DurationGood	Supports subscription filter for the DurationGood Aggregate.
Server	Aggregate *Subscription* – DurationBad	Supports subscription filter for the DurationBad Aggregate.
Server	Aggregate *Subscription* – PercentGood	Supports subscription filter for the PercentGood Aggregate.
Server	Aggregate *Subscription* – PercentBad	Supports subscription filter for the PercentBad Aggregate.
Server	Aggregate *Subscription* – WorstQuality	Supports subscription filter for the WorstQuality Aggregate.

Category	Title	Description
Server	Aggregate *Subscription* – WorstQuality2	Supports subscription filter for the WorstQuality2 Aggregate.
Server	Aggregate *Subscription* – AnnotationCount	Supports subscription filter for the AnnotationCount Aggregate.
Server	Aggregate *Subscription* – StandardDeviationSample	Supports subscription filter for the StandardDeviationSample Aggregate.
Server	Aggregate *Subscription* – VarianceSample	Supports subscription filter for the VarianceSample Aggregate.
Server	Aggregate *Subscription* – StandardDeviationPopulation	Supports subscription filter for the StandardDeviationPopulation Aggregate.
Server	Aggregate *Subscription* – VariancePopulation	Supports subscription filter for the VariancePopulation Aggregate.
Server	Aggregate *Subscription* – Custom	The *Server* supports subscribing to custom Aggregates that do not have standard tests defined. These Aggregates are listed as untested by this *ConformanceUnit*.
Client	Aggregate – *Client* Usage	Uses Historical access to Aggregate which requires at least one of the defined Aggregates is supported as defined in Part 13.
Client	Aggregate – *Client* Interpolative	Uses Historical access to the Interpolative Aggregate.
Client	Aggregate – *Client* Average	Uses Historical access to the Average Aggregate.
Client	Aggregate – *Client* TimeAverage	Uses Historical access to the TimeAverage Aggregate.
Client	Aggregate – *Client* TimeAverage2	Uses Historical access to the TimeAverage2 Aggregate.
Client	Aggregate – *Client* Total	Uses Historical access to the Total Aggregate.
Client	Aggregate – *Client* Total2	Uses Historical access to the Total2 Aggregate.
Client	Aggregate – *Client* Minimum	Uses Historical access to the Minimum Aggregate.
Client	Aggregate – *Client* MinimumActualTime	Uses Historical access to the MinimumActualTime Aggregate.
Client	Aggregate – *Client* Minimum2	Uses Historical access to the Minimum2 Aggregate.
Client	Aggregate – *Client* MinimumActualTime2	Uses Historical access to the MinimumActualTime2 Aggregate.
Client	Aggregate – *Client* Maximum	Uses Historical access to the Maximum Aggregate.
Client	Aggregate – *Client* MaximumActualTime	Uses Historical access to the MaximumActualTime Aggregate.
Client	Aggregate – *Client* Maximum2	Uses Historical access to the Maximum2 Aggregate.
Client	Aggregate – *Client* MaximumActualTime2	Uses Historical access to the MaximumActualTime2 Aggregate.
Client	Aggregate – *Client* Range	Uses Historical access to the Range Aggregate.
Client	Aggregate – *Client* Range2	Uses Historical access to the Range2 Aggregate.
Client	Aggregate – *Client* Count	Uses Historical access to the Count Aggregate.
Client	Aggregate – *Client* DurationInStateZero	Uses Historical access to the DurationInStateZero Aggregate.
Client	Aggregate – *Client* DurationInStateNonZero	Uses Historical access to the DurationInStateNonZero Aggregate.
Client	Aggregate – *Client* NumberOfTransitions	Uses Historical access to the NumberOfTransitions Aggregate.
Client	Aggregate – *Client* Start	Uses Historical access to the Start Aggregate.
Client	Aggregate – *Client* StartBound	Uses Historical access to the StartBound Aggregate.
Client	Aggregate – *Client* End	Uses Historical access to the End Aggregate.
Client	Aggregate – *Client* EndBound	Uses Historical access to the EndBound Aggregate.
Client	Aggregate – *Client* Delta	Uses Historical access to the Delta Aggregate.
Client	Aggregate – *Client* DeltaBounds	Uses Historical access to the DeltaBounds Aggregate.

Category	Title	Description
Client	Aggregate – *Client* DurationGood	Uses Historical access to the DurationGood Aggregate.
Client	Aggregate – *Client* DurationBad	Uses Historical access to the DurationBad Aggregate.
Client	Aggregate – *Client* PercentGood	Uses Historical access to the PercentGood Aggregate.
Client	Aggregate – *Client* PercentBad	Uses Historical access to the PercentBad Aggregate.
Client	Aggregate – *Client* WorstQuality	Uses Historical access to the WorstQuality Aggregate.
Client	Aggregate – *Client* WorstQuality2	Uses Historical access to the WorstQuality2 Aggregate.
Client	Aggregate – *Client* AnnotationCount	Uses Historical access to the AnnotationCount Aggregate.
Client	Aggregate – *Client* StandardDeviationSample	Uses Historical access to the StandardDeviationSample Aggregate.
Client	Aggregate – *Client* VarianceSample	Uses Historical access to the VarianceSample Aggregate.
Client	Aggregate – *Client* StandardDeviationPopulation	Uses Historical access to the StandardDeviationPopulation Aggregate.
Client	Aggregate – *Client* VariancePopulation	Uses Historical access to the VariancePopulation Aggregate.
Client	Aggregate – *Client* Custom Aggregates	The *Client* can make use of all custom Aggregates in the list of Aggregates, via Historical access, exposed by the *Server*. This includes displaying or utilizing the data in some manner.
Client	Aggregate *Subscription* – *Client* Filter	Subscribes for data using Aggregate filters which requires at least one of the Aggregates defined in Part 13 is supported.
Client	Aggregate *Subscription* – *Client* Interpolative	Subscribes for data using the Interpolative Aggregate filter.
Client	Aggregate *Subscription* – *Client* Average	Subscribes for data using the Average Aggregate filter.
Client	Aggregate *Subscription* – *Client* TimeAverage	Subscribes for data using the TimeAverage Aggregate filter.
Client	Aggregate *Subscription* – *Client* TimeAverage2	Subscribes for data using the TimeAverage2 Aggregate filter.
Client	Aggregate *Subscription* – *Client* Total	Subscribes for data using the Total Aggregate filter.
Client	Aggregate *Subscription* – *Client* Total2	Subscribes for data using the Total2 Aggregate filter.
Client	Aggregate *Subscription* – *Client* Minimum	Subscribes for data using the Minimum Aggregate filter.
Client	Aggregate *Subscription* – *Client* MinimumActualTime	Subscribes for data using the MinimumActualTime Aggregate filter.
Client	Aggregate *Subscription* – *Client* Minimum2	Subscribes for data using the Minimum2 Aggregate filter.
Client	Aggregate *Subscription* – *Client* MinimumActualTime2	Subscribes for data using the MinimumActualTime2 Aggregate filter.
Client	Aggregate *Subscription* – *Client* Maximum	Subscribes for data using the Maximum Aggregate filter.
Client	Aggregate *Subscription* – *Client* MaximumActualTime	Subscribes for data using the MaximumActualTime Aggregate filter.
Client	Aggregate *Subscription* – *Client* MaximumActualTime2	Subscribes for data using the MaximumActualTime2 Aggregate filter.
Client	Aggregate *Subscription* – *Client* Maximum2	Subscribes for data using the Maximum2 Aggregate filter.
Client	Aggregate *Subscription* – *Client* Range	Subscribes for data using the Range Aggregate filter.
Client	Aggregate *Subscription* – *Client* Range2	Subscribes for data using the Range2 Aggregate filter.

Category	Title	Description
Client	Aggregate *Subscription* – *Client* Count	Subscribes for data using the Count Aggregate filter.
Client	Aggregate *Subscription* – *Client* DurationInStateZero	Subscribes for data using the DurationInStateZero Aggregate filter.
Client	Aggregate *Subscription* – *Client* DurationInStateNonZero	Subscribes for data using the DurationInStateNonZero Aggregate filter.
Client	Aggregate *Subscription* – *Client* NumberOfTransitions	Subscribes for data using the NumberOfTransitions Aggregate filter.
Client	Aggregate *Subscription* – *Client* Start	Subscribes for data using the Start Aggregate filter.
Client	Aggregate *Subscription* – *Client* StartBound	Subscribes for data using the StartBound Aggregate filter.
Client	Aggregate *Subscription* – *Client* End	Subscribes for data using the End Aggregate filter.
Client	Aggregate *Subscription* – *Client* EndBound	Subscribes for data using the EndBound Aggregate filter.
Client	Aggregate *Subscription* – *Client* Delta	Subscribes for data using the Delta Aggregate filter.
Client	Aggregate *Subscription* – *Client* DeltaBounds	Subscribes for data using the DeltaBounds Aggregate filter.
Client	Aggregate *Subscription* – *Client* DurationGood	Subscribes for data using the DurationGood Aggregate filter.
Client	Aggregate *Subscription* – *Client* DurationBad	Subscribes for data using the DurationBad Aggregate filter.
Client	Aggregate *Subscription* – *Client* PercentGood	Subscribes for data using the PercentGood Aggregate filter.
Client	Aggregate *Subscription* – *Client* PercentBad	Subscribes for data using the PercentBad Aggregate filter.
Client	Aggregate *Subscription* – *Client* WorstQuality	Subscribes for data using the WorstQuality Aggregate filter.
Client	Aggregate *Subscription* – *Client* WorstQuality2	Subscribes for data using the WorstQuality2 Aggregate filter.
Client	Aggregate *Subscription* – *Client* AnnotationCount	Subscribes for data using the AnnotationCount Aggregate filter.
Client	Aggregate *Subscription* – *Client* StandardDeviationSample	Subscribes for data using the StandardDeviationSample Aggregate filter.
Client	Aggregate *Subscription* – *Client* VarianceSample	Subscribes for data using the VarianceSample Aggregate filter.
Client	Aggregate *Subscription* – *Client* StandardDeviationPopulation	Subscribes for data using the StandardDeviationPopulation Aggregate filter.
Client	Aggregate *Subscription* – *Client* VariancePopulation	Subscribes for data using the VariancePopulation Aggregate filter.
Client	Aggregate *Subscription* – *Client* Custom Aggregates	The *Client* supports subscribing to all custom Aggregates in the list of Aggregates exposed by the *Server*. This includes displaying or utilizing the data in some manner.

Table 19 describes auditing related items that can be profiled. Most full function *Servers* would support these features, although some resource constrained *Servers* may not provide this functionality. *Clients* that are security aware or are used to support security logging would support these features

Table 19 – Auditing

Category	Title	Description
Server	Auditing Base	Support AuditEvents. The list of supported AuditEvents shall be verified during certification testing and will be shown in the certification test result. Base AuditEvents are defined in Part 3 and in Part 5.
Client	Auditing *Client* Audit ID	*Client* supports generating AuditEvents ids and providing them to *Servers*.
Client	Auditing *Client* Subscribes	The *Client* supports subscribing for AuditEvents and storing / processing them in a secure manner.

Table 20 describes Redundancy related items that are profiled. *Servers* that support redundancy would support appropriate *ConformanceUnits* based on the type of redundancy they support. *Clients* that are capable of handling redundancy would support the appropriate *ConformanceUnits* based of the type of redundancy they support.

Table 20 – Redundancy

Category	Title	Description
Server	Redundancy *Server*	Supports *Server* based redundancy.
Server	Redundancy *Server* Transparent	Supports transparent *Server* redundancy.
Client	Redundancy *Client*	*Client* supports *Client* redundancy. *Clients* that support *Client* redundancy can failover to another *Client* (requires some out of band communication).
Client	Redundancy *Client* Switch	*Clients* supporting this *ConformanceUnit* monitor the redundancy status for non-transparent redundancy *Servers* and switch to the backup *Server* when they recognize a change in server status.

Table 21 describes items for a Global *Discovery Server* (GDS). *Servers* that act as a GDS would support these *ConformanceUnits*.

Table 21 – Global Discovery Server

Category	Title	Description
Global Directory *Service*	GDS Application Directory	Supports the Directory *Object* with all Methods like RegisterApplication and QueryServers.
Global Directory *Service*	GDS Query Applications	Supports the QueryApplications *Method* on the Directory *Object* specified in Part 12.
Global Directory *Service*	GDS LDS-ME Connectivity	The GDS can be configured to use specific LDS-ME installations for semi-automatic application registration for all *Servers* on a subnet.
Global Directory *Service*	GDS *Certificate* Manager Pull Model	This Conformance Unit requires support of the complete Information Model and *Services* for *Certificate* management including the Pull Model as specified in Part 12.
Global Directory *Service*	GDS *Certificate* Manager Push Model	This Conformance Unit requires use of the complete Information model and *Services* for the *Certificate* management Push Model as specified in UA Part 12.
Global Directory *Service*	GDS Key Credential *Service* Pull Model	This Conformance Unit requires support of the complete Information Model and *Services* for KeyCredential Pull Management as specified in UA Part 12.
Global Directory *Service*	GDS Key Credential *Service* Push Model	This Conformance Unit requires use of the complete Information model and *Services* for KeyCredential Push Management as specified in UA Part 12.
Global Directory *Service*	GDS Authorization *Service Server*	This Conformance Unit requires support of AuthorizationServiceType Objects as specified in Part 12. UA Clients use the RequestAccessToken Method on these Objects to request an Access Token from an Identity Provider.

5.5 Miscellaneous

The following table describes miscellaneous *ConformanceUnits.*

Table 22 – Miscellaneous

Category	Title	Description
Server	Documentation – Supported *Profiles*	The documentation includes a description of the profiles supported by the product. This description includes the level of Certification testing the product has passed.
Server	Documentation – Multiple Languages	The documentation is available in multiple languages. The results of this conformance unit include the list of supported languages.
Server	Documentation – Users Guide	The application includes documentation that describes the available functionality provided by the application. For *Servers* it includes a summary of all functionality provided by the *Server*.
Server	Documentation – On-line	The documentation provided by the application is available in electronic format as part of the application. The electronic documentation, could be a WEB page, installed document or CD/DVD, but in all case it can be accessed from the application or from a link installed with the application.
Server	Documentation – Installation	The application includes installation instructions that are sufficient to easily install the application. This includes descriptions of any and all possible configuration items. Instructions for loading or configuring security related items such as Application Instance *Certificates*.
Server	Documentation – Trouble Shooting Guide	The application includes documentation that describes typical problems a user may encounter and actions that the user could perform to resolve the problem. It could also describe tip, tricks or other actions that could help a user diagnose or fix a problem. It could also describe tools or other items that can be used in diagnosing or repairing problems. The actual Trouble Shooting Guide can be part of other documentation, but should be complete enough to provide useful information to a novice user.
Client	Documentation *Client* – Supported *Profiles*	The documentation includes a description of the profiles supported by the product. This description includes any software certificates that describes the level of Certification testing the product has passed.
Client	Documentation *Client* – Multiple Languages	The documentation is available in multiple languages. The results of this conformance unit include the list of supported languages.
Client	Documentation *Client* – Users Guide	The application includes documentation that describes the available functionality provided by the application. For client applications this includes any operator restrictions or general functionality that the client application makes use of.
Client	Documentation *Client* – On-line	The documentation provided by the application is available in electronic format as part of the application. The electronic documentation could be a WEB page, installed document or CD/DVD, but in all cases it can be accessed from the application or from a link installed with the application.
Client	Documentation *Client* – Installation	The application includes installation instructions that are sufficient to easily install the application. This includes descriptions of any and all possible configuration items. Instructions for loading or configuring security related items such as Application Instance *Certificates*.
Client	Documentation *Client* – Trouble Shooting Guide	The application includes documentation that describes typical problems a user may encounter and actions that the user could perform to resolve the problem. It could also describe tips, tricks or other actions that could help a user diagnose or fix a problem. It could also describe tools or other items that can be used in diagnosing or repairing problems. The actual Trouble Shooting Guide can be part of other documentation, but should be complete enough to provide useful information to a novice user.

Category	Title	Description
Security	Best Practice – Timeouts	The user is able to configure reasonable timeouts for Secure Channels, sessions and subscriptions to limit Denial of *Service* and resource consumption issues (see Part 2 for additional details).
Security	Best Practice – Strict *Message* Handling	The application assures that messages that are illegally or incorrectly formed are rejected with appropriate error code or appropriate actions as specified in Part 4 and Part 6.
Security	Best Practice – Random Numbers	All random numbers that are required for security use appropriate cryptographic library based random number generators.
Security	Best Practice – Administrative Access	The *Server* and *Client* allow for appropriate restriction of access to administrative personnel. This includes multiple levels of administrative access on platforms that support multiple administrative roles (such as Windows or Linux).
Security	Best Practice – *Alarm* Handling	A *Server* should restrict critical alarm functionality to users that have the appropriate rights to perform these actions. This would include disabling or alarms, shelving of alarms and generation of dialog messages. It would also include other security related functionality such maintaining appropriate timeouts for shelving and dialogs and preventing an overload of dialog messages.
Security	Best Practice – Audit Events	Subscriptions for Audit Events are restricted to authorized personnel. A *Server* may also reject a *Subscription* for Audit Events that is not over a Secure Channel if one is available.
Security	Best Practice – Audit Events *Client*	Audit tracking system connects to a *Server* using a Secure Channel and under the appropriate administrative rights to allow access to Audit Events.

6 Profiles

6.1 Overview

This section includes a listing of the categories that a *Profile* can be grouped into, a list of named *Profiles* and the detailed listing of each *Profile* including directly defined *ConformanceUnits* and any sub *Profiles* that are included in the *Profile*.

6.2 Profile list

Table 23 lists *Profiles*. The *Profile* table is ordered by *Profile* category and then alphabetically by the name of the *Profile*. The table includes a list of categories the *Profile* is associated with and a URI. The URI is used to uniquely identify a *Profile*. The URI shall be able to be used to access the information provided in this document with regard to the given *Profile* in an on-line display.

An application (*Client* or *Server*) shall implement all of the *ConformanceUnits* in a *Profile* in order to be compliant with the *Profile*. Some *Profiles* contain optional *ConformanceUnits*. An optional *ConformanceUnit* means that an application has the option to not support the *ConformanceUnit*. However, if supported, the application shall pass all tests associated with the *ConformanceUnit*. For example, some *ConformanceUnits* require specific information model items to be available. They are, therefore, listed as optional in order to allow for the information model items to be omitted. If a *Server* desires to be listed as supporting the optional *ConformanceUnit* then it shall include any required information model items in the configuration provided for certification testing. The test result that is generated by the certification testing lists all optional *ConformanceUnits* and whether they are supported or not by the tested UA application. Some *ConformanceUnits* also include lists of supported DataTypes or optional Subtypes, the list are handled in the same manner as optional *ConformanceUnits*. All reporting requirements for optional *ConformanceUnits* also apply to these lists of supported DataTypes or Subtypes.

