Integration of reasoning in Web of Things framework

M2, 30 ECTS, July 2021

Aditya Das

Internship Advisor: Dr. Kamal Singh
Academic Advisor: Prof. Pierre Maret

0.1 Acknowledgement

I would like to greatly thank my internship advisor, Dr. Kamal Singh for helping me continuously at every step of the way throughout the internship. As a novice in scientific research, I thank him for his generous time in helping me overcome the challenges and difficulties that were encountered while pursuing and understanding the topic of the internship and the methodology to start and continue this project. I would also like to thank him for taking the time in helping me write this report accurately and correctly in terms of scientific methodology. I would also like to additionally thank my academic advisor, Prof. Pierre Maret for his constant encouragement towards my interest in research and development.

June 27, 2021

Contents

List of Figures

Abstract

This report is in the context of project CoSWoT (Constrained semantic web of things) that is being undertaken at the Université de Lyon. The idea of the work undertaken is to test different kinds of sensors effectively and find out what sensor is suitable in what kind of platform with connectivity. That apart, use case study of a "Smart Lab" system would also be implemented to test the effectiveness of our theory of devising this structure.

0.2 Introduction

Internet of Things (IoT) is considered to have a lot of potential and modern day application in several areas. However, the potential of this new kind of technology is being held back by the non-implementation of different kinds of sensors in a common framework for a practical use-case.The ambient-intelligent care rooms for the future have a wide range of Internet of Things devices such as sensor with consistent data generation capabilities. All of these can be used for a variety of functionalities such as monitoring of weather and various environmental parameters (example: light intensity, temperature), bodily parameters (examples: heart/pulse rate). The advantage of the IoT is that the data streams originating from the various sensors and devices can be combined with this knowledge to derive new knowledge about the environment and the patient's current condition[14]. This enables devices to achieve situation- and context-awareness, and enables better support of the nursing staff in their activities[13].

There are many semantic languages, such as the Resource Decription language (RDF) and the Web Ontology Language (OWL) which allows us to use this semantic framework through ontologies.[3] An ontology is a semantic model that formally describes the concepts in a particular domain, their relationships and attributes.[10] The use of ontology can significantly help us in the representation of heterogeneous data in a uniform way. The advancement of ontologies that are specific to IoT devices such as the Semantic Sensor Network(SSN)[5] that is used to enrich IoT data. However, semantically reasoning over a very large or a very complex data can be taxing on the system. Therefore, the system might not be able to keep up with continuous data streams when we take into consideration of a smart lab or a smart healthcare hospital system. [8]. A slow system can be potentially detrimental when we consider the harm it might be causing over its overall functioning. In an example we can cite, a smart lab might potentially need to ring an alarm for an emergency immediately. By immediately, it can be within a matter of 2 seconds. For each situation or use case, the urgency can be defined differently. Furthermore, due to the constraint on the resources available to function, the time complexity can be a bigger issue in this case.

To tackle the issue with performing real-time analysis, two research trends have emerged, being stream reasoning and cascading reasoning. Stream reasoning[8] tries to incorporate semantic reasoning techniques in stream processing techniques. It defines a data stream as a sequence of time-annotated

items ordered according to temporal criteria, and studies the application of inference techniques to such streaming data[8]. Cascading reasoning [23] exploits the trade-off between reasoning complexity and data stream velocity by constructing a processing hierarchy of reasoners. Hence, there is the need for a platform using these techniques to solve the issues in smart healthcare. The complex way in which semantics functions depends a lot on the underlying language that is used which means the expressiveness of the language.

Web of Thing (WoT) has been primarily modeled to overcome application layer fragmentation in the application layer [11]. A "thing" is identified by its URI, which can be used as an identifier for the semantic interoperability between the supplier and the consumers. A Web of Thing platform uses some standardization norms set by international organization like WC3. CoSWoT (Constraint Semantic Web of Things) is a research project sponsored by the University of Lyon which seeks to counter fragmentation in the application layer. The main objectives of the CoSWoT Project revolve around the following:

1. **Use of Ontologies** - This revolves around the usage of ontologies to create generalised models for heterogeneous devices. The main concern surrounding this is since different devices send data in different formats, a question of interoperability arises and standardisation organizations (such as W3C) aims at countering that. However, questions on whether existing ontologies have been adequetly to target application domains or the applicability of data stream principles developed in variety of protocols and standards remains to be explored.

