Applied Laser Technology

Vol. 27, No.2,April (2020), pp.01–05

Evaluation of a Semantic Thing-to-Service Matching Approach in Industrial IoT Environments

Dr D Arun Kumar[1,] A Valli Basha [2] S L Pratap Reddy[3], R V Sree Hari[4]

[2] Asst. Professor, Department of ECE, K. S. R. M College of Engineering(A), Kadapa

[1,3,4] Associate Professor, Department of ECE, K. S. R. M College of Engineering(A), Kadapa

Abstract—

While Industrial IoT platforms enable improved data-driven decision-making, which in turn increases productivity in manufacturing and other commercial proprietary industries when combined with specific IoT hardware, data interchange and provisioning between the data sources and platform services continue to be a challenge. We present and extensively detail a software-based open-source solution we call Thing to Service Matching (TSMatch), which allows semantic matching at a granular level between available IoT data and services. The study also discusses the proposed solution's deployment in two separate Aerospace production scenarios and provides an evaluation of the solution's performance in a testbed environment..

Key words

Some of the concepts that might be indexed in relation to the Internet of Things are: data sources, semantic matching, use cases, and the IoT application.

INTRODUCTION

Industrial Internet of Things (IoT) systems provide a number of benefits, including the opportunity to better use data gathered in industrial settings to boost production and company operations. Data collecting has advanced, yet it remains challenging to analyze and use the data obtained. Densely populated IoT infrastructures full of sensors and actuators are essential to the successful operation of IoT systems. Due to its vendor-specific nature, proprietary nature, and dependence on certain IoT hardware or cyber-physical systems, deploying and maintaining an IoT platform may be difficult. The additional, specialized human interaction required to move the data to the Cloud might make the process lengthy, error-prone, and/or costly [1]. The vendor-based strategy has resulted in a fragmented IoT environment, making interoperability a pressing concern from both a communication and an application perspective.

Concerned with the second issue, this research seeks to find a way to automate the transfer of information between IoT Things (data sources) and the services provided by an IoT platform. Thing to Service Matching (TSMatch) is a piece of open-source software we created to help with this issue. Fine-grained semantic matching between services and collected IoT data is within the scope of TSMatch's capabilities. As a result, TSMatch streamlines the process of connecting current IoT networks to external services by automating the matching and provisioning steps. This work contributes a description of the publicly available TSMatch1 engine,

(ii) an assessment of TSMatch in a simulated IIoT setup (testbed); and (iii) evidence that TSMatch can be used in industrial scenarios with realistic operational conditions. The next steps of the paper are outlined below. Section II describes the related work, while Section III explains how TSMatch's primary pieces of code interact together. In Section IV, we get into the nitty-gritty of the actual procedure. Furthermore, the study demonstrates in Part V that the system may be readily connected with other IoT systems like EFPF2. Then, in an experimental context, we evaluate TSMatch's supplementary processing time and time to completion. The results and discussion of those results may be found in Section VI. The conclusion and next steps are outlined in Section VII.

