OPC UA:
The Information Backbone of the Industrial Internet

geautomation.com

1 Introduction

Automation systems built using GE's Industrial Internet Control System (IICS) technology are comprised of applications distributed on different hardware platforms and unified with a common information backbone. This backbone uses the OPC Unified Architecture (OPC UA) framework. OPC UA is a secure, platform-independent, scalable, and object-oriented architecture for representing and communicating information.

By using OPC UA, information can be modeled so that applications can inherently derive its meaning and consequently make better decisions based on that meaning. This enables applications to gain intelligence that can lead to new and exciting outcomes in data management.

In addition, OPC UA provides a mechanism to protect the confidentiality and integrity of information and to determine whether applications are trustworthy--a fundamental need of the Industrial Internet.

This paper will provide an overview of the OPC UA and how it is used within IICS to generate beneficial outcomes for customers.

2 OPC UA Overview

OPC UA is a specification developed and maintained by the OPC Foundation. The mission of the OPC Foundation is to manage a global organization in which users, vendors, and consortia collaborate to create data transfer standards for secure and reliable interoperability in industrial automation.

The OPC UA specification is composed of several parts. Parts 1 through 5 form the basic concepts of OPC UA and are defined independently of the implementation. These parts describe the Security Model, Address Space Model, Information Model and Services used for OPC UA applications. The Services and Security Model are described with abstract definitions that can be mapped to specific implementations.

Part 6 defines mappings of the abstract specifications in Parts 1 through 5 to technologies used for implementation. This includes technologies for implementing data encoding, security protocols, and transport protocols necessary to create a real application.

Part 7 breaks down the features of OPC UA into conformance units. A set of conformance units define a profile. An OPC UA application should be built to comply with one of the defined profiles based on requirements of the application and the resources available on the device that will host the application. This allows scaling OPC UA so it can be deployed on small devices that comply with a profile with a small set of conformance units to very large devices that comply with a profile with a large set of conformance units.

Parts 8 through 11 extend the basic concepts in Parts 1 through 5 to cover the functionality found in OPC Classic. This covers Data Access, Alarms & Conditions, and Historical Data Access. These parts describe extensions to the OPC UA Information Model associated with the OPC Classic functionalities; e.g. Part 8 describes the information model for Data Access.

There are other parts to the specification. The organization of the specification allows it to evolve with new parts to address new and emerging requirements. For instance, an important extension to the specification is Part 14 – PubSub. This extension addresses use cases for controller-to-controller communication, public subscriptions, and integration with message brokers.

The remainder of this section will explain how the specification came to be and its core concepts.

2.1 The Emergence of OPC UA

In the mid-'90s, the OPC Foundation published three separate specifications referred to collectively as OPC Classic. Included in OPC Classic are specifications for Data Access, Alarm & Events, and Historical Data Access. OPC Classic was quickly adopted as a mechanism to abstract industrial specific protocols into a standardized interface that allowed software like HMI/SCADA to communicate with a range of devices. This enabled the seamless integration of automation products from different vendors into one system. But, OPC Classic had drawbacks, such as dependence on Microsoft COM/DCOM technology, three separate data models, and limited protection against unauthorized data access. As industrial automation evolved, these and other drawbacks made it clear that OPC Classic had to be replaced.

The replacement of OPC Classic started with the initial release of OPC UA in 2006. OPC UA was created to remove the limitations imposed by OPC Classic, e.g. dependence on Microsoft technology, and to address emerging requirements for security, communication across firewalls, and support of complex data structures. Where OPC Classic had separate data models that made it difficult to connect different types of information, OPC UA unified all information into a single AddressSpace. Beyond unification of data, OPC UA features a service-oriented architecture that integrates all the functionality of OPC Classic into one extensible framework--making it ideal for the Industrial Internet's information backbone.

2.2 What is OPC UA?

OPC UA specifies how information is modeled and communicated in a system like the Industrial Internet. It specifies abstract services that can be implemented in language-specific APIs and mapped to different communication stacks. Keeping the service definitions abstract allows OPC UA to be extended over time to new and emerging technologies without changing the underlying design. It also specifies an AddressSpace model that provides a standard way for servers to represents data in an object-oriented manner to clients.

At its core, the OPC UA architecture defines clients and servers as interacting partners where clients consume information that servers provide.

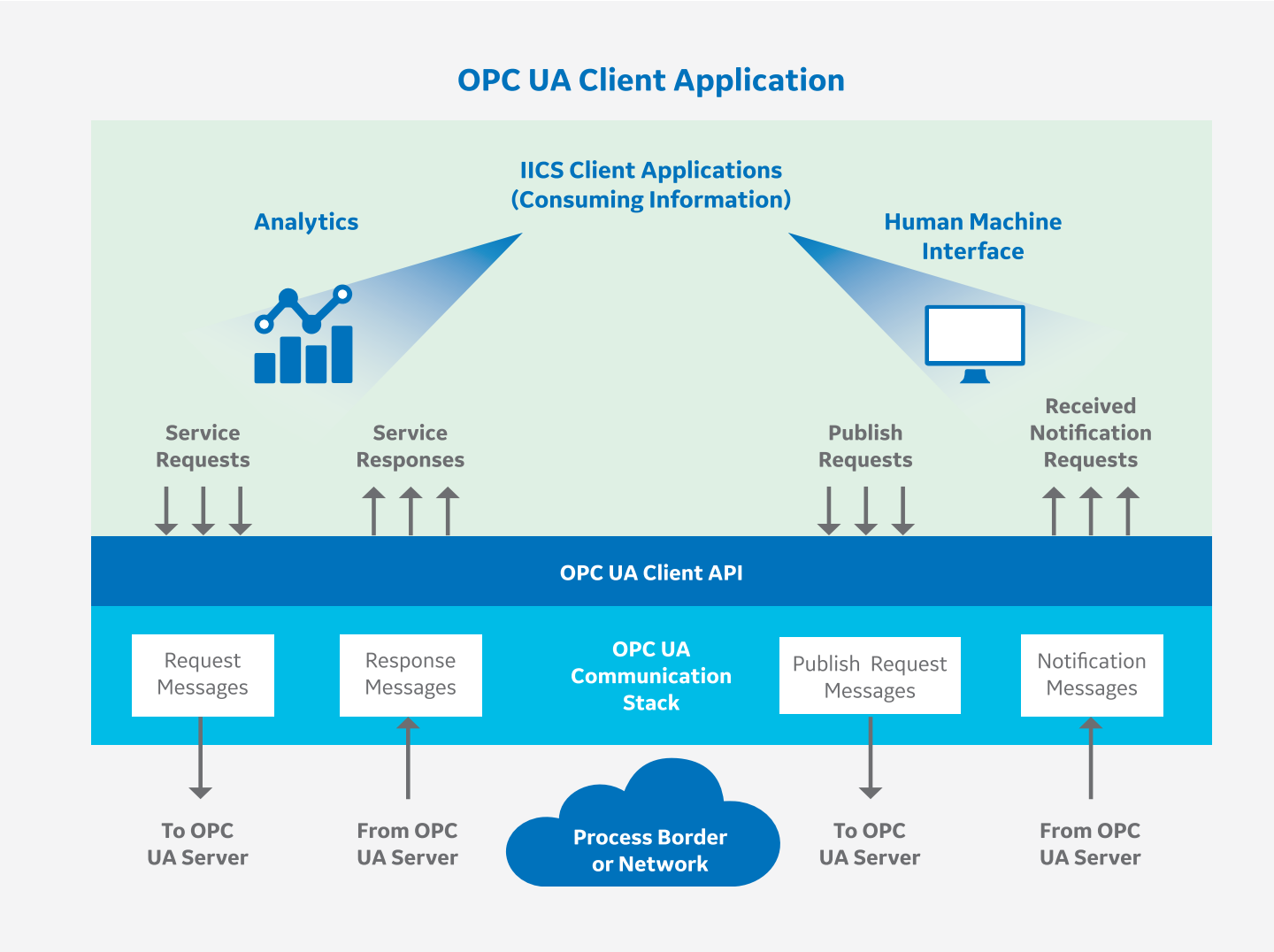

Figure 1 - OPC UA Client

2.3 The OPC UA Client

IICS client applications, such as analytics and human-machine interfaces, fulfill specific use cases that bring benefit to end-users. For example, the human-machine interface might request all the information required to display a report or to trend related process variables in a chart. These client applications use OPC UA as the infrastructure to interact with servers to access information required by the use cases.

