# An Extended SNMP Based IoT Context-Aware Model for Dynamic Adaptability of Embedded Systems Software

Camille Jaggernauth
Novax Industries Corp.
l202-1525 Cliveden Ave.
Delta, B.C,
Canada V3M 6L2
camille.jaggernauth@novax.com

Douglas Gubbe
Novax Industries Corp.
202-1525 Cliveden Ave.
Delta, B.C,
Canada V3M 6L2

***Abstract*—This paper introduces a context-aware IoT (Internet of Things) model featuring context collector, context controller and cognitive engine components. This model supplies internet connectivity to previously published research, forming a complete, distributed, system solution for dynamic adaptability of embedded systems software for resource constrained hardware. We present the updated system model and proof of concept results for a traffic control system using the SCOOT algorithm (Split Cycle and Offset Optimization Technique).**

***Keywords*—IoT; SNMP; context-aware; dynamic adaptability; embedded systems software**

## I. Introduction

In our previous research we introduced a dynamically adaptable context-aware architecture which improves the application software's flexibility and responsiveness according different user requirements, varying operational conditions and easy update of embedded system software. This architecture was designed for resource constrained, low footprint (code-space), single processor, potentially energy-aware, solutions e.g. wireless sensor networks, mature systems, cost-aware solutions and legacy implementations. Context is defined as changeable and characterizing information such as sensor data [1],[2],[3].

In this work we add an IoT (Internet of Things) component based on extended SNMP (Simple Network Management Protocol) and the original context-aware model [5]. The combined model features the original ITS (Intelligent Transportation System) component and a new IoT component (Figure 1). ITS systems provide innovative services for better traffic and transportation management. An ITS system may comprise audible pedestrian systems, traffic controller hardware, remote SCOOT in-stations and out-stations and central management software. SCOOT is an algorithm that optimizes traffic control operation by using traffic queue occupancy and local conditions including error conditions to modify traffic signal controller hardware signal operation (red/green/amber lights operation, walk, don't walk, flashing don't walk audible pedestrian states) [4].

The innovation in the IoT model is its enabling of internet connectivity of the resourced constrained hardware that it is serially tethered to. The IoT model's design as part of a complete system solution ensures the overall architecture's code uniformity, readability, modularity, extensibility and maintainability. The design shares the advantages of the ITS model in that it is low latency (optimized operation), low code complexity (as per McCabe's complexity metric) and can share the same configuration files and utilities for configuring the cognitive engine (reducing development costs).

## II. IoT Context-Aware Model and Sytem Setup

The IoT model is shown in Figure 1. The context-collectors are sensor/system inputs. In this paper the context collectors are the green light data, error info and scoot data as shown in Table 1. There is also local context data from the ITS hardware BAC (Button Audio Conflict) and BS (Button Stuck) as shown in Table 3.The context-controllers are control sequences and are the Force and Demand data shown in Table 2 and local ITS controllers vibe, led and logging as shown in Table 3. The cognitive engine is responsible for enabling dynamic adaptability using Fuzzy Cognitive Map logic – by using phi/delta operators to assign user configurable weights to the different combination of collectors to determine a final result e.g. error flag [1]. In Table 3 the user selects what values of BAC and BS determine which particular output. This logic is coded into a logic map which is interpreted and executed by the cognitive engine. In the IoT model the context-collector is integrated with the cognitive engine which also performs buffering and averaging of SCOOT samples.

SNMP was chosen because the oid (object identifier) components of its mibs (management information database) map directly to our existing model's context element. The traffic controller mib in this paper is a public mib. SNMP's extensibility allows for the seamless implementation of the collector and controller software functionality. The cognitive engine is a stand-alone linux based program.

SNMP comprises a feature-rich set including an snmpd agent daemon (used by context controllers/collectors), trapd/inform daemon (used by context collectors), get/set command line applications (used by context controllers) . The SNMP snmpd agent can be extended in numerous ways. In this paper we used a) recompiling the agent to include context mib modules and b) using bash scripts in the pass mode. The ITS hardware platform is a MCU 8-bit, 128 KB Flash embedded processor. The IoT micro-controller hardware platform is a 75MHz ARM processor running Linux w/ 8 MB Flash, 16 MB SDRAM.