Table 23 – Profile list

Profile	Related Category	URI
Core Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/CoreFacet
Core 2017 Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/Core2017Facet
Sessionless *Server* Facet	*Server*	http://opcfoundation.org/UA-*Profile*/*Server*/SessionLess
Reverse Connect Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ReverseConnect
Base Server Behaviour Facet	Server	http://opcfoundation.org/UA-Profile/Server/Behaviour
Request State Change Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/RequestStateChange
Subnet Discovery Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/SubnetDiscovery
Global Certificate Management Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/GlobalCertificateManagement
Authorization Service Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/AuthorizationServiceConfiguration
KeyCredential Service Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/KeyCredentialManagement
Attribute WriteMask Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/AttributeWriteMask
File Access Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/FileAccess
Documentation Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/Documentation
Embedded DataChange Subscription Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/EmbeddedDataChangeSubscription
Standard DataChange Subscription Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/StandardDataChangeSubscription
Standard DataChange Subscription 2017 Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/StandardDataChangeSubscription2017
Enhanced DataChange Subscription Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/EnhancedDataChangeSubscription
Enhanced DataChange Subscription 2017 Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/EnhancedDataChangeSubscription2017
Durable Subscription Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/DurableSubscription
Data Access Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/DataAccess
ComplexType Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ComplexTypes
ComplexType 2017 Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ComplexTypes2017
Standard Event Subscription Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/StandardEventSubscription
Address Space Notifier Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/AddressSpaceNotifier
A & C Base Condition Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACBaseCondition
A & C Refresh2 Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACRefresh2
A & C Address Space Instance Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACAddressSpaceInstance
A & C Enable Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACEnable
A & C AlarmMetrics Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACAlarmMetrics
A & C Alarm Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACAlarm
A & C Acknowledgeable Alarm Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACAckAlarm
A & C Exclusive Alarming Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACExclusiveAlarming
A & C Non-Exclusive Alarming Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACNon-ExclusiveAlarming
A & C Previous Instances Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACPreviousInstances
A & C Dialog Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACDialog
A & C CertificateExpiration Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/ACCertificateExpiration
A & E Wrapper Facet	Server	http://opcfoundation.org/UA-Profile/Server/AEWrapper
Method Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/Methods
Auditing Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/Auditing
Node Management Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/NodeManagement
User Role Base Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/UserRoleBase
User Role Management Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/UserRoleManagement
State Machine Server Facet	Server	http://opcfoundation.org/UA-*Profile*/*Server*/StateMachine
Client Redundancy Server Facet	Server	http://opcfoundation.org/UA-*Profile*/*Server*/ClientRedundancy
Redundancy Transparent Server Facet	Server	http://opcfoundation.org/UA-*Profile*/*Server*/TransparentRedundancy
Redundancy Visible Server Facet	Server	http://opcfoundation.org/UA-*Profile*/*Server*/VisibleRedundancy
Historical Raw Data Server Facet	Server	http://opcfoundation.org/UA-*Profile*/*Server*/HistoricalRawData
Historical Aggregate Server Facet	Server	http://opcfoundation.org/UA-*Profile*/*Server*/AggregateHistorical
Historical Data AtTime Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalDataAtTime
Historical Access Modified Data Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalModifiedData
Historical Annotation Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalAnnotation

Profile	Related Category	URI
Historical Data Insert Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalDataInsert
Historical Data Update Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalDataUpdate
Historical Data Replace Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalDataReplace
Historical Data Delete Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalDataDelete
Historical Access Structured Data Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalStructuredData
Base Historical Event Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/BaseHistoricalEvent
Historical Event Update Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalEventUpdate
Historical Event Replace Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalEventReplace
Historical Event Insert Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalEventInsert
Historical Event Delete Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/HistoricalEventDelete
Aggregate Subscription Server Facet	Server	http://opcfoundation.org/UA-Profile/Server/AggregateSubscription
Nano Embedded Device Server Profile	Server	http://opcfoundation.org/UA-Profile/Server/NanoEmbeddedDevice
Nano Embedded Device 2017 Server Profile	Server	http://opcfoundation.org/UA-Profile/Server/NanoEmbeddedDevice2017
Micro Embedded Device Server Profile	Server	http://opcfoundation.org/UA-Profile/Server/MicroEmbeddedDevice
Micro Embedded Device 2017 Server Profile	Server	http://opcfoundation.org/UA-Profile/Server/MicroEmbeddedDevice2017
Embedded UA Server Profile	Server	http://opcfoundation.org/UA-Profile/Server/EmbeddedUA
Embedded 2017 UA Server Profile	Server	http://opcfoundation.org/UA-Profile/Server/EmbeddedUA2017
Standard UA Server Profile	Server	http://opcfoundation.org/UA-Profile/Server/StandardUA
Standard 2017 UA Server Profile	Server	http://opcfoundation.org/UA-Profile/Server/StandardUA2017
Core *Client* Facet	*Client*	http://opcfoundation.org/UA-*Profile*/*Client*/Core
Core 2017 *Client* Facet	*Client*	http://opcfoundation.org/UA-*Profile*/*Client*/Core2017
Sessionless *Client* Facet	*Client*	http://opcfoundation.org/UA-*Profile*/*Client*/SessionLess
Reverse Connect Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ReverseConnect
Base Client Behaviour Facet	Client	http://opcfoundation.org/UA-Profile/Client/Behaviour
Discovery Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/Discovery
Subnet Discovery Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/SubnetDiscovery
Global Discovery Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/GlobalDiscovery
Global Certificate Management Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/GlobalCertificateManagement
KeyCredential Service Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/KeyCredentialManagement
Access Token Request Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/AccessTokenRequest
AddressSpace Lookup Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/AddressSpaceLookup
Request State Change Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/RequestStateChange
File Access Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/FileAccess
Entry Level Support 2015 Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/EntryLevelSupport2015
Multi-Server Client Connection Facet	Client	http://opcfoundation.org/UA-Profile/Client/MultiServer
Documentation – Client	Client	http://opcfoundation.org/UA-Profile/Client/Documentation
Attribute Read Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/AttributeRead
Attribute Write Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/AttributeWrite
DataChange Subscriber Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/DataChangeSubscriber
Durable Subscription Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/DurableSubscription
DataAccess Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/DataAccess
Event Subscriber Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/EventSubscriber
Base Event Processing Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/BaseEventProcessing
Notifier and Source Hierarchy Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/NotifierAndSourceHierarchy
A & C Base Condition Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACBaseCondition
A & C Refresh2 Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACRefresh2
A & C Address Space Instance Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACAddressSpaceInstance
A & C Enable Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACEnable
A & C AlarmMetrics Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACAlarmMetrics
A & C Alarm Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACAlarm
A & C Exclusive Alarming Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACExclusiveAlarming
A & C Non-Exclusive Alarming Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACNon-ExclusiveAlarming
A & C Previous Instances Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACPreviousInstances
A & C Dialog Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACDialog
A & C CertificateExpiration Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/ACCertificateExpiration
A & E Proxy Facet	Client	http://opcfoundation.org/UA-Profile/Client/AEProxy
Method Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/Method
Auditing Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/Auditing

Profile	Related Category	URI
Node Management Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/NodeManagement
Advanced Type Programming Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/TypeProgramming
User Role Management Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/UserRoleManagement
State Machine Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/StateMachine
Diagnostic Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/Diagnostic
Redundant Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/Redundancy
Redundancy Switch Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/RedundancySwitch
Historical Access Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalAccess
Historical Data AtTime Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalAccessAtTime
Historical Aggregate Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalAccessAggregate
Historical Annotation Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalAnnotation
Historical Access Modified Data Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalAccessModifiedData
Historical Data Insert Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalInsertData
Historical Data Update Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalUpdateData
Historical Data Replace Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalReplaceData
Historical Data Delete Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalDeleteData
Historical Access Client Server Timestamp Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalServerTimeStamp
Historical Structured Data Access Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalAccessStructuredData
Historical Structured Data AtTime Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalAtTimeStructuredData
Historical Structured Data Modified Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalModifiedStructuredData
Historical Structured Data Insert Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalInsertStructuredData
Historical Structured Data Update Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalUpdateStructuredData
Historical Structured Data Replace Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalReplaceStructuredData
Historical Structured Data Delete Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalDeleteStructuredData
Historical Events Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalEvents
Historical Event Insert Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalInsertEvents
Historical Event Update Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalUpdateEvents
Historical Event Replace Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalReplaceEvents
Historical Event Delete Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/HistoricalDeleteEvents
Aggregate Subscriber Client Facet	Client	http://opcfoundation.org/UA-Profile/Client/AggregateSubscriber
Standard UA Client Profile	Client	http://opcfoundation.org/UA-Profile/Client/Standard
Standard UA Client 2017 Profile	Client	http://opcfoundation.org/UA-Profile/Client/Standard2017
UA-TCP UA-SC UA-Binary	Transport	http://opcfoundation.org/UA-Profile/Transport/uatcp-uasc-uabinary
HTTPS UA-Binary	Transport	http://opcfoundation.org/UA-Profile/Transport/https-uabinary
HTTPS UA-XML	Transport	http://opcfoundation.org/UA-Profile/Transport/https-uasoapxml
HTTPS UA-JSON	Transport	http://opcfoundation.org/UA-Profile/Transport/https-uajson
WSS UA-SC UA-Binary	Transport	http://opcfoundation.org/UA-Profile/Transport/wss-uasc-uabinary
WSS UA-JSON	Transport	http://opcfoundation.org/UA-Profile/Transport/wss-uajson
Security User Access Control Full	Security, Server	http://opcfoundation.org/UA-Profile/Security/UserAccessFull
Security User Access Control Base	Security, Server	http://opcfoundation.org/UA-Profile/Security/UserAccessBase
Security Time Synchronization	Security	http://opcfoundation.org/UA-Profile/Security/TimeSync
Best Practice – Audit Events	Security, Server	http://opcfoundation.org/UA-Profile/Security/BestPracticeAuditEvents
Best Practice – Alarm Handling	Security, Server	http://opcfoundation.org/UA-Profile/Security/BestPracticeAlarmHandling
Best Practice – Random Numbers	Security	http://opcfoundation.org/UA-Profile/Security/BestPracticeRandomNumbers
Best Practice – Timeouts	Security	http://opcfoundation.org/UA-Profile/Security/BestPracticeTimeouts
Best Practice – Administrative Access	Security	http://opcfoundation.org/UA-Profile/Security/BestPracticeAdministrativeAccess
Best Practice – Strict Message Handling	Security, Server	http://opcfoundation.org/UA-Profile/Security/BestPracticeStrictMessage
Best Practice – Audit Events Client	Client, Security	http://opcfoundation.org/UA-Profile/Security/BestPracticeAuditEventsClient
TransportSecurity – TLS 1.2	Security	http://opcfoundation.org/UA-Profile/TransportSecurity/TLS-1-2
TransportSecurity – TLS 1.2 with PFS	Security	http://opcfoundation.org/UA-Profile/TransportSecurity/TLS-1-2-PFS
SecurityPolicy – None	Security	http://opcfoundation.org/UA/SecurityPolicy#None

Profile	Related Category	URI
SecurityPolicy [A] - Aes128-Sha256-RsaOaep	Security	http://opcfoundation.org/UA/SecurityPolicy#Aes128_Sha256_RsaOaep
SecurityPolicy [B] – Basic256Sha256	Security	http://opcfoundation.org/UA/SecurityPolicy#Basic256Sha256
SecurityPolicy - Aes256-Sha256-RsaPss	Security	http://opcfoundation.org/UA/SecurityPolicy#Aes256_Sha256_RsaPss
User Token – Anonymous Facet	Security	http://opcfoundation.org/UA-Profile/Security/UserToken/Anonymous
User Token – User Name Password Server Facet	Security, Server	http://opcfoundation.org/UA-Profile/Security/UserToken/Server/UserNamePassword
User Token – X509 Certificate Server Facet	Security, Server	http://opcfoundation.org/UA-Profile/Security/UserToken/Server/X509Certificate
User Token – Issued Token Server Facet	Security, Server	http://opcfoundation.org/UA-Profile/Security/UserToken/Server/IssuedToken
User Token – Issued Token Windows Server Facet	Security, Server	http://opcfoundation.org/UA-Profile/Security/UserToken/Server/IssuedTokenWindows
User Token – JWT Server Facet	Security	http://opcfoundation.org/UA-Profile/Security/UserToken/Server/JsonWebToken
User Token – User Name Password Client Facet	Client, Security	http://opcfoundation.org/UA-Profile/Security/UserToken/Client/UserNamePassword
User Token – X509 Certificate Client Facet	Client, Security	http://opcfoundation.org/UA-Profile/Security/UserToken/Client/X509Certificate
User Token – Issued Token Client Facet	Client, Security	http://opcfoundation.org/UA-Profile/Security/UserToken/Client/IssuedToken
User Token – Issued Token Windows Client Facet	Client, Security	http://opcfoundation.org/UA-Profile/Security/UserToken/Client/IssuedTokenWindows
User Token – JWT Client Facet	Security	http://opcfoundation.org/UA-Profile/Security/UserToken/Client/JsonWebToken
Global Discovery Server Profile	Global Directory Service, Server	http://opcfoundation.org/UA-Profile/Server/GlobalDiscovery
Global Discovery Server 2017 Profile	Global Directory Service, Server	http://opcfoundation.org/UA-Profile/Server/GlobalDiscovery2017
Global Discovery and Certificate Management Server	Global Directory Service, Server	http://opcfoundation.org/UA-Profile/Server/GlobalDiscoveryAndCertificateManagement
Global Discovery and Certificate Mgmt 2017 Server	Global Directory Service, Server	http://opcfoundation.org/UA-Profile/Server/GlobalDiscoveryAndCertificateManagement2017
Global Certificate Management Client Profile	Client, Global Directory Service	http://opcfoundation.org/UA-Profile/Client/GlobalCertificateManagement
Global Certificate Management Client 2017 Profile	Client, Global Directory Service	http://opcfoundation.org/UA-Profile/Client/GlobalCertificateManagement2017
Global Service Authorization Request Server Facet	Global Directory Service	http://opcfoundation.org/UA-Profile/Server/GlobalServiceAuthorization
Global Service KeyCredential Pull Facet	Global Directory Service	http://opcfoundation.org/UA-Profile/Server/GlobalServiceKeyCredentials
Global Service KeyCredential Push Facet	Global Directory Service	http://opcfoundation.org/UA-Profile/Client/GlobalServiceKeyCredentials

The contents of each of the listed *Profiles* will be described in a tabular form in a separate section. Each table may contain references to additional *Profiles* and or *ConformanceUnits*. If

a *Profile* is referenced it means that it is completely included. The *ConformanceUnits* are referenced using their name and conformance group. For the details of the *ConformanceUnit* the reader should examine the *ConformanceUnit* details in the appropriate conformance group section.

6.3 Conventions for Profile definitions

Profiles have the following naming conventions:

- *Profiles* intended for OPC UA *Servers* contain the term *Server* in their titles,
- *Profiles* intended for OPC UA *Clients* contain the term *Client* in their titles
- The term Facet in the title of a *Profile* indicates that this *Profile* is expected to be part of another larger *Profile* or concerns a specific aspect of OPC UA. *Profiles* with the term Facet in their title are expected to be combined with other *Profiles* to define the complete functionality of an OPC UA *Server*, *Client*, Publisher, or Subscriber.

6.4 Profile versioning

Versioning of *Profile* is accomplished with a naming convention. Whenever a profile is revised, the year of the new revision is added to the name. Example:

Version 1	**Core *Server* Facet**
Version 2	Core 2017 *Server* Facet

6.5 Applications

A vendor that is developing a UA application, whether it is a *Server* application or a *Client* application, shall review the list of available *Profiles*. From this list the vendor shall select the *Profiles* that include the functionality required by the application. Typically this will be multiple *Profiles*. Conformance to a single *Profile* may not yield a complete application. In most cases multiple *Profiles* are needed to yield a useful application. All *Servers* and *Clients* shall support at least a core *Profile* (Core *Server Facet* or Core *Client Facet*) and at least one Transport *Profile*

For example an HMI *Client* application may choose to support the “Core *Client Facet*”, the “UA-TCP UA-SC UA-Binary” *Profile*, the “Data Access *Client Facet*”, the “DataChange Subscriber *Client* Facet” and the “*Attribute* Write *Client Facet*”. If the *Client* is to be *TestLab* tested then it would also support “Base *Client* Behaviour” *Profile*. This list of *Profiles* would allow the *Client* to communicate with an OPC UA *Server* using UA-TCP/UA Security/UA binary. It would be able to subscribe for data, write to data and would support the DA data model. It would also follow the best practice guideline for behaviour. Figure 2 illustrates the *Profile* hierarchy that this application may contain: This figure is only an illustration and the represented *Profiles* may change.

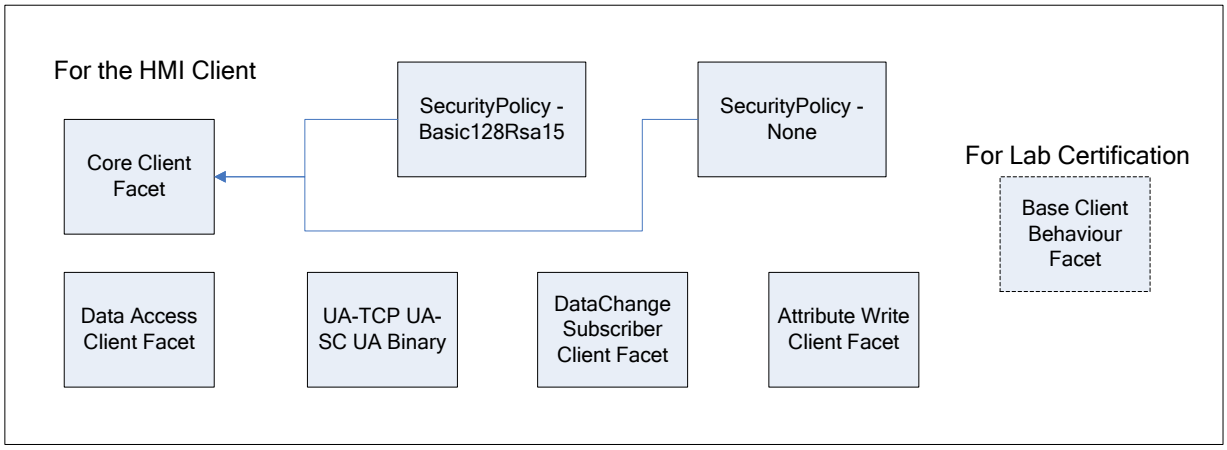

Figure 2 – HMI Client sample

All *Clients* should take into account the types of *Servers* and *Server Profiles* that they are targeted to support. Some *Servers* might not support *Subscriptions* and *Clients* should be able to fall back to Read *Services.*

A special case is a generic *Client* that is designed to communicate with a large number of *Servers* and therefore able to perform a broad range of functionality. ”Standard UA *Client Profile”* has been defined for this kind of *Clients*.

Many *Clients*, however, will be specialized and do not need all of the functionality in the ”Standard UA *Client Profile”* and thus would only support the limited set of functionality they require. A trend *Client*, for example, would only need functionality to subscribe to or read data.

Another example is an embedded device OPC UA *Server* application that may choose to support “Embedded UA *Server*” *Profile* and the “DataAccess *Server* Facet” *Profile*. This device would be a resource constrained device that would support UA-TCP, UA-Security, UA Binary encoding, data subscriptions and the DA data model. It may not support the optional attribute write. Figure 3 illustrates the hierarchy that this application may contain: This figure is just an illustration and the represented *Profiles* may change.

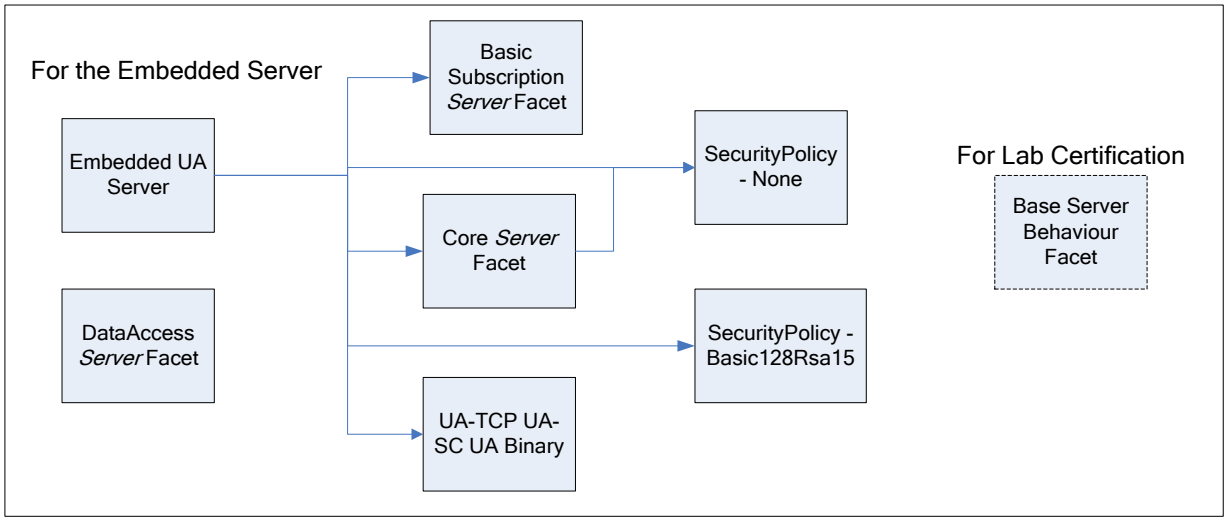

Figure 3 – Embedded Server sample

Another simple system *Server* application may choose to support: “Standard UA *Server*” *Profile* and the “DataAccess *Server* Facet” *Profile*. If the *Server* is to be lab tested then it would also support “Base *Server* Behaviour” *Profile*. This device would be a mid-level OPC UA *Server* that would support all that the embedded *Server* in the previous example supported and it would

add support for an enhance level of the subscription service and support for writes. Figure 4 illustrates the hierarchy that this application may contain: This figure is just an illustration and the represented *Profile* may change.

