Communicating with Legacy Devices in a Connected World

A key objective of the Internet of Things (IoT) is to connect various devices and sensors deployed in industrial, energy grid, building, and many other environments to the Internet and cloud. These components communicate to the outside world using a wide variety of protocols (e.g., MQTT and Modbus) and APIs, which IoT application developers must implement as well on Intel gateways. Writing and testing this code can take considerable time and effort, but streamlining this task is the development framework offered by Solution Family*. The Solution Family team has worked to provide a tool-based IoT development platform designed to help developers create intelligent devices at the edge, with or without the need for a central service.

The Solution Family development framework enables rapid, extensible, and secure IoT development in minutes while preserving existing investments and reducing development costs by 80 to 90 percent. In a relatively short time, even workers on the factory floor can use the tool to retrieve data from its source, implement data rules, and send the processed data to a cloud service. The tool’s simple user interface is very easy to use and makes operational data available to non-technical personnel. Solution Family designed this application platform to ease the development of IoT applications, the optimization of operational environments, and the creation of value-added services without having to write a single line of code.

Intel® Gateway Solutions for the Internet of Things (Intel® Gateway Solutions for the IoT) is designed to provide Internet connectivity for the 85 percent of all legacy and existing devices1 that today are isolated and unconnected. The gateway aggregates, filters, processes, and secures sensor network data before sending it to the cloud or an enterprise. It also integrates technologies and protocols for networking, embedded control, enterprise-grade security, and manageability.

This solution brief describes how to deploy the Solution Family development framework on Intel gateways.
Beyond Traditional SDKs

The Solution Family development framework delivers many clear advantages over traditional software development kits (SDKs), with up to a 90 percent savings in initial development and future upgrade costs. It provides developers with a native data model, graphically-defined rules and aggregation, drag-and-drop data to a cloud or storage service, and graphically-created visualization. In comparison, traditional SDKs typically require developers to learn the APIs, create the application source code, and build the application and tests, and these efforts incur high costs that can rarely be leveraged between projects. With respect to maintenance, the Solution Family development framework has tools to upgrade the product and user applications over time, which developers must do themselves with traditional SDKs.

Solution Family* Architecture Features

The Solution Family offering consists of an application agent, called Solution Engine*, that serves as an implementation mechanism, and a set of application development tools, called Solution Builder* and Dashboard Builder*, used for end application-focused design and development. The relationships between devices and dashboards can be seen in Figure 1.

![Figure 1. Solution Family* Components](image)

The Solution Family IoT Approach, illustrated in Figure 2, drives the user through five steps in creating their IoT application. The user starts by getting data from appliances through protocol adapters. Next, this data is analyzed through analytics and aggregation. Data is then sent to and from the cloud. This new knowledge is then used to control the application. Finally, the user can access the on any device. The whole IoT application is developed entirely through tools with Solution Builder and Dashboard Builder.
As illustrated in Figure 3, Solution Builder sends rules, statistics, temperature, data collectors, etc. to the appropriate modules and applications in Solution Engine via REST APIs. All operational data collected from the adapters, analytics, and rules is transmitted from the Solution Engine to Solution Builder or other applications via a separate REST interface.
Solution Engine, shown in Figure 3, is a cross-platform implementation that can execute on any platform meeting the Common Language Infrastructure (CLI), such as C# or VB.NET. It has three communication channels:

- A standards-compliant REST interface enables any client to retrieve data. In addition, an open source Client API is available to aid the development of client applications.

- Clients can communicate via a "virtual adapter", allowing an application to publish data through the Solution Engine. Virtual adapter source code is delivered with the Solution Engine code base and provides a starting point for anyone developing custom models or hardware interfaces to extend their solution.

- Solution Builder provides the core development tool that developers use to create rules, run analytics, record data, and analyze their networked Solution Engines without writing code.

The Solution Family product architecture includes adapters for interfacing to specific standards-based hardware or models. Adapters are dynamically loaded and can be pushed to the Solution Engine in real-time, providing the simplest way to expose data to the operator working with the Solution Builder tool in data driven environments. The Solution Family product provides templates for creating adapters, modules, analytics providers, user tool views, and database/cloud storage providers to extend the capabilities of the Solution Family architecture in all dimensions as needed by the application.

Building a Solution
The following provides a step-by-step guide for creating a simple deployment with the Solution Family product running on a Linux*-based Intel gateway, like Intel Gateway Solutions for the IoT. This section assumes the gateway has a network connection to the Internet, can resolve public IP addresses, and is able to use FTP to retrieve files from a remote server.

The Solution Family Team created a guide to the installation Solution Engine and, if necessary, Mono on a Linux* OS image. Installation details and common pitfalls are provided in the Installing Solution Family Products (Linux Ed.).pdf, downloadable at ftp://ftp.solution-family.com/Docs/.

Setting up Simulated Modbus via Solution Builder
Solution Builder has a Windows*- based GUI interface that connects to Solution Engine running on the target Intel gateway platform. The following step connects to a PLC represented by a Modbus TCP Simulator and Visualize the Memory Registers within the PLC.

Step 1: Install Modbus simulator
Modbus simulator can be downloaded from ftp://realtimeonline.com/Release/Modbus%20Sim/, and it should be installed along with Solution Builder.

Step 2: Setup Software Modbus on Intel Gateways
Follow the instructions in the HOL-204-Using-Modbus.pdf to setup the emulated TCP/IP software Modbus through Solution Builder.
End-to-End Data Flow

Step 1: Setup Data Collector on Intel Gateway
Follow the instructions on HOL-103-Introduction-to-Data-Collection.pdf to setup cloud services.

Step 2: Setup Cloud & Data Viewer
Follow the instructions on HOL-121-Pushing-Data-to-the-DreamFactory-Cloud.pdf to setup the cloud infrastructure to enable data viewing through the cloud.


Further hands-on-labs, installation guides, articles, and presentations that walk developers through creation of entire IoT applications are available at http://blog.solution-family.com.