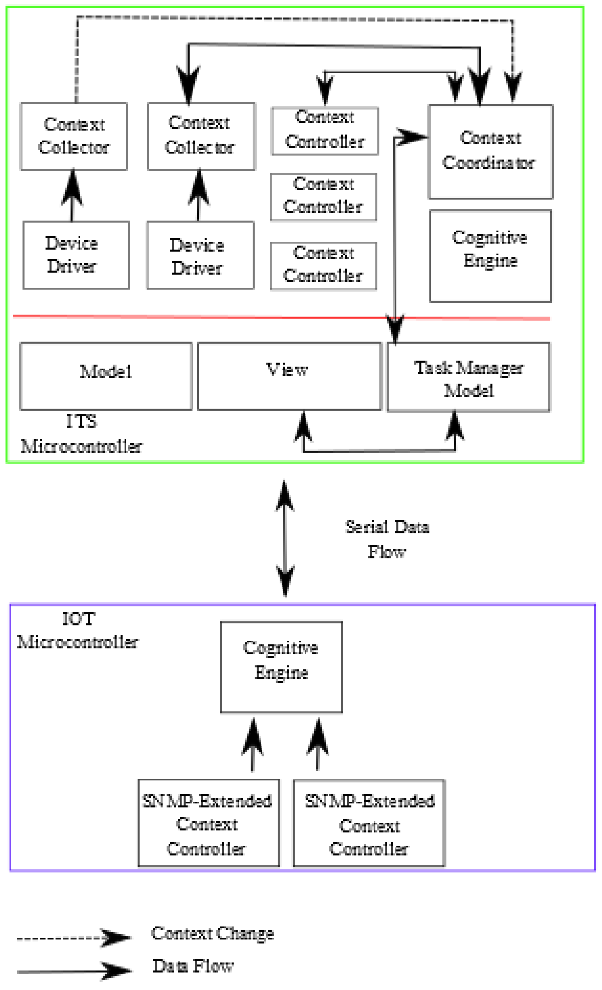

Figure 1. System Architecture.

## III Firmware Architecture Design

Table 1 and 2 show the message flow between the ITS/IoT micro-controller (serial) (column 2) and the remote SCOOT system (SNMP) (column 3). The local traffic controller hardware supplies Green Light data, error conditions and SCOOT detector occupancy and the remote SCOOT system responds with Force and Demand data for the traffic controller hardware. Figure 2 shows the pseudo code for the context controllers. Additional (not shown) context-controllers include SCOOT reporting intervals, port ids, ip addresses. Figure 3 shows the pseudo code for the cognitive engine. Table 3 shows how local error conditions can be combined to provide different SCOOT errors (setting different priority errors) as well as local ITS responses (actuating vibe, led, logging). Table 3 is usually coded into a logic map which is downloaded to both the ITS and IOT hardware and interpreted by their respective cognitive engines.

TABΛE I. Request Message

| Field | Serial Data | SNMP OID |
|---|---|---|
| Byte 1 | | |
| Bits 1-4 | Address | 1.3.6.1.4.1.13267.3.2.2.2 |
| | | |
| Byte 2 | | |
| Bits 1-8 | Force Stages 1-8 | 1.3.6.1.4.1.13267.3.2.4.2.1.5 |
| | | |
| Byte 3 | | |
| Bits 1-4 | Demand Stages 1-4 | 1.3.6.1.4.1.13267.3.2.4.2.1.4 |

TABΛE II. Response Message

| Field | Serial Data | SNMP OID |
|---|---|---|
| Byte 1 | | |
| Bits 1-4 | Address | 1.3.6.1.4.1.13267.3.2.2.2 |
| | | |
| Byte 2 | | |
| Bits 1-8 | Green Stages 1-8 | 1.3.6.1.4.1.13267.3.2.5.1.1.3 |
| | | |
| Byte 3 | | |
| Bits 8 | Critical Error Condition | 1.3.6.1.4.1.13267.3.2.5.1.1.29 |
| Bits 7 | Medium Error Condition | 1.3.6.1.4.1.13267.3.2.5.1.1.31 |
| Bits 6 | Low Error Condition | 1.3.6.1.4.1.13267.3.2.5.1.1.33 |
| | | |
| Bytes 4-7 | | |
| Bits 1-8 | Scoot Detector Data N | 1.3.6.1.4.1.13267.3.2.5.1.1.32 |

```
Based on the oid received
    Store the value in a oid specific file
    Value is not persistent
    Exit
```

Figure 2. Pseudo code for IoT model context controller.

```
Based on the type of input argument
    Either
        Report “still alive” using snmp inform
    Or
        Fetch the logic map
        Parse and interpret logic map
        Evaluate map for setting error conditions
        Update snmp inform args with new values
        If ITS values have changed
            Update snmp inform args with new values
Send snmp inform message if there are new args
```

Figure 3. Pseudo code for IoT model cognitive engine.

TABAE III. COGNITIVE MAP LOGIC

| BAC | BS | Action |
|---|---|---|
| 1 | 1 | Turn off led<br>SNMP Critical Error |
| 0 | 1 | Turn off vibe<br>SNMP Medium Error |
| 1 | 0 | Log error condition<br>SNMP Low Error |

## IV RESULTS

Figure 4 and 5 show the operation of the context collectors' and context controllers' message flow. Figure 4 shows the wire-shark capture of snmp set message from the the remote SCOOT server to set the local traffic controller's force bits.

```
104 71.801000 a.b.c.d w.x.y.z SNMP 177
set-request 1.3.6.1.4.1.13267.3.2.4.2.1.5
```

Figure 4. SNMP context-controller messages

Figure 5 shows the wire-shark capture of the context-collector messages sent using snmp inform to update the remote SCOOT server. Specifically a low error condition is reported as well as new SCOOT data.

## V CONCLUSION

We have demonstrated a context-aware IOT model suitable for adding internet connectivity for resource hardware for an ITS industry scenario. The use of the cognitive engine in the IoT model allows for the adding of user configurable local context to the SCOOT algorithm. Future work includes research into augmenting the system model's security.

```
Object Name: 1.3.6.1.2.1.1.3.0
Value (Timeticks): 93761
Object Name: 1.3.6.1.6.3.1.1.4.1.0
Value (OID): 1.3.6.1.4.1.13267.3.2.6.1
ObjectName: 1.3.6.1.4.1.13267.3.2.5.1.1.3
Value (OctetString): 00
ObjectName: 1.3.6.1.4.1.13267.3.2.5.1.1.33
Value (Integer32): 0
ObjectName: 1.3.6.1.4.1.13267.3.2.5.1.1.32
Value (OctetString): 0100030005010002
```

Figure 5. SNMP context-controller messages

## ACKNOWLEDGMENT

The authors wish to thank Novax Industries Corporation for the use of their ITS (Intelligent Transportation System) platform for proof of concept implementation.

## REFERENCES

[1] C. Jaggernauth, B. Kaminska and D. Gubbe, “A Context-Aware Model for Dynamic Adaptability of Software for Embedded Systems,” International Journal of Computer (IJC), vol. 19, mo 1, pp. 91-113, November 2015.

[2] A.K. Dey, G.D. Abowd and D. Salber, “A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications.” Hum.-Comput. Interact, vol. 16-2, pp 97-166, 2001.

[3] P. Inverardi and M. Tivoli, “The Future of Software: Adaptation and Dependability.” ISSSE 2006-2008, LNCS 5413, Springer-Verlag Berlin, Heidelberg, pp 1-31, 2009.

[4] SCOOT. Scoot Systems. Internet: http://www.scoot-utc.com/, [Oct 27, 2016]..

[5] SNMP. Net-SNMP. Internet: http://www.net-snmp.org/, [Oct 27, 2016].

## BIOGRAPHIES

**Camille Jaggernauth** (PhD) is the Senior Software Engineer at Novax Industries Corp.

**Douglas Gubbe** (DiplT) is the CTO at Novax Industries Corp.