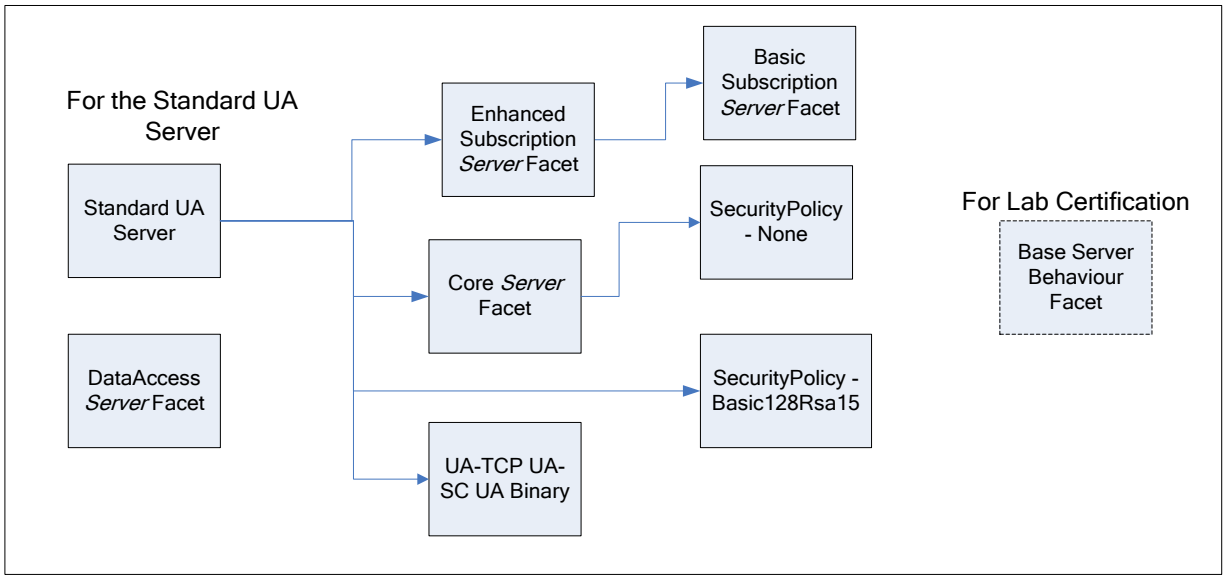

Figure 4 – Standard UA Server sample

If the example HMI *Client* were to connect to either of the example *Servers*, it may have to adjust its behavior based on the *Profile* reported by the respective *Servers*. If the HMI *Client* were communicating with the embedded device it would not be able to perform any write operations. It may also have to limit the number of subscriptions or sessions based on the performance limits of the *Server*. If the HMI *Client* is connected to the Standard *Server* it would be able to open additional windows, have higher limits on performance related items and it would be able to allow writes.

6.6 Profile tables

6.6.1 Introduction

The following sections describe *Profiles* in a tabular format.

Each table contains three columns. The first column is a description of the conformance group that the *ConformanceUnit* is part of. This allows the reader to easily find the *ConformanceUnit*. This column may also state "*Profile*" in which case the listed item is not a *ConformanceUnit*, but an included *P rofile.* The second column is a brief description of the *ConformanceUnit* or included *Profile*. The last column indicates if the *ConformanceUnit* is optional or required.

6.6.2 Core Server Facet

Table 24 describes the details of the Core *Server* Facet. This Facet defines the core functionality required for any UA *Server* implementation. The core functionality includes the ability to discover endpoints, establish secure communication channels, create Sessions, browse the *AddressSpace* and read and/or write to *Attributes* of *Nodes*. The key requirements are: support for a single *Session*, support for the *Server* and *Server* Capabilities *Object*, all mandatory *Attributes* for *Nodes* in the *AddressSpace*, and authentication with UserName and Password. This Facet has been extended with additional Base Information *ConformanceUnits*. They are optional to provide backward compatibility. In the future the *ConformanceUnit* "Base Info *Server* Capabilities" will become required, and so it is highly recommended that all *Servers* support it. For broad applicability, it is recommended that *Servers* support multiple transport and security *Profiles*.

Table 24 – Core Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	SecurityPolicy – None	False
Profile	User Token – User Name Password *Server* Facet	False
Address Space Model	Address Space Base	False
Attribute Services	*Attribute* Read	False
Attribute Services	*Attribute* Write Index	True
Attribute Services	*Attribute* Write Values	True
Base Information	Base Info Core Structure	False
Base Information	Base Info OptionSet	True
Base Information	Base Info Placeholder Modelling Rules	True
Base Information	Base Info *Server* Capabilities	True
Base Information	Base Info ValueAsText	True
Discovery Services	*Discovery* Find *Servers* Self	False
Discovery Services	*Discovery* Get Endpoints	False
Security	Security – No Application Authentication	True
Security	Security Administration	True
Session Services	*Session* Base	False
Session Services	*Session* General *Service* Behaviour	False
Session Services	*Session* Minimum 1	False
View *Services*	View Basic	False
View *Services*	View Minimum Continuation Point 01	False
View *Services*	View RegisterNodes	False
View *Services*	View TranslateBrowsePath	False

6.6.3 Core 2017 Server Facet

Table 25 describes the details of the Core 2017 *Server* Facet. This Facet defines the core functionality required for any UA *Server* implementation. The core functionality includes the ability to discover endpoints, establish secure communication channels, create Sessions, browse the *AddressSpace* and read and/or write to *Attributes* of *Nodes*. The key requirements are: support for a single *Session*, support for the *Server* and *Server* Capabilities *Object*, all mandatory *Attributes* for *Nodes* in the *AddressSpace*, and authentication with UserName and Password. For broad applicability, it is recommended that *Servers* support multiple transport and security *Profiles*. This Facet supersedes the “Core *Server* Facet”.

Table 25 – Core 2017 Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	SecurityPolicy – None	False
Profile	User Token – User Name Password *Server* Facet	False
Address Space Model	Address Space Atomicity	False
Address Space Model	Address Space Base	False
Address Space Model	Address Space Full Array Only	False
Attribute Services	*Attribute* Read	False
Attribute Services	*Attribute* Write Index	True
Attribute Services	*Attribute* Write Values	True
Base Information	Base Info Core Structure	False
Base Information	Base Info Estimated Return Time	True
Base Information	Base Info OptionSet	True
Base Information	Base Info Placeholder Modelling Rules	True
Base Information	Base Info Selection List	True
Base Information	Base Info *Server* Capabilities	True
Base Information	Base Info ValueAsText	True
Discovery Services	*Discovery* Find *Servers* Self	False
Discovery Services	*Discovery* Get Endpoints	False
Security	Security Administration	True
Security	Security Role *Server* Authorization	True
Session Services	*Session* Base	False

Group	Conformance Unit / *Profile* Title	Optional
Session Services	*Session* General *Service* Behaviour	False
Session Services	*Session* Minimum 1	False
View *Services*	View Basic	False
View *Services*	View Minimum Continuation Point 01	False
View *Services*	View RegisterNodes	False
View *Services*	View TranslateBrowsePath	False

6.6.4 Sessionless Server Facet

Table 26 describes the details of the Sessionless *Server* Facet. Defines the use of Sessionless *Service* invocation in a *Server*.

Table 26 – Sessionless Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Discovery Services	*Discovery* Get Endpoints SessionLess	False
Session Services	*Session* Sessionless Invocation	False

6.6.5 Reverse Connect Server Facet

Table 27 describes the details of the Reverse Connect *Server* Facet. This Facet defines support of reverse connectivity in a *Server*. Usually, a connection is opened by the *Client* before starting the UA-specific handshake. This will fail, however, when *Servers* are behind firewalls with no open ports to connect to. In the reverse connectivity scenario, the *Server* opens the connection and starts with a ReverseHello message requesting that the *Client* establish a Secure Channel using this connection.

Table 27 – Reverse Connect Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Protocol and Encoding	Protocol Reverse Connect *Server*	False

6.6.6 Base Server Behaviour Facet

Table 28 describes the details of the Base *Server* Behaviour Facet. This Facet defines best practices for the configuration and management of *Servers* when they are deployed in a production environment. It provides the ability to enable or disable certain protocols and to configure the *Discovery Server* and specify where this *Server* shall be registered.

Table 28 – Base Server Behaviour Facet

Group	Conformance Unit / *Profile* Title	Optional
Discovery Services	*Discovery* Configuration	False
Protocol and Encoding	Protocol Configuration	False
Security	Security Administration	False
Security	Security Administration – XML Schema	False
Security	Security *Certificate* Administration	False

6.6.7 Request State Change Server Facet

Table 29 describes the details of the Request State Change *Server* Facet. This Facet specifies the support of the RequestServerStateChange *Method*.

Table 29 – Request State Change Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info RequestServerStateChange *Method*	False

6.6.8 Subnet Discovery Server Facet

Table 30 describes the details of the Subnet *Discovery Server* Facet. Support of this Facet enables discovery of the *Server* on a subnet using mDNS. This functionality is only applicable when *Servers* do not register with an LDS.

Table 30 – Subnet Discovery Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Discovery Services	*Discovery Server* Announcement using mDNS	False

6.6.9 Global Certificate Management Server Facet

Table 31 describes the details of the Global *Certificate* Management *Server* Facet. This Facet defines the capability to interact with a Global *Certificate* Management *Server* to obtain an initial or renewed *Certificate* and Trust Lists.

Table 31 – Global Certificate Management Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Push Model for Global *Certificate* and TrustList Management	False

6.6.10 Authorization Service Server Facet

Table 32 describes the details of the Authorization *Service Server* Facet. This Facet defines the support for configuring the necessary information to validate access tokens when presented by a Client during session establishment. Access Tokens are issued by Authorization *Services*.

Table 32 – Authorization Service Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Authorization Service Configuration Server	False

6.6.11 KeyCredential Service Server Facet

Table 33 describes the details of the KeyCredential *Service Server* Facet. This Facet defines the capability to interact with a KeyCredential *Service* to obtain KeyCredentials. For example KeyCredentials are needed to access an Authorization *Service* or a Broker. The KeyCredential *Service* is typically part of a system-wide tool, like a GDS that also manages Applications, Access Tokens, and *Certificates*.

Table 33 – KeyCredential Service Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Push Model for KeyCredential *Service*	False

6.6.12 Attribute WriteMask Server Facet

Table 34 describes the details of the *Attribute* WriteMask *Server* Facet. This Facet defines the capability to update characteristics of individual *Nodes* in the *AddressSpace* by allowing writing to *Node Attributes*. It requires support for authenticating user access as well as providing information related to access rights in the *AddressSpace* and actually restricting the access rights as described.

Table 34 – Attribute WriteMask Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Security User Access Control Base	False
Address Space Model	Address Space UserWriteMask	False
Address Space Model	Address Space UserWriteMask Multilevel	True
Address Space Model	Address Space WriteMask	False

6.6.13 File Access Server Facet

Table 35 describes the details of the File Access *Server* Facet. This Facet specifies the support of exposing File information via the defined FileType. This includes reading of file as well as optionally writing of file data.

Table 35 – File Access Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info FileType Base	False
Base Information	Base Info FileType Write	True

6.6.14 Documentation Server Facet

Table 36 describes the details of the Documentation *Server* Facet. This Facet defines a list of user documentation that a server application should provide.

Table 36 – Documentation Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Documentation – Installation	False
Miscellaneous	Documentation – Multiple Languages	True
Miscellaneous	Documentation – On-line	True
Miscellaneous	Documentation – Supported *Profiles*	True
Miscellaneous	Documentation – Trouble Shooting Guide	True
Miscellaneous	Documentation – Users Guide	False

6.6.15 Embedded DataChange Subscription Server Facet

Table 37 describes the details of the Embedded DataChange *Subscription Server* Facet. This Facet specifies the minimum level of support for data change notifications within subscriptions. It includes limits which minimize memory and processing overhead required to implement the Facet. This Facet includes functionality to create, modify and delete Subscriptions and to add, modify and remove Monitored Items. As a minimum for each *Session*, *Servers* shall support one *Subscription* with up to two items. In addition, support for two parallel Publish requests is required. This Facet is geared for a platform such as the one provided by the Micro Embedded Device *Server Profile* in which memory is limited and needs to be managed.

Table 37 – Embedded DataChange Subscription Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Monitored Item *Services*	Monitor Basic	False
Monitored Item *Services*	Monitor Items 2	False
Monitored Item *Services*	Monitor QueueSize_1	False
Monitored Item *Services*	Monitor Value Change	False
Subscription Services	*Subscription* Basic	False
Subscription Services	*Subscription* Minimum 1	False
Subscription Services	*Subscription* Publish Discard Policy	False
Subscription Services	*Subscription* Publish Min 02	False

6.6.16 Standard DataChange Subscription Server Facet

Table 38 describes the details of the Standard DataChange *Subscription Server* Facet. This Facet specifies the standard support of subscribing to data changes. This Facet extends features and limits defined by the Embedded Data Change *Subscription* Facet. As a minimum, *Servers* shall support 2 Subscriptions with at least 100 items for at least half of the required Sessions. The 100 items shall be supported for at least half of the required Subscriptions. Queuing with up to two queued entries is required. Support of five parallel Publish requests per *Session* is required. This Facet also requires the support of the triggering service. This Facet has been updated to include optional *ConformanceUnits* to allow for backward compatibility. These optional *ConformanceUnits* are highly recommended, in that in a future release they will be made mandatory.

Table 38 – Standard DataChange Subscription Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Embedded DataChange *Subscription Server* Facet	False

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info GetMonitoredItems *Method*	True
Method Services	*Method* Call	True
Monitored Item *Services*	Monitor Items 10	False
Monitored Item *Services*	Monitor Items 100	False
Monitored Item *Services*	Monitor MinQueueSize_02	False
Monitored Item *Services*	Monitor Triggering	False
Monitored Item *Services*	Monitored Items Deadband Filter	False
Subscription Services	*Subscription* Minimum 02	False
Subscription Services	*Subscription* Publish Min 05	False

6.6.17 Standard DataChange Subscription 2017 Server Facet

Table 39 describes the details of the Standard DataChange *Subscription* 2017 *Server* Facet. This Facet specifies the standard support of subscribing to data changes and extends features and limits defined by the Embedded Data Change *Subscription* Facet. See *ConformanceUnits* for these limits. Note that the *Method* Call *Service* is only required for the *Methods* defined in this Facet. This Facet supersedes the “Standard DataChange *Subscription Server* Facet”.

Table 39 – Standard DataChange Subscription 2017 Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Embedded DataChange *Subscription Server* Facet	False
Base Information	Base Info GetMonitoredItems *Method*	False
Base Information	Base Info ResendData *Method*	False
Method Services	*Method* Call	False
Monitored Item *Services*	Monitor Items 10	False
Monitored Item *Services*	Monitor Items 100	False
Monitored Item *Services*	Monitor MinQueueSize_02	False
Monitored Item *Services*	Monitor Triggering	False
Monitored Item *Services*	Monitored Items Deadband Filter	False
Subscription Services	*Subscription* Minimum 02	False
Subscription Services	*Subscription* Publish Min 05	False

6.6.18 Enhanced DataChange Subscription Server Facet

Table 40 describes the details of the Enhanced DataChange *Subscription Server* Facet. This Facet specifies an enhanced support of subscribing to data changes. It is part of the Standard UA *Server Profile*. This Facet increases the limits defined by the Standard Data Change *Subscription* Facet.

Table 40 – Enhanced DataChange Subscription Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Standard DataChange *Subscription Server* Facet	False
Monitored Item *Services*	Monitor Items 500	False
Monitored Item *Services*	Monitor MinQueueSize_05	False
Subscription Services	*Subscription* Minimum 05	False
Subscription Services	*Subscription* Publish Min 10	False

6.6.19 Enhanced DataChange Subscription 2017 Server Facet

Table 41 describes the details of the Enhanced DataChange *Subscription* 2017 *Server* Facet. This Facet specifies an enhanced support of subscribing to data changes. It is part of the Standard UA *Server* 2017 *Profile*. This Facet increases the limits defined by the Standard Data Change *Subscription* 2017 *Server* Facet.

Table 41 – Enhanced DataChange Subscription 2017 Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Standard DataChange *Subscription* 2017 *Server* Facet	False
Monitored Item *Services*	Monitor Items 500	False
Monitored Item *Services*	Monitor MinQueueSize_05	False
Subscription Services	*Subscription* Minimum 05	False
Subscription Services	*Subscription* Publish Min 10	False

6.6.20 Durable Subscription Server Facet

Table 42 describes the details of the Durable *Subscription Server* Facet. This Facet specifies support of durable storage of data and events even when *Clients* are disconnected. This Facet implies support of any of the DataChange or *Event Subscription* Facets.

Table 42 – Durable Subscription Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Subscription Services	*Subscription* Durable	False
Subscription Services	*Subscription* Durable StorageLevel High	True
Subscription Services	*Subscription* Durable StorageLevel Medium	True
Subscription Services	*Subscription* Durable StorageLevel Small	True

6.6.21 Data Access Server Facet

Table 43 describes the details of the Data Access *Server* Facet. This Facet specifies the support for an Information Model used to provide industrial automation data. This model defines standard structures for analog and discrete data items and their quality of service. This Facet extends the Core *Server* Facet which includes support of the basic *AddressSpace* behaviour.

Table 43 – Data Access Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Data Access	Data Access AnalogItems	True
Data Access	Data Access ArrayItemType	True
Data Access	Data Access Complex Number	True
Data Access	Data Access DataItems	False
Data Access	Data Access DoubleComplex Number	True
Data Access	Data Access MultiState	True
Data Access	Data Access MultiStateValueDiscrete	True
Data Access	Data Access PercentDeadband	True
Data Access	Data Access Semantic Changes	True
Data Access	Data Access TwoState	True

6.6.22 ComplexType Server Facet

Table 44 describes the details of the ComplexType *Server* Facet. This Facet extends the Core *Server* Facet to include *Variables* with Complex Data, i.e. data that are composed of multiple elements such as a structure and where the individual elements are exposed as component variables. Support of this Facet requires the implementation of structured DataTypes and *Variables* that make use of these DataTypes. The Read, Write and Subscriptions service set shall support the encoding and decoding of these structured DataTypes. As an option the *Server* can also support alternate encodings, such as an XML encoding when the binary protocol is currently used and vice-versa.

Table 44 – ComplexType Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space Complex Data Dictionary	False
Attribute Services	*Attribute* Alternate Encoding	True
Attribute Services	*Attribute* Read Complex	False

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Write Complex	False
Monitored Item *Services*	Monitor Alternate Encoding	True

6.6.23 ComplexType 2017 Server Facet

Table 45 describes the details of the ComplexType 2017 *Server* Facet. This Facet extends the Core *Server* Facet to include *Variables* with structured data, i.e. data that are composed of multiple elements such as a structure and where the individual elements are exposed as component variables. Support of this Facet requires the implementation of structured DataTypes and *Variables* that make use of these DataTypes. The Read, Write and Subscriptions service set shall support the encoding and decoding of these structured DataTypes. As an option the *Server* can also support alternate encodings, such as an XML encoding when the binary protocol is currently used and vice-versa.

Table 45 – ComplexType 2017 Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space DataTypeDefinition *Attribute*	False
Attribute Services	*Attribute* Alternate Encoding	True
Attribute Services	*Attribute* Read Complex	False
Attribute Services	*Attribute* Write Complex	False
Monitored Item *Services*	Monitor Alternate Encoding	True
Monitored Item Services	Monitor Complex Value	True

6.6.24 Standard Event Subscription Server Facet

Table 46 describes the details of the Standard *Event Subscription Server* Facet. This Facet specifies the standard support for subscribing to events and is intended to supplement any of the FullFeatured *Profiles*. Support of this Facet requires the implementation of *Event* Types representing the Events that the *Server* can report and their specific fields. It also requires at least the *Server Object* to have the *EventNotifier Attribute* set. It includes the *Services* to Create, Modify and Delete Subscriptions and to Add, Modify and Remove Monitored Items for *Object Nodes* with an "*EventNotifier Attribute*". Creating a monitoring item may include a filter that includes SimpleAttribute FilterOperands and a select list of Operators. The operators include: Equals, IsNull, GreaterThan, LessThan, GreaterThanOrEqual, LessThanOrEqual, Like, Not, Between, InList, And, Or, Cast, BitwiseAnd, BitwiseOr and TypeOf. Support of more complex filters is optional. This Facet has been updated to include several optional Base Information *ConformanceUnits*. These *ConformanceUnits* are optional to allow for backward compatibility, in the future these optional *ConformanceUnits* will become required, and so it is highly recommended that all servers support them.

