Creating Time Series-Based Metadata for Semantic IoT Web Services

Kasper Apajalahti

Aalto University School of Science, Finland,
kasper.apajalaht@aalto.fi
https://seco.cs.aalto.fi/projects/mobile5g/

Abstract. In the near future, the Internet of things (IoT) will rapidly change and automate tasks in our everyday life. IoT networks have sensors measuring the environment and automated agents changing it with respect to predefined objectives. Modeling agents as web services requires lots of metadata from the environment in order to define the desired performance in a specific context. For this purpose, we propose an automatic measurement-based metadata creation method that analyses multivariate time series gathered from the sensors during agents change the environment. The time series analysis uses a cumulative sum algorithm (CuSum) to detect events and association rule learning to find temporal patterns. We evaluate our system with a Long-Term Evolution (LTE) simulator having mobile phones corresponding to IoT devices, LTE macro cells as the data source, and the Self-Organised Network (SON) functions as the automated agents in the network. Our experiments give promising results and show that the metadata creation process can be utilised to characterise IoT agents.

1 Introduction

Internet of Things (IoT) services combine sensors and IoT devices into applications that solve predefined use cases. Some services provide access to data gathered from sensors and others analyse the data and perform actions on the environment with respect to some objectives [8], for example, to increase the temperature of a room or signal quality of a mobile device. A key challenge in IoT is the interoperability between services and applications [5]. The services may be deployed to a variety of environments. Yet, the semantics of services should be defined with relevant metadata in order to find similarities in services across network domains. Analogously to a cellular network, one domain might want to prevent an anomalous behaviour by understanding how other domains have experienced and solved an issue in a similar context. For example, a sudden bad weather may be the root cause for terminals (IoT devices) to increase the load of nearby indoor cells (data sources) as people escape rain indoors. In this case, web services accessing local weather data combined with network-specific data and agent services would be necessary to act proactively to handle the upcoming load peak indoors.

A general challenge with web services is the need to involve domain experts in developing the services [14]. Particularly, semantic IoT services have development costs in modeling the services with relevant metadata [12] and linking it to other ontologies in order to make services discovered and invoked [9]. The potential development costs also lead to a cold-start problem among the web service modelers: the SWS system cannot recommend suitable and interoperable services before developers in several domains have used their resources to manually model them. Hence, there is a need to automate the service development process.

In this paper, we propose a time series-based metadata creation process for semantic IoT services. The process is evaluated in a Long Term Evolution (LTE) network simulator environment having mobile devices that measure the quality of service and LTE macro cells that periodically report aggregated measurements. With respect to the LTE networks, the Self-Organised Networks (SON) has brought automation to the management tasks [11]. The SON functions can be viewed as specialised agents controlling the LTE cell configurations with respect to predefined objectives. The simulator contains simple SON functions which optimise the network performance and our goal is to characterise their behaviour with the metadata creation process.

The paper consists of the following parts. Related work is discussed in Section 2. After that, Section 3 gives an overview of the SWS methodology in the scope of IoT and cellular networks. Section 4 explains the theory and statistical methods used to create time series-based metadata for the services. Section 5 presents the experiments for the metadata creation process in an LTE simulator. Finally, Section 6 concludes the paper.

2 Related Work

Bytyçi et. al [4] propose a method that combines association rule learning and ontologies to mine patterns from water quality measurement data. They managed to enrich the mining results by first populating the context ontology with sensor data and then using the ontology as an input to the association rule learning.

Fan et. al [6] use association rule learning for sensor-based constructions to find contextual patterns from sensor measurements [6]. The experiments show that temporal patterns can be identified with respect to time metadata, such as a public holiday, weekday, or weekend.

Galushka et. al [7] examine data mining techniques for smart home environment. The authors present techniques both to transform time series into labeled segments and to use association rule learning to find temporal patterns from them.

