AFDtek (A Division of Fastek International Inc.)
245 Riviera Drive, Unit 1, Markham ON L3R 5J9, Canada
Tel: 905.474.0909 | Fax: 905.474.0809

Systems Integration Basics

Introduction

Systems integration is a process whereby a cohesive system is created from components that were not specifically designed to work together. Components of an integrated system are often systems in their own right. This paper discusses systems integration in terms related to the building management industry but many of the concepts and caveats are equally applicable to other industries.

A building management project frequently requires the integration of separate specialized systems, controllers, and infrastructure technology into a building management system. The requirement for integration can arise for a number of reasons. Some of those are:

- Another vendor's specialized controller must be added to a system. For example, add a boiler controller to the building automation system.
- Two buildings of a campus may have systems from different vendors. The owner requires the two systems to be managed from a "single seat." In other words, a single operator sitting in front of a single workstation must be able to operate all the systems.
- Several stand-alone building systems must be integrated into a single facility management system in order to enable automatic responses to events. For example, swiping a badge on the security system card reader must turn on lights and HVAC for an office floor.

The facility owner's expectation is for the integration to be cost-effective and the result to be a single cohesive system.

The information in this document is applicable to integrating a single field level controller to a system, integrating peer systems and integrating systems to a higher level management system or a carrier infrastructure such as an IT network or the Internet.

Use this document:

- To help understand issues that must be addressed when designing an integration project
- As an aid in gathering information for an integration project
- To help make decisions as to the most appropriate approach to use for integration

Key Concepts

ASCII

The American Standard Code for Information Interchange (ASCII) is a widely used data format for the representation of English and Western European text characters.

API

An Application Programming Interface (API) is a defined and documented software interface that a software program may use to interact with a system or a specific sub-system or feature of a system.

The OPC Data Access and OPC Alarms and Events APIs (defined by the OPC Foundation) are examples.

Integration

Integration is the process of inter-connecting one system with another system in order to provide a useful exchange of information, data and/or control between the systems.

Interface

An interface is a point where interaction occurs between two systems, devices, programs, etc. To be useful for integration, an interface must have stable and well-defined characteristics. A standard interface is one that is defined and documented by a recognized professional or industry organization. The RS-232 interface is an example of a standard hardware interface – it is currently managed by the Telecommunications Industry Association (TIA).

The term "interface" is also used to refer to a hardware or software component that translates or converts one interface into another interface.

Media

Media is the medium through which data is conveyed. Media has specific physical properties such as voltage levels, bandwidth and frequency response. A telephone line is a type of media characterized only by its physical properties.

More complex media may provide a choice of cabling or interconnections with different sets of physical properties. For example, standard available Ethernet media includes several specifications for twisted pair wire and fiber optics cables. The common element among these various types of physical media is the Media Access Control (MAC) protocol. A MAC protocol manages access to the media for outgoing messages and manages address recognition of received messages. A MAC protocol is an integral component of the implementation of sophisticated media such as Ethernet.

When integrating separate devices; determining and achieving a common media is one of the steps to a successful integration. For popular media such as Ethernet, products are available that convert from one type of physical media to another.

Middleware

Middleware is software used to "glue" together disparate systems without those systems needing to know anything about each other. Knowledge of both systems is contained in the middleware.

Protocol

A protocol is a set of rules governing the exchange or transmission of data electronically between devices. A protocol is like a language; both devices must understand the same language in order to exchange data.

A complex system has multiple levels of interactions. The different levels will likely use different protocols, each optimized for the nature of the interactions required at that level. For example, the network level might interact using Ethernet. The operating systems might interact using TCP/IP. The applications programs might interact using BACnet.

System

A system is a group of hardware and software components that work together in a cohesive manner to perform a task.

A system may be as simple as a single controller or as complex as the telephone network. A personal computer with a video card, memory, a disk drive and an operating system is also a system.

System Integration

System integration is a process whereby a cohesive system is created from components that were not specifically designed to work together. Components of an integrated system are often systems in their own right.

Scope of Data Exchange

Although the requirements for integration of two systems may sometimes be met simply by running their workstation software concurrently on the same computer, the reason for the majority of integration projects is to share data between two or more systems.