Table 46 – Standard Event Subscription Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space Events	False
Base Information	Base Info Device Failure	True
Base Information	Base Info EventQueueOverflow EventType	True
Base Information	Base Info Progress Events	True
Base Information	Base Info SemanticChange	True
Base Information	Base Info System Status	True
Base Information	Base Info System Status Underlying System	True
Monitored Item *Services*	Monitor Basic	False
Monitored Item *Services*	Monitor Complex *Event* Filter	True
Monitored Item *Services*	Monitor Events	False
Monitored Item *Services*	Monitor Items 10	False
Monitored Item *Services*	Monitor QueueSize_ServerMax	False
Subscription Services	*Subscription* Basic	False
Subscription Services	*Subscription* Minimum 02	False
Subscription Services	*Subscription* Publish Discard Policy	False
Subscription Services	*Subscription* Publish Min 05	False

6.6.25 Address Space Notifier Server Facet

Table 47 describes the details of the Address Space Notifier *Server* Facet. This Facet requires the support of a hierarchy of *Object Nodes* that are notifiers and *Nodes* that are event sources. The hierarchy is commonly used as a way to organize a plant into areas that can be managed by different operators.

Table 47 – Address Space Notifier Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space Notifier Hierarchy	False
Address Space Model	Address Space Source Hierarchy	False

6.6.26 A & C Base Condition Server Facet

Table 48 describes the details of the A & C Base *Condition Server* Facet. This Facet requires basic support for *Conditions*. Information about *Conditions* is provided through *Event* notifications and thus this Facet builds upon the Standard *Event Subscription Server* Facet. *Conditions* that are in an “interesting” state (as defined by the *Server*) can be refreshed using the Refresh *Method*, which requires support for the *Method Server* Facet. Optionally the server may also provide support for *Condition* classes

Table 48 – A & C Base Condition Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	*Method Server* Facet	False
Profile	Standard *Event Subscription Server* Facet	False
Alarms and *Conditions*	A & C Basic	False
Alarms and *Conditions*	A & C *Condition* Sub-Classes	True
Alarms and *Conditions*	A & C ConditionClasses	True
Alarms and *Conditions*	A & C Refresh	False

6.6.27 A & C Refresh2 Server Facet

Table 49 describes the details of the A & C Refresh2 *Server* Facet. This Facet enhances the A & C Base *Condition Server* Facet with support of the ConditionRefresh2 *Method*.

Table 49 – A & C Refresh2 Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Server* Facet	False
Alarms and *Conditions*	A & C Refresh2	False

6.6.28 A & C Address Space Instance Server Facet

Table 50 describes the details of the A & C Address Space Instance *Server* Facet. This Facet specifies the support required for a *Server* to expose *Alarms* and *Conditions* in its *AddressSpace*. This includes the A & C *AddressSpace* information model.

Table 50 – A & C Address Space Instance Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Alarms and *Conditions*	A & C Instances	False

6.6.29 A & C Enable Server Facet

Table 51 describes the details of the A & C Enable *Server* Facet. This Facet requires the enabling and disabling of *Conditions*. This Facet builds upon the A&C Base *Condition Server* Facet. Enabling and disabling also requires that instances of these ConditionTypes exist in the *AddressSpace* since the enable *Method* can only be invoked on an instance of the *Condition*

Table 51 – A & C Enable Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Server* Facet	False
Alarms and *Conditions*	A & C Enable	False
Alarms and *Conditions*	A & C Instances	False

6.6.30 A & C AlarmMetrics Server Facet

Table 52 describes the details of the A & C AlarmMetrics *Server* Facet. This Facet requires support for AlarmMetrics. AlarmMetrics expose status and potential issues in the alarm system. A *Server* can provide these metrics at various levels (operator station, plant area, overall system etc.).

Table 52 – A & C AlarmMetrics Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Alarms and *Conditions*	A & C *Alarm* Metrics	False

6.6.31 A & C Alarm Server Facet

Table 53 describes the details of the A & C *Alarm Server* Facet. This Facet requires support for *Alarms*. *Alarms* extend the ConditionType by adding an Active state which indicates when something in the system requires attention by an Operator. This Facet builds upon the A&C Base *Condition Server* Facet. This facet requires that discrete AlarmTypes be supported, it also allows for optional support of shelving, alarm comments and other discrete AlarmTypes such as Trip or Off-Normal.

Table 53 – A & C Alarm Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Server* Facet	False
Alarms and *Conditions*	A & C *Alarm*	False
Alarms and *Conditions*	A & C Audible Sound	True
Alarms and *Conditions*	A & C Comment	True
Alarms and *Conditions*	A & C Discrepancy	True
Alarms and *Conditions*	A & C Discrete	False
Alarms and *Conditions*	A & C First in Group *Alarm*	True
Alarms and *Conditions*	A & C OffNormal	True
Alarms and *Conditions*	A & C On-Off Delay	True
Alarms and *Conditions*	A & C Out Of Service	True
Alarms and *Conditions*	A & C Re-Alarming	True
Alarms and *Conditions*	A & C Shelving	True
Alarms and *Conditions*	A & C Silencing	True
Alarms and *Conditions*	A & C Suppression	True
Alarms and *Conditions*	A & C Suppression by Operator	True
Alarms and *Conditions*	A & C SystemOffNormal	True
Alarms and *Conditions*	A & C Trip	True

6.6.32 A & C Acknowledgeable Alarm Server Facet

Table 54 describes the details of the A & C Acknowledgeable *Alarm Server* Facet. This Facet requires support for Acknowledgement of active *Alarms*. This Facet builds upon the A & C *Alarm Server* Facet. Acknowledgement requires support of the Acknowledge *Method* and the Acknowledged state. Support of the Confirmed state and the Confirm *Method* is optional.

Table 54 – A & C Acknowledgeable Alarm Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C *Alarm Server* Facet	False
Alarms and *Conditions*	A & C Acknowledge	False
Alarms and *Conditions*	A & C Confirm	True

6.6.33 A & C Exclusive Alarming Server Facet

Table 55 describes the details of the A & C Exclusive Alarming *Server* Facet. This Facet requires support for *Alarms* with multiple sub-states that identify different limit *Conditions*. This facet builds upon the A&C *Alarm Server* Facet. The term exclusive means only one sub-state can be active at a time. For example, a temperature exceeds the HighHigh limit the associated exclusive LevelAlarm will be in the HighHigh sub-state and not in the High sub-state. This Facet requires that a *Server* support at least one of the optional *Alarm* models: Limit, RateOfChange or Deviation.

Table 55 – A & C Exclusive Alarming Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C *Alarm Server* Facet	False
Alarms and *Conditions*	A & C Exclusive Deviation	True
Alarms and *Conditions*	A & C Exclusive Level	True
Alarms and *Conditions*	A & C Exclusive Limit	False
Alarms and *Conditions*	A & C Exclusive RateOfChange	True

6.6.34 A & C Non-Exclusive Alarming Server Facet

Table 56 describes the details of the A & C Non-Exclusive Alarming *Server* Facet. This Facet requires support for *Alarms* with multiple sub-states that identify different limit *Conditions*. This Facet builds upon the A&C *Alarm Server* Facet. The term non-exclusive means more than one sub-state can be active at a time. For example, if a temperature exceeds the HighHigh limit the associated non-exclusive LevelAlarm will be in both the High and the HighHigh sub-state. This Facet requires that a server support at least one of the optional alarm models: Limit, RateOfChange or Deviation.

Table 56 – A & C Non-Exclusive Alarming Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C *Alarm Server* Facet	False
Alarms and *Conditions*	A & C Non-Exclusive Deviation	True
Alarms and *Conditions*	A & C Non-Exclusive Level	True
Alarms and *Conditions*	A & C Non-Exclusive Limit	False
Alarms and *Conditions*	A & C Non-Exclusive RateOfChange	True

6.6.35 A & C Previous Instances Server Facet

Table 57 describes the details of the A & C Previous Instances *Server* Facet. This Facet requires support for *Conditions* with previous states that still require action on the part of the operator. This Facet builds upon the A&C Base *Condition Server* Facet. A common use case for this Facet is a safety critical system that requires that all *Alarms* be acknowledged even if it the original problem goes away and the *Alarm* returns to the inactive state. In these cases, the previous state with active *Alarm* is still reported by the *Server* until the Operator acknowledges it. When a *Condition* has previous states it will produce events with different Branch identifiers. When previous state no longer needs attention the branch will disappear.

Table 57 – A & C Previous Instances Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Server* Facet	False
Alarms and *Conditions*	A & C Branch	False

6.6.36 A & C Dialog Server Facet

Table 58 describes the details of the A & C Dialog *Server* Facet. This Facet requires support of Dialog *Conditions*. This Facet builds upon the A & C BaseCondition *Server* Facet Dialogs are ConditionTypes used to request user input. They are typically used when a *Server* has entered some state that requires intervention by a *Client*. For example, a *Server* monitoring a paper machine indicates that a roll of paper has been wound and is ready for inspection. The *Server*

would activate a Dialog *Condition* indicating to the user that an inspection is required. Once the inspection has taken place the user responds by informing the *Server* of an accepted or unaccepted inspection allowing the process to continue.

Table 58 – A & C Dialog Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Server* Facet	False
Alarms and *Conditions*	A & C Dialog	False

6.6.37 A & C CertificateExpiration Server Facet

Table 59 describes the details of the A & C CertificateExpiration *Server* Facet. This Facet requires support of the CertificateExpirationAlarmType. It is used to inform *Clients* when the *Server*'s *Certificate* is within the defined expiration period.

Table 59 – A & C CertificateExpiration Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Server* Facet	False
Alarms and *Conditions*	A & C Acknowledge	False
Alarms and *Conditions*	A & C *Alarm*	False
Alarms and *Conditions*	A & C CertificateExpiration	False
Alarms and *Conditions*	A & C Comment	True
Alarms and *Conditions*	A & C Confirm	True
Alarms and *Conditions*	A & C Shelving	True

6.6.38 A & E Wrapper Facet

Table 60 describes the details of the A & E Wrapper Facet. This Facet specifies the requirements for a UA *Server* that wraps an OPC *Alarm* & *Event* (AE) *Server* (COM). This *Profile* identifies the sub-set of the UA *Alarm* & *Condition* model which is provided by the COM OPC AE specification. It is intended to provide guidance to developers who are creating servers that front end existing applications. It is important to note that some OPC A&E COM *Servers* may not support all of the functionality provided by an OPC UA A&C server, in these cases similar functionality maybe available via some non-OPC interface. For example if an A&E COM server does not support sending *Alarm* Acknowledgement messages to the system that it is obtaining alarm information from, this functionality may be available via some out of scope features in the underlying *Alarm* system. Another possibility is that the underlying system does not require acknowledgements or automatically acknowledges the alarm.

Table 60 – A & E Wrapper Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space Events	False
Address Space Model	Address Space Notifier Hierarchy	False
Address Space Model	Address Space Source Hierarchy	False
Alarms and *Conditions*	A & C Acknowledge	False
Alarms and *Conditions*	A & C *Alarm*	False
Alarms and *Conditions*	A & C Basic	False
Alarms and *Conditions*	A & C ConditionClasses	False
Alarms and *Conditions*	A & C Refresh	False
Alarms and *Conditions*	A & E Wrapper Mapping	False
Monitored Item *Services*	Monitor Basic	False
Monitored Item *Services*	Monitor Complex *Event* Filter	False
Monitored Item *Services*	Monitor Events	False
Monitored Item *Services*	Monitor Items 2	False
Monitored Item *Services*	Monitor QueueSize_ServerMax	False
Subscription Services	*Subscription* Basic	False
Subscription Services	*Subscription* Minimum 1	False
Subscription Services	*Subscription* Publish Discard Policy	False
Subscription Services	*Subscription* Publish Min 02	False

6.6.39 Method Server Facet

Table 61 describes the details of the *Method Server* Facet. This Facet specifies the support of *Method* invocation via the Call service. Methods are “lightweight” functions which are similar to the methods of a class found in any object-oriented programming language. A *Method* can have its scope bounded by an owning *Object* or an owning *ObjectType*. Methods with an *ObjectType* as their scope are similar to static methods in a class.

Table 61 – Method Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Method*	False
Method Services	*Method* Call	False

6.6.40 Auditing Server Facet

Table 62 describes the details of the Auditing *Server* Facet. This Facet requires the support of Auditing which includes the Standard *Event Subscription Server* Facet. Support of this Facet requires that Audit Events be produced when a client performs some action to change the state of the server, such as changing the *AddressSpace*, inserting or updating a value etc. The auditEntryId passed by the *Client* is a field contained in every Audit *Event* and allows actions to be traced across multiple systems. The Audit *Event* Types and their fields must be exposed in the *Server*’s *AddressSpace*

Table 62 – Auditing Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Standard *Event Subscription Server* Facet	False
Auditing	Auditing Base	False

6.6.41 Node Management Server Facet

Table 63 describes the details of the *Node* Management *Server* Facet. This Facet requires the support of the *Services* that allow the *Client* to add, modify and delete *Nodes* in the *AddressSpace*. These *Services* provide an interface which can be used to configure *Servers*. This means all changes to the *AddressSpace* are expected to persist even after the *Client* has disconnected from the *Server*

Table 63 – Node Management Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space Base	False
Base Information	Base Info Model Change	False
Base Information	Base Info Type System	False
Node Management Services	*Node* Management Add *Node*	False
Node Management Services	*Node* Management Add Ref	False
Node Management Services	*Node* Management Delete *Node*	False
Node Management Services	*Node* Management Delete Ref	False

6.6.42 User Role Base Server Facet

Table 64 describes the details of the User Role Base *Server* Facet. This Facet defines support of the OPC UA Information Model to expose configured user roles and permissions.

Table 64 – User Role Base Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security Role *Server* Base	False

6.6.43 User Role Management Server Facet

Table 65 describes the details of the User Role Management *Server* Facet. This Facet defines support of the OPC UA approach to manage user roles and permissions and to grant access to *Nodes* and *Services* based on the assigned roles and permissions.

Table 65 – User Role Management Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	User Role Base *Server* Facet	False
Security	Security Role *Server* DefaultRolePermissions	False
Security	Security Role *Server* IdentityManagement	False
Security	Security Role *Server* Management	False
Security	Security Role *Server* Restrict Applications	True
Security	Security Role *Server* Restrict Endpoints	True
Security	Security Role *Server* RolePermissions	True
Security	Security Role Well Known	False

6.6.44 State Machine Server Facet

Table 66 describes the details of the State Machine *Server* Facet. This Facet defines support of StateMachines based on the types in UA Part 5.

Table 66 – State Machine Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info Available States and Transitions	True
Base Information	Base Info Finite State Machine Instance	True
Base Information	Base Info State Machine Instance	False

6.6.45 Client Redundancy Server Facet

Table 67 describes the details of the *Client* Redundancy *Server* Facet. This Facet defines the *Server* actions that are required for support of redundant *Clients*. Support of this Facet requires the implementation of the TransferSubscriptions *Service* which allows the transfer of Subscriptions from one *Client*'s *Session* to another *Client*'s *Session*.

Table 67 – Client Redundancy Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Subscription Services	*Subscription* Transfer	False

6.6.46 Redundancy Transparent Server Facet

Table 68 describes the details of the Redundancy Transparent *Server* Facet. This Facet requires support for transparent redundancy. If *Servers* implement transparent redundancy then the failover from one *Server* to another is transparent to the *Client* such that the *Client* is unaware that a failover has occurred; the *Client* does not need to do anything at all to keep data flowing. This type of redundancy is usually a hardware solution.

Table 68 – Redundancy Transparent Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Redundancy	Redundancy *Server* Transparent	False

6.6.47 Redundancy Visible Server Facet

Table 69 describes the details of the Redundancy Visible *Server* Facet. This Facet specifies the support for non-transparent redundancy. Failover for this type of redundancy requires the *Client* to monitor *Server* status and to switch to a backup *Server* if it detects a failure. The *Server* shall expose the methods of failover it supports (cold, warm or hot). The failover method tells the *Client* what it must do when connecting to a *Server* and when a failure occurs. Cold

redundancy requires a *Client* to reconnect to a backup *Server* after the initial *Server* has failed. Warm redundancy allows a *Client* to connect to multiple *Servers*, but only one *Server* will be providing values. In hot redundancy multiple *Servers* are able to provide data and a *Client* can connect to multiple *Servers* for the data.

Table 69 – Redundancy Visible Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Redundancy	Redundancy *Server*	False

6.6.48 Historical Raw Data Server Facet

Table 70 describes the details of the Historical Raw Data *Server* Facet. This Facet defines the basic functionality when supporting historical data access for raw data.

Table 70 – Historical Raw Data Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Read	False
Historical Access	Historical Access Data Max *Nodes* Read Continuation Point	False
Historical Access	Historical Access Read Raw	False
Historical Access	Historical Access ServerTimestamp	True

6.6.49 Historical Aggregate Server Facet

Table 71 describes the details of the Historical Aggregate *Server* Facet. This Facet indicates that the server supports aggregate processing to produce derived values from raw historical data.

Table 71 – Historical Aggregate Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Aggregates	Aggregate – AnnotationCount	True
Aggregates	Aggregate – Average	True
Aggregates	Aggregate – Count	True
Aggregates	Aggregate – Custom	True
Aggregates	Aggregate – Delta	True
Aggregates	Aggregate – DeltaBounds	True
Aggregates	Aggregate – DurationBad	True
Aggregates	Aggregate – DurationGood	True
Aggregates	Aggregate – DurationInStateNonZero	True
Aggregates	Aggregate – DurationInStateZero	True
Aggregates	Aggregate – End	True
Aggregates	Aggregate – EndBound	True
Aggregates	Aggregate – Interpolative	True
Aggregates	Aggregate – Maximum	True
Aggregates	Aggregate – Maximum2	True
Aggregates	Aggregate – MaximumActualTime	True
Aggregates	Aggregate – MaximumActualTime2	True
Aggregates	Aggregate – Minimum	True
Aggregates	Aggregate – Minimum2	True
Aggregates	Aggregate – MinimumActualTime	True
Aggregates	Aggregate – MinimumActualTime2	True
Aggregates	Aggregate – NumberOfTransitions	True
Aggregates	Aggregate – PercentBad	True
Aggregates	Aggregate – PercentGood	True
Aggregates	Aggregate – Range	True
Aggregates	Aggregate – Range2	True
Aggregates	Aggregate – StandardDeviationPopulation	True
Aggregates	Aggregate – StandardDeviationSample	True

Group	Conformance Unit / *Profile* Title	Optional
Aggregates	Aggregate – Start	True
Aggregates	Aggregate – StartBound	True
Aggregates	Aggregate – TimeAverage	True
Aggregates	Aggregate – TimeAverage2	True
Aggregates	Aggregate – Total	True
Aggregates	Aggregate – Total2	True
Aggregates	Aggregate – VariancePopulation	True
Aggregates	Aggregate – VarianceSample	True
Aggregates	Aggregate – WorstQuality	True
Aggregates	Aggregate – WorstQuality2	True
Aggregates	Aggregate Historical Configuration	True
Aggregates	Aggregate Master Configuration	False
Attribute Services	*Attribute* Historical Read	False
Historical Access	Historical Access Aggregates	False
Historical Access	Historical Access Data Max *Nodes* Read Continuation Point	False

6.6.50 Historical Data AtTime Server Facet

Table 72 describes the details of the Historical Data AtTime *Server* Facet. This Facet indicates that the historical *Server* supports reading data by specifying specific timestamps.

Table 72 – Historical Data AtTime Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Read	False
Historical Access	Historical Access Data Max *Nodes* Read Continuation Point	False
Historical Access	Historical Access Time Instance	False

6.6.51 Historical Access Modified Data Server Facet

Table 73 describes the details of the Historical Access Modified Data *Server* Facet. This Facet defines support of reading modified historical values (values that where modified or inserted).

Table 73 – Historical Access Modified Data Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Read	False
Historical Access	Historical Access Modified Values	False

6.6.52 Historical Annotation Server Facet

Table 74 describes the details of the Historical Annotation *Server* Facet. This Facet defines support for the storage and retrieval of annotations for historical data.

Table 74 – Historical Annotation Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Read	False
Attribute Services	*Attribute* Historical Update	False
Historical Access	Historical Access Annotations	False

6.6.53 Historical Data Insert Server Facet

Table 75 describes the details of the Historical Data Insert *Server* Facet. This Facet includes Historical Data Insert functionality.