Labbaci et. al [13] analyse web service logs and interactions between web services to learn frequent compositions and substitutions of services in a web service system. Their method has a similar focus on analysing the past data related to the web service invocations in order to learn characteristics of their behaviour. Such as in our work, they use association rule learning to find the most frequent item sets from the web service-related data.

The related work focuses on association rule learning of measurement-based sensor data and web service logs. However, none of the earlier works learn association rules from sensor measurements in order to characterise IoT services, which is the focus of this paper.

3 Semantic web service model

3.1 Methodology

The core idea in the Semantic Web Service (SWS) is to discover, compose, and invoke services with respect to user requests [2]. This idea applies both to IoT and cellular networks: requests based on the multivariate measurements of IoT devices or mobile phones need to be handled with an action that causes a desired impact to the measured values. Figure 1 describes a simplified web service ontology, where arrows depict *"hasElement"* relations. It also explains the analogy to IoT and cellular networks, such as the service corresponds to an agent, operation to an action, and effects to changes in the metric values. The architecture of the model is adapted from a simple SWS model, WSMO-lite [15]. A service has operations that aim to change the status of its environment.

Effects play a central role in using the ontology; the service model is used by linking the effects of requests and operations. For example, the goal of a network operation could be to enhance customer satisfaction for mobile users during a rush hour. This context-specific intention is mapped to some metric effects, such as an increase in the throughput and balancing the usage of physical resource blocks (PRBs). Based on the network measurements, some service operations are known to produce the requested context-specific effects and thus, they are mapped as responses to the request.

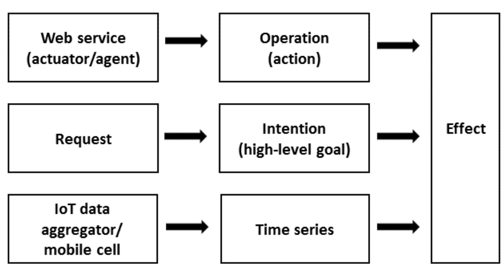

Fig. 1. Service ontology constructs for a service (top), request (middle), and environment/object status (bottom).

As domains may have dedicated data and means of management, there is a need to also understand the semantics of cross-domain effects. For example, domains might have services that monitor threshold values for different key performance indicators (KPIs). Yet, some KPIs are associated (correlate) in the given context and are part of the same intention. Thus, both of the services would be valid solutions to a request having similar effect as an objective. More details about the dependency modeling of effects are defined in earlier research [3].

3.2 Cross-domain request

An example of a cross-domain user request from mobile network management is illustrated in Figure 2. An operator from the network X wants to request better Quality of Service (QoS) in some context, such as during a public event. The requested intention can be achieved by increasing the throughput. Another network domain Y has deployed a SON function as a web service with an operation that is known to increase Reference Signal Received Power (RSRP) during a public event. Knowing the semantic similarity between the same cross-domain KPIs and that the throughput and RSRP in the domain Y are statistically associated, the given service operation can be discovered and mapped as a response to the user request. Altogether, even though a problem and solution would be in different networks and might address different parameters, the SWS system finds a request-operation mapping relation with respect to semantic modeling and statistical dependencies.

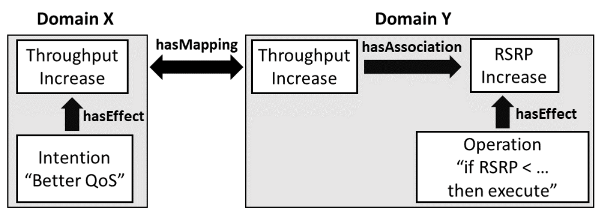

Fig. 2. An example of a cross-domain request-operation mapping.

In order to have a functional SWS system in a distributed and multi-domain sensor network, contextual metadata is needed, for example, the relevant operation-specific effects that enable the discovery of the web services. Creating and maintaining these manually is resource-intensive. Moreover, metadata mappings (such as corresponding KPIs) between different environments need to be resolved before the full benefit and utilization of the services can be realised. Automation for the service modeling and metadata addition can reduce the cost of deployment significantly by utilising SWS systems. In the next subsection, we introduce a process to address this problem with automatic metadata creation methods.