Successful data exchange requires the use of a common protocol – a protocol that both systems understand. The data being exchanged must be accessible on both systems via that protocol. Choosing an appropriate protocol first requires understanding the scope of the integration requirements. Once the scope is known, you are then able to evaluate the available protocols to determine whether they provide the best opportunity for integration.

The following, in order of increasing complexity of implementation, are examples of different scope of integration:

- Display or store text messages received from the integrated system
- Generate executive system events from text messages received from the integrated system
- Display point data values received from the integrated system
- Integrate points into the native format of the executive system for display and event monitoring
- Command or change values of integrated points
- Integrate alarm and event messaging into the executive system's alarm management feature
- Execute high level management functions from the executive system: examples include; system configuration, database generation and archiving

You should generally avoid implementing an integration of greater scope than required. If a custom solution will be required, this is particularly important. Making the integration "more complete" than is actually required may unnecessarily increase the complexity of the integration, is usually more expensive to implement and may not be fully testable.

The following points illustrate the protocol evaluation process:

- Determine whether any protocols of the systems being integrated support the level of data exchange required for integration – if you cannot get data in or out of a system, integration will not be possible
- Choose a common protocol that supports the level of data exchange required

- If there is no common protocol, look for an available gateway product that can convert from one supported data exchange capable protocol to another
- If a single protocol conversion is not possible, look for available gateway products that can convert both supported data exchange capable protocols to a common protocol
- If protocol conversion is not achievable with off-the-shelf products or cannot meet the requirements for data exchange, custom software will be required to integrate the systems

In some cases, it may not be cost effective or may not even be possible to provide the level of integration the requirements call for. If this is the case, it will be necessary to modify or reduce integration expectations.

Data Exchange Management

When two systems are integrated, one of the systems is generally "in control" of the data exchange. The system in control generally initiates the data exchange and will issue commands or other controlling messages to the system being controlled. Note that the system in control of the data exchange does not have to be the system that is the "master system" i.e. the system from which the integrated system is managed.

The available communications protocols may determine which system is in control of the data exchange. For example, if the communications protocol is a half-duplex polling protocol such as the Modbus protocol, one device must be the "master" of the protocol. The master is in control of the communications.

When systems are integrated with a peer-to-peer communications protocol such as BACnet, either system may initiate communications. This in effect means that control moves from one system to another dependent upon the activities of the systems. Through choices made regarding the design of the integration, the system designer may be able to choose which system is "in control". For example, if the application programs in the design have a client/server architecture, the client is generally "in control" of the data exchange. The designer may be able to choose where the clients and servers reside in the integrated system architecture.

It is important to establish where control will reside for each interface. A system will usually offer more than one interface that has the potential for being a point of integration. However, not all interfaces will have equal capability for control. The ability or degree to which an interface can control or be controlled may affect the choices of integration method and the level in the system architecture at which to accomplish integration. These choices can also affect the cost of integration.

Shared Media

Most systems have components distributed in separate physical "boxes" at varying distances from one another. Media is a necessary part of the system to interconnect the boxes. Traditionally, installation of a system has included the installation of dedicated media for the exclusive use of the system.

Today it is quite common to install a system using shared media. Shared media is used concurrently by often unrelated systems and is frequently owned or under the jurisdiction of a third party. Any system that uses shared media must be considered an integrated system, even if the systems using the media are not explicitly integrated with one another. Examples of shared media include:

- A LAN or WAN

- A broadband network
- The Telephone network
- The Internet
- Radio spectrum (wireless)

The first two examples are usually private media – i.e. they are under the ownership of the site the system is installed on. Note, however, that jurisdiction over the media often resides with a different person or department than those responsible for the system being installed.

When devices being interconnected are in different locations separated by public or a third-party's private property, it will usually be necessary to use a public shared media such as the telephone system or the Internet.

Wireless connections are a special type of shared media. What is shared is the space through which the signals travel and the frequency spectrum the wireless technology uses. A wireless connection is inherently less reliable than a wired connection; however, a wireless solution may be suitable for:

- Temporary connections
- Making connections to difficult-to-reach locations
- Hand-held or portable devices
- Line-of-sight access across public or private property

Possible wireless media include:

- Wi-Fi – local area network products based on the IEEE 802.11 specification
- Bluetooth – wireless technology managed by the Bluetooth SIG
- ZigBee – wireless technology managed by the ZigBee Alliance
- VHF or UHF radio
- Microwave links
- Optical links

Wi-Fi is an off-the-shelf technology – generally used for short range wireless connectivity to Ethernet networks. Bluetooth is an off-the-shelf technology generally used for direct short range connectivity between two devices – for example, between a computer and a printer.