Figure 1 illustrates this interaction pattern where client applications make requests to the OPC UA Client API. The OPC UA Communication Stack then translates these requests into messages that are sent to the server through the underlying communication framework. The server processes the received requests and sends a response back to the OPC UA Communication Stack on the client. The OPC UA Communication Stack then delivers it to the client application through the OPC UA Client API.

OPC UA clients can subscribe to receive notifications when an event occurs or when a data value changes. In this case, the events and data values are referred to as monitored items. The server will monitor these items and send notifications to the client in response to a publish request message from the client. This is the preferred method for clients to receive cyclical updates of variable values.

2.4 The OPC UA Server

OPC UA servers expose information for clients to find and consume. The collection of information that servers make available to clients is called the AddressSpace. The AddressSpace standardizes how objects are represented. These objects are defined in terms of variables, methods, and their relationships to other objects as shown in Figure 3.

The OPC UA AddressSpace unifies the three classic data models (Data Access, Alarm & Events, and Historical Data Access) into one information model. This unification makes it easy to connect the dots between data values that are read to events that are raised based on those data values.

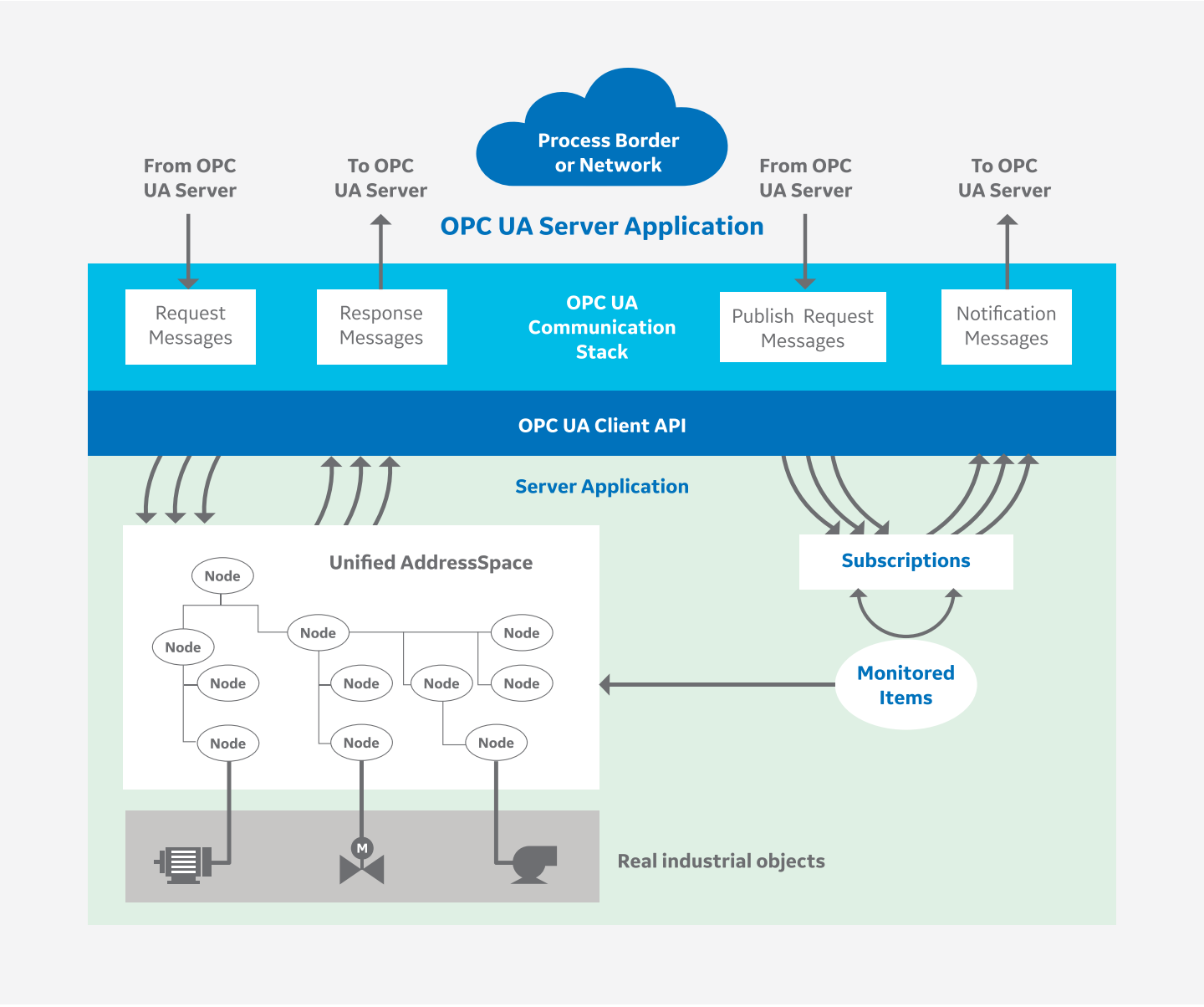

Figure 2 - OPC UA Server

OPC UA objects and the things that comprise them – like variables and methods - are represented as nodes in the AddressSpace as shown in Figure 2. Nodes are described by attributes and interconnected by references to form a graph-like model that can represent simple to complex industrial objects such as motors, valves, and pumps. OPC UA defines different classes of nodes as shown in Figure 4. Each node in a server's AddressSpace is an instance of one of these NodeClasses.

Clients access the attributes of nodes, such as the value attribute of a Variable NodeClass, by using the services provided by the server, such as the *Read* service.

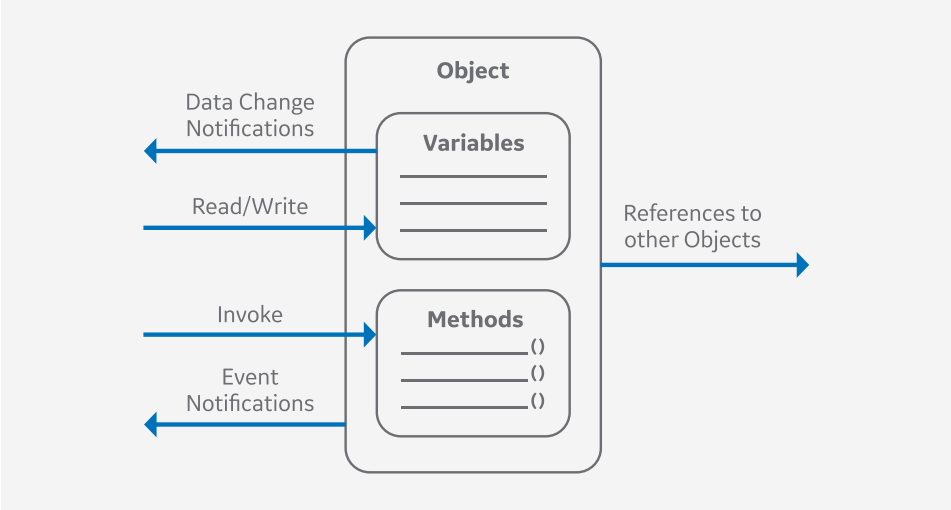

Figure 3 - OPC UA Object Model (Part 3)