Table 75 – Historical Data Insert Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Update	False

Group	Conformance Unit / *Profile* Title	Optional
Historical Access	Historical Access Insert Value	False
Historical Access	Historical Access ServerTimestamp	True

6.6.54 Historical Data Update Server Facet

Table 76 describes the details of the Historical Data Update *Server* Facet. This Facet includes Historical Data Update functionality.

Table 76 – Historical Data Update Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Update	False
Historical Access	Historical Access ServerTimestamp	True
Historical Access	Historical Access Update Value	False

6.6.55 Historical Data Replace Server Facet

Table 77 describes the details of the Historical Data Replace *Server* Facet. This Facet includes Historical Data Replace functionality.

Table 77 – Historical Data Replace Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Update	False
Historical Access	Historical Access Replace Value	False
Historical Access	Historical Access ServerTimestamp	True

6.6.56 Historical Data Delete Server Facet

Table 78 describes the details of the Historical Data Delete *Server* Facet. This Facet includes Historical Data Delete functionality.

Table 78 – Historical Data Delete Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Update	False
Historical Access	Historical Access Delete Value	False

6.6.57 Historical Access Structured Data Server Facet

Table 79 describes the details of the Historical Access Structured Data *Server* Facet. This Facet indicates that the *Server* supports storage and retrieval of structured values for all supported access types. If a listed access type is supported then the corresponding optional *ConformanceUnit* shall be supported.

Table 79 – Historical Access Structured Data Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Historical Access	Historical Access Structured Data Delete	True
Historical Access	Historical Access Structured Data Insert	True
Historical Access	Historical Access Structured Data Read Modified	True
Historical Access	Historical Access Structured Data Read Raw	False
Historical Access	Historical Access Structured Data Replace	True
Historical Access	Historical Access Structured Data Time Instance	True
Historical Access	Historical Access Structured Data Update	True

6.6.58 Base Historical Event Server Facet

Table 80 describes the details of the Base Historical *Event Server* Facet. This Facet defines the server requirements to support basic Historical *Event* functionality, including simple filtering and general access.

Table 80 – Base Historical Event Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Read	False
Historical Access	Historical Access *Event* Max Events Read Continuation Point	False
Historical Access	Historical Access Events	False

6.6.59 Historical Event Update Server Facet

Table 81 describes the details of the Historical *Event* Update *Server* Facet. This Facet includes Historical *Event* update access functionality.

Table 81 – Historical Event Update Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Update	False
Historical Access	Historical Access Update *Event*	False

6.6.60 Historical Event Replace Server Facet

Table 82 describes the details of the Historical *Event* Replace *Server* Facet. This Facet includes Historical *Event* replace access functionality.

Table 82 – Historical Event Replace Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Update	False
Historical Access	Historical Access Replace *Event*	False

6.6.61 Historical Event Insert Server Facet

Table 83 describes the details of the Historical *Event* Insert *Server* Facet. This Facet includes Historical *Event* insert access functionality.

Table 83 – Historical Event Insert Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Update	False
Historical Access	Historical Access Insert *Event*	False

6.6.62 Historical Event Delete Server Facet

Table 84 describes the details of the Historical *Event* Delete *Server* Facet. This Facet includes Historical *Event* delete access functionality

Table 84 – Historical Event Delete Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute* Historical Update	False
Historical Access	Historical Access Delete *Event*	False

6.6.63 Aggregate Subscription Server Facet

Table 85 describes the details of the Aggregate *Subscription Server* Facet. This Facet defines the handling of the aggregate filter when subscribing for *Attribute* values.

Table 85 – Aggregate Subscription Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Standard DataChange *Subscription Server* Facet	False
Aggregates	Aggregate *Subscription* – AnnotationCount	True
Aggregates	Aggregate *Subscription* – Average	True
Aggregates	Aggregate *Subscription* – Count	True

Group	Conformance Unit / *Profile* Title	Optional
Aggregates	Aggregate *Subscription* – Custom	True
Aggregates	Aggregate *Subscription* – Delta	True
Aggregates	Aggregate *Subscription* – DeltaBounds	True
Aggregates	Aggregate *Subscription* – DurationBad	True
Aggregates	Aggregate *Subscription* – DurationGood	True
Aggregates	Aggregate *Subscription* – DurationInStateNonZero	True
Aggregates	Aggregate *Subscription* – DurationInStateZero	True
Aggregates	Aggregate *Subscription* – End	True
Aggregates	Aggregate *Subscription* – EndBound	True
Aggregates	Aggregate *Subscription* – Filter	False
Aggregates	Aggregate *Subscription* – Interpolative	True
Aggregates	Aggregate *Subscription* – Maximum	True
Aggregates	Aggregate *Subscription* – Maximum2	True
Aggregates	Aggregate *Subscription* – MaximumActualTime	True
Aggregates	Aggregate *Subscription* – MaximumActualTime2	True
Aggregates	Aggregate *Subscription* – Minimum	True
Aggregates	Aggregate *Subscription* – Minimum2	True
Aggregates	Aggregate *Subscription* – MinimumActualTime	True
Aggregates	Aggregate *Subscription* – MinimumActualTime2	True
Aggregates	Aggregate *Subscription* – NumberOfTransitions	True
Aggregates	Aggregate *Subscription* – PercentBad	True
Aggregates	Aggregate *Subscription* – PercentGood	True
Aggregates	Aggregate *Subscription* – Range	True
Aggregates	Aggregate *Subscription* – Range2	True
Aggregates	Aggregate *Subscription* – StandardDeviationPopulation	True
Aggregates	Aggregate *Subscription* – StandardDeviationSample	True
Aggregates	Aggregate *Subscription* – Start	True
Aggregates	Aggregate *Subscription* – StartBound	True
Aggregates	Aggregate *Subscription* – TimeAverage	True
Aggregates	Aggregate *Subscription* – TimeAverage2	True
Aggregates	Aggregate *Subscription* – Total	True
Aggregates	Aggregate *Subscription* – Total2	True
Aggregates	Aggregate *Subscription* – VariancePopulation	True
Aggregates	Aggregate *Subscription* – VarianceSample	True
Aggregates	Aggregate *Subscription* – WorstQuality	True
Aggregates	Aggregate *Subscription* – WorstQuality2	True
Monitored Item *Services*	Monitor Aggregate Filter	False

6.6.64 Nano Embedded Device Server Profile

Table 86 describes the details of the Nano Embedded Device *Server Profile*. This *Profile* is a FullFeatured *Profile* intended for chip level devices with limited resources. This *Profile* is functionally equivalent to the Core *Server* Facet and defines the OPC UA TCP binary protocol as the required transport profile.The support of Diagnostic *Objects* and *Variables* is optional for this *Profile* despite it being defined as “mandatory” in UA Part 5. Support of Diagnostic *Objects* and *Variables* is mandatory in some higher level *Profiles*.
Exposing types in the *AddressSpace* is optional for this *Profile* except if custom types (i.e. types that are derived from well-known *ObjectTypes*, VariableTypes, *ReferenceType* or DataTypes) are used. Exposing all supported types in the *AddressSpace* is mandatory in some higher level *Profiles*.

Table 86 – Nano Embedded Device Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Core *Server* Facet	False
Profile	UA-TCP UA-SC UA-Binary	False

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info Custom Type System	True
Base Information	Base Info Diagnostics	True

6.6.65 Nano Embedded Device 2017 Server Profile

Table 87 describes the details of the Nano Embedded Device 2017 *Server Profile*. This *Profile* is a FullFeatured *Profile* intended for chip level devices with limited resources. This *Profile* is functionally equivalent to the Core *Server* Facet and defines the OPC UA TCP binary protocol as the required transport profile.The support of Diagnostic *Objects* and *Variables* is optional for this *Profile* despite it being defined as “mandatory” in UA Part 5. Support of Diagnostic *Objects* and *Variables* is mandatory in some higher level *Profiles*. Exposing types in the *AddressSpace* is optional for this *Profile* except if custom types (i.e. types that are derived from well-known *ObjectTypes*, VariableTypes, *ReferenceType* or DataTypes) are used. Exposing all supported types in the *AddressSpace* is mandatory in some higher level *Profiles*.
This profile supersedes the “Nano Embedded Device *Server Profile*”.

Table 87 – Nano Embedded Device 2017 Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Core 2017 *Server* Facet	False
Profile	UA-TCP UA-SC UA-Binary	False
Base Information	Base Info Custom Type System	True
Base Information	Base Info Diagnostics	True

6.6.66 Micro Embedded Device Server Profile

Table 88 describes the details of the Micro Embedded Device *Server Profile*. This *Profile* is a FullFeatured *Profile* intended for small devices with limited resources. This *Profile* builds upon the Nano Embedded Device *Server Profile*. The most important additions are: support for subscriptions via the Embedded Data Change *Subscription Server* Facet and support for at least two sessions. A complete Type System is not required; however, if the *Server* implements any non-UA types then these types and their super-types must be exposed.

Table 88 – Micro Embedded Device Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Embedded DataChange *Subscription Server* Facet	False
Profile	Nano Embedded Device *Server Profile*	False
Session Services	*Session* Minimum 2 Parallel	False

6.6.67 Micro Embedded Device 2017 Server Profile

Table 89 describes the details of the Micro Embedded Device 2017 *Server Profile*. This *Profile* is a FullFeatured *Profile* intended for small devices with limited resources. This *Profile* builds upon the Nano Embedded Device *Server Profile*. The most important additions are: support for subscriptions via the Embedded Data Change *Subscription Server* Facet and support for at least two sessions. A complete Type System is not required; however, if the *Server* implements any non-UA types then these types and their super-types must be exposed.
This profile supersedes the “Micro Embedded Device *Server Profile*”.

Table 89 – Micro Embedded Device 2017 Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Embedded DataChange *Subscription Server* Facet	False
Profile	Nano Embedded Device 2017 *Server Profile*	False
Session Services	*Session* Minimum 2 Parallel	False

6.6.68 Embedded UA Server Profile

Table 90 describes the details of the Embedded UA *Server Profile*. This *Profile* is a FullFeatured *Profile* that is intended for devices with more than 50 MBs of memory and a more powerful processor. This *Profile* builds upon the Micro Embedded Device *Server Profile*. The most important additions are: support for security via the Security Policy – Basic128Rsa15 Facet, and support for the Standard DataChange *Subscription Server* Facet. This *Profile* also requires that servers expose all OPC-UA types that are used by the *Server* including their components and their super-types.

Table 90 – Embedded UA Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Micro Embedded Device *Server Profile*	False
Profile	SecurityPolicy – Basic128Rsa15	False
Profile	Standard DataChange *Subscription Server* Facet	False
Base Information	Base Info Engineering Units	True
Base Information	Base Info Placeholder Modelling Rules	True
Base Information	Base Info Type System	False
Security	Security Default ApplicationInstance Certificate	False

6.6.69 Embedded 2017 UA Server Profile

Table 91 describes the details of the Embedded 2017 UA *Server Profile*. This *Profile* is a FullFeatured *Profile* that is intended for devices with more than 50 MBs of memory and a more powerful processor. This *Profile* builds upon the Micro Embedded Device *Server Profile*. The most important additions are: support for security via the Security Policies and support for the Standard DataChange *Subscription Server* Facet. This *Profile* also requires that Servers expose all OPC-UA types that are used by the *Server* including their components and their super-types. This profile supersedes the “Embedded Device *Server Profile*”.

Table 91 – Embedded 2017 UA Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Micro Embedded Device 2017 *Server Profile*	False
Profile	Standard DataChange *Subscription* 2017 *Server* Facet	False
Base Information	Base Info Engineering Units	True
Base Information	Base Info Type System	False
Security	Security – No Application Authentication	True
Security	Security Default ApplicationInstance Certificate	False
Security	Security Policy Required	False

6.6.70 Standard UA Server Profile

Table 92 describes the details of the Standard UA *Server Profile*. This *Profile* is a FullFeatured *Profile* that defines a minimum set of functionality required for PC based OPC UA servers. Such a server must provide the base *AddressSpace* structure with type nodes, instance nodes and diagnostic information. The *Server* must provide connection establishment through the OPC UA TCP binary protocol with security and the creation of at least 50 parallel sessions. It includes view services like browsing and the attribute services for reading and writing of current values. In addition, the monitoring of data changes is included with a minimum of 5 subscriptions for half of the required sessions (total 225) and a minimum of 500 monitored items for half of the subscriptions (total 56250).

Table 92 – Standard UA Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Embedded UA *Server Profile*	False
Profile	Enhanced DataChange *Subscription Server* Facet	False
Profile	User Token – X509 *Certificate Server* Facet	False
Attribute Services	*Attribute* Write StatusCode & Timestamp	True
Base Information	Base Info Diagnostics	False

Group	Conformance Unit / *Profile* Title	Optional
Discovery Services	*Discovery* Register	False
Discovery Services	*Discovery* Register2	True
Session Services	*Session* Cancel	False
Session Services	*Session* Change User	True
Session Services	*Session* Minimum 50 Parallel	False
View *Services*	View Minimum Continuation Point 05	False

6.6.71 Standard 2017 UA Server Profile

Table 93 describes the details of the Standard 2017 UA *Server Profile*. This *Profile* is a FullFeatured *Profile* that defines a minimum set of functionality required for PC based OPC UA servers. Compared to the embedded profiles, the *Profile* requires higher limits for Sessions, Subscriptions and Monitored Items. It also requires support of diagnostic information. This profile supersedes the "Standard UA *Server Profile*".

Table 93 – Standard 2017 UA Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Embedded 2017 UA *Server Profile*	False
Profile	Enhanced DataChange *Subscription* 2017 *Server* Facet	False
Profile	User Token – X509 *Certificate Server* Facet	False
Attribute Services	*Attribute* Write StatusCode & Timestamp	True
Base Information	Base Info Diagnostics	False
Discovery Services	*Discovery* Register	False
Discovery Services	*Discovery* Register2	False
Session Services	*Session* Cancel	False
Session Services	*Session* Change User	True
Session Services	*Session* Minimum 50 Parallel	False
View *Services*	View Minimum Continuation Point 05	False

6.6.72 Core Client Facet

Table 94 describes the details of the Core *Client* Facet. This Facet defines the core functionality required for any *Client*. This Facet includes the core functions for Security and *Session* handling.

Table 94 – Core Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	SecurityPolicy – Basic128Rsa15	False
Profile	SecurityPolicy – None	False
Profile	User Token – User Name Password *Client* Facet	False
Profile	User Token – X509 *Certificate Client* Facet	False
Base Information	Base Info *Client* Estimated Return Time	True
Security	Security Administration	False
Session Services	*Session Client* Base	False
Session Services	*Session Client* Cancel	True
Session Services	*Session Client* Detect Shutdown	False
Session Services	*Session Client* General *Service* Behaviour	False
Session Services	*Session Client* Impersonate	True
Session Services	*Session Client* KeepAlive	False
Session Services	*Session Client* Renew NodeIds	True

6.6.73 Core 2017 Client Facet

Table 95 describes the details of the Core 2017 *Client* Facet. This Facet defines the core functionality required for any *Client*. This Facet includes the core functions for Security and *Session* handling.
This Facet supersedes the Core *Client* Facet.

Table 95 – Core 2017 Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	SecurityPolicy – None	False
Profile	User Token – User Name Password *Client* Facet	False
Profile	User Token – X509 *Certificate Client* Facet	False
Base Information	Base Info *Client* Estimated Return Time	True
Base Information	Base Info *Client* Selection List	True
Security	Security Administration	False
Security	Security Policy Required	False
Session Services	*Session Client* Auto Reconnect	False
Session Services	*Session Client* Base	False
Session Services	*Session Client* Cancel	True
Session Services	*Session Client* Detect Shutdown	False
Session Services	*Session Client* General *Service* Behaviour	False
Session Services	*Session Client* Impersonate	True
Session Services	*Session Client* KeepAlive	False
Session Services	*Session Client* Renew NodeIds	True

6.6.74 Sessionless Client Facet

Table 96 describes the details of the Sessionless *Client* Facet. Defines the use of Sessionless *Service* invocation in a *Client*.

Table 96 – Sessionless Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Discovery Services	*Discovery Client* Get Endpoints SessionLess	True
Session Services	*Session Client* SessionLess *Service* Calls	False

6.6.75 Reverse Connect Client Facet

Table 97 describes the details of the Reverse Connect *Client* Facet. This Facet defines support of reverse connectivity in a *Client*. Usually, a connection is opened by the *Client* before starting the UA-specific handshake. This will fail, however, when *Servers* are behind firewalls. In the reverse connectivity scenario, the *Client* accepts a connection request and a ReverseHello message from a *Server* and establishes a Secure Channel using this connection.

Table 97 – Reverse Connect Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Protocol and Encoding	Protocol Reverse Connect *Client*	False

6.6.76 Base Client Behaviour Facet

Table 98 describes the details of the Base *Client* Behaviour Facet. This Facet indicates that the *Client* supports behaviour that *Clients* shall follow for best use by operators and administrators. They include allowing configuration of an endpoint for a server without using the discovery service set; Support for manual security setting configuration and behaviour with regard to security issues; support for Automatic reconnection to a disconnected server. These behaviours can only be tested in a test lab. They are best practice guidelines.

Table 98 – Base Client Behaviour Facet

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info *Client* Remote *Nodes*	True
Discovery Services	*Discovery Client* Configure Endpoint	False
Security	Security Administration	False
Security	Security Administration – XML Schema	False
Security	Security *Certificate* Administration	False
Session Services	*Session Client* Auto Reconnect	True
Subscription Services	*Subscription Client* Multiple	False

Group	Conformance Unit / *Profile* Title	Optional
Subscription Services	*Subscription Client* Publish Configurable	False

6.6.77 Discovery Client Facet

Table 99 describes the details of the *Discovery Client* Facet. This Facet defines the ability to discover *Servers* and their Endpoints.

Table 99 – Discovery Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Discovery Services	*Discovery Client* Configure Endpoint	False
Discovery Services	*Discovery Client* Find *Servers* Basic	False
Discovery Services	*Discovery Client* Find *Servers* Dynamic	False
Discovery Services	*Discovery Client* Find *Servers* with URI	True
Discovery Services	*Discovery Client* Get Endpoints Basic	False
Discovery Services	*Discovery Client* Get Endpoints Dynamic	False

6.6.78 Subnet Discovery Client Facet

Table 100 describes the details of the Subnet *Discovery Client* Facet. Support of this Facet enables discovery of the *Server* on a subnet.

Table 100 – Subnet Discovery Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Discovery Services	*Discovery Client* Find *Servers* on Network	False
Discovery Services	*Discovery Client* Find *Servers* on Network using LDS-ME	True
Discovery Services	*Discovery Client* Find *Servers* on Network using mDNS	True

6.6.79 Global Discovery Client Facet

Table 101 describes the details of the Global *Discovery Client* Facet. Support of this Facet enables system-wide discovery of *Servers* using a Global *Discovery Server* (GDS).

Table 101 – Global Discovery Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Discovery Services	*Discovery Client* Find Applications in GDS	True
Discovery Services	*Discovery Client* Find *Servers* in GDS	False

6.6.80 Global Certificate Management Client Facet

Table 102 describes the details of the Global *Certificate* Management *Client* Facet. This Facet defines the capability to interact with a Global *Certificate* Management *Server* to obtain an initial or renewed *Certificate* and Trust Lists.

Table 102 – Global Certificate Management Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Pull Model for Global Certificate and TrustList Management	False

6.6.81 KeyCredential Service Client Facet

Table 103 describes the details of the KeyCredential *Service Client* Facet. This Facet defines the capability to interact with a KeyCredential *Service* to obtain KeyCredentials. For example KeyCredentials are needed to access an Authorization *Service* or a Broker. The KeyCredential *Service* is typically part of a system-wide tool, like a GDS that also manages Applications, Access Tokens, and *Certificates*.

Table 103 – KeyCredential Service Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Pull Model for KeyCredential *Service*	False

6.6.82 Access Token Request Client Facet

Table 104 describes the details of the Access Token Request *Client* Facet. A *Client* Facet for using the RequestAccessToken *Method* on an Authorization *Server* (defined in Part 12) to request such a token.

Table 104 – Access Token Request Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Authorization *Service Client*	False

6.6.83 AddressSpace Lookup Client Facet

Table 105 describes the details of the *AddressSpace* Lookup *Client* Facet. This Facet defines the ability to navigate through the *AddressSpace* and includes basic *AddressSpace* concepts, view and browse functionality and simple attribute read functionality.

