MODULE 2.1: FRAMEWORK

Framework

… something composed of parts fitted together and united
… a structural frame
… a basic structure

Objectives

- Explain what a universal framework is
- Describe how various smart devices can be seamlessly integrated
- Recognize how data & information collection can be automated

Framework benefits	1. Overcomes incompatibility of competing technologies and communication protocols (e.g., BACnet, Lon, etc.) 2. Connects devices together 3. Obtains real-time access to data and status that originates at the device level 4. Integrates data from multiple points-of-origin into a common graphical user interface (GUI) 5. Equips operators with the ability to monitor and/or control systems and equipment 6. Enables enterprise-wide and Internet access to equipment, systems and buildings 7. Automates trend data collection (histories), alarming and notification
Web browser	You can use the Niagara framework to monitor, manage and control (in real time) any building or system over the Internet using a web browser like Internet Explorer or Mozilla Firefox.

QUESTIONS FOR REVIEW:

1. *How does the Niagara framework overcome the problem of device incompatibility?*
2. *What are the benefits of having a universal framework like Niagara^{AX}?*
3. *TRUE or FALSE? Once having integrated a building's or system's diverse systems and devices, the power of the Niagara Framework allows you to monitor, manage and control them from a centralized monitoring station, or from anywhere in the world through a web browser.*

MODULE 2.2: NORMALIZATION

To cause to conform to a standard or norm

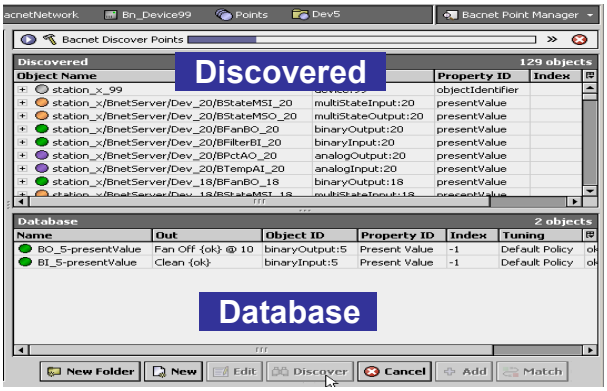
In the case of Niagara^{AX}, that standard is called BAJA.

Objectives

- Understand the concept of normalization.
- Describe the role of the Common Object Model in normalization.
- Identify the 8 Niagara object types used in the Common Object Model.
- Explain the overall process of discovering what is in a smart device and how those items become Niagara objects.

BAJA	*Building Automation Java Architecture* The core standard (open specification) to which the Niagara framework conforms
Common Object Model	The Niagara^{AX} Common Object Model makes it possible for diverse connected systems to talk to each other and to the enterprise. The Framework takes the data elements such as the following from the various devices: ▪ Inputs ▪ Outputs ▪ Setpoints ▪ Schedules ▪ Control parameters and processes these items into normalized software components. This conversion normalizes the attributes of the devices (both data and behavior), creating a database of objects that talk to and work coherently with each other in real time.
Niagara objects	The device drivers for a particular family of smart devices specify how individual data points map to a collection of **8 simple Niagara objects**. These data points can be any data from a remote object, such as: ▪ space temperature ▪ lighting status ▪ valve position ▪ pump speed The decision to establish a connection to one or more data points then becomes a matter of selecting which of these Niagara objects is best suited for the task. Those objects can then be configured and presented graphically to more easily monitor and control. These objects are **integral to the driver architecture** that is the foundation on which all device integration rests. They are either read-only or writable and are color-coded for ease of identification: ▪ **Boolean** - green ▪ **Numeric** - purple ▪ **Enumerated** - orange ▪ **String** - gray

Read-only	**READ-ONLY** objects have output only, and are used for **monitoring** only. They can be fed as inputs to writable objects.
Writable	**WRITABLE** objects represent data items that can be written (programmed) as well as read. - 16 different input properties correspond to priority levels such as *emergency* and *operator override*.
Boolean objects	Boolean objects represent a binary value with only 2 possible states, typically coded as a TRUE or FALSE condition. However, each condition can be shown in a way that is most meaningful for the application: ▪ **ON / OFF** ▪ **YES / NO** ▪ **OPEN / CLOSED** ▪ **OCCUPIED / UNOCCUPIED** *Boolean objects are color-coded GREEN.*
Numeric objects	Numeric objects represent an analog value such as a: ▪ Temperature ▪ Current ▪ Rate ▪ (or similar floating point number) *Numeric objects are color-coded PURPLE.*
Enumerated (Enum) objects	Enumerated objects (enum) represent multiple states (more than one) such as a multi-speed fan or pump (or varying count: integers → 1, 2, 3, etc.) ▪ **OFF / SLOW / FAST** *Enumerated objects are color-coded ORANGE.*
String objects	**String objects** represent one or more **ASCII characters**, often with literal meaning. **ASCII characters** *ASCII stands for **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange. ASCII is a code for representing English characters as numbers, with each letter assigned a number from 0 to 127.* For example, the ASCII code for uppercase M is 77. Most computers use ASCII codes to represent text, which makes it possible to transfer data from one computer to another. *String objects are color-coded GRAY.*

Discovery process	The process by which Niagara "learns" *which data points exist in any given smart device* Before data originating in a remote device, like a chill plant controller, can be used, the device and information from that device must be pulled into the Niagara software. **Discover** allows you to find devices and data items (points/components) that are defined using a driver's framework. Online "device learns" are possible using the **Device Manager** for many drivers. Data points are discovered using the **Points Manager**. Whenever available, this method is the easiest way to accurately add devices and device components to the station database. Most device/point learns in Niagara[AX] are a 2-step process where you first: 1. Discover device/point candidates to include in the station database. 2. Select and Add from those candidates, creating device components in the network. 
Workplace[AX]	WorkPlace[AX] (Tridium's branded version of the Workbench) provides an integrated development environment (IDE) for non-programmers to develop their own customized applications. The Workbench has features of a file explorer and a computer-aided design (CAD) application. It allows Niagara[AX] installation or maintenance professional to graphically review and edit the contents and behavior of a Niagara[AX] **station**, as well as the configuration of a Niagara **platform** -- the computer on which the station is running. At a high level, the main window (from left to right) of the Workbench is typically split into 2 sections: **1. SIDEBAR:** a. Navigation tree - displays the station as a tree structure b. Palettes - displayed under the Navigation tree; used to add additional functionality and logic to the station **2. VIEWPANE:** Displays additional information and details about whatever is selected in the Navigation tree, and can display a number of different views

 TOOLS	▪ ***Device Manager*** ▪ ***Points Manager*** *(unique to each network type)*
 JOB AIDS	▪ ***Discovery Process*** ▪ ***Niagara Objects*** ▪ ***Normalization***
 LAB	*You will have opportunity to practice discovering networks, devices and objects in the Niagara Technical Certification Program on-site training course.*

QUESTIONS FOR REVIEW:

1. *For the Common Object Model, normalization is the process by which a complex smart device is …?*
2. *Given an example, match it to the appropriate type of Niagara object.*
3. *TRUE or FALSE? Discovery is the process in which Niagara detects the objects in a smart device or smart devices in a network that can be graphically represented in a station.*