Wi-Fi and Bluetooth are the only wireless technologies that would not require special engineering. Other wireless solutions will require custom engineering and installation from a service provider knowledgeable of the technology.

Platform Sharing

Platform sharing is the use of a single computer to concurrently run the application software of different systems. Platform sharing is generally limited to software running on general-purpose computers running a standard operating system such as Microsoft Windows. Platform sharing is seldom possible on controllers.

There are two major scenarios where an integrated system will have platform sharing.

Simple Platform Sharing

The simplest platform sharing scenario is where data exchange between different systems is not required but the owner does want to have "single seat" management of the systems. This will

require installation and concurrent operation of the workstation software of different systems on the same computer. This could be described as “simple” platform sharing.

In order to share a platform for this scenario, the following conditions must be met:

- All systems’ software must be compatible with the same operating system. For example; Windows XP Professional
- All systems’ software must be compatible with the same revisions of shared platform software technology or the technology supports simultaneous use of multiple versions. Shared technology examples include; Microsoft SQL Server, Microsoft .NET Framework, JAVA
- All systems’ software must be compatible with the same revisions of shared application software. Examples of application software that might be shared include; Microsoft Internet Explorer, Microsoft Office, ICONICS Genesis
- Platform hardware must meet the minimum requirements stated for the software that has the highest minimum hardware requirements. For concurrent operation, it may be necessary to increase the minimum hardware requirements beyond this level

Complex Platform Sharing

The other platform sharing scenario, “complex” platform sharing, is where software on the workstation platform is used to facilitate data exchange between the systems. This scenario, in addition to requiring the simple platform sharing conditions be met, also requires support for or the availability of APIs that give access to the features and data the integration requires.

API support may be divided into two broad categories, standard APIs and application APIs.

Standard APIs

Standard APIs are APIs provided by the platform specifically for the interchange of data between applications. The systems’ application software running on the workstation platform must be able to interact with a standard API in order to use it for integration. Standard APIs generally are platform specific. Examples of standard APIs found on Windows platforms include:

- Dynamic Data Exchange (DDE)
- ActiveX Controls (one system must provide ActiveX components, the other system must be capable of using them)
- ODBC (for database sharing)
- TCP/IP

Some applications that support standard APIs provide user interfaces for configuring and using the API. If a user interface is not provided, it will generally be necessary to implement programming in the application’s scripting language in order to use the API.

Application APIs

An application API is an API exposed by the application software itself. These APIs are application specific, generally proprietary and usually unique to the application. In some cases, commercial integration products may be available that use the APIs. Usually, however, a custom solution will be needed. The solution may be implemented with:

- A program written in a scripting language provided with one or more of the systems being integrated
- A commercial middleware application

- A custom application program written in a commercial programming language such as Java, Visual Basic, C++, C#, etc. designed to "glue" the systems together

Custom programming is a complex topic. Consider consulting with an integration-consulting service before specifying or implementing a custom programmed solution.

System Performance

Integration must always take into consideration the performance expectations for the integration and the performance available from the systems being integrated. The following specification requirements for integrated system performance should be known:

- The maximum acceptable end-to-end delay from the initiation of an event to when the event is reported to the user
- The maximum acceptable end-to-end delay from the initiation of an event to when a programmed response to the event is activated - programmed responses include; interlocks, energy management features and custom processes
- The number of data points that will be integrated
- The maximum acceptable refresh rate for displayed data

Every system has a finite ability to process data. When a system's performance limits are approached, its performance will likely degrade. If a system's performance limits are exceeded, the system behavior may become unpredictable.

The capabilities of the systems being integrated and the bandwidth of the integration media will determine the performance capability of the integrated system. The possible performance of the integrated system may be less than the performance called for in the project specification.

There are two types of performance limiting factors to be aware of. The first is the media's bandwidth. The media bandwidth limits:

- The aggregate rate of data exchange between devices
- The event response time of the system

The practical effect of a media bandwidth limit will be to limit the number of devices the media can support. If the media bandwidth is exceeded, the number of devices on the media must be reduced. Note that with shared media, traffic from other, unrelated, systems on the media will also contribute to the aggregate data rate on the media.