Table 105 – AddressSpace Lookup Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Base	False
Attribute Services	*Attribute Client* Read Base	False
Attribute Services	*Attribute Client* Remote *Nodes Attribute* Access	True
Base Information	Base Info *Client* Basic	False
Base Information	Base Info *Client* GetMonitoredItems *Method*	True
View *Services*	View *Client* Basic Browse	False
View *Services*	View *Client* Basic ResultSet Filtering	False
View *Services*	View *Client* RegisterNodes	True
View *Services*	View *Client* Remote *Nodes* Browse	True
View *Services*	View *Client* Remote *Nodes* Translate Browse	True
View *Services*	View *Client* TranslateBrowsePath	True

6.6.84 Request State Change Client Facet

Table 106 describes the details of the Request State Change *Client* Facet. This Facet specifies the ability to invoke the RequestServerStateChange *Method*.

Table 106 – Request State Change Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info *Client* RequestServerStateChange	False

6.6.85 File Access Client Facet

Table 107 describes the details of the File Access *Client* Facet. This Facet defines the ability to use File transfer via the defined FileType. This includes reading and optionally writing.

Table 107 – File Access Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info *Client* FileType Base	False
Base Information	Base Info *Client* FileType Write	True

6.6.86 Entry Level Support 2015 Client Facet

Table 108 describes the details of the Entry Level Support 2015 *Client* Facet. This Facet defines the ability to interoperate with low-end *Servers*, in particular *Servers* that support the Nano Embedded *Profile* but in general *Servers* with defined limits.

Table 108 – Entry Level Support 2015 Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info *Client* Honour Operation Limits	False
Base Information	Base Info *Client* Type Pre-Knowledge	False
Session Services	*Session Client* Single *Session*	False
Subscription Services	*Subscription Client* Fallback	False

6.6.87 Multi-Server Client Connection Facet

Table 109 describes the details of the Multi-*Server Client* Connection Facet. This Facet defines the ability for simultaneous access to multiple *Servers*.

Table 109 – Multi-Server Client Connection Facet

Group	Conformance Unit / *Profile* Title	Optional
Session Services	*Session Client* Multiple Connections	False

6.6.88 Documentation – Client

Table 110 describes the details of the Documentation – *Client*. This Facet provides a list of user documentation that a *Client* application should provide.

Table 110 – Documentation – Client

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Documentation *Client* – Installation	False
Miscellaneous	Documentation *Client* – Multiple Languages	True
Miscellaneous	Documentation *Client* – On-line	True
Miscellaneous	Documentation *Client* – Supported *Profiles*	True
Miscellaneous	Documentation *Client* – Trouble Shooting Guide	True
Miscellaneous	Documentation *Client* – Users Guide	False

6.6.89 Attribute Read Client Facet

Table 111 describes the details of the *Attribute* Read *Client* Facet. This Facet defines the ability to read *Attribute* values of *Nodes*.

Table 111 – Attribute Read Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Atomicity	True
Address Space Model	Address Space *Client* Complex Data Dictionary	True
Address Space Model	Address Space *Client* DataTypeDefinition *Attribute*	True
Address Space Model	Address Space *Client* Full Array Only	True
Attribute Services	*Attribute Client* Read Base	False
Attribute Services	*Attribute Client* Read Complex	True
Attribute Services	*Attribute Client* Read with proper Encoding	True

6.6.90 Attribute Write Client Facet

Table 112 describes the details of the *Attribute* Write *Client* Facet. This Facet defines the ability to write *Attribute* values of *Nodes*.

Table 112 – Attribute Write Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Atomicity	True
Address Space Model	Address Space *Client* Complex Data Dictionary	True
Address Space Model	Address Space *Client* DataTypeDefinition *Attribute*	True

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Full Array Only	True
Attribute Services	*Attribute Client* Write Base	False
Attribute Services	*Attribute Client* Write Complex	True
Attribute Services	*Attribute Client* Write Quality & Timestamp	True

6.6.91 DataChange Subscriber Client Facet

Table 113 describes the details of the DataChange Subscriber *Client* Facet. This Facet defines the ability to monitor *Attribute* values for data change.

Table 113 – DataChange Subscriber Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Atomicity	True
Address Space Model	Address Space *Client* Complex Data Dictionary	True
Address Space Model	Address Space *Client* DataTypeDefinition *Attribute*	True
Address Space Model	Address Space *Client* Full Array Only	True
Base Information	Base Data *Client* ResendData *Method*	True
Base Information	Base Info *Client* GetMonitoredItems *Method*	True
Monitored Item *Services*	Monitor *Client* by Index	False
Monitored Item *Services*	Monitor *Client* Complex Value	True
Monitored Item *Services*	Monitor *Client* Deadband Filter	True
Monitored Item *Services*	Monitor *Client* Modify	True
Monitored Item *Services*	Monitor *Client* Trigger	True
Monitored Item *Services*	Monitor *Client* Value Change	False
Subscription Services	*Subscription Client* Basic	False
Subscription Services	*Subscription Client* Modify	True
Subscription Services	*Subscription Client* Multiple	True
Subscription Services	*Subscription Client* Republish	False

6.6.92 Durable Subscription Client Facet

Table 114 describes the details of the Durable *Subscription Client* Facet. This Facet specifies use of durable Subscriptions. It implies support of any of the DataChange or *Event* Subscriber Facets.

Table 114 – Durable Subscription Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Subscription Services	*Subscription Client* Durable	False

6.6.93 DataAccess Client Facet

Table 115 describes the details of the DataAccess *Client* Facet. This Facet defines the ability to utilize the DataAccess Information Model, i.e., industrial automation data like analog and discrete data items and their quality of service.

Table 115 – DataAccess Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Base	False
Address Space Model	Address Space *Client* Complex Data Dictionary	True
Attribute Services	*Attribute Client* Read Base	False
Attribute Services	*Attribute Client* Read Complex	True
Attribute Services	*Attribute Client* Read with proper Encoding	True
Data Access	Data Access *Client* AnalogItems	True
Data Access	Data Access *Client* Basic	False
Data Access	Data Access *Client* Deadband	True
Data Access	Data Access *Client* MultiState	True

Group	Conformance Unit / *Profile* Title	Optional
Data Access	Data Access *Client* MultiStateValueDiscrete	True
Data Access	Data Access *Client* SemanticChange	True
Data Access	Data Access *Client* TwoState	True

6.6.94 Event Subscriber Client Facet

Table 116 describes the details of the *Event* Subscriber *Client* Facet. This Facet defines the ability to subscribe for *Event Notifications*. This includes basic *AddressSpace* concept and the browsing of it, adding events and event filters as monitored items and adding subscriptions.

Table 116 – Event Subscriber Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Base	False
Monitored Item *Services*	Monitor *Client* Complex *Event* Filter	True
Monitored Item *Services*	Monitor *Client Event* Filter	False
Monitored Item *Services*	Monitor *Client* Events	False
Monitored Item *Services*	Monitor *Client* Modify	True
Monitored Item *Services*	Monitor *Client* Trigger	True
Subscription Services	*Subscription Client* Basic	False
Subscription Services	*Subscription Client* Modify	True
Subscription Services	*Subscription Client* Multiple	True
Subscription Services	*Subscription Client* Republish	False
View *Services*	View *Client* Basic Browse	True
View *Services*	View *Client* TranslateBrowsePath	True

6.6.95 Base Event Processing Client Facet

Table 117 describes the details of the Base *Event* Processing *Client* Facet. This Facet defines the ability to subscribe for and process basic OPC UA Events. The *Client* has to support at least one of the Events in the Facet.

Table 117 – Base Event Processing Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	*Event* Subscriber *Client* Facet	False
Base Information	Base Info *Client* Change Events	True
Base Information	Base Info *Client* Device Failure	True
Base Information	Base Info *Client* Progress Events	True
Base Information	Base Info *Client* System Status	True
Base Information	Base Info *Client* System Status Underlying System	True
Base Information	Base Info *Event* Processing	False

6.6.96 Notifier and Source Hierarchy Client Facet

Table 118 describes the details of the Notifier and Source Hierarchy *Client* Facet. This Facet defines the ability to find and use a hierarchy of *Objects* that are event notifier and *Nodes* that are event sources in the *Server AddressSpace*.

Table 118 – Notifier and Source Hierarchy Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	*Event* Subscriber *Client* Facet	False
Address Space Model	Address Space *Client* Notifier Hierarchy	False
Address Space Model	Address Space *Client* Source Hierarchy	False
Subscription Services	*Subscription Client* Publish Configurable	False

6.6.97 A & C Base Condition Client Facet

Table 119 describes the details of the A & C Base *Condition Client* Facet. This Facet defines the ability to use the *Alarm* and *Condition* basic model. This includes the ability to subscribe for Events and to initiate a Refresh *Method*.

Table 119 – A & C Base Condition Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	*Event* Subscriber *Client* Facet	False
Profile	*Method Client* Facet	False
Alarms and *Conditions*	A & C Basic *Client*	False
Alarms and *Conditions*	A & C *Condition* Sub-Classes *Client*	True
Alarms and *Conditions*	A & C ConditionClasses *Client*	False
Alarms and *Conditions*	A & C Refresh *Client*	False

6.6.98 A & C Refresh2 Client Facet

Table 120 describes the details of the A & C Refresh2 *Client* Facet. This Facet enhances the A & C Base *Condition Server* Facet with the ability to initiate a ConditionRefresh2 *Method*.

Table 120 – A & C Refresh2 Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Client* Facet	False
Alarms and *Conditions*	A & C Refresh2 *Client*	False

6.6.99 A & C Address Space Instance Client Facet

Table 121 describes the details of the A & C Address Space Instance *Client* Facet. This Facet defines the ability to use *Condition* instances in the *AddressSpace*.

Table 121 – A & C Address Space Instance Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Alarms and *Conditions*	A & C Instances *Client*	False

6.6.100 A & C Enable Client Facet

Table 122 describes the details of the A & C Enable *Client* Facet. This Facet defines the ability to enable and disable *Alarms*.

Table 122 – A & C Enable Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Client* Facet	False
Alarms and *Conditions*	A & C Enable *Client*	False

6.6.101 A & C AlarmMetrics Client Facet

Table 123 describes the details of the A & C AlarmMetrics *Client* Facet. This Facet defines the ability to use the AlarmMetrics model, i.e. understand and use the collected alarm metrics at any level in the HasNotifier hierarchy.

Table 123 – A & C AlarmMetrics Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Alarms and *Conditions*	A & C *Alarm* Metrics *Client*	False

6.6.102 A & C Alarm Client Facet

Table 124 describes the details of the A & C *Alarm Client* Facet. This Facet defines the ability to use the alarming model (the AlarmType or any of the sub-types).

Table 124 – A & C Alarm Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Client* Facet	False
Alarms and *Conditions*	A & C Acknowledge *Client*	False
Alarms and *Conditions*	A & C *Alarm Client*	False
Alarms and *Conditions*	A & C Audible Sound *Client*	True
Alarms and *Conditions*	A & C Comment *Client*	True
Alarms and *Conditions*	A & C Confirm *Client*	True
Alarms and *Conditions*	A & C Discrepancy Client	True
Alarms and *Conditions*	A & C Discrete *Client*	False
Alarms and *Conditions*	A & C First in Group *Alarm Client*	True
Alarms and *Conditions*	A & C OffNormal *Client*	True
Alarms and *Conditions*	A & C On-Off Delay *Client*	True
Alarms and *Conditions*	A & C Out Of *Service Client*	True
Alarms and *Conditions*	A & C Re-Alarming *Client*	True
Alarms and *Conditions*	A & C Shelving *Client*	True
Alarms and Conditions	A & C Silencing Client	True
Alarms and *Conditions*	A & C Suppression by Operator *Client*	True
Alarms and Conditions	A & C Suppression Client	True
Alarms and *Conditions*	A & C SystemOffNormal *Client*	True
Alarms and *Conditions*	A & C Trip *Client*	True

6.6.103 A & C Exclusive Alarming Client Facet

Table 125 describes the details of the A & C Exclusive Alarming *Client* Facet. This Facet defines the ability to use the exclusive *Alarm* model. This includes understanding the various subtypes such as ExclusiveRateOfChangeAlarm, ExclusiveLevelAlarm and ExclusiveDeviationAlarm.

Table 125 – A & C Exclusive Alarming Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C *Alarm Client* Facet	False
Alarms and *Conditions*	A & C Exclusive Deviation *Client*	True
Alarms and *Conditions*	A & C Exclusive Level *Client*	True
Alarms and *Conditions*	A & C Exclusive Limit *Client*	False
Alarms and *Conditions*	A & C Exclusive RateOfChange *Client*	True

6.6.104 A & C Non-Exclusive Alarming Client Facet

Table 126 describes the details of the A & C Non-Exclusive Alarming *Client* Facet. This Facet defines the ability to use the non-exclusive *Alarm* model. This includes understanding the various subtypes such as NonExclusiveRateOfChangeAlarm, NonExclusiveLevelAlarm and NonExclusiveDeviationAlarm.

Table 126 – A & C Non-Exclusive Alarming Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C *Alarm Client* Facet	False
Alarms and *Conditions*	A & C Non-Exclusive Deviation *Client*	True
Alarms and *Conditions*	A & C Non-Exclusive Level *Client*	True
Alarms and *Conditions*	A & C Non-Exclusive Limit *Client*	False
Alarms and *Conditions*	A & C Non-Exclusive RateOfChange *Client*	True

6.6.105 A & C Previous Instances Client Facet

Table 127 describes the details of the A & C Previous Instances *Client* Facet. This Facet defines the ability to use previous instances of *Alarms*. This implies the ability to understand branchIds.

Table 127 – A & C Previous Instances Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Client* Facet	False
Alarms and *Conditions*	A & C Branch *Client*	False

6.6.106 A & C Dialog Client Facet

Table 128 describes the details of the A & C Dialog *Client* Facet. This Facet defines the ability to use the dialog model. This implies the support of *Method* invocation to respond to dialog messages.

Table 128 – A & C Dialog Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Client* Facet	False
Alarms and *Conditions*	A & C Dialog *Client*	False

6.6.107 A & C CertificateExpiration Client Facet

Table 129 describes the details of the A & C CertificateExpiration *Client* Facet. This Facet defines the ability to use the CertificateExpirationAlarmType.

Table 129 – A & C CertificateExpiration Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	A & C Base *Condition Client* Facet	False
Alarms and *Conditions*	A & C Acknowledge *Client*	True
Alarms and *Conditions*	A & C *Alarm Client*	False
Alarms and *Conditions*	A & C CertificateExpiration *Client*	False
Alarms and *Conditions*	A & C Comment *Client*	True
Alarms and *Conditions*	A & C Confirm *Client*	True
Alarms and *Conditions*	A & C Shelving *Client*	True

6.6.108 A & E Proxy Facet

Table 130 describes the details of the A & E Proxy Facet. This Facet describes the functionality used by a default A & E *Client* proxy. A *Client* exposes this Facet so that a *Server* may be able to better understand the commands that are being issued by the *Client*, since this Facet indicates that the *Client* is an A&E Com *Client*.

Table 130 – A & E Proxy Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Base	False
Alarms and *Conditions*	A & C Acknowledge *Client*	False
Alarms and *Conditions*	A & C *Alarm Client*	False
Alarms and *Conditions*	A & C Basic *Client*	False
Alarms and *Conditions*	A & C ConditionClasses *Client*	False
Alarms and *Conditions*	A & C Discrete *Client*	False
Alarms and *Conditions*	A & C Exclusive Deviation *Client*	False
Alarms and *Conditions*	A & C Exclusive Level *Client*	False
Alarms and *Conditions*	A & C Exclusive Limit *Client*	False
Alarms and *Conditions*	A & C Exclusive RateOfChange *Client*	False
Alarms and *Conditions*	A & C Instances *Client*	False
Alarms and *Conditions*	A & C Non-Exclusive Deviation *Client*	False
Alarms and *Conditions*	A & C Non-Exclusive Level *Client*	False
Alarms and *Conditions*	A & C Non-Exclusive Limit *Client*	False
Alarms and *Conditions*	A & C Non-Exclusive RateOfChange *Client*	False
Alarms and *Conditions*	A & C OffNormal *Client*	False
Alarms and *Conditions*	A & C Refresh *Client*	False

Group	Conformance Unit / *Profile* Title	Optional
Alarms and *Conditions*	A & C SystemOffNormal *Client*	True
Alarms and *Conditions*	A & C Trip *Client*	False
Attribute Services	*Attribute Client* Read Base	False
Base Information	Base Info *Client* Basic	False
Base Information	Base Info *Client* Change Events	False
Discovery Services	*Discovery Client* Configure Endpoint	False
Discovery Services	*Discovery Client* Find *Servers* Basic	False
Discovery Services	*Discovery Client* Find *Servers* Dynamic	False
Discovery Services	*Discovery Client* Find *Servers* with URI	False
Discovery Services	*Discovery Client* Get Endpoints Basic	False
Discovery Services	*Discovery Client* Get Endpoints Dynamic	False
Method Services	*Method Client* Call	False
Monitored Item *Services*	Monitor *Client* Complex *Event* Filter	False
Monitored Item *Services*	Monitor *Client Event* Filter	False
Monitored Item *Services*	Monitor *Client* Events	False
Security	Security Administration	False
Security	Security Administration – XML Schema	False
Security	Security *Certificate* Administration	False
Session Services	*Session Client* Auto Reconnect	False
Subscription Services	*Subscription Client* Basic	False
Subscription Services	*Subscription Client* Multiple	False
Subscription Services	*Subscription Client* Publish Configurable	False
Subscription Services	*Subscription Client* Republish	False
View *Services*	View *Client* Basic Browse	False
View *Services*	View *Client* Basic ResultSet Filtering	False
View *Services*	View *Client* TranslateBrowsePath	False

6.6.109 Method Client Facet

Table 131 describes the details of the *Method Client* Facet. This Facet defines the ability to call arbitrary Methods.

Table 131 – Method Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Method Services	*Method Client* Call	False

6.6.110 Auditing Client Facet

Table 132 describes the details of the Auditing *Client* Facet. This Facet defines the ability to monitor Audit Events.

Table 132 – Auditing Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	*Event* Subscriber *Client* Facet	False
Auditing	Auditing *Client* Audit ID	False
Auditing	Auditing *Client* Subscribes	False

6.6.111 Node Management Client Facet

Table 133 describes the details of the *Node* Management *Client* Facet. This Facet defines the ability to configure the *AddressSpace* of an OPC UA *Server* through OPC UA *Node* Management *Service* Set.

Table 133 – Node Management Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Base	False

Group	Conformance Unit / *Profile* Title	Optional
Node Management Services	*Node* Management *Client*	False

6.6.112 Advanced Type Programming Client Facet

Table 134 describes the details of the Advanced Type Programming *Client* Facet. This Facet defines the ability to use the type model and process the instance *AddressSpace* based on the type model. For example a client may contain generic displays that are based on a type, in that they contain a relative path from some main type. On call up this main type is matched to an instance and all of display items are resolved based on the provided type model.

Table 134 – Advanced Type Programming Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Base	False
Base Information	Base Info *Client* Basic	False
Base Information	Base Info *Client* Type Programming	False
View *Services*	View *Client* TranslateBrowsePath	False

6.6.113 User Role Management Client Facet

Table 135 describes the details of the User Role Management *Client Facet*. This *Facet* defines knowledge of the OPC UA *Information Model* for user roles and permissions and the use of the *Methods* to manage them.

Table 135 – User Role Management Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security Role *Client* Base	False
Security	Security Role *Client* DefaultRolePermissions	False
Security	Security Role *Client* Management	False
Security	Security Role *Client* Restrict Applications	True
Security	Security Role *Client* Restrict Endpoints	True
Security	Security Role *Client* RolePermissions	False

6.6.114 State Machine Client Facet

Table 136 describes the details of the State Machine *Client* Facet. This Facet defines the ability to use state machines based on the StateMachineType or a sub-type.

Table 136 – State Machine Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Base Information	Base Info Client Available States and Transitions	True
Base Information	Base Info *Client* Finite State Machine Instance	True
Base Information	Base Info *Client* State Machine Instance	False

6.6.115 Diagnostic Client Facet

Table 137 describes the details of the Diagnostic *Client* Facet. This Facet defines the ability to read and process diagnostic information that is part of the OPC UA information model.

Table 137 – Diagnostic Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space *Client* Base	False
Base Information	Base Info *Client* Basic	False
Base Information	Base Info *Client* Diagnostics	False

6.6.116 Redundant Client Facet

Table 138 describes the details of the Redundant *Client* Facet. This Facet defines the ability to use the redundancy feature available for redundant *Clients*.