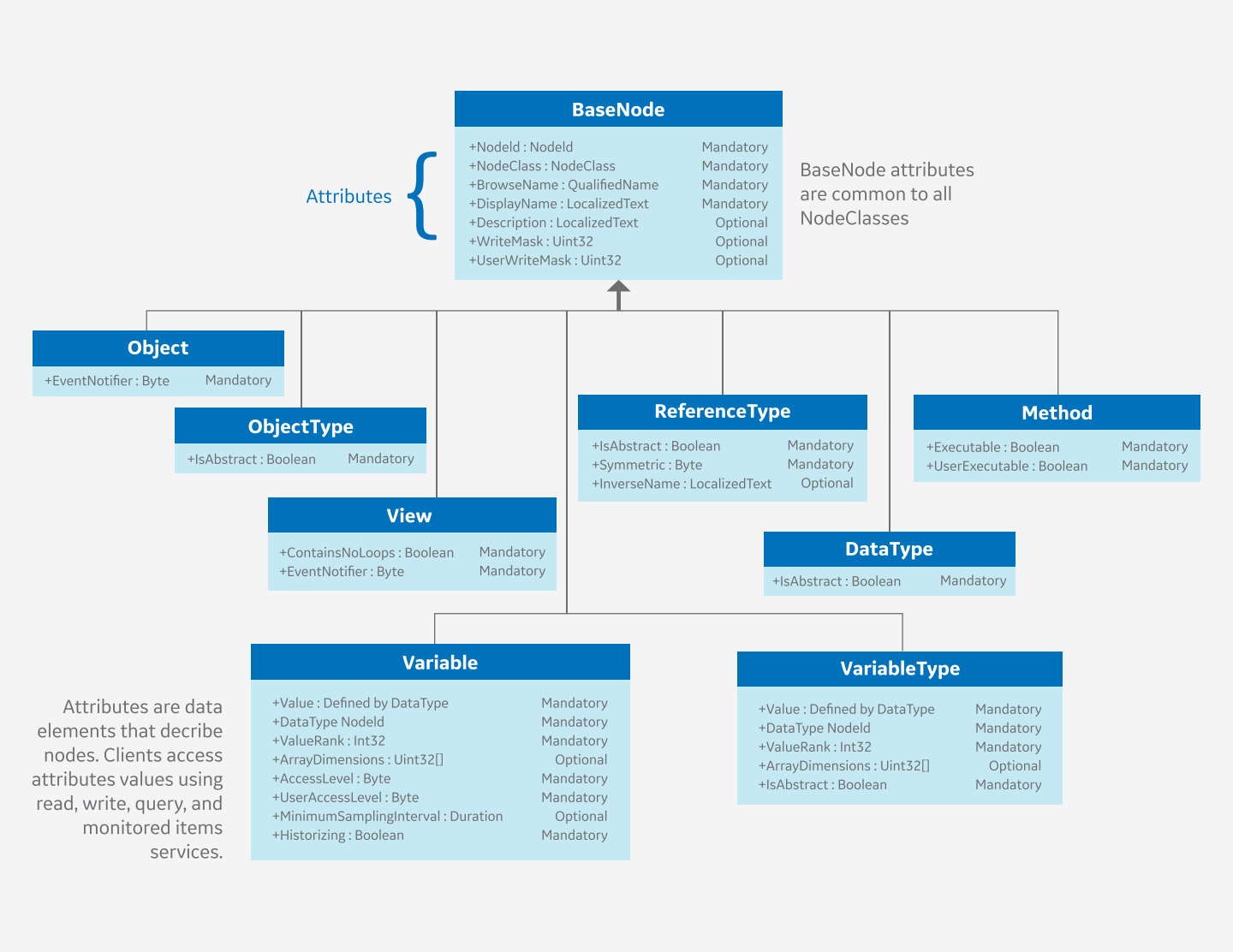

Figure 4 - OPC UA Node Classes

2.5 OPC UA Information Models

Because OPC UA uses object-oriented techniques, it can formulate an information model that serves a specific problem domain. By using ObjectType, VariableType, DataType, and ReferenceType NodeClasses, an information model can be constructed where the type definitions convey meaning and can represent entities found in the problem domain. This allows vendors and standards organizations to create a known object model that client applications and tools can be written against. This is one of the benefits of IICS. IICS has an integrated object model built using OPC UA objects that allows systems that comprise IICS--from the PLC to the SCADA system to the cloud platform--to find the right information and present that information in the right context at the right time.

For example, Figure 5[1]shows an ObjectType node representing a motorized valve. The variable, method, and object nodes under the ObjectType node is called *InstanceDeclaration*. If all nodes under an ObjectType node are mandatory, then every instance of that ObjectType will have the same *InstanceDeclaration.* Therefore, knowledge of an ObjectType's *InstanceDeclaration* can be used to create HMI widgets and apps that can be used for all instances of that ObjectType. For example, an HMI widget to visualize the status of a motorized valve and an app to determine when the valve needs to have preventive maintenance can be created using the information given by the type MotorizedValveType.

Along with exposing information, an OPC UA server provides clients with a sophisticated set of services, including discovery services, subscription services, query services, and node management services as defined in Part 4.

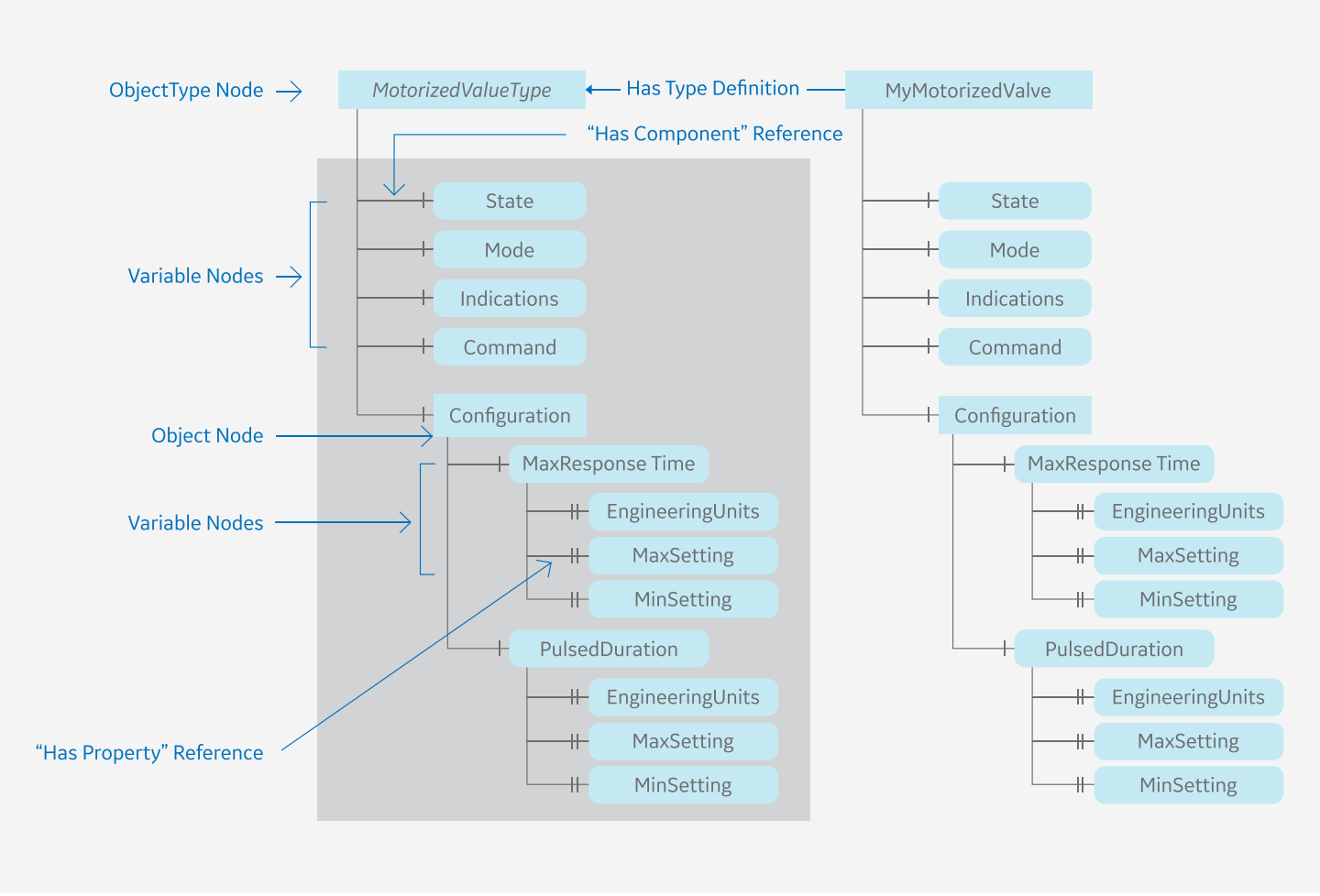

Figure 5 - OPC ObjectType Definition for a Motorized Operated Valve

[1]OPC UA provides a graphical notation for nodes based on their NodeClass. That is each NodeClass has a unique graphical form that can be used to express OPC UA data models. This graphical notation is used in Figure 5 and is defined in Part 3 of the specification.

2.6 OPC UA Communication Stack

The OPC UA Communication Stack, as shown in Figure 6, provides low-level functionality for both the OPC UA client and server applications. To send a message the stack provides functionality to encode, secure, and format the message. To receive a message, the stack provides functionality to decode, decrypt, and unpack the message. The implementation of this functionality is defined in Part 6 of the specification. One possible implementation is shown in Figure 6 where the low-level functions are mapped to an optimized binary protocol based on TCP.