2. **Inclusion of Distributed and Embedded Reasoning in the Platforms** - Questions in this regard arises on how to provide reasoning capabilities on a platform that is so heavily fragmented because of the heterogeneous nature of the devices and the data types.

0.2.1 Use IoT Devices for Smart Reconstruction and Web of Things

There are several different kinds of IoT devices that have come up over the last several years. These have been primarily used to build up smart infrastructure for the need of technological and societal progression.By 2030 100 million devices are supposed to be connected to the internet.[19] The challenge of handling the vast amount of real-time data was upfront and many

people came up with possible solutions to the problem. [24] The lead to the coming up of the Web of Things solution to the problem.[15] This basically consists of the following:

1. **Thing Description** - It is basically a machine reading meta data that *describes* the contents of the thing and its action affordances .

2. **Binding Templates** - This provides the informational context on how to provide the necessary network interfaces for protocols used and an IoT-based ecosystem.

3. **Scripting APIs** - The purpose of scripting APIs is to simply WoT development models. This aspect of the WoT abstract architecture is optional and is primarily important to aid vendors with portability.

4. **Security Guidelines** - An optional building block of the WoT architecture. This is primarily important to provide security measures in a WoT system architecture.

   A more detailed explanation of the most important component - namely the Thing Description is explained in the subsections that follow.

   Fig 1 describes a regular abstract architecture of a Thing Description. It provides us with different kinds of interactions in a WoT platform, namely *Thing-to-Thing*, *Thing-to-Cloud* and *Thing-to-Gateway* interactions [1].

0.2.2 Rules and Reasoning

Rules and *Reasoning* in computer science and semantic technology refers to infer logical consequences from a given set of axioms or a group of asserted facts. As simple rule can be described as follows:

if $Temperature \geq 20$ **then**
  $Red \leftarrow LED$
else
  $Blue \leftarrow LED$
end if

From the above example of a simple rule, we can see that it can be seen that rules and reasoning can be used to infer that an LED should glow red when the temperature is greater than 20 and glow blue when the temperature is below 20.

0.3 Objective of the Report

- This report contains information and evidence with regards to the smart reconstruction and a possible use case that can be reconstructed using various scenarios. In this work, we have first explored existing sensors in use and their possible installation. We have then explored smart framework use cases set up by others using similar methodologies. These were deconstructed and methods for possible reconstruction for our very own use cases were explored.

0.4 Related Works

0.4.1 Sensors/IoT Devices in Use

There have of course been several IoT devices that have been introduced in the market recently with each having different set of capabilities. Given our specific requirement, it is extremely important for us to explore the different kinds of sensors available in the market. Some of which are as follows:

(a) **Arduino Uno** - Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; it needs to be connected to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.[2]

(b) **Pycom-SiPy** - Pycom SiPy is a triple bearer MicroPython-enabled micro controller with Sigfox, Wi-Fi, and Bluetooth.[4]

(c) **ESP8266** - It is a very cost-effective and highly integrated Wi-Fi MCU for IoT applications.ESP8266EX is integrated with a 32-bit Tensilica processor, standard digital peripheral interfaces, antenna switches, RF balun, power amplifier, low noise receive amplifier, filters and power management modules.It achieves extra-low power consumption and reaches a maximum clock speed of 160 MHz. The Real-Time Operating System (RTOS) and Wi-Fi stack allow about 80 percent of the processing power to be available for user application programming and development.[17]