Table 138 – Redundant Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Redundancy	Redundancy *Client*	False
Subscription Services	*Subscription Client* TransferSubscriptions	True

6.6.117 Redundancy Switch Client Facet

Table 139 describes the details of the Redundancy Switch *Client* Facet. A *Client* that supports this Facet supports monitoring the redundancy status for non-transparent redundant *Servers* and switching to the backup *Server* when they recognize a change.

Table 139 – Redundancy Switch Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Redundancy	Redundancy *Client* Switch	False

6.6.118 Historical Access Client Facet

Table 140 describes the details of the Historical Access *Client* Facet. This Facet defines the ability to read, process, and update historical data.

Table 140 – Historical Access Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Read	False
Historical Access	Historical Access *Client* Browse	False
Historical Access	Historical Access *Client* Read Raw	False

6.6.119 Historical Data AtTime Client Facet

Table 141 describes the details of the Historical Data AtTime *Client* Facet. This Facet defines the ability to access data at specific instances in time.

Table 141 – Historical Data AtTime Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Access *Client* Facet	False
Historical Access	Historical Access *Client* Time Instance	False

6.6.120 Historical Aggregate Client Facet

Table 142 describes the details of the Historical Aggregate *Client* Facet. This Facet defines the ability to read historical data by specifying the needed aggregate. This implies consideration of the list of aggregates supported by the *Server*.

Table 142 – Historical Aggregate Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Aggregates	Aggregate – *Client* AnnotationCount	True
Aggregates	Aggregate – *Client* Average	True
Aggregates	Aggregate – *Client* Count	True
Aggregates	Aggregate – *Client* Custom Aggregates	True
Aggregates	Aggregate – *Client* Delta	True
Aggregates	Aggregate – *Client* DeltaBounds	True
Aggregates	Aggregate – *Client* DurationBad	True
Aggregates	Aggregate – *Client* DurationGood	True
Aggregates	Aggregate – *Client* DurationInStateNonZero	True

Group	Conformance Unit / *Profile* Title	Optional
Aggregates	Aggregate – *Client* DurationInStateZero	True
Aggregates	Aggregate – *Client* End	True
Aggregates	Aggregate – *Client* EndBound	True
Aggregates	Aggregate – *Client* Interpolative	True
Aggregates	Aggregate – *Client* Maximum	True
Aggregates	Aggregate – *Client* Maximum2	True
Aggregates	Aggregate – *Client* MaximumActualTime	True
Aggregates	Aggregate – *Client* MaximumActualTime2	True
Aggregates	Aggregate – *Client* Minimum	True
Aggregates	Aggregate – *Client* Minimum2	True
Aggregates	Aggregate – *Client* MinimumActualTime	True
Aggregates	Aggregate – *Client* MinimumActualTime2	True
Aggregates	Aggregate – *Client* NumberOfTransitions	True
Aggregates	Aggregate – *Client* PercentBad	True
Aggregates	Aggregate – *Client* PercentGood	True
Aggregates	Aggregate – *Client* Range	True
Aggregates	Aggregate – *Client* Range2	True
Aggregates	Aggregate – *Client* StandardDeviationPopulation	True
Aggregates	Aggregate – *Client* StandardDeviationSample	True
Aggregates	Aggregate – *Client* Start	True
Aggregates	Aggregate – *Client* StartBound	True
Aggregates	Aggregate – *Client* TimeAverage	True
Aggregates	Aggregate – *Client* TimeAverage2	True
Aggregates	Aggregate – *Client* Total	True
Aggregates	Aggregate – *Client* Total2	True
Aggregates	Aggregate – *Client* Usage	False
Aggregates	Aggregate – *Client* VariancePopulation	True
Aggregates	Aggregate – *Client* VarianceSample	True
Aggregates	Aggregate – *Client* WorstQuality	True
Aggregates	Aggregate – *Client* WorstQuality2	True
Historical Access	Historical Access *Client* Read Aggregates	False

6.6.121 Historical Annotation Client Facet

Table 143 describes the details of the Historical Annotation *Client* Facet. This Facet defines the ability to retrieve and write annotations for historical data.

Table 143 – Historical Annotation Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Access *Client* Facet	False
Profile	Historical Data Update *Client* Facet	False
Historical Access	Historical Access *Client* Annotations	False

6.6.122 Historical Access Modified Data Client Facet

Table 144 describes the details of the Historical Access Modified Data *Client* Facet. This Facet defines the ability to access prior historical data (values that were modified or inserted).

Table 144 – Historical Access Modified Data Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Access *Client* Facet	False
Historical Access	Historical Access *Client* Read Modified	False

6.6.123 Historical Data Insert Client Facet

Table 145 describes the details of the Historical Data Insert *Client* Facet. This Facet defines the ability to insert historical data.

Table 145 – Historical Data Insert Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Updates	False
Historical Access	Historical Access *Client* Data Insert	False

6.6.124 Historical Data Update Client Facet

Table 146 describes the details of the Historical Data Update *Client* Facet. This Facet defines the ability to update historical data.

Table 146 – Historical Data Update Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Updates	False
Historical Access	Historical Access *Client* Data Update	False

6.6.125 Historical Data Replace Client Facet

Table 147 describes the details of the Historical Data Replace *Client* Facet. This Facet defines the ability to replace historical data.

Table 147 – Historical Data Replace Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Updates	False
Historical Access	Historical Access *Client* Data Replace	False

6.6.126 Historical Data Delete Client Facet

Table 148 describes the details of the Historical Data Delete *Client* Facet. This Facet defines the ability to delete historical data.

Table 148 – Historical Data Delete Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Updates	False
Historical Access	Historical Access *Client* Data Delete	False

6.6.127 Historical Access Client Server Timestamp Facet

Table 149 describes the details of the Historical Access *Client Server* Timestamp Facet. This Facet defines the ability to request and process *Server* timestamps, in addition to source timestamps.

Table 149 – Historical Access Client Server Timestamp Facet

Group	Conformance Unit / *Profile* Title	Optional
Historical Access	Historical Access *Client Server* Timestamp	False

6.6.128 Historical Structured Data Access Client Facet

Table 150 describes the details of the Historical Structured Data Access *Client* Facet. This Facet defines the ability to read structured values for historical nodes.

Table 150 – Historical Structured Data Access Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Access *Client* Facet	False
Historical Access	Historical Access *Client* Structure Data Raw	False

6.6.129 Historical Structured Data AtTime Client Facet

Table 151 describes the details of the Historical Structured Data AtTime *Client* Facet. This Facet defines the ability to read structured values for historical nodes at specific instances in time.

Table 151 – Historical Structured Data AtTime Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Data AtTime *Client* Facet	False
Historical Access	Historical Access *Client* Structure Data Time Instance	False

6.6.130 Historical Structured Data Modified Client Facet

Table 152 describes the details of the Historical Structured Data Modified *Client* Facet. This Facet defines the ability to read structured values for prior historical data (values that were modified or inserted).

Table 152 – Historical Structured Data Modified Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Access Modified Data *Client* Facet	False
Historical Access	Historical Access *Client* Structure Data Read Modified	False

6.6.131 Historical Structured Data Insert Client Facet

Table 153 describes the details of the Historical Structured Data Insert *Client* Facet. This Facet defines the ability to insert structured historical data.

Table 153 – Historical Structured Data Insert Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Data Insert *Client* Facet	False
Historical Access	Historical Access *Client* Structure Data Insert	False

6.6.132 Historical Structured Data Update Client Facet

Table 154 describes the details of the Historical Structured Data Update *Client* Facet. This Facet defines the ability to update structured historical data.

Table 154 – Historical Structured Data Update Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Data Update *Client* Facet	False
Historical Access	Historical Access *Client* Structure Data Update	False

6.6.133 Historical Structured Data Replace Client Facet

Table 155 describes the details of the Historical Structured Data Replace *Client* Facet. This Facet defines the ability to replace structured historical data.

Table 155 – Historical Structured Data Replace Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Data Replace *Client* Facet	False
Historical Access	Historical Access *Client* Structure Data Replace	False

6.6.134 Historical Structured Data Delete Client Facet

Table 156 describes the details of the Historical Structured Data Delete *Client* Facet. This Facet defines the ability to remove structured historical data.

Table 156 – Historical Structured Data Delete Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	Historical Data Delete *Client* Facet	False
Historical Access	Historical Access *Client* Structure Data Delete	False

6.6.135 Historical Events Client Facet

Table 157 describes the details of the Historical Events *Client* Facet. This Facet defines the ability to read Historical Events, including simple filtering.

Table 157 – Historical Events Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Read	False
Historical Access	Historical Access *Client* Read Events	False

6.6.136 Historical Event Insert Client Facet

Table 158 describes the details of the Historical *Event* Insert *Client* Facet. This Facet defines the ability to insert historical events.

Table 158 – Historical Event Insert Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Updates	False
Historical Access	Historical Access *Client Event* Inserts	False

6.6.137 Historical Event Update Client Facet

Table 159 describes the details of the Historical *Event* Update *Client* Facet. This Facet defines the ability to update historical events.

Table 159 – Historical Event Update Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Updates	False
Historical Access	Historical Access *Client Event* Updates	False

6.6.138 Historical Event Replace Client Facet

Table 160 describes the details of the Historical *Event* Replace *Client* Facet. This Facet defines the ability to replace historical events.

Table 160 – Historical Event Replace Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Updates	False
Historical Access	Historical Access *Client Event* Replaces	False

6.6.139 Historical Event Delete Client Facet

Table 161 describes the details of the Historical *Event* Delete *Client* Facet. This Facet defines the ability to delete Historical events.

Table 161 – Historical Event Delete Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Attribute Services	*Attribute Client* Historical Updates	False
Historical Access	Historical Access *Client Event* Deletes	False

6.6.140 Aggregate Subscriber Client Facet

Table 162 describes the details of the Aggregate Subscriber *Client* Facet. This Facet defines the ability to use the aggregate filter when subscribing for *Attribute* values.

Table 162 – Aggregate Subscriber Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Aggregates	Aggregate *Subscription – Client* AnnotationCount	True
Aggregates	Aggregate *Subscription – Client* Average	True
Aggregates	Aggregate *Subscription – Client* Count	True
Aggregates	Aggregate *Subscription – Client* Custom Aggregates	True
Aggregates	Aggregate *Subscription – Client* Delta	True
Aggregates	Aggregate *Subscription – Client* DeltaBounds	True
Aggregates	Aggregate *Subscription – Client* DurationBad	True
Aggregates	Aggregate *Subscription – Client* DurationGood	True
Aggregates	Aggregate *Subscription – Client* DurationInStateNonZero	True
Aggregates	Aggregate *Subscription – Client* DurationInStateZero	True
Aggregates	Aggregate *Subscription – Client* End	True
Aggregates	Aggregate *Subscription – Client* EndBound	True
Aggregates	Aggregate *Subscription – Client* Filter	False
Aggregates	Aggregate *Subscription – Client* Interpolative	True
Aggregates	Aggregate *Subscription – Client* Maximum	True
Aggregates	Aggregate *Subscription – Client* Maximum2	True
Aggregates	Aggregate *Subscription – Client* MaximumActualTime	True
Aggregates	Aggregate *Subscription – Client* MaximumActualTime2	True
Aggregates	Aggregate *Subscription – Client* Minimum	True
Aggregates	Aggregate *Subscription – Client* Minimum2	True
Aggregates	Aggregate *Subscription – Client* MinimumActualTime	True
Aggregates	Aggregate *Subscription – Client* MinimumActualTime2	True
Aggregates	Aggregate *Subscription – Client* NumberOfTransitions	True
Aggregates	Aggregate *Subscription – Client* PercentBad	True
Aggregates	Aggregate *Subscription – Client* PercentGood	True
Aggregates	Aggregate *Subscription – Client* Range	True
Aggregates	Aggregate *Subscription – Client* Range2	True
Aggregates	Aggregate *Subscription – Client* StandardDeviationPopulation	True
Aggregates	Aggregate *Subscription – Client* StandardDeviationSample	True
Aggregates	Aggregate *Subscription – Client* Start	True
Aggregates	Aggregate *Subscription – Client* StartBound	True
Aggregates	Aggregate *Subscription – Client* TimeAverage	True
Aggregates	Aggregate *Subscription – Client* TimeAverage2	True
Aggregates	Aggregate *Subscription – Client* Total	True
Aggregates	Aggregate *Subscription – Client* Total2	True
Aggregates	Aggregate *Subscription – Client* VariancePopulation	True
Aggregates	Aggregate *Subscription – Client* VarianceSample	True
Aggregates	Aggregate *Subscription – Client* WorstQuality	True
Aggregates	Aggregate *Subscription – Client* WorstQuality2	True
Monitored Item *Services*	Monitor *Client* Aggregate Filter	False

Group	Conformance Unit / *Profile* Title	Optional
Monitored Item *Services*	Monitor *Client* by Index	False
Monitored Item *Services*	Monitor *Client* Modify	True
Monitored Item *Services*	Monitor *Client* Value Change	False
Subscription Services	*Subscription Client* Basic	False
Subscription Services	*Subscription Client* Modify	True
Subscription Services	*Subscription Client* Multiple	True
Subscription Services	*Subscription Client* Republish	True

6.6.141 Standard UA Client Profile

Table 163 describes the details of the Standard UA *Client Profile*. This *Profile* is a FullFeatured *Profile* that defines a minimum set of functionality required for generic OPC UA *Clients*. Such a *Client* shall be able to use local, subnet and global discovery. It shall be able to maintain a connection with a single *Session* (as required for nano embedded *Servers*). If Subscriptions are used, the *Client* shall respect the limits of *Servers* with limited resources. If a *Server* does not support Subscriptions, the *Client* shall provide read access as fallback. The *Client* must provide connection establishment through the OPC UA TCP binary protocol with and without security.

Table 163 – Standard UA Client Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	*AddressSpace* Lookup *Client* Facet	False
Profile	*Attribute* Read *Client* Facet	False
Profile	*Attribute* Write *Client* Facet	False
Profile	Base *Client* Behaviour Facet	False
Profile	Core *Client* Facet	False
Profile	DataChange Subscriber *Client* Facet	False
Profile	*Discovery Client* Facet	False
Profile	Entry Level Support 2015 *Client* Facet	False
Profile	Global *Certificate* Management *Client* Facet	False
Profile	Global *Discovery Client* Facet	False
Profile	*Method Client* Facet	False
Profile	SecurityPolicy [B] – Basic256Sha256	False
Profile	SecurityPolicy – Basic256	False
Profile	Subnet *Discovery Client* Facet	False
Profile	UA-TCP UA-SC UA-Binary	False
Profile	User Token – Anonymous Facet	False

6.6.142 Standard UA Client 2017 Profile

Table 164 describes the details of the Standard UA *Client* 2017 *Profile*. This *Profile* is a FullFeatured *Profile* that defines a minimum set of functionality required for generic OPC UA *Clients*. Such a *Client* shall be able to use local, subnet and global discovery. It shall be able to maintain a connection with a single *Session* (as required for nano embedded *Servers*). If *Subscriptions* are used, the *Client* shall respect the limits of *Servers* with limited resources. If a *Server* does not support *Subscriptions*, the *Client* shall provide read access as fallback. The *Client* must provide connection establishment through the OPC UA TCP binary protocol with and without security.
This *Profile* supersedes the “Standard UA *Client Profile*”

Table 164 – Standard UA Client 2017 Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	*AddressSpace* Lookup *Client* Facet	False
Profile	*Attribute* Read *Client* Facet	False
Profile	*Attribute* Write *Client* Facet	False
Profile	Base *Client* Behaviour Facet	False
Profile	Core 2017 *Client* Facet	False
Profile	DataChange Subscriber *Client* Facet	False
Profile	*Discovery Client* Facet	False

Group	Conformance Unit / *Profile* Title	Optional
Profile	Entry Level Support 2015 *Client* Facet	False
Profile	Global *Certificate* Management *Client* Facet	False
Profile	Global *Discovery Client* Facet	False
Profile	*Method Client* Facet	False
Profile	Subnet *Discovery Client* Facet	False
Profile	UA-TCP UA-SC UA-Binary	False
Profile	User Token – Anonymous Facet	False

6.6.143 UA-TCP UA-SC UA-Binary

Table 165 describes the details of the UA-TCP UA-SC UA-Binary. This transport Facet defines a combination of network protocol, security protocol and message encoding that is optimized for low resource consumption and high performance. It combines the simple TCP based network protocol UA-TCP 1.0 with the binary security protocol UA-SecureConversation 1.0 and the binary message encoding UA-Binary 1.0.

Table 165 – UA-TCP UA-SC UA-Binary

Group	Conformance Unit / *Profile* Title	Optional
Protocol and Encoding	Protocol UA TCP	False
Protocol and Encoding	UA Binary Encoding	False
Protocol and Encoding	UA Secure Conversation	False

6.6.144 HTTPS UA-Binary

Table 166 describes the details of the HTTPS UA-Binary. This transport Facet defines a combination of network protocol, security protocol and message encoding that balances compatibility with widely used HTTPS transport and a compact UA-Binary encoded message for added performance. It is expected that this transport will be used to support installations where firewalls only permit HTTPS or where a WEB browser is used as *Client*. This transport requires that one of the TransportSecurity *Profiles* for TLS be provided.

Table 166 – HTTPS UA-Binary

Group	Conformance Unit / *Profile* Title	Optional
Protocol and Encoding	Protocol HTTPS	False
Protocol and Encoding	UA Binary Encoding	False
Security	Security TLS General	False

6.6.145 HTTPS UA-XML

Table 167 describes the details of the HTTPS UA-XML. This transport Facet defines a combination of network protocol, security protocol and message encoding that uses HTTPS transport and a SOAP XML encoded message for use with standard SOAP V1.2 toolkits. This transport requires that one of the TransportSecurity *Profiles* for TLS be provided.

Table 167 – HTTPS UA-XML

Group	Conformance Unit / *Profile* Title	Optional
Protocol and Encoding	Protocol HTTPS	False
Protocol and Encoding	UA SOAP-XML Encoding	False
Security	Security TLS General	False

6.6.146 HTTPS UA-JSON

Table 168 describes the details of the HTTPS UA-JSON. This transport Facet defines a combination of network protocol, security protocol and message encoding that uses HTTPS transport and a UA-JSON encoded message. This transport requires that one of the TransportSecurity *Profiles* for TLS be provided.

Table 168 – HTTPS UA-JSON

Group	Conformance Unit / *Profile* Title	Optional
Protocol and Encoding	JSON Reversible Encoding	False
Protocol and Encoding	Protocol HTTPS	False
Security	Security TLS General	False

6.6.147 WSS UA-SC UA-Binary

Table 169 describes the details of the WSS UA-SC UA-Binary. This transport Facet defines a combination of network protocol, security protocol and message encoding that uses WSS transport as a tunnel for UA-SecureConversation and UA-Binary encoded messages. Although transport security is available in WSS via TLS, additional message security can be used to assure end-to-end security.

Table 169 – WSS UA-SC UA-Binary

Group	Conformance Unit / *Profile* Title	Optional
Protocol and Encoding	Protocol Web Sockets	False
Protocol and Encoding	UA Binary Encoding	False
Protocol and Encoding	UA Secure Conversation	False
Security	Security TLS General	False

6.6.148 WSS UA-JSON

Table 170 describes the details of the WSS UA-JSON. This transport Facet defines a combination of network protocol, security protocol and message encoding that uses WSS transport with UA-JSON encoded messages.

Table 170 – WSS UA-JSON

Group	Conformance Unit / *Profile* Title	Optional
Protocol and Encoding	JSON Reversible Encoding	False
Protocol and Encoding	Protocol Web Sockets	False
Security	Security TLS General	False

6.6.149 Security User Access Control Full

Table 171 describes the details of the Security User Access Control Full. A *Server* that supports this profile supports restricting multiple levels of access to all *Nodes* in the *AddressSpace* based on the validated user.

Table 171 – Security User Access Control Full

Group	Conformance Unit / *Profile* Title	Optional
Profile	Security User Access Control Base	False
Address Space Model	Address Space User Access Level Full	False

6.6.150 Security User Access Control Base

Table 172 describes the details of the Security User Access Control Base. A *Server* that supports this profile supports restricting some level of access to some *Nodes* in the *AddressSpace* based on the validated user.

Table 172 – Security User Access Control Base

Group	Conformance Unit / *Profile* Title	Optional
Address Space Model	Address Space User Access Level Base	False
Security	Security User IssuedToken Kerberos	True
Security	Security User IssuedToken Kerberos Windows	True
Security	Security User Name Password	False
Security	Security User X509	True

6.6.151 Security Time Synchronization

Table 173 describes the details of the Security Time Synchronization. This Facet indicates that the application supports the minimum required level of time synchronization to ensure secure communication. One of the optional time synchronization conformance units must be supported.