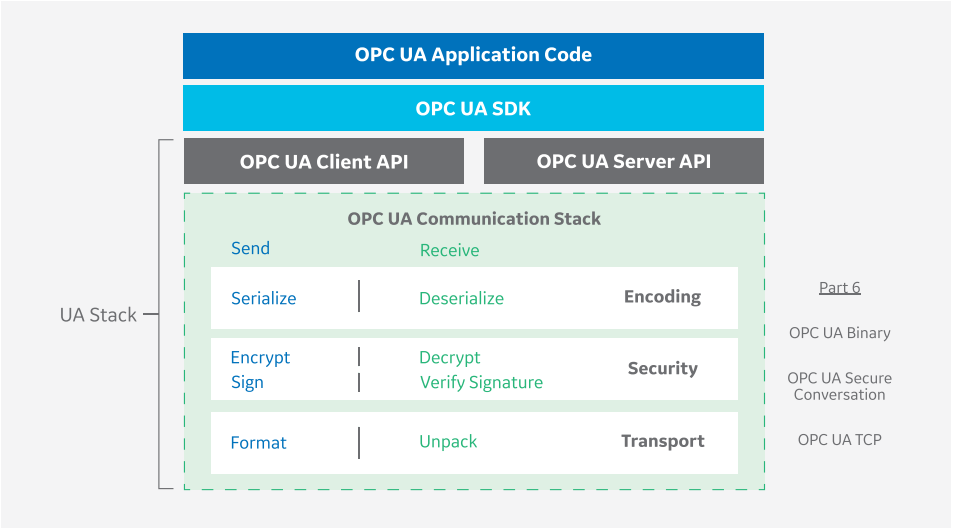

Figure 6 - OPC UA Communication Stack

2.7 System of Systems

The platform independence and scalability of OPC UA make it a critical technology for the Industrial Internet. An intelligent device with an embedded OPC UA server and client can achieve bi-directional communication with other intelligent devices. An aggregation server can concentrate, normalize, and enrich information from underlying servers and then make that aggregated information available to high-level clients as shown in Figure 7. The aggregation server plays an important role in minimizing the number of connections that resource-limited devices need to manage.

The various systems that comprise the Industrial Internet will run on a variety of operating systems; an OPC UA aggregation server might run on Windows whereas an OPC UA embedded server might run on VxWorks. The ability of OPC UA to provide secure data exchange independent of platform and operating-system is essential to converge disparate systems into one secure system. The resulting chain of systems, from low-level devices to SCADA systems to enterprise applications, are integrated with OPC UA to form a system of systems.

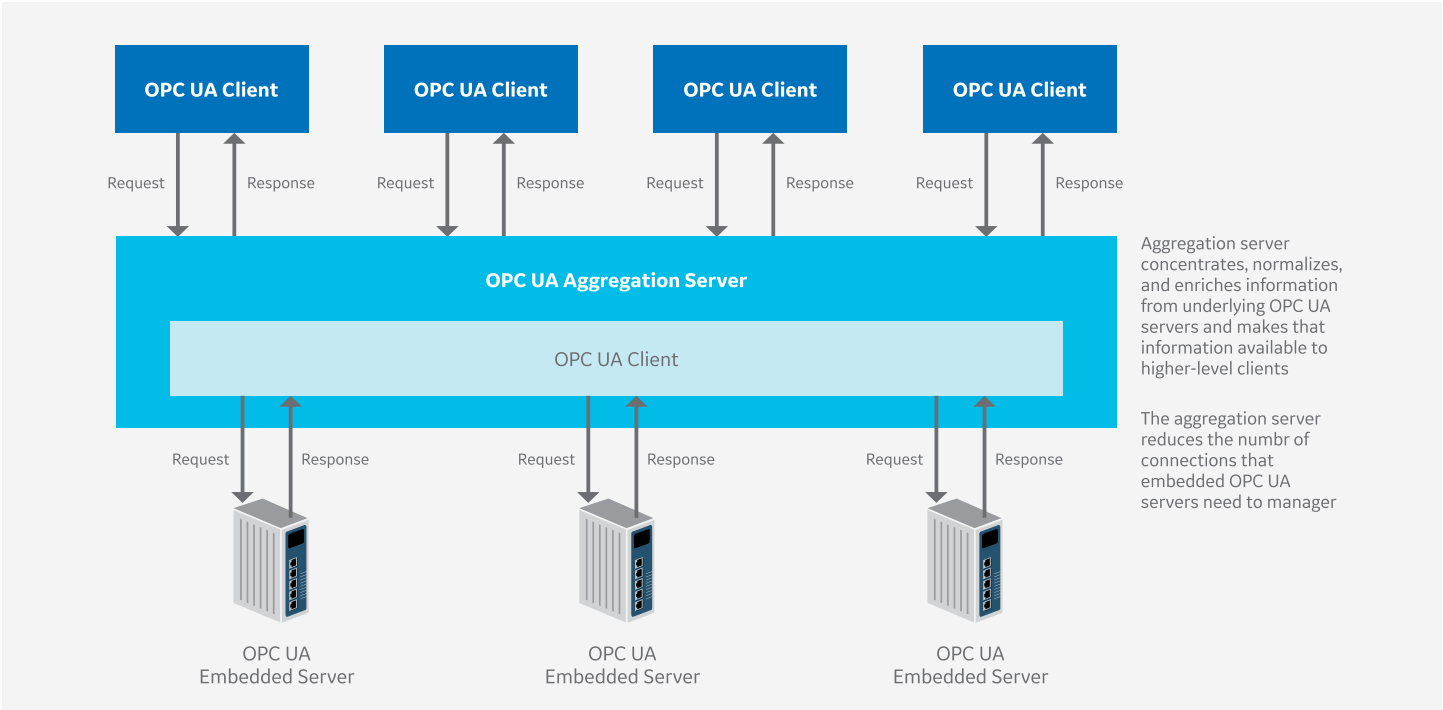

Figure 7 – OPC UA Aggregation Server

2.8 OPC UA Security

Security is an integral part of OPC UA technology. An OPC UA server provides a set of services dedicated to creating a secure connection. Once created, it will apply the security protocol to messages between the client and server to ensure the integrity and confidentiality of messages.

An overview of how a secure connection is achieved between an OPC UA server and client is provided to bring insights into the workings of OPC UA security. Table 1 provides a list of security-related terms.

Term	Definition
Application Instance Certificate	This X.509v3 certificate identifies an instance of an OPC UA application running on a host. This type of certificate is used to open a secure channel.
Certificate	An electronic document with information affirmed by a trusted party. The information includes such things as signature algorithm, validation period, and the public key. Its primary usage is to distribute public keys used to implement the security policies of a system.
Endpoint	Physical address available on a network that allows clients to access one or more services provided by a server.
Global Discovery Server	Used to manage certificates and discover server endpoints within an OPC UA system.
Public Key Cryptography (PKC)	A cryptography technique that uses a paired public and private key. The public key is made available to other applications and the private key is kept secret. For example, an OPC UA client would use the server's public key to encrypt a request message. Upon receipt of the request message, the server would use its private key to decrypt the message. PKC is used to sign data as follows. An OPC UA client would use its private key to sign a request message. Upon receipt of the request message, the server would use the client's public key to verify the signature.
Secret	Information that is used to derive cryptographic keys for securing messages using symmetric cryptography. Secrets from the client and server are used to derive symmetric keys and are exchanged during the creation of a Secure Channel.
Secure Channel	A communication path established between an OPC UA client and server that have authenticated each other using certain OPC UA services and for which security parameters have been negotiated and applied.
Security Mode	The mode to secure messages between a client and server. The possible modes are SignAndEncrypt, Sign, and None.
Symmetric Key	A key shared by the server and client to encrypt and decrypt messages. This technique is much less CPU intensive then PKC.
X509.v3	A specific format of a certificate defined by x.509.

Table 1 - Security Related OPC UA Terms

The overview of creating an OPC UA secure connection is described as follows:

1. Find available endpoints to establish a *Secure Channel*

a) If an OPC UA client does not have pre-configured information on how to connect to a server, it can send an unsecured request to the discovery endpoint of the server to get descriptions of the server's available endpoints. The discovery endpoint of a server is either well-known or obtained from a Global Discovery Server. Included in the returned endpoint descriptions is all the information required for the client to establish a *Secure Channel* between itself and the endpoint including the server's *Application Instance Certificate* and the supported *Security Mode.*

b) The client selects an available endpoint that it can handle from a security perspective and validates the server's *Application Instance Certificate* for the endpoint to ensure that it is trustworthy.

