CalQLogic Inc.

Los Angeles, CA

IoT TECHNOLOGY SPECIAL

CIOReview

The Navigator for Enterprise Solutions

Key Concepts to Support Creation of Powerful IoT Apps

Key Concepts to Support Creation of Powerful IoT Apps

By Dr. Donald Cohen, Senior Scientist and
Dr. K. Narayanaswamy, CTO & Co founder, Cs3 Inc.

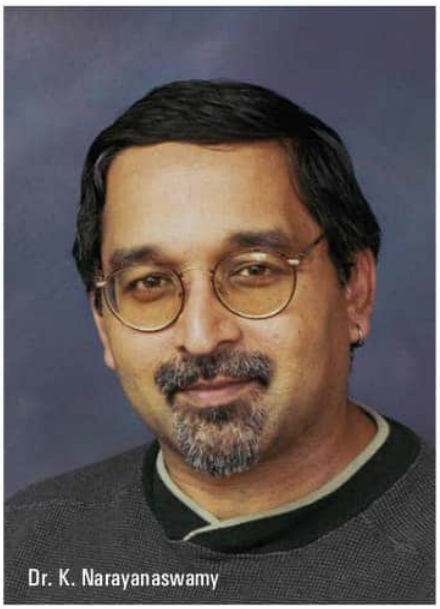
We have considerable experience in integrating enterprise tools that were not designed a-prior to interoperate. Many insights we learned in the enterprise software integration domain are applicable as key concepts to support the development of today's "machine-to-machine" IoT applications. Listed below are the key concepts to support creation of powerful IoT applications.

IoT Apps Need Uniform Data Representation and a Query Language

To motivate the discussion, consider the development of a contemporary Home Automation Controller (HAC) that controls many home appliances made by different vendors and interacts with services such as power, utilities and security monitoring provided by outside providers. The HAC needs to retrieve data from the devices and services with which it interacts. Ideally, each device would provide a means to retrieve its data, for example: thermostat setting, external temperature, power consumption and others factors. The HAC would be easier to build if all the devices support a uniform high-level data representation and a standard access protocol such as HTTP to retrieve the data. Otherwise, one has to build adapters to retrieve the data from different devices. A uniform data representation greatly helps humans who need to work with the HAC as well. Of the commonly used standards, we generally prefer relational representations for device data (familiar from logic, logic programming, and relational databases). Our assessment is that relational representation offer superior composability through the power of logic as a query language, compared to, say, XML/JSON data representation, and its more navigational style of data retrieval.

IoT Apps are Innately Event-Driven

A HAC needs to sense the data from the devices in the environment, and react to the


Dr. K. Narayanaswamy

perceived state or state change by using the controls available to the HAC, for example by changing the thermostat setting for the home or sending the owner an alarm. We use the term event to denote any state change. Events are different from ordinary data in that each event is associated with a time or the time the change that took place (or was observed). We define applications that need to react to state changes as event-driven. Exactly how the HAC should react to specific events needs to be customizable by the user (home owner). In addition, user initiated actions (e.g., setting the thermostat to 69F) can be treated as just another kind of event that the HAC must handle.

Objects in the IoT world should ideally notify other IoT objects about changes in their state. Rather than waste resources by sending all notifications to all objects (most of whom would ignore most notifications), it is common to use a protocol like "publish/subscribe", whereby event notifications are sent only to objects with expressed interest in that event. This is much more efficient than the alternative of polling the devices. Notifications take the same relational form as the data, enabling the recipient to examine it and decide how to respond.

IoT Apps Benefit from Sophisticated Event Management

Along with other standard services, an IoT operating system should support a common event management substrate (EMS) in order to support device integration. It is not that exotic for operating systems to provide a pervasive integration mechanism; tool builders have counted on "pipes" in UNIX as a universal integration mechanism for decades. Analogically, IoT applications are easier to create using publish/subscribe with a powerful EMS that supports logical queries over relational expressions and events, and incorporates a rich set of event types:

- **Primitive Events** represent changes in the real state of the world. So, in the HAC example, refrigerator or ambient temperature changes or a signal from the power company or a break-in alarm from a window sensor are all examples of primitive events.

- **Composite Events** are defined using a logic-based query language. We have found it advantageous for developers to use standard logical connectives and quantifiers of logic to define composite events. Two other important kinds of composite events prove to be very useful: 1) Various forms of historical reference (which require the substrate to store past events). This is useful in specifying composite events such as: Alarm Override PIN is not correctly entered within 2 minutes of the door opening event 2) State changes specified in 2-state logic can be the basis for events. Example: A refrigerator's thermostat in Fahrenheit setting is decreased by more than 10 percent -- this requires comparing the previous temperature value to the new temperature value.

An EMS with powerful composite event support simplifies IoT app development by obviating the need to write code to detect these events.

IoT App Development Is Easier with Event-Triggered Rules

Many IoT applications are natural to specify as a set of event-triggered rules. It is, therefore, advantageous for the EMS to directly support this

> "Uniform data representation and advanced event support greatly facilitate rapid IoT app development"


Dr. Donald Cohen

construct. Each rule has a trigger, the event that activates it, and a response, the code that is automatically invoked when the rule is activated.

rule < rule-name > **trigger** event-name (parameter-1, parameter-2, …, parameter-n)

response < response code that uses parameter-1 through parameter-n >

Note that the rule's trigger event ("event-name") can be a primitive or composite event, making the construct very powerful. Rules are modular and can be added, modified, or removed to change the IoT application logic. Rules enable rapid development of all IoT apps, and enhance the adaptability of apps in accommodating changing requirements.CR