(d) **Raspberry Pi** - Raspberry Pi is a series of small single-board computers (SBCs) developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom. The Raspberry Pi project originally leaned towards the promotion of teaching basic computer science in schools and in developing countries. The original model became more popular than anticipated, selling outside its target market for uses such as robotics. It is widely used in many areas, such as for weather monitoring, because of its low cost, modularity, and open design. It is typically used by computer and electronic hobbyists, due to its adoption of HDMI and USB devices.[16]

(e) **iot.mozilla.org** - This is an IoT Platform rather than a device that was explored. In our case, this worked as a "Gateway" along with the Raspberry Pi. WebThings is Mozilla's open source implementation of the Web of Things, including the WebThings Gateway and the WebThings Framework.It provides a web-based user interface to monitor and control smart home devices, a rules engine to automate them and an add-ons system to extend the gateway with support for a wide range of existing smart home devices.[21]

0.4.2 Stream Reasoning Frameworks

Data Stream Management Systems (DSMS) and Complex Event Processing (CEP) systems allow to query homogeneous streaming data structured according to a fixed data model [19]. However, in contrast to Semantic Web reasoners, DSMS and CEP systems are not able to deal with heterogeneous data sources and lack support for the integration of domain knowledge in a standardized fashion. Therefore,

stream reasoning [15] has emerged as a challenging research area that focuses on the adoption of semantic reasoning techniques for streaming data. The first prototypes of RDF Stream Processing (RSP) engines [20] mainly focus on stream processing. The most well-known examples of RSP engines are C-SPARQL [21] and CQELS [22], but others also exist, such as EP-SPARQL [23] and SPARQLStream [24]. Because a continuous data stream has no defined ending, a window is placed on top of the data stream. A continuous query is registered once and produces results continuously over time as the streaming data passes through the window. As such, these RSP engines can filter and query a continuous flow of data and can provide real-time answers. Each of these engines has different semantics and is tailored towards different use cases. Other solutions, e.g., Sparkwave [25] and INSTANS [26], use extensions of the RETE algorithm [27] for pattern matching.

0.4.3 Cascade Reasoning Frameworks

The concept of cascading reasoning [22] was proposed to exploit the trade-off between reasoning complexity and data stream velocity. The aim is to construct a processing hierarchy of reasoners. At lower levels, high frequency data streams are filtered with little or no reasoning, to reduce the volume and rate of the data. At higher levels, more complex reasoning is possible, as the change frequency has been further reduced. This approach avoids feeding high frequency streaming data directly to complex reasoning algorithms. In the vision of cascading reasoning, streams are first fed to one or more RSP engines, and then to more expressive semantic reasoners. The concept of cascading reasoners fits nicely with the current trend in IoT architectures towards Fog computing [6], where the edge is introduced as an intermediate layer between data acquisition and the cloud-based processing layer. The edge allows filtering and aggregation of data, resulting in reduced network congestions, less latency and improved scalability. In addition, it enables to process the data close to its source, which in turn can improve the response time, as it allows to rapidly act on events occurring in the environment. As such, fast and possibly less accurate derivations and actions can be made at the edge of the network. These intermediate results can then be combined and forwarded to the cloud

for further, more complex and less time-stringent processing. In this way, more privacy is also enabled. For example, these intermediate results can filter out sensitive data, to avoid sending it over the network. Recently, much research effort has been put into Fog computing [9], and Fog computing frameworks, such as FogFrame [20], are being designed. These frameworks focus on the creation of a dynamic software environment to execute services in a distributed way. They are useful for system deployment, execution and monitoring, but not sufficient to support cascading reasoning. In addition, a generic framework is required that enables cascading reasoning across the IoT fog. Different distributed semantic reasoning frameworks exist, e.g., DRAGO, LarKC, Marvin and WebPIE. However, they all have limitations in terms of applicability for the IoT [12]. First, they do not consider the heterogeneous nature of an IoT network. In particular, the use of low-end devices and networking aspects are not considered. Both criteria are, however, essential to Fog computing. Second, as they distribute reasoning evenly across nodes, cascading reasoning is not considered. Third, they do not focus on complex reasoning. Hence, these frameworks cannot be used as such. Recent stream reasoning research also touches the area of Fog computing and devices with limited resources. Various relevant topics are addressed, from publishing RDF streams from smartphones [18], over optimizing semantic reasoning in memory-constrained environments, to optimizing the format to exchange and query RDF data in these environments. These results can be useful for a cascading reasoning system, but do not solve the need for a generic cascading reasoning framework.