2. Open *Secure Channel* to selected endpoint

a) A client then makes a request to open a *Secure Channel* to the selected endpoint of the server in accordance with the *Security Mode.* If the *Security Mode* is None, then the request message is unsecure. If the Security Mode is either Sign or SignAndEncrypt, then the request message is secured using *Public Key Cryptography (PKC).* In the request message, the client sends its *Application Instance Certificate* and a secret.

b) Once the open *Secure Channel* request message is received by the server, it validates the *Application Instance Certificate* of the client to ensure that it is trustworthy. The client's *Application Instance Certificate* is provided in the unencrypted part of the request message. If the client is trustworthy, then the server uses PKC to decrypt and verify the signature of the request message. The server then sends a response message, which includes a secret, to the client, secured in a similar fashion to the request.

c) The client receives the secured response from the server and can, therefore, decrypt and verify the signature of the server using PKC. The use of PKC during the open *Secure Channel* request-response message exchange is necessary so that: (1) the client can authenticate the server, (2) the server can authenticate the client, and (3) secrets can be exchanged between the client and server so that *Symmetric Keys* can be derived.

d) If *Symmetric Keys* are derived on both the client and server, then a *Secure Channel* is established. The *Symmetric Keys* are used for encrypting and signing all subsequent messages on the *Secure Channel.*

3. Create Session

a. The client makes a request to the server to create a session on top of the *Secure Channel.*

b. In response to this request, the server returns a session identifier and authentication token. The session identifier is used to identify the session in audit logs and in the server's AddressSpace. The authentication token is used to associate an incoming request with a session.

4. Activate Session

a) The client makes a request to the server to activate the session. The purpose of this request is to associate a user identity with a session. Therefore, the request includes the user's credentials along with proof that the request is coming from the same client that created the session.

b) Once the server receives the request, it validates the user credentials and that the client is the same that created the session. If these validations succeed, the session is activated and a connection between the client and server is established.

Figure 8 shows the secure connection that is achieved by the four-step process described above. Once the secure connection is achieved, the client can access the information from the server's AddressSpace through secure request/response messaging protocol.

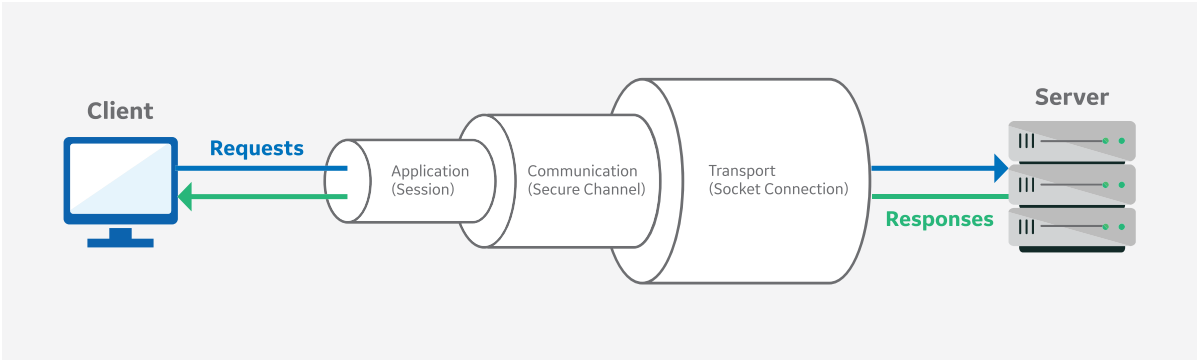

Figure 8 - Communication Channels

Figure 9 shows how the communication channels that form a connection between a client and server relate to the layers in the OPC UA security architecture.

The Application Layer is used to transmit information, settings, and commands in a session. It also manages user authentication and authorization. The Communication Layer protects the integrity and confidentiality of information by means of the *Secure Channel*. It also authenticates OPC UA applications. The Transport Layer handles the transmission, reception, and transport of information that is provided by the Communication Layer.

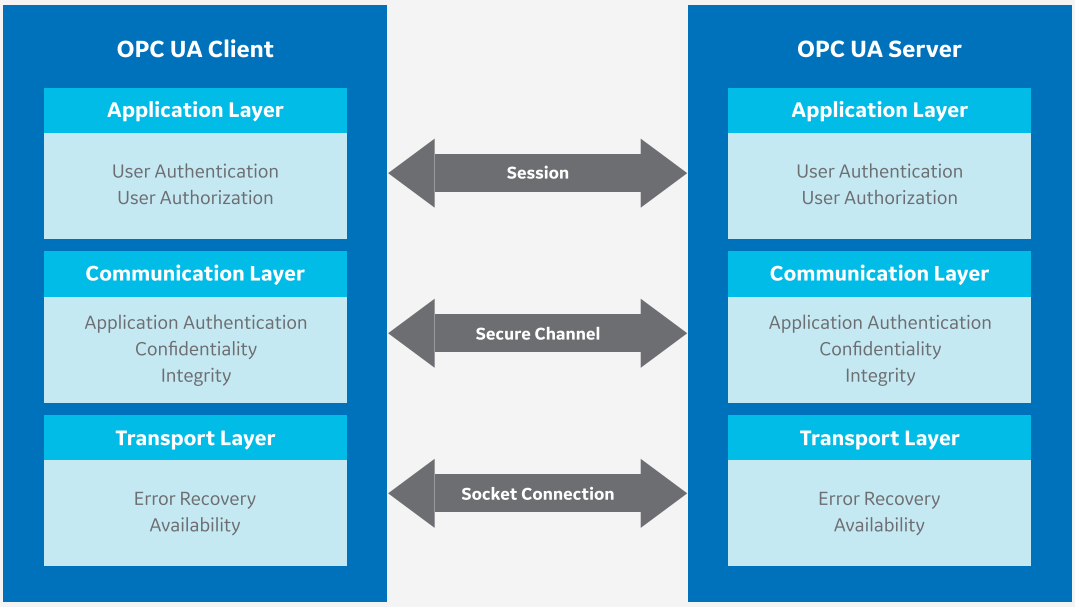

Figure 9 - OPC UA Security Architecture

2.9 High Availability with OPC UA

OPC UA enables servers, clients, and networks to be redundant by providing services to achieve redundancy in a standardized manner.

OPC UA clients, especially ones that aggregate data from several underlying servers, typically need to be fault tolerant. To meet this requirement OPC UA allows identically configured clients to work as a redundant pair. If one client fails then the other client starts consuming the same information, ideally without any information loss. This high-availability mechanism has one client active and the other in a backup mode. The backup client monitors status information of the active client in the server's AddressSpace. When the active client fails, the status information changes and the backup client takes actions to ensure the continuation of information flow between the client and server.

OPC UA server redundancy allows clients to have multiple sources to obtain the same information. OPC UA has two modes of server redundancy: transparent and non-transparent. In transparent redundancy, the fail-over from one server to another does not require any actions from the client; actually, the client is unaware that the failure has occurred. In non-transparent redundancy, the failure from one server to another requires the client to take actions to ensure the continuation of information flow between the client and server.

Network redundancy provides multiple paths for client and servers to communicate thus eliminating a single point of failure.

3 OPC UA in the Context of IICS

IICS provides an intelligent and secure infrastructure for customers to create their control applications. These qualities in part are achieved with OPC UA. The next sections will illustrate how OPC UA in the context of IICS brings intelligence to data and security to a system.

3.1 IICS Information Model

IICS at its core has an object model that represents industrial equipment and the entities that are used to control equipment, such as control modules. These entities are modeled using OPC UA constructs as defined in Figure 4. The object model provides semantics around the data for things like a level transmitter. This object definition enables HMI graphics, alarms, apps, and logic to work seamlessly together using the same information model for a given ObjectType. This concept is expanded beyond simple devices, like transmitters, to include more complex structures that can be used to create a complete application. This brings new opportunities to compose high-quality applications in less time.

The IICS Information Model provides semantic value to data by defining:

- ObjectTypes that represent entities in a domain
- Concrete names for properties using the BrowseName attribute
- ReferenceTypes that convey the relationship between two nodes; e.g. "HasCommandVar" ReferenceType is used to reference an object to one of its command variables. The reference forms the predicate between the source node and targeted node.
- VariableTypes that provide context and restrictions for usage

Where OPC Classic provided minimal semantics such as tag name and engineering units, OPC UA provides a whole extensible infrastructure for information modeling. Leveraging this infrastructure to its fullest has the potential to transform industrial automation by providing semantic interoperability from the device to the cloud. Semantic interoperability allows client applications to generate reports, HMI screens, controls logic and analytics automatically and when required on demand. This is powerful because clients not only receive data identified by tag name, but information that can be interpreted so that the client can make decisions real-time on how to convey that information to facilitate better outcomes.