4 Methods to create service metadata from sensor measurements

4.1 The data sequence of the process

The measurement-based metadata is added as effects to the service operations and it is used to bind requests to operations. Our process analyses statistically the behaviour of a service operation in a given context while it is executed. Figure 3 illustrates the data sequence of processing metadata for a service operation based on realised actions and measured metric values both before and after the actions have been taken. In the beginning, the user decides how an operation is defined. For example, a service could be an algorithm and an operation a set of parameter values. The actions fulfilling this criteria are retrieved from the database (step 1) and based on their timestamps, a time series for every available metric of the operated object (such as the mobile cell) is retrieved from the measurement database (step 2). Time series are analysed with an event detection method (step 3) and the detected events are sent to the association learning component (step 4). This component detects whether one or more metrics have temporal correlation. Finally, the associated events are sent to the ontology and the service operation is populated with these events (step 5). The events are later used as service operation effects, as described in Figure 1.

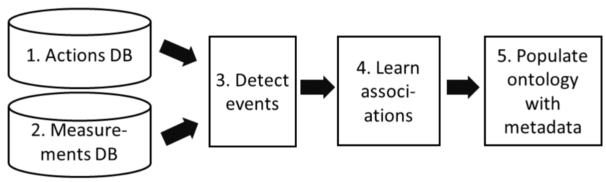

Fig. 3. Process flow for identifying associated events and adding them as operation metadata.

4.2 Event detection with CuSum

The CuSum algorithm is a statistical quality control method that can be used to detect value changes in a time series. The basic concept is to cumulatively sum up changes between data points and a comparison value and flag a change if the sum exceeds a predefined threshold value. The Equation 1 describes how to detect increasing event in our system. The equation contains a max of zero and the cumulative sum of value s_h, the data point x_t, and the combined comparison value of mean and standard deviation, μ and σ, calculated from the time series. σ is used as a threshold sum value for increasing trends.

$$s_h = max(0, s_h + x_t - \mu - \sigma) \tag{1}$$

For analysing decreasing trends in a time series, the Equation 2 is used instead. Compared with the earlier equation, now a min operator is used and CuSum contains a positive sign for the σ. The threshold sum for detecting a decreasing trend is $-\sigma$.

$$s_l = min(0, s_l + x_t - \mu + \sigma) \tag{2}$$

When CuSum is executed for all operation-specific actions, the outcome is a dataset where each row depicts a single action having a list of measurement events it produced. From this dataset, we may further learn operation-specific patterns between measurement events.

4.3 Temporal pattern mining with Apriori

Association rule learning is a data mining method that learns rules between the sets of items in a database. The idea is to analyse the co-occurrence of items in a database row and to use some measure and threshold to find out relevant rules. The simplest measure is the support, which is calculated as a proportion of the database rows containing the given set of items. [10] In our use case, the support is the proportion of detection timestamps containing a set of metric events. Thus, it indicates the frequency of the events occurring simultaneously in the given context.

In addition to support, confidence is another measure to determine associations between items. The Equation 3 shows the definition of the confidence. It can be interpreted as an if/then pattern: if set of events X occurs, then set of events Y also occurs. As it can be seen, the measure indicates the proportion of X (the support of X) that also contains Y (the support of X and Y). [10]

$$conf(X \rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)} \tag{3}$$

In this system, the objective is to learn support and confidence values for measurement events occurred during a set of actions made by an agent. For this purpose, we use an open-source implementation[1] that of a well-known Apriori algorithm (see [1] for further details).