Table 173 – Security Time Synchronization

Group	Conformance Unit / *Profile* Title	Optional
Security	Security Time Synch – Configuration	False
Security	Security Time Synch – NTP / OS Based support	True
Security	Security Time Synch – UA based support	True

6.6.152 Best Practice – Audit Events

Table 174 describes the details of the Best Practice – Audit Events. Subscriptions for Audit Events shall be restricted to authorized personnel.

Table 174 – Best Practice – Audit Events

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Best Practice – Audit Events	False

6.6.153 Best Practice – Alarm Handling

Table 175 describes the details of the Best Practice – *Alarm* Handling. A *Server* should restrict critical alarm handling functionality to users that have the appropriate rights to perform these actions.

Table 175 – Best Practice – Alarm Handling

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Best Practice – *Alarm* Handling	False

6.6.154 Best Practice – Random Numbers

Table 176 describes the details of the Best Practice – Random Numbers. All random numbers that are required for security should use appropriate cryptographic library based random number generators.

Table 176 – Best Practice – Random Numbers

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Best Practice – Random Numbers	False

6.6.155 Best Practice – Timeouts

Table 177 describes the details of the Best Practice – Timeouts. The administrator should be able to configure reasonable timeouts for Secure Channels, Sessions and Subscriptions. Setting these timeouts allows limiting Denial of *Service* attacks and overload issues.

Table 177 – Best Practice – Timeouts

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Best Practice – Timeouts	False

6.6.156 Best Practice – Administrative Access

Table 178 describes the details of the Best Practice – Administrative Access. The *Server* and *Client* allow restricting the use of certain *Services* and access to parts of the *AddressSpace* to administrative personnel. This includes multiple level of administrative access on platforms that support multiple administrative roles (such as Windows or Linux).

Table 178 – Best Practice – Administrative Access

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Best Practice – Administrative Access	False

6.6.157 Best Practice – Strict Message Handling

Table 179 describes the details of the Best Practice – Strict *Message* Handling. *Server* and *Client* reject messages that are incorrectly formed as specified in Part 4 and Part 6.

Table 179 – Best Practice – Strict Message Handling

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Best Practice – Strict *Message* Handling	False

6.6.158 Best Practice – Audit Events Client

Table 180 describes the details of the Best Practice – Audit Events *Client*. Audit Tracking system connect to a *Server* using a Secure Channel and under the appropriate authorization to allow access to Audit Events.

Table 180 – Best Practice – Audit Events Client

Group	Conformance Unit / *Profile* Title	Optional
Miscellaneous	Best Practice – Audit Events *Client*	False

6.6.159 TransportSecurity – TLS 1.2

Table 181 describes the details of the TransportSecurity – TLS 1.2. This Facet defines a transport security for configurations with high security needs. It makes use of TLS 1.2 and uses TLS_RSA_WITH_AES_256_CBC_SHA256. As computing power increases, security algorithms are expected to expire. NIST provides guidelines for expected expiration dates for individual algorithms. These guidelines provide recommended dates at which the algorithm should be replaced or upgraded to a more secure algorithm. They do not indicate a failure of the algorithm. NIST has no recommendations for this TransportSecurity. It is recommended that *Servers* and *Clients* support all security profiles and developers provide the recommended profile as a default. It is up to an administrator to configure the actual exposed TransportSecurity *Profiles*.

Table 181 – TransportSecurity – TLS 1.2

Group	Conformance Unit / *Profile* Title	Optional
Security	Security TLS_RSA with AES_256_CBC_SHA256	False

6.6.160 TransportSecurity – TLS 1.2 with PFS

Table 182 describes the details of the TransportSecurity – TLS 1.2 with PFS. This Facet defines a transport security for configurations with high security needs and perfect forward secrecy (PFS). It makes use of TLS 1.2 and uses TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 or TLS_DHE_RSA_WITH_AES_256_CBC_SHA256.

As computing power increases, security algorithms are expected to expire. NIST provides guidelines for expected expiration dates for individual algorithms. These guidelines provide recommended dates at which the algorithm should be replaced or upgraded to a more secure algorithm. They do not indicate a failure of the algorithm. NIST has no recommendations for this TransportSecurity. It is recommended that *Servers* and *Clients* support all security profiles and developers provide the recommended profile as a default. It is up to an administrator to configure the actual exposed TransportSecurity *Profiles*.

Table 182 – TransportSecurity – TLS 1.2 with PFS

Group	Conformance Unit / *Profile* Title	Optional
Security	Security TLS_DHE_RSA with AES_nnn_CBC_SHA256	False

6.6.161 SecurityPolicy – None

Table 183 describes the details of the SecurityPolicy – None. This security Facet defines a security policy used for configurations with the lowest security needs. This security policy can affect the behaviour of the CreateSession and *ActivateSession Services*. It also results in a SecureChannel which has no channel security. By default this security policy should be disabled if any other security policies are available.

Table 183 – SecurityPolicy – None

Group	Conformance Unit / *Profile* Title	Optional
Security	AsymmetricEncryptionAlgorithm_None	False
Security	AsymmetricSignatureAlgorithm_None	False
Security	KeyDerivationAlgorithm_None	False
Security	Security None CreateSession ActivateSession	False
Security	Security None CreateSession ActivateSession 1.0	True
Security	SecurtyPolicy_None_Limits	False
Security	SymmetricEncryptionAlgorithm_None	False
Security	SymmetricSignatureAlgorithm_None	False

6.6.162 SecurityPolicy – Basic128Rsa15

SecurityPolicy – Basic128Rsa15 has been deprecated in v1.04 since the hash algorithm Sha-1 is not considered secure anymore.

6.6.163 SecurityPolicy – Basic256

SecurityPolicy – Basic128Rsa15 has been deprecated in v1.04 since the hash algorithm Sha-1 is not considered secure anymore.

6.6.164 SecurityPolicy [A] - Aes128-Sha256-RsaOaep

Table 184 describes the details of the SecurityPolicy [A] - Aes128-Sha256-RsaOaep. This security Facet defines a security policy for configurations with average security needs. It requires a PKI infrastructure. As computing power increases, security policies are expected to expire. NIST provides guidelines for expected expiration dates for individual algorithms. These guidelines provide recommended dates at which the algorithm should be replaced or upgraded to a more secure algorithm. They do not indicate a failure of the algorithm. This security policy has no published end dates as of this time. It is recommended that *Servers* and *Clients* support all security profiles and support configurability of actual exposed and default security policies.

Table 184 – SecurityPolicy [A] - Aes128-Sha256-RsaOaep

Group	Conformance Unit / *Profile* Title	Optional
Security	Aes128-Sha256-RsaOaep_Limits	False
Security	AsymmetricEncryptionAlgorithm_RSA-OAEP-SHA1	False
Security	AsymmetricSignatureAlgorithm_RSA-PKCS15-SHA2-256	False
Security	CertificateSignatureAlgorithm_RSA-PKCS15-SHA2-256	False
Security	KeyDerivationAlgorithm_P-SHA2-256	False
Security	Security *Certificate* Validation	False
Security	Security Encryption Required	False
Security	Security Signing Required	False
Security	SymmetricEncryptionAlgorithm_AES128-CBC	False
Security	SymmetricSignatureAlgorithm_HMAC-SHA2-256	False

6.6.165 SecurityPolicy [B] – Basic256Sha256

Table 185 describes the details of the SecurityPolicy [B] – Basic256Sha256. This security Facet defines a security policy for configurations with high security needs. It requires a PKI infrastructure.

As computing power increases, security policies are expected to expire. NIST provides guidelines for expected expiration dates for individual algorithms. These guidelines provided recommended dates at which the algorithm should be replaced or upgraded to a more secure

algorithm. They do not indicate a failure of the algorithm. This security policy has no published end dates as of this time. It is recommended that *Servers* and *Clients* support all security profiles and developers provide the recommended profile as a default. It is up to an administrator to configure the actual exposed security policies.

Table 185 – SecurityPolicy [B] – Basic256Sha256

Group	Conformance Unit / *Profile* Title	Optional
Security	AsymmetricEncryptionAlgorithm_RSA-OAEP-SHA1	False
Security	AsymmetricSignatureAlgorithm_RSA-PKCS15-SHA2-256	False
Security	Basic256Sha256_Limits	False
Security	CertificateSignatureAlgorithm_RSA-PKCS15-SHA2-256	False
Security	KeyDerivationAlgorithm_P-SHA2-256	False
Security	Security *Certificate* Validation	False
Security	Security Encryption Required	False
Security	Security Signing Required	False
Security	SymmetricEncryptionAlgorithm_AES256-CBC	False
Security	SymmetricSignatureAlgorithm_HMAC-SHA2-256	False

6.6.166 SecurityPolicy - Aes256-Sha256-RsaPss

Table 186 describes the details of the SecurityPolicy - Aes256-Sha256-RsaPss. This security Facet defines a security policy for configurations with a need for high security. It requires a PKI infrastructure. As computing power increases, security policies are expected to expire. NIST provides guidelines for expected expiration dates for individual algorithms. These guidelines provide recommended dates at which the algorithm should be replaced or upgraded to a more secure algorithm. They do not indicate a failure of the algorithm. This security policy has no published end dates as of this time. It is recommended that *Servers* and *Clients* support all security profiles and support configurability of actual exposed and default security policies.

Table 186 – SecurityPolicy - Aes256-Sha256-RsaPss

Group	Conformance Unit / *Profile* Title	Optional
Security	Aes256-Sha256-RsaPss_Limits	False
Security	AsymmetricEncryptionAlgorithm_RSA-OAEP-SHA2-256	False
Security	AsymmetricSignatureAlgorithm_RSA-PSS -SHA2-256	False
Security	CertificateSignatureAlgorithm_ RSA-PKCS15-SHA2-256	False
Security	KeyDerivationAlgorithm_P-SHA2-256	False
Security	Security *Certificate* Validation	False
Security	Security Encryption Required	False
Security	Security Signing Required	False
Security	SymmetricEncryptionAlgorithm_AES256-CBC	False
Security	SymmetricSignatureAlgorithm_HMAC-SHA2-256	False

6.6.167 User Token – Anonymous Facet

Table 187 describes the details of the User Token – Anonymous Facet. This Facet indicates that anonymous User Tokens are supported.

Table 187 – User Token – Anonymous Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security User Anonymous	False

6.6.168 User Token – User Name Password Server Facet

Table 188 describes the details of the User Token – User Name Password *Server* Facet. This Facet indicates that a user token that is comprised of a username and password is supported. This user token can affect the behaviour of the *ActivateSession Service*.

Table 188 – User Token – User Name Password Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security Invalid user token	False
Security	Security User Name Password	False

6.6.169 User Token – X509 Certificate Server Facet

Table 189 describes the details of the User Token – X509 *Certificate Server* Facet. This Facet indicates that the use of an X509 certificates to identify users is supported.

Table 189 – User Token – X509 Certificate Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security Invalid user token	False
Security	Security User X509	False

6.6.170 User Token – Issued Token Server Facet

Table 190 describes the details of the User Token – Issued Token *Server* Facet. This Facet indicates that a User Token that is comprised of an issued token is supported.

Table 190 – User Token – Issued Token Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security Invalid user token	False
Security	Security User IssuedToken Kerberos	False

6.6.171 User Token – Issued Token Windows Server Facet

Table 191 describes the details of the User Token – Issued Token Windows *Server* Facet. This Facet further refines the User Token - Issued Token to indicate a windows implementation of Kerberos

Table 191 – User Token – Issued Token Windows Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Profile	User Token – Issued Token *Server* Facet	False
Security	Security User IssuedToken Kerberos Windows	False

6.6.172 User Token – JWT Server Facet

Table 192 describes the details of the User Token – JWT *Server* Facet. This Facet defines support for JSON Web Tokens (JWT) to identify the user during *Session* setup. A JWT is the Access Token format which OPC UA requires when using OAuth2.

Table 192 – User Token – JWT Server Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Azure Identity Provider Authority *Profile*	True
Security	OAuth2 Authority *Profile*	True
Security	OPC UA Authority *Profile*	True
Security	Security Invalid user token	False
Security	Security User JWT IssuedToken	False
Security	Security User JWT Token Policy	False

6.6.173 User Token – User Name Password Client Facet

Table 193 describes the details of the User Token – User Name Password *Client* Facet. This Facet defines the ability to use a user token that is comprised of a username and password.

Table 193 – User Token – User Name Password Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security User Name Password *Client*	False

6.6.174 User Token – X509 Certificate Client Facet

Table 194 describes the details of the User Token – X509 *Certificate Client* Facet. This Facet defines the ability to use an X509 certificates to identify users.

Table 194 – User Token – X509 Certificate Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security User X509 *Client*	False

6.6.175 User Token – Issued Token Client Facet

Table 195 describes the details of the User Token – Issued Token *Client* Facet. This Facet defines the ability to use the User Token - Issued Token (Kerberos) to connect to a *Server*.

Table 195 – User Token – Issued Token Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security User IssuedToken Kerberos *Client*	False

6.6.176 User Token – Issued Token Windows Client Facet

Table 196 describes the details of the User Token – Issued Token Windows *Client* Facet. This Facet defines the ability to use the User Token - Issued Token (Windows implementation of Kerberos) to connect to a *Server*

Table 196 – User Token – Issued Token Windows Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Security User IssuedToken Kerberos Windows *Client*	False

6.6.177 User Token – JWT Client Facet

Table 197 describes the details of the User Token – JWT *Client* Facet. This Facet defines the ability to use JSON Web Tokens (JWT) as user identification during *Session* setup. JWTs are used to request an access token from an external Authorization *Service*.

Table 197 – User Token – JWT Client Facet

Group	Conformance Unit / *Profile* Title	Optional
Security	Azure Identity Provider Authority *Profile*	True
Security	OAuth2 Authority *Profile*	True
Security	OPC UA Authority *Profile*	True
Security	Security User JWT IssuedToken *Client*	False
Security	Security User JWT Token Policy *Client*	False

6.6.178 Global Discovery Server Profile

Table 198 describes the details of the Global *Discovery Server Profile*. This *Profile* is a FullFeatured *Profile* that covers the necessary *Services* and Information Model of a UA *Server* that acts as a GDS.

Table 198 – Global Discovery Server Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Core *Server* Facet	False
Profile	*Method Server* Facet	False

Group	Conformance Unit / *Profile* Title	Optional
Profile	SecurityPolicy – Basic128Rsa15	False
Profile	SecurityPolicy – Basic256	False
Profile	Standard DataChange *Subscription Server* Facet	False
Profile	UA-TCP UA-SC UA-Binary	False
Profile	User Token – X509 *Certificate Server* Facet	False
GDS	GDS Application Directory	False
GDS	GDS LDS-ME Connectivity	False
Security	Security Default ApplicationInstance Certificate	False
Session Services	*Session* Minimum 50 Parallel	False

6.6.179 Global Discovery Server 2017 Profile

Table 199 describes the details of the Global *Discovery Server* 2017 *Profile*. This *Profile* is a FullFeatured *Profile* that covers the necessary *Services* and Information Model of a UA *Server* that acts as a GDS.
This *Profile* supersedes the “Global *Discovery Server Profile*”.

Table 199 – Global Discovery Server 2017 Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Core 2017 *Server* Facet	False
Profile	*Method Server* Facet	False
Profile	Standard DataChange *Subscription* 2017 *Server* Facet	False
Profile	UA-TCP UA-SC UA-Binary	False
GDS	GDS Application Directory	False
GDS	GDS LDS-ME Connectivity	False
GDS	GDS Query Applications	False
Security	Security Default ApplicationInstance *Certificate*	False
Security	Security Policy Required	False
Session Services	*Session* Minimum 50 Parallel	False

6.6.180 Global Discovery and Certificate Management Server

Table 200 describes the details of the Global *Discovery* and *Certificate* Management *Server*. This *Profile* is a FullFeatured *Profile* that covers the necessary *Services* and Information Model of a UA *Server* that acts as a GDS and a global *Certificate* Manager.

Table 200 – Global Discovery and Certificate Management Server

Group	Conformance Unit / *Profile* Title	Optional
Profile	Auditing *Server* Facet	False
Profile	File Access *Server* Facet	False
Profile	Global *Discovery Server Profile*	False
Profile	SecurityPolicy [B] – Basic256Sha256	False
Profile	Standard *Event Subscription Server* Facet	False
GDS	GDS *Certificate* Manager Pull Model	False
GDS	GDS *Certificate* Manager Push Model	False

6.6.181 Global Discovery and Certificate Mgmt 2017 Server

Table 201 describes the details of the Global *Discovery* and *Certificate* Mgmt 2017 *Server*. This *Profile* is a FullFeatured *Profile* that covers the necessary *Services* and Information Model of a UA *Server* that acts as a GDS and a global *Certificate* Manager.
This *Profile* supersedes the “Global *Discovery* and *Certificate* Management *Server*”.

Table 201 – Global Discovery and Certificate Mgmt 2017 Server

Group	Conformance Unit / *Profile* Title	Optional
Profile	Auditing *Server* Facet	False
Profile	File Access *Server* Facet	False
Profile	Global *Discovery Server* 2017 *Profile*	False

Group	Conformance Unit / *Profile* Title	Optional
Profile	Standard *Event Subscription Server* Facet	False
GDS	GDS *Certificate* Manager Pull Model	False

6.6.182 Global Certificate Management Client Profile

Table 202 describes the details of the Global *Certificate* Management *Client Profile*. This *Profile* is a FullFeatured *Profile* that uses the Push Model for the management of *Certificates* and Trust Lists.

Table 202 – Global Certificate Management Client Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Core *Client* Facet	False
Profile	*Discovery Client* Facet	False
Profile	Entry Level Support 2015 *Client* Facet	False
Profile	File Access *Client* Facet	False
Profile	*Method Client* Facet	False
Profile	SecurityPolicy [B] – Basic256Sha256	False
Profile	SecurityPolicy – Basic256	False
Profile	UA-TCP UA-SC UA-Binary	False
GDS	GDS *Certificate* Manager Push Model	False
Security	Security Default ApplicationInstance Certificate	False

6.6.183 Global Certificate Management Client 2017 Profile

Table 203 describes the details of the Global *Certificate* Management *Client* 2017 *Profile*. This *Profile* is a FullFeatured *Profile* that uses the Push Model for the management of *Certificates* and Trust Lists.
This *Profile* supersedes the “Global *Certificate* Management *Client Profile*”.

Table 203 – Global Certificate Management Client 2017 Profile

Group	Conformance Unit / *Profile* Title	Optional
Profile	Core 2017 *Client* Facet	False
Profile	*Discovery Client* Facet	False
Profile	Entry Level Support 2015 *Client* Facet	False
Profile	File Access *Client* Facet	False
Profile	*Method Client* Facet	False
Profile	UA-TCP UA-SC UA-Binary	False
GDS	GDS *Certificate* Manager Push Model	False
Security	Security Default ApplicationInstance *Certificate*	False

6.6.184 Global Service Authorization Request Server Facet

Table 204 describes the details of the Global *Service* Authorization Request *Server* Facet. This Facet defines the capability of a *Server* (like a GDS) to provide access tokes to OPC UA *Clients* via an Authorization *Service* as defined in UA Part 12.

Table 204 – Global Service Authorization Request Server Facet

Group	Conformance Unit / *Profile* Title	Optional
GDS	GDS Authorization *Service Server*	False

6.6.185 Global Service KeyCredential Pull Facet

Table 205 describes the details of the Global Service KeyCredential Pull Facet. This Facet requires providing the *Information Model* for Pull Management as defined in UA Part 12. For example KeyCredentials are needed to access an Authorization *Service* or a Broker. OPC UA *Clients* use this Information Model to request and update KeyCredentials they need.

Table 205 – Global Service KeyCredential Pull Facet

Group	Conformance Unit / *Profile* Title	Optional
GDS	GDS Key Credential *Service* Pull Model	False

6.6.186 Global Service KeyCredential Push Facet

Table 206 describes the details of the Global Service KeyCredential Push Facet. This Facet requires the use of KeyCredential Push Management functions to set or update credentials in an OPC UA Server. For example KeyCredentials are needed to access an Authorization *Service* or a Broker. This OPC UA *Server* in turn has to provide the KeyCredentialConfigurationType *Objects* that represent required credentials.

Table 206 – Global Service KeyCredential Push Facet

Group	Conformance Unit / *Profile* Title	Optional
GDS	GDS Key Credential *Service* Push Model	False

_____