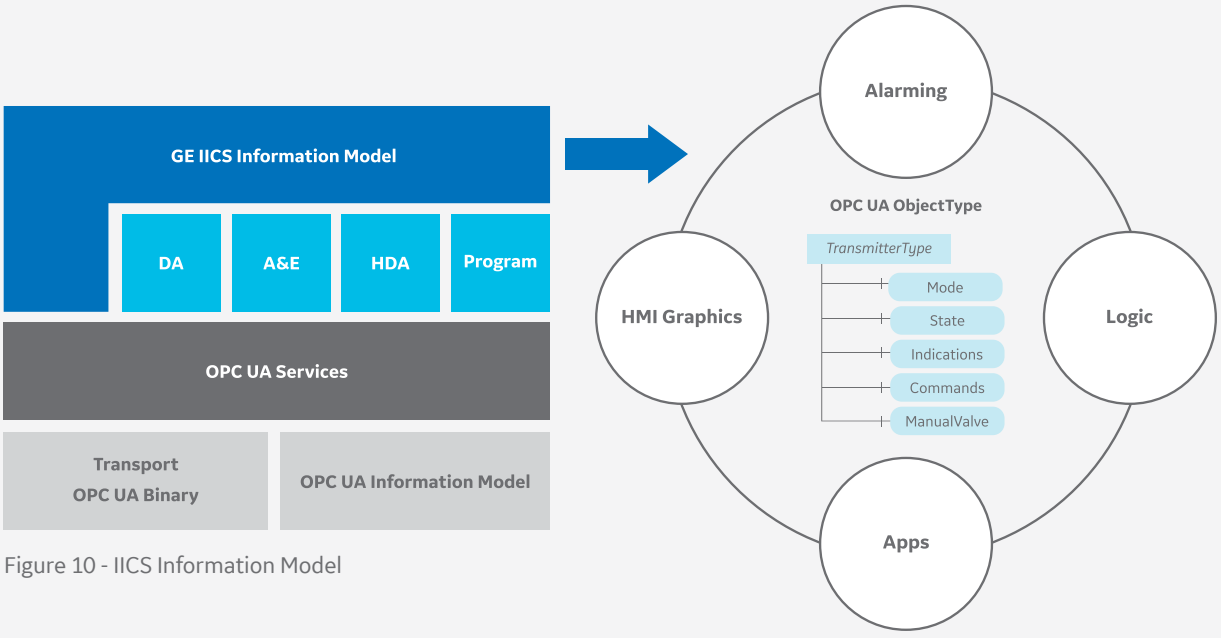

Figure 10 - IICS Information Model

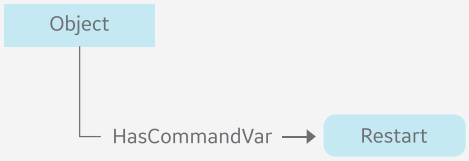

Figure 11 - Example ReferenceType

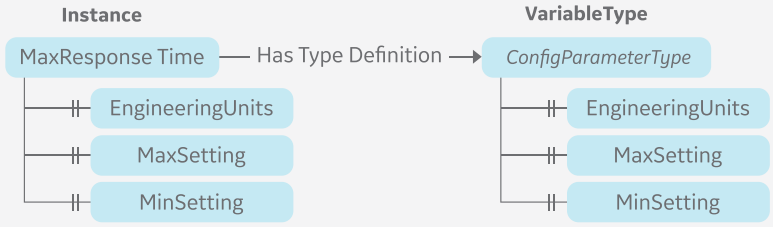

Figure 12 - Example VariableType

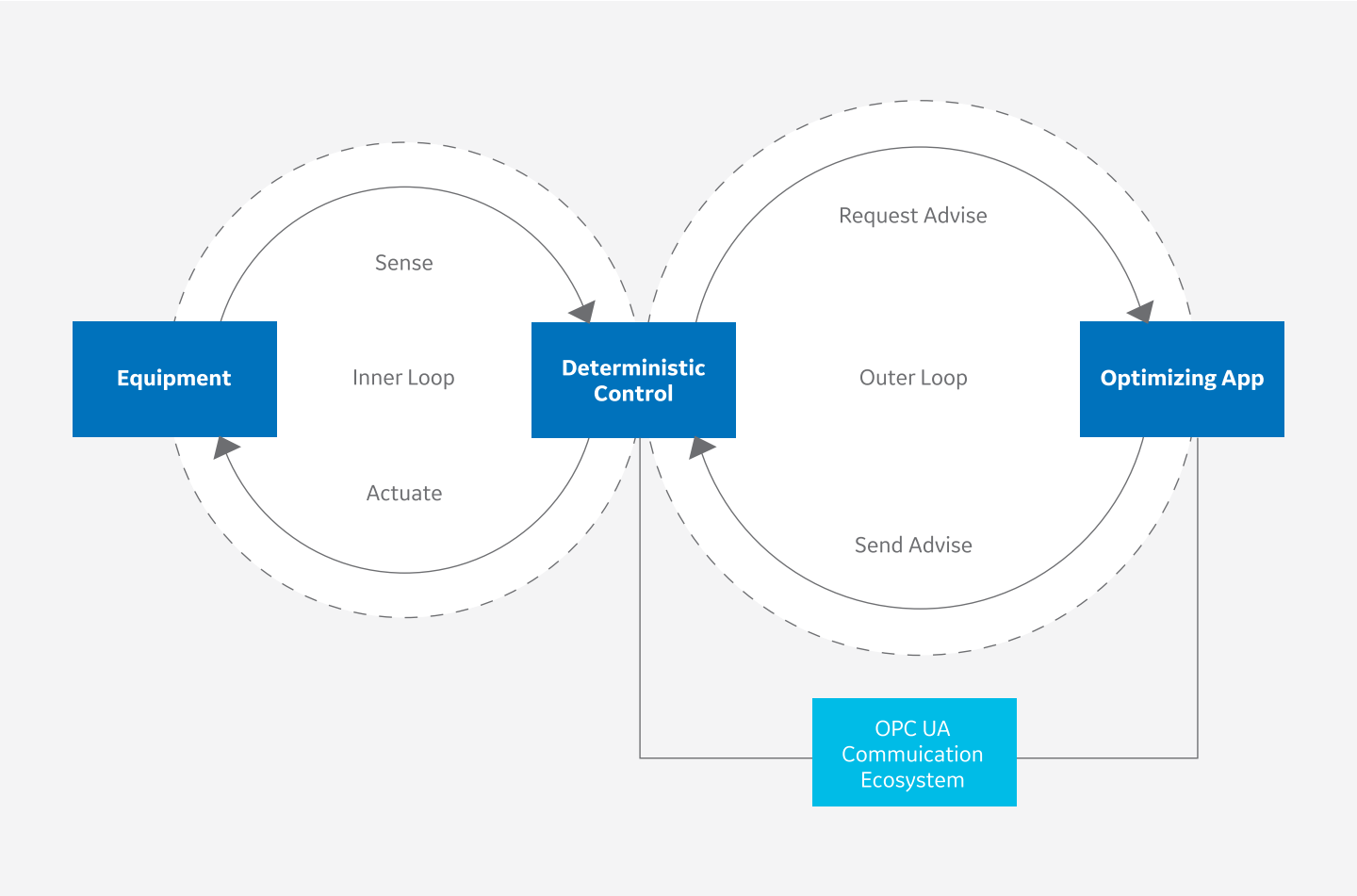

Figure 13 - Inner and Outer Loop Control Strategy

3.2 OPC UA and Inner/Outer Loop Control

A key feature of IICS is a built-in ecosystem to support inner and outer loop control strategies. As shown in Figure 13, an inner loop provides deterministic control and an outer loop provides non-deterministic advice to the inner loop to achieve a goal, such as minimizing a cost function.

The inner loop runs on a PACSystem* controller with an embedded OPC UA server. The outer loop runs on a variety of platforms that use GE's Industrial Object Server technology. The Industrial Object Server is a software stack that includes an OPC UA aggregation server, which aggregates information from underlying servers and then exposes that information through an OPC UA server to client apps. The client apps access the aggregated information by using language-specific APIs provided in the Industrial Object Server's software stack as shown in Figure 14.

One of the advantages of the inner and outer loop ecosystem is the ability to couple the knowledge of OPC UA type definitions with the OPC UA services for discovering information. The type definition indicates to the system what information to seek, and the discovery services give the system the means to find that information. This gives IICS the ability to configure and deploy apps in a semi-automated manner.

One example would be using services that find specific nodes based on a type definition. In this case, the app developer would reference variables in the app based on knowledge of the *InstanceDeclaration* of the ObjectType needed by the app; i.e. reference the app variables in the context of the ObjectType. Then a discovery service, specifically TranslateBrowsePathToNodeId, can be used to map concrete variables to the app variables. This example is illustrated in Figure 15.