5 Evaluation

5.1 Case study: LTE network simulation with SON functions

The applicability of our metadata creation process is evaluated with an LTE network simulator. The simulator environment comprises 20 LTE base stations with 32 LTE cells covering an area with a radius of 5 km. The simulator creates Performance Management (PM) data reports that contain cell level KPIs such as the average cell throughput, radio link failures (RLFs), average Reference Signal Received Power (RSRP), the overall usage of the Per Resource Blocks (PRB), and average channel quality indicator (CQI) level. The cell level KPIs are aggregations of the measurements made by the user equipments (UEs) that constantly report the experienced signal status to a cell they are attached to. The PM data of the cells are reported periodically in 15 minute intervals in simulation time. The time series gathered from the PM data contains 10 time steps before and after the actions have been executed or activated.

Table 1 describes the scenarios created for our experiments. The idea was to create network issues with similar objectives but in different contexts; in all scenarios, users have issues getting the required throughput level, but the required actions differ significantly from each other.

In the coverage problem, the UEs are located uniformly in an area where the coverage is insufficient and the solution is to increase TXP to enhance the coverage and therefore the overall throughput level. The second scenario, local overload, has a few hundred UEs located in a small area near one base station hosting three cells. Now the throughput should be increased by adjusting the antenna tilt angles (remote electrical tilt, RET) towards the group of UEs. The third scenario, mobile overload, has 500 uniformly located background UEs and a group of 500 UEs constantly moving in the

[1] https://github.com/asaini/Apriori

Table 1. Simulation scenarios

Scenario	UEs	Objective	Solution
Coverage problem	1000	Inc. Thr.	Increase power
Local overload	350	Inc. Thr.	Downtilt
Mobile overload	500	Inc. Thr.	Balance load

simulated area causing abrupt load peaks in the cells. An increase in the throughput in this case should be achieved by balancing the load between the nearby cells.

5.2 Context-specific support values

Figure 4 presents the action-specific support values in different scenarios and KPIs. Figure presents one subplot for each scenario and each subplot presents KPI-specific support values for each action. Positive support value indicates a support measurement for an increase and negative a decrease. For example, the first five bars show support values for the increasing and decreasing events for the KPIs when no action has been taken in the coverage problem scenario. The first bar shows that increasing events for CQI has been measured with a support value of 0.12 and decreasing events with a value of 0.09. With respect to these experiments, a threshold level of ±0.15 (marked with two dashed lines) is suitable for labelling action-specific KPI effects.

In general, we may conclude that the distribution of the scenario- and action-specific support values show that the detection of single KPI events works well as the values are plausible with respect to the actions. Especially, the throughput values show that the best agents in every scenario also enhance the throughput in the network, which is the desired outcome. Also, the fact that the number of false positive support values (values when no action is taken) is low, indicates an adequate performance of the CuSum method.

5.3 Context-specific associations and their applicability as metadata

The association rules for every scenario-specific action were generated with a minimum support level of 0.15 and confidence level of 0.70. Figure 5 shows the quantities of associations learned among the recorded events. With the given parameters, the best agents also generate the most associations between the KPI effects, whereas few associations are produced from other agents in the coverage problem. This is a desired outcome as our goal is to highlight the best matching agents in different contexts.

The final step in the metadata creation process is to populate the ontology instances with relevant events and associations. As defined in the Section 4.1, the populated ontology instances are web service operations: three scenario-specific operations for each web service (network agent). Finally, we may examine the request (ontology queries) that naps the queried effects with the relevant service operation metadata. Table 2 illustrates the examples of combining association rules that we tested and verified to retrieve correct mappings to the operations. For example, if a request contains associations from $IncTHR$ (throughput) to $DecRLF$ and from $DecRLF$ to $DecPRB$, it gives a unique