The second performance limitation is imposed by the system devices' capacity to process data. A device's capacity is exceeded when it receives, in a period of time, more data than it can process in that time. In this situation, the device's event response time will increase. The device's operation may even be disrupted and data may be lost. Processing overload can occur when:

- Two interconnected systems are mismatched in data processing capability
- A system receives data from multiple sources at an aggregate data rate that exceeds its ability to process the data
- A system receives data asynchronously – it cannot exercise control over when data is received

It is highly desirable to integrate systems in a way that allows the system receiving data to control the rate at which data is received. The following data exchange mechanisms, or a combination of them, can provide data flow control:

- Polling for data – a master/slave mechanism
- Request/response data flow – a peer-to-peer mechanism

- Buffering received data – used with asynchronous data transfers

If the integration allows a system to control the data flow into the system, the system's performance limits, and its response as it approaches the limits, can be known and managed.

The physical limits of systems indirectly define the performance limits of the systems. The onset of unacceptable performance degradation marks the performance limits. These limits generally occur some time before absolute physical limits are reached. The performance limits are difficult to quantify because they usually result from the sum of a number of variable factors. These factors include:

- The rate at which events occur in the system
- Available system memory
- Processor speed
- Media bandwidth
- The amount of data being exchanged
- The number and type of active features and processes (i.e. trending, demand limiting, control loops, etc.)

Manufacturers are generally unable to provide precise figures for performance limits. Instead, they will make recommendations for maximum point counts. There usually is an inherent assumption that the point count limit applies to a "typical" mix of point types for the market or application the system is designed for – for example, the HVAC market. Unfortunately, performance limits usually can only be determined by experience. Nevertheless, an attempt should be made to quantify the systems' limits; if for no other reason than to avoid attempting the impossible. The limits to know include:

- The bandwidth of the media used to interconnect the systems; this is generally expressed in megabits per second (Mb/s) but could be expressed in other forms such as bits per second (b/s) or megabytes per second (MB/s)
- The maximum amount of data each system or device can support
- The remaining available data capacity of the systems
- The maximum rate at which each system or device can accept data
- The maximum amount of data a system can accept in a single data packet

The reciprocal performance information required is the rate at which a system will generate data. This is also difficult to quantify. The data event rate is dependent upon the operational activity of the system. Again, experience is usually the best means of determining this information. The limits to know include:

- The maximum rate at which the systems will generate messages or events
- The maximum rate at which messages will be sent between systems

When system performance is degraded due to exceeding system limitations, it will be necessary to limit or reduce the scope of integration.

Data Format

Integration must take into consideration the format of the data being exchanged. Data interchange between systems is generally in one of two forms: textual messages or data messages.

Textual messages are usually produced in ASCII format. In the simplest integration, the receiving system will be required to display the unaltered text on a workstation, print it or record it in a

database. The receiving system does not have to understand the data – the "processing" of data is done by the system user when the message is viewed.

If integration requires the receiving system to act on data contained in a textual message, the receiving system requires knowledge of the message content. The system will have to parse the message text looking either for specific words or for words at specific places in the message. Parsing will likely require custom programming. The parsed data will be used to generate an event or update a point.

Data messages are data carried in a data protocol. If the systems being integrated support a common data protocol, for example BACnet, it may be possible to integrate without customization.

When the systems do not support a common protocol, a protocol conversion is required. Commercial products are available from various vendors for some conversions. When a commercial product is not available, a custom solution will be required to convert from one protocol to another.

Data Resolution

Integration must take into consideration the resolution of data being exchanged. This is primarily an issue for analog data and for time data.

Depending upon the design or capabilities of the system or device being integrated, the value of analog data may be available only in its raw form (for example, digital counts), or it may already be processed to the representation value for a specific point. (For example, the value of a temperature point may be provided directly as a °F temperature value.) If the analog value is in its raw form, it may be necessary to convert the value to another format before it can be used. If the analog data has a representative value, the precision of the value (typically indicated by whether a decimal place is included with the value) may have to be changed.