Another scenario is the ability of an app to discover objects of a certain type. For example, an app could provide optimization methods for pumps. When deployed, the app could discover all objects of type "pump" and then subscribe to their data. The same mechanism could be used to automatically fulfill many use cases, from populating reports with data to creating all HMI graphical objects for a given type, ultimately leading to a self-configuring system.

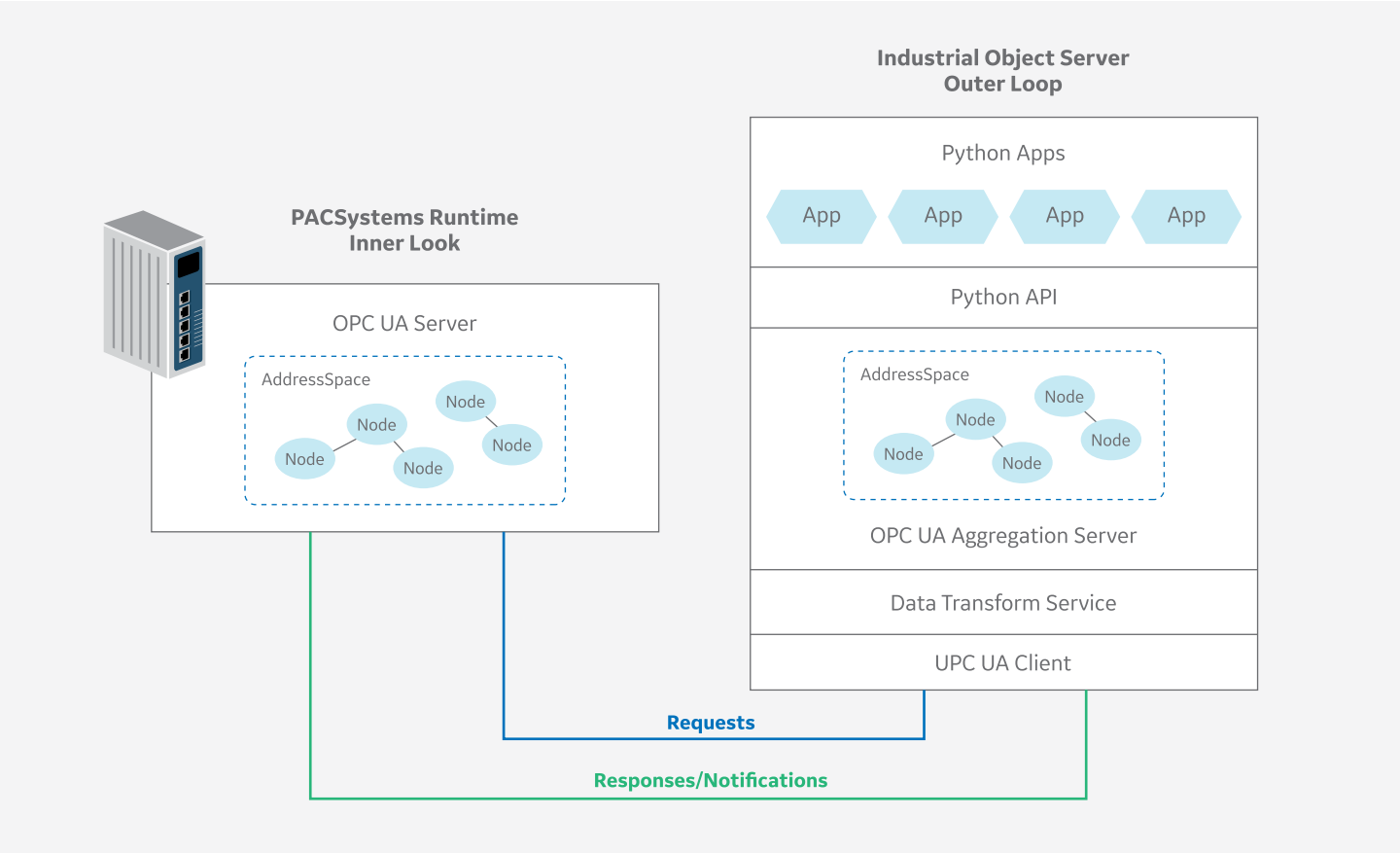

Figure 14 - Industrial Object Server

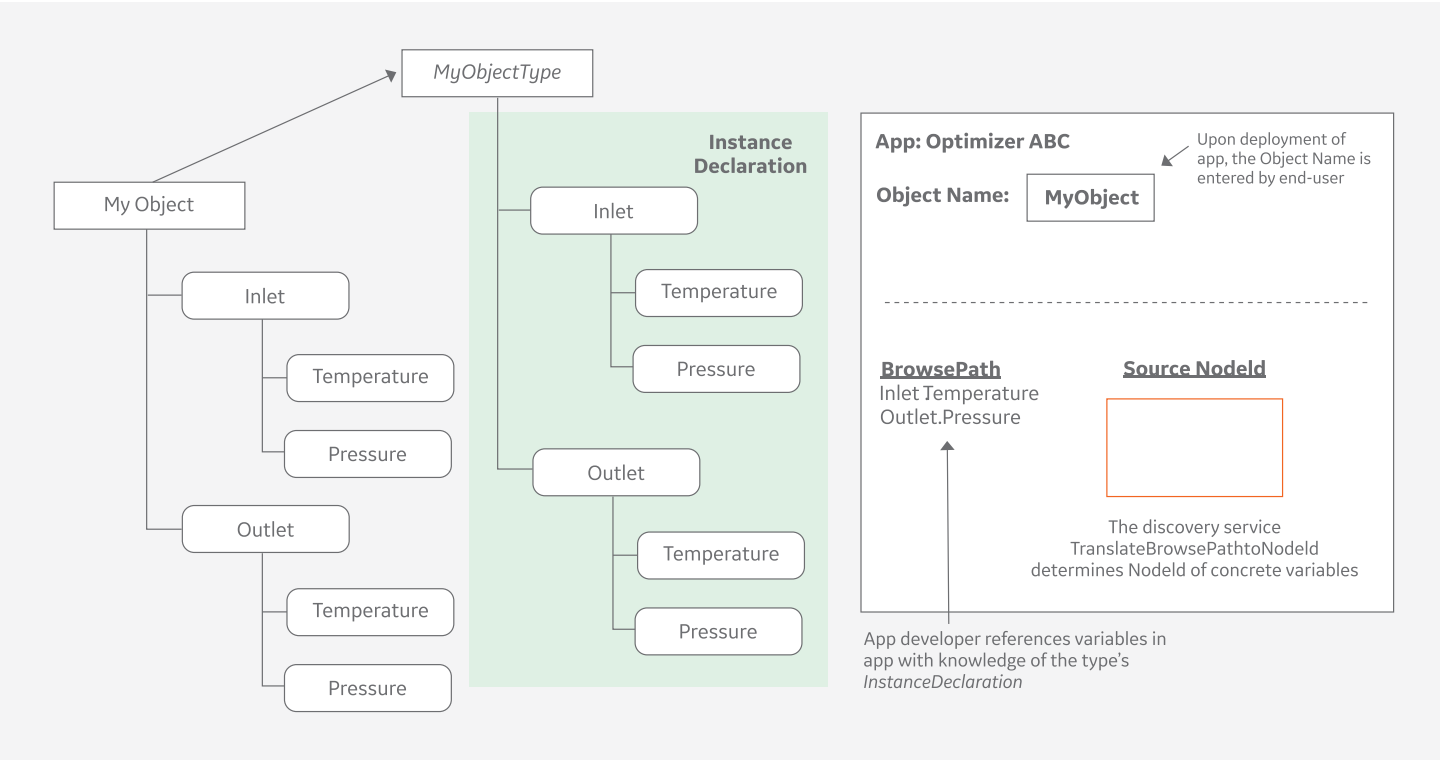

Figure 15 - Use InstanceDeclaration Knowledge to Help App Deployment

3.3 IICS Security using OPC UA

As described in Section 2.8, the establishment of a secure channel uses *Public Key Cryptography (PKC)*. This security protocol uses certificates to store public keys and other information needed for PKC operations. Managing certificates is of utmost importance, but this complex task requires an infrastructure. To address this, GE created an infrastructure based on Part 12 of the OPC UA specification that is referred to as the Global Discovery Server (GDS). The GDS facilitates secure communication on the Industrial Internet by:

- Providing a mechanism for clients to discover available OPC UA servers and their security configuration
- Managing and distributing certificates and trust lists for OPC UA applications
- Acting as a certificate authority for an OPC UA system

Figure 16 illustrates a system where an OPC UA client application and OPC UA server application use the GDS to enable secure communication between them. The components in the system are defined in Table 2.