0.4.4 Use Case Study - The Smart Hospital

The authors in [7] implemented a Cascade Reasoning Framework for a smart healthcare system. They realised the vision of cascading reasoning through a framework, using stream reasoning techniques and following the Fog computing principles. Stream reasoning techniques are required in order to infer real-time knowledge and actions from the voluminous and heterogeneous background knowledge and streaming data sources. However, current stream reasoning solutions fail to combine real-time and expressive reasoning. Incorporating them in a

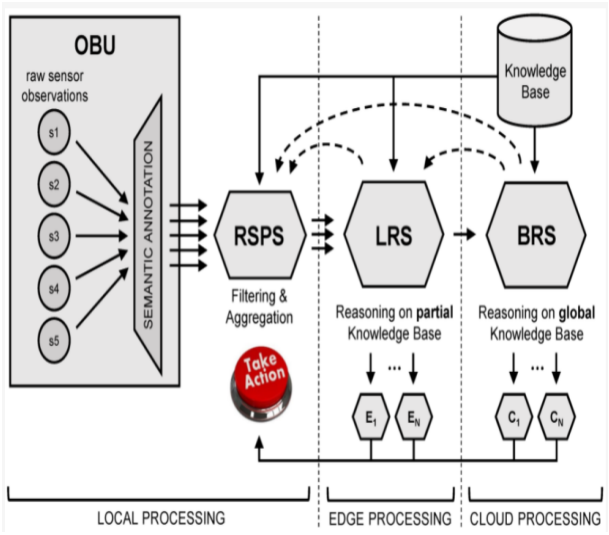

Figure 1: High-level architecture of the proposed cascading reasoning framework. The blocks represent the several components, the arrows indicate how the data flows through these components. The dotted arrows indicate possible feedback loops to preceding components.

cascading fashion is a possible solution. In addition, cascading reasoning and Fog computing principles offer the potential to solve the existing issues in smart healthcare. The central knowledge base contains the domain ontologies and static context information. For example, in healthcare, the knowledge base includes information on existing medical domain knowledge and a semantic version of the EHR of patients. The information in the knowledge base can be managed in a centralized database system, which can be an RDF triple store. In this case, the triple store contains mappings of the existing data architecture to the supported ontological formats. In addition, the information can also be managed in a regular database system that supports ontology-based data access (OBDA) [52]. The OBU refers to the infrastructure used to monitor the given environment. This infrastructure can consist of WSNs, BANs and other sensor platforms. The task of the OBU is threefold: (i) capture the raw sensor observations, (ii) semantically annotate these observations and (iii) push the resulting set of RDF triples on its corresponding output RDF data stream. This set of RDF triples should consist of a reference to the sensor producing the observation,

the observed value and an associated timestamp. The RSPS, LRS and BRS components are the stream processing and reasoning components. By configuration, only the relevant parts of the central knowledge base are available on each RSPS and LRS component, as these components typically do not need to know all domain knowledge and/or context information of the full system. On each BRS component, the full central knowledge base is available. Updates to the knowledge bases are coordinated from the BRS component(s). The RSPS is situated locally and contains an RSP engine. The input of this engine consists of the RDF data streams produced by the OBU, or another RSPS. The task of the RSP engine is to perform some initial local processing of these input data streams, by aggregating and filtering the data according to its relevance. On the RSPS, little to no reasoning is done on the data, depending on the use case. The output of the RSP engine is one or more RDF streams of much lower frequency, containing the interesting and relevant data that is considered for further processing and reasoning. The LRS is situated in the edge of the network. Here, a local reasoner is running that is capable of performing more expressive reasoning, e.g., OWL 2 RL or OWL 2 DL reasoning. It takes as input the output RDF stream(s) of the RSPS, or another LRS. As the velocity of these streams is typically lower than the original stream, computation time of the reasoning can be reduced. The service has two main responsibilities. First, it can push reasoning results and/or data patterns to another LRS, or a BRS in the cloud, for further processing. Again, the output stream typically is of lower frequency than the input streams. Second, it can also push results to one or more other external components that are capable of performing some first local actions. This allows for fast intervention, before the results are further processed deeper in the network. The BRS is situated in the cloud. It also consists of an expressive reasoner that has access to the full central knowledge base. Typically, a small number of BRS components exist in the system, compared to several LRS and even more RSPS components. The reasoning performed by the BRS can be much more complex, as it is working on data streams of much lower frequency compared to the local and edge components. This enables to derive and define intelligent and useful insights and actions. These insights and action commands can be forwarded to other BRS or external components, which can then act upon the received information or