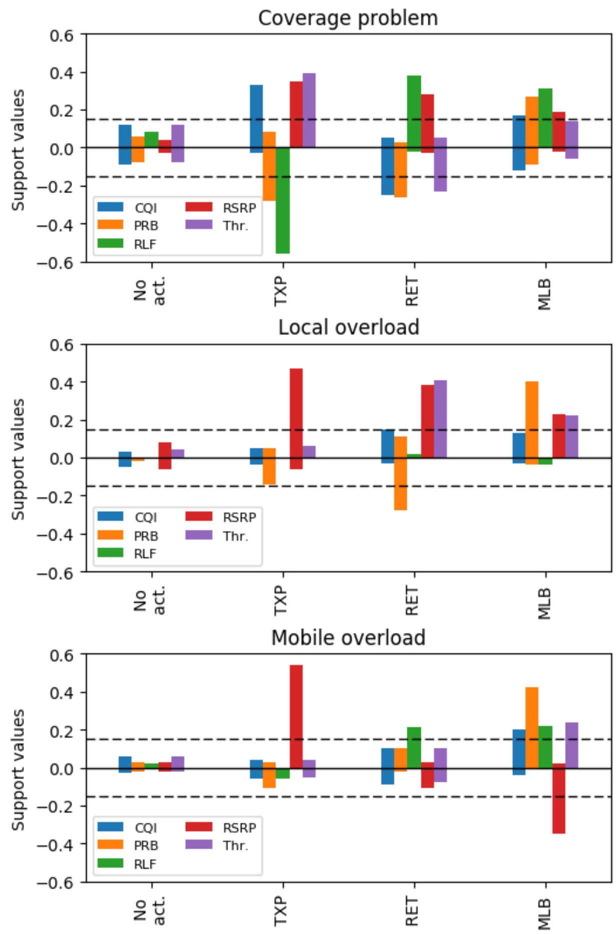

Fig. 4. Support values for action- and KPI-specific events in three scenarios.

mapping to the TXP agent operation on a coverage problem scenario. Similarly, the two other rows show rules that are also unique among all rule sets and that corresponds to the best solution in the scenario. In addition to the association rules shown in the table, the request query may include effects that pass the minimum support level (e.g. "increase the throughput") or negations of undesired effects (e.g. "do not decrease the throughput").

Altogether, the demonstrated associations indicate that we managed to distinguish the important agent operations and scenarios from each other with our metadata creation process.

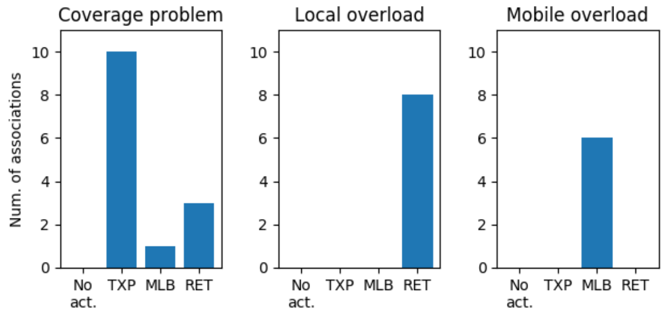

Fig. 5. Scenario-specific quantities of associations for service operations. Threshold for support is 0.15 and for confidence 0.70.

Table 2. Unique set of rules that characterise suitable agents in every scenario.

Scenario	Action	Matching rules
Coverage problem	TXP	$IncTHR \rightarrow DecRLF$, $DecRLF \rightarrow DecPRB$
Local overload	RET	$IncTHR \rightarrow DecPRB$, $IncRSRP \rightarrow DecPRB$
Mobile overload	MLB	$IncTHR \rightarrow IncPRB$, $DecRSRP \rightarrow IncPRB$

6 Conclusions and future work

We proposed a method of creating time series-based metadata for services that operate in IoT networks. The metadata creation process is a combination of statistical methods, event detection and association rule learning, and it is based on analysing multivariate time series gathered from the network elements while some actions (service operations) are executed. The process was evaluated with a Long Term Evolution (LTE) simulator where automated agents (web services) configure the antenna parameters of LTE macro cells in order to enhance the network quality. We created three simulation scenarios and evaluated the results of three agents in those. Our experiments show that the presented metadata creation process works on these scenarios; all suitable service operations can be characterised with the generated metadata. For future work, we examine and compare different event detection and pattern mining methods and evaluate them in more complex IoT environments.

