An Application of Automata Cloud Processing

By: Jason Walker

Advisors
Official: Dr. George Markowsky
Unofficial: Zach Hutchinson Ph.D student

A capstone project proposal for partial fulfillment for the degree of
Bachelors of Science in Computer Science
University of Maine Orono
September 2016

Description

I am developing a cloud based automata processing unit. In the most basic sense the system will take in automata depicted using an XML document and a desired state. The system will then output a series of actions or transitions needed to be in that state. At the core of my project will be the automata processor. This system will be integrated with a RESTful service and web socket server that will allow quick communication between the client and the server.

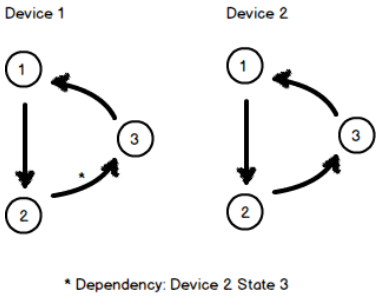
Background

My interest in the Internet of Things actually started during my first semester here at the University of Maine. I was enrolled in an introductory New Media course with Mike Scott. He was introducing the idea of disruptive technology and IoT came up in that lecture. From there I was captivated. How could the internet communicate with the everyday objects? I knew it was currently happening because there were systems such as the Nest thermostat. Phillips and Honeywell also had their line of smart devices. I saw one fatal flaw in the whole field. The flaw was that each manufacturer was creating their own dedicated system to run their devices. This meant that each set of devices was going to require their own app on your phone or tablet. This idea seemed like an obvious flaw to me. From there I started dreaming of a framework were this was not required. Why not let someone handle the overarching communication layer and hardware manufacturers use the system? This concept is the main driver behind this project. My goal is to allow individuals to create devices both simple and complex that the system can easily handle.

Objectives

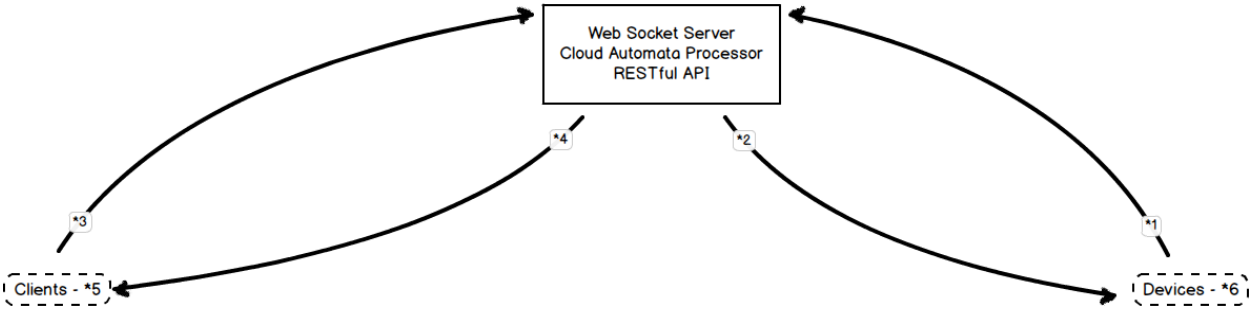
Stated previously the goal is to build a system that can handle complex automaton. Most machines (lightbulbs, thermostats, traffic lights, etc.) can be modeled by graphing the automaton. The graph can then be manipulated using various graphing algorithms. My main algorithm of choice will be Dijkstra's algorithm to find the shortest path from one state to another. In this implementation the graphs will be directed and weighted (all weighted one). I will also be implementing grouping and dependencies. Meaning that devices can be group to do a user defined set of actions (all lights in this room turn on). Dependencies are when one state requires another state in another automaton. The implementation will traverse multiple graphs when there are dependencies.

Example:



This diagram represents two automata graphs. The current state of Device 1 is state 1. The current state of Device 2 is state 1 also. Let the transition between each state be fn12, fn23 and fn31. The transition on Device 1 has a dependency that Device 2 need to be in state 3. When we run our algorithm to find the needed actions to take we will end up with: `[D1-fn12, D2-fn12, D2-fn23, D1-23]`. This queue represents the actions needed to be taken by the devices to be in the desired state.

The cloud automata processor only requires an XML document describing the automata. The device itself will receive a command to execute a function described in the XML document.



Basic Architecture Diagram

The diagram above describes the basic architecture as I currently have it planned. The client (*5) can be a desktop web application or a mobile device. This is what triggers the desired state. When you tap a button on the GUI that is meant to turn the device on or off it sends a request (*3) to the servers. The servers are integrated and connected to the database. The database holds the automata graphs and the current state of the machine. The automata processor will then run the algorithm to find the actions needed and store them in a queue. The web socket server (*2) will send the commands to the devices (*6) to handle one by one until the queue is empty.

You can expect the final project implementation to include a web GUI for communicating with the servers. A demo of the project would include adding new devices (pre wired) to the service and being able to control it from the web GUI. There would be a demonstration of the grouping and dependency functions. My personal growth objectives would be to learn more about advanced automata. I would be happy to present this project to potential employers so that I may start a career in the field of IoT.

Methods

I will primarily be using Node.js to build the server side functionality. Node.js can be used to scale. It has less overhead that Apache. A process manager like PM2 can be used to help Node.js scale as needed. It can also be used to scale to multiple cores as well. Being a front end developer my experience with JavaScript is fairly good. Node.js is JavaScript therefore I am able to apply my knowledge. The client side web application will be mostly JavaScript as well by using libraries such as Backbone.js. I will also use Bower for package management.

In order to have a fully functioning graph, I will need to implement some basic data structures such as the Graph, Vertex, and Edge. I will also implement a Queue and other various needed data structures.