commands. In some use cases, it might be useful to provide feedback to preceding components in the chain. This is possible in the current architecture, by the use of feedback loops. This feedback can be seen as messages, e.g., events or queries, in the opposite direction of the normal data flow.When deploying the architecture, each component in the system should be configured. To this end, the observer concept is used: each component, including external components, can register itself as an observer to an output stream of another component. In this way, the system components can be linked in any possible way. Hence, using the generic architecture, an arbitrary network of components can be constructed and configured. It should be noted that this is a push-based architecture, where the outputs of each component are immediately pushed to the input stream(s) of its observers. Note that this architecture assumes the following prerequisites: (i) the security of the architecture has been set up and ensured; (ii) no loss of connectivity to the cloud is assumed; (iii) each component runs on a node with at least 1 GB of memory resources; and (iv) each component is available at all times.

In this paper, a cascading reasoning framework is proposed, which can be used to support responsive ambient-intelligent healthcare interventions. A generic cascading architecture is presented that allows for constructing a pipeline of reasoning components, which can be hosted locally, in the edge of the network, and in the cloud, corresponding to the Fog computing principles. The architecture is implemented and evaluated on a pervasive health use case situated in hospital care, where medically diagnosed patients are constantly monitored. A performance evaluation has shown that the total system latency is lower than 5 s in almost all cases, allowing for fast intervention by a nurse in case of an alarming situation. It is discussed how the cascading reasoning platform solves existing issues in smart healthcare, by offering the possibility to perform personalized time-critical decision-making, by enabling the usage of heterogeneous devices with limited resources, and by allowing for flexible privacy management. Additional advantages include reduced network traffic, saving of back-end resources for high priority situations, improved responsiveness and autonomy, and removal of a single point of failure. Future work offers some interesting pathways. First, it should be researched how to deal with connectivity losses and

noisy sensor observations, which are current system drawbacks. Second, a large scale evaluation of the platform should be performed with multiple devices and different healthcare scenarios in realistic conditions. To this end, data collection of representative patient profiles and healthcare scenarios is currently ongoing. Third, the framework can be extended to adaptively distribute a federated cascading reasoning system across the IoT fog, also taking into account the scaling of the full system.

0.5 Implementation

0.5.1 Implementation of Different Sensors

0.5.2 IoT Mozilla Gateway with Raspberry Pi

0.5.3 C-SPARQL Testing and Implementation

0.6 Results and Discussion

0.7 Remaining Work

0.8 Conclusion

Bibliography

[1] Web of things (wot) architecture.

[2] Yusuf Abdullahi Badamasi. The working principle of an arduino. In *2014 11th international conference on electronics, computer and computation (ICECCO)*, pages 1–4. IEEE, 2014.

[3] Payam Barnaghi, Wei Wang, Cory Henson, and Kerry Taylor. Semantics for the internet of things: early progress and back to the future. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(1):1–21, 2012.

[4] José María Cámara Nebreda et al. Pycom-multinetwork devices, part one: Mqtt via wifi. 2018.