For analog data, the following need to be considered:

- The format of the analog data
- The precision of analog data that will be received from a device
- The precision of analog values that must be provided for analog data commands sent to a device
- The data display precision for analog data

For time data, there are two issues. The first issue is whether time data is supplied with or stored with data such as point value samples and events. This time data is known as a time stamp. The second issue is with the resolution of the time stamp.

Time stamps are rarely included with data. If data is required to be time stamped, the application must time stamp the data when it is sampled or received. Time stamp information that must be determined includes:

- Are data samples time stamped
- The data time stamp resolution (i.e. minutes, seconds, 10th seconds or 100th seconds)
- Are system messages time stamped
- The message time stamp resolution
- Are archived data time stamped
- The archived data time stamp resolution
- Will time stamped data be from different time zones

- Will time stamped data time be adjusted for day light savings time

Time stamp resolution is a factor in accurately comparing a sequence of events. A sequence of events happening within a few seconds cannot be accurately assessed if the resolution of the time stamps is one minute. Time stamp resolution is also an important compatibility factor for data samples from multiple sources stored in a common database. For example:

- The database schema must be able to handle the finest resolution of time supplied with the data
- All data samples must be stored with time stamps of the same resolution
- Database searches must be able to correctly return data based on time search keys and the data must be time sorted in correct order

When time data is stored in a database, it must be normalized to the time resolution the database supports and it must be normalized to Coordinated Universal Time (UTC).

Data Synchronization

Data synchronization is required when an integrated system has two (or more) copies of the same data. There will be the original data at the data source and a duplicate of it within the integrated system. Synchronization ensures the duplicate data is a true copy of the source data. The scope of integration and the availability of common data exchange protocols affects whether data duplication must be used in the integration design. As a general rule, data duplication should be avoided. Reasons include:

- Extra commissioning work – creating the duplicate data will mean extra work is being done
- Extra maintenance – changes in the original data set will require making complementary changes where the duplicate data resides
- Reduced system reliability due to the risk of having inaccurate or out of date duplicate data – data synchronization is difficult to maintain
- Reduced system reliability due to greater system complexity

Situations where the integration design will generally have to maintain duplicate copies of data in an integrated system include:

- The integration cannot use rules to transform or map data from the format used in the source system to a different format in the integrated system
- The integrated system is required to compute new data or data points from the data received from the source system
- The integrated system must be able to display data from the source system when that system is not connected

Duplicate data may exist as data points in the integrated system database or in databases associated with integration hardware or software located with either the integrated system or the source system.

There are two principal ways that duplicate data may be kept in synchronization with the source data; by periodically polling all the data of the source or by the source reporting changes in data (exception reporting). Each method has both strengths and weaknesses.

Updates by Polling

The advantages of updating duplicate data by periodically polling all source data include:

- Duplicate data will automatically remain synchronized

Updating data by polling all the data has a number of disadvantages:

- When duplicating a large amount of data, the refresh time for the data will be long – the overall performance is constrained by the communications bandwidth
- The integrated system does not receive updates in real time
- The integrated system must evaluate all data for value changes, alarm conditions and other reporting criteria – this processing burden may affect the performance of the integrated system
- Synchronization within the data may be lost due to the time required to transfer the data – events will occur during the time of the transfer that may affect data that has already been transferred
- The possibility of loss of synchronization within the data may negate the advantage of automatic synchronization between the integrated system and the source data

Updates by Exception

Updating by exception means that the integrated system receives only data that has changed. Exception data may be periodically requested by the integrated system or may be asynchronously "pushed" from the source system. Exception data asynchronously "pushed" from the source has the advantage of being delivered in real-time. The advantages of sharing data by exception include:

- It may be possible to have real-time data updates
- The integrated system requires less processing bandwidth
- Less communications bandwidth is required between the integrated system and the data source

Updating data by exception has some disadvantages. These include:

- When communications is established between the systems, a data synchronization process must be executed that will ensure the complete set of duplicate data matches the source data
- The time required to synchronize the complete set of duplicate data may be lengthy
- The synchronization process must be repeated each time communications between the systems is interrupted and then restored
- There may be conditions under which an update is "lost", i.e. it is not received by the integrated system, and goes undetected – this will cause the integrated system to be out of synchronization with the source data

Conclusion

Systems integration is a complex topic with many factors to consider. Execution of a successful integration will require knowledge and understanding of the customer's needs weighed against a careful balance of compromises.