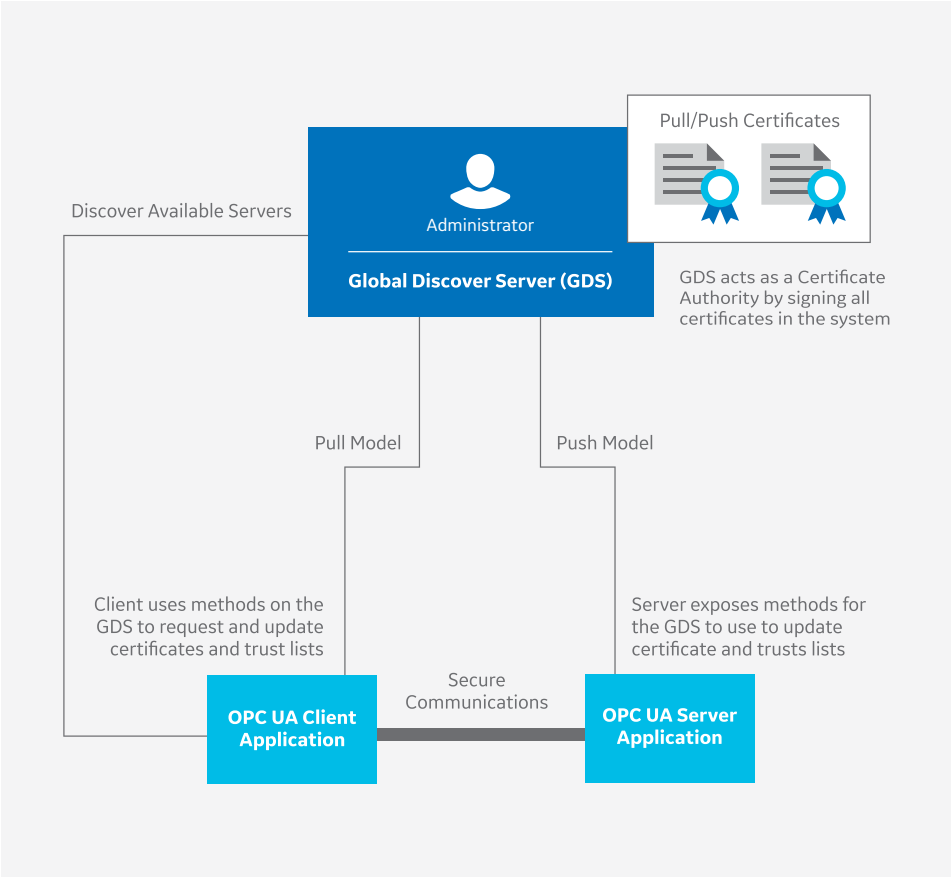

Figure 16 – OPC UA Applications Using GDS

Component	Definition
OPC UA Client Application	An application that consumes services and information from an OPC UA server (see Figure 1 and Figure 16)
OPC UA Server Application	An application that provides services and information to an OPC UA client (see Figure 2 and Figure 16)
Certificate Authority	A trusted entity that can issue certificates. The Global Discovery Server can act as the Certificate Authority for an OPC UA system.
Global Discovery Server	Global Discovery Server is used to manage certificates and discover server endpoints (see Figure 16).

Table 2 - GDS System Components

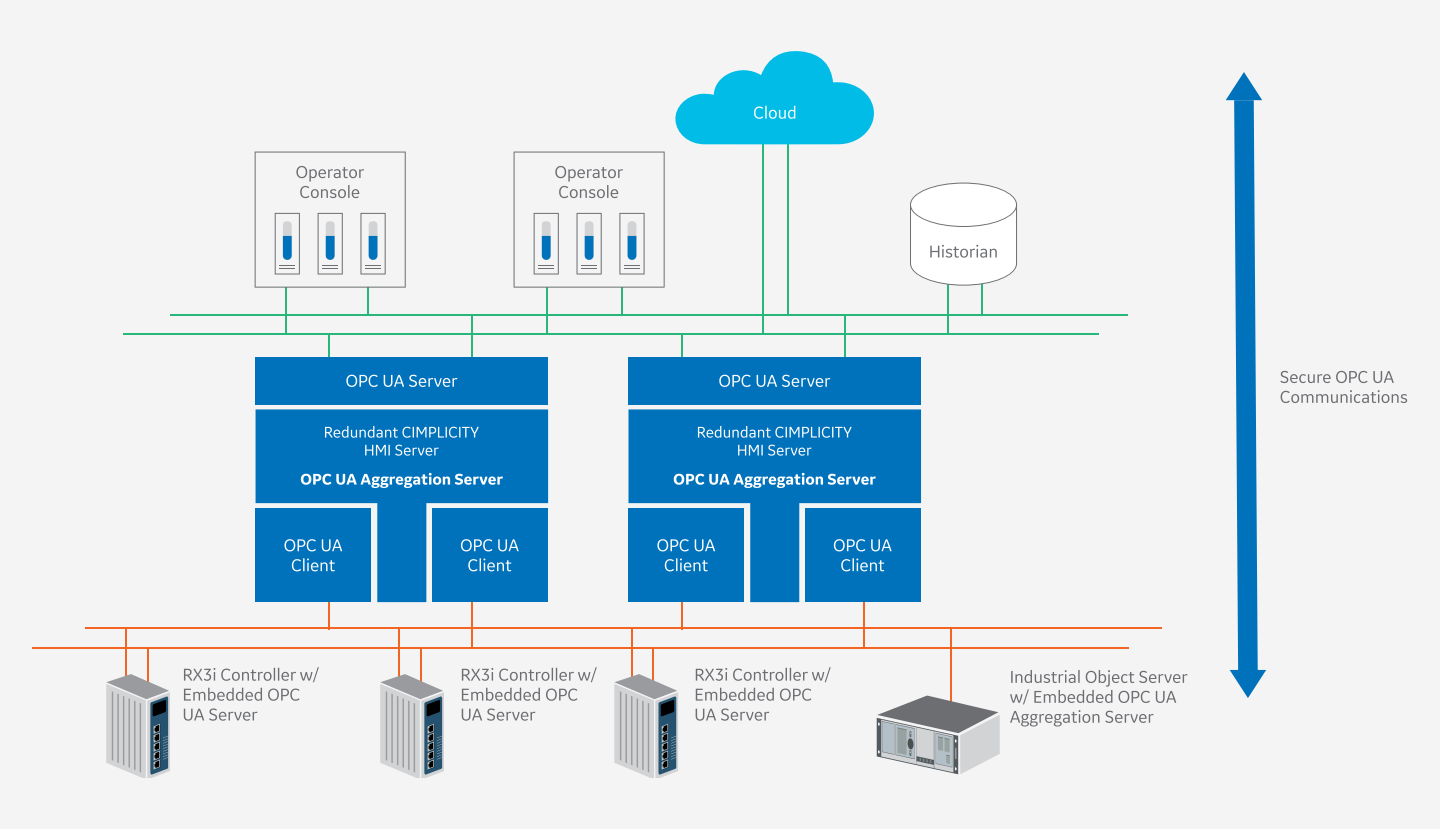

Figure 17 - OPC UA System Architecture

3.4 IICS Architecture with OPC UA

Figure 17 shows a system architecture based on RX3i and CIMPLICITY HMI/SCADA technology that uses OPC UA as its information backbone. Some differentiating features of this architecture are:

- RX3i hypervised platform makes it possible to run the inner and outer loop on the same box (see Section 3.2)
- RX3i hardware-based security including trusted boot, coupled with OPC UA secure messaging provides comprehensive security at multiple levels
- CIMPLICITY provides built-in OPC UA client and server redundancy to achieve high-availability

4 Conclusion

Keeping information secure is critical in the age of the Industrial Internet. OPC UA allows distributed applications to communicate securely by protecting the confidentiality and integrity of the communication. It also makes it possible for applications to authenticate each other using *Public Key Cryptography*.

The intelligence of a system is commensurate with its ability to derive meaning from its data. OPC UA information modeling makes it possible to add semantics to data so that its meaning can be automatically derived by applications. This results in self-composing client applications that are easy to develop and support.

Beyond security and information modeling, OPC UA features:

- Support for multiple platforms
- Discovery of information and resources
- High availability
- Extensibility (e.g. adding Pub/Sub)
- Scalability (e.g. embedded profile)

Together, these attributes of OPC UA make it the ideal communication backbone for GE's Industrial Internet Control System.

Imagination at work

02.18 GFT909