Acknowledgment

We thank Vilho Räisänen and Eero Hyvönen for comments and support on early versions of this paper. This work is partially funded by Nokia.

References

1. Agrawal, R., Srikant, R., et al.: Fast algorithms for mining association rules. In: Proceedings of the 20th VLDB Conference. pp. 487–499 (1994)

2. Ankolekar, A., Burstein, M., Hobbs, J.R., Lassila, O., Martin, D., McDermott, D., McIlraith, S.A., Narayanan, S., Paolucci, M., Payne, T., Sycara, K.: DAML-S: Web Service Description for the Semantic Web, pp. 348–363. Springer Berlin Heidelberg, Berlin, Heidelberg (2002). https://doi.org/10.1007/3-540-48005-6_27
3. Apajalahti, K., Niiranen, J., Kapoor, S., Räisänen, V.: Sharing performance measurement events across domains. In: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM). pp. 463–469 (May 2017). https://doi.org/10.23919/INM.2017.7987313
4. Bytyçi, E., Ahmedi, L., Kurti, A.: Association Rule Mining with Context Ontologies: An Application to Mobile Sensing of Water Quality, pp. 67–78. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-49157-8_6
5. Da Xu, L., He, W., Li, S.: Internet of things in industries: A survey. IEEE Transactions on Industrial Informatics **10**(4), 2233–2243 (Nov 2014). https://doi.org/10.1109/TII.2014.2300753
6. Fan, C., Xiao, F., Yan, C.: A framework for knowledge discovery in massive building automation data and its application in building diagnostics. Automation in Construction **50**, 81 – 90 (2015). https://doi.org/http://dx.doi.org/10.1016/j.autcon.2014.12.006
7. Galushka, M., Patterson, D., Rooney, N.: Temporal Data Mining for Smart Homes, pp. 85–108. Springer Berlin Heidelberg, Berlin, Heidelberg (2006). https://doi.org/10.1007/11788485_6
8. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems **29**(7), 1645 – 1660 (2013). https://doi.org/http://dx.doi.org/10.1016/j.future.2013.01.010
9. Gyrard, A., Serrano, M., Atemezing, G.A.: Semantic web methodologies, best practices and ontology engineering applied to internet of things. In: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT). pp. 412–417 (Dec 2015). https://doi.org/10.1109/WF-IoT.2015.7389090
10. Hahsler, M., Grün, B., Hornik, K.: arules - a computational environment for mining association rules and frequent item sets. Journal of Statistical Software, Articles **14**(15), 1–25 (2005). https://doi.org/10.18637/jss.v014.i15, https://www.jstatsoft.org/v014/i15
11. Hämäläinen, S., *et al.*: LTE Self-Organising Networks (SON): Network Management Automation for Operational Efficiency. John Wiley & Sons (2011)
12. Heß, A., Kushmerick, N.: Learning to Attach Semantic Metadata to Web Services, pp. 258–273. Springer Berlin Heidelberg, Berlin, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39718-2_17
13. Labbaci, H., Medjahed, B., Aklouf, Y.: Learning interactions from web service logs. In: Benslimane, D., Damiani, E., Grosky, W.I., Hameurlain, A., Sheth, A., Wagner, R.R. (eds.) Database and Expert Systems Applications. pp. 275–289. Springer International Publishing, Cham (2017)
14. Pautasso, C., Zimmermann, O., Leymann, F.: Restful web services vs. "big"' web services: Making the right architectural decision. In: Proceedings of the 17th International Conference on World Wide Web. pp. 805–814. WWW '08, ACM, New York, NY, USA (2008). https://doi.org/10.1145/1367497.1367606
15. W3C: WSMO-Lite: Lightweight Semantic Descriptions for Services on the Web, W3C Recommendation, World Wide Web Consortium (2010). Available: https://www.w3.org/Submission/WSMO-Lite/ (accessed May 2018)