[5] Michael Compton, Payam Barnaghi, Luis Bermudez, Raul Garcia-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, et al. The ssn ontology of the w3c semantic sensor network incubator group. *Journal of Web Semantics*, 17:25–32, 2012.

[6] Amir Vahid Dastjerdi, Harshit Gupta, Rodrigo N Calheiros, Soumya K Ghosh, and Rajkumar Buyya. Fog computing: Principles, architectures, and applications. In *Internet of things*, pages 61–75. Elsevier, 2016.

[7] Mathias De Brouwer, Femke Ongenae, Pieter Bonte, and Filip De Turck. Towards a cascading reasoning framework to support responsive ambient-intelligent healthcare interventions. *Sensors*, 18(10):3514, 2018.

[8] Daniele Dell'Aglio, Emanuele Della Valle, Frank van Harmelen, and Abraham Bernstein. Stream reasoning: A survey and outlook. *Data Science*, 1(1-2):59–83, 2017.

[9] Julien Gedeon, Jens Heuschkel, Lin Wang, and Max Mühlhäuser. Fog computing: Current research and future challenges. *1. GI/ITG KuVS Fachgespräche Fog Computing*, pages 1–4, 2018.

[10] Thomas R Gruber. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993.

[11] Dominique Guinard, Vlad Trifa, and Erik Wilde. A resource oriented architecture for the web of things. In *2010 Internet of Things (IOT)*, pages 1–8. IEEE, 2010.

[12] Amelie Gyrard, Soumya Kanti Datta, Christian Bonnet, and Karima Boudaoud. A semantic engine for internet of things: Cloud, mobile devices and gateways. In *2015 9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 336–341. IEEE, 2015.

[13] Femke Ongenae, Jeroen Famaey, Stijn Verstichel, Saar De Zutter, Steven Latré, Ann Ackaert, Piet Verhoeve, and Filip De Turck. Ambient-aware continuous care through semantic context dissemination. *BMC medical informatics and decision making*, 14(1):1–27, 2014.

[14] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE communications surveys & tutorials*, 16(1):414–454, 2013.

[15] Dave Raggett. The web of things: Challenges and opportunities. *Computer*, 48(5):26–32, 2015.

[16] Matt Richardson and Shawn Wallace. *Getting started with raspberry PI.* " O'Reilly Media, Inc.", 2012.

[17] Marco Schwartz. *Internet of Things with ESP8266*. Packt Publishing Ltd, 2016.

[18] Yehia Abo Sedira, Riccardo Tommasini, and Emanuele Della Valle. Mobilewave: Publishing rdf streams from smartphones. In *International Semantic Web Conference (Posters, Demos & Industry Tracks)*, 2017.

[19] Saurabh Singh, Pradip Kumar Sharma, Seo Yeon Moon, and Jong Hyuk Park. Advanced lightweight encryption algorithms for

iot devices: survey, challenges and solutions. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–18, 2017.

[20] Olena Skarlat, Kevin Bachmann, and Stefan Schulte. Fogframe: Iot service deployment and execution in the fog. *KuVS-Fachgespräch Fog Comput*, 1:5–8, 2018.

[21] Erich Stark, Frank Schindler, Erik Kučera, Oto Haffner, and Alena Kozáková. Adapter implementation into mozilla webthings iot platform using javascript. In *2020 Cybernetics & Informatics (K&I)*, pages 1–7. IEEE.

[22] Heiner Stuckenschmidt, Stefano Ceri, Emanuele Della Valle, and Frank Van Harmelen. Towards expressive stream reasoning. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2010.

[23] Xiang Su, Pingjiang Li, Jukka Riekki, Xiaoli Liu, Jussi Kiljander, Juha-Pekka Soininen, Christian Prehofer, Huber Flores, and Yuhong Li. Distribution of semantic reasoning on the edge of internet of things. In *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9. IEEE, 2018.

[24] Roy Woodhead, Paul Stephenson, and Denise Morrey. Digital construction: From point solutions to iot ecosystem. *Automation in Construction*, 93:35–46, 2018.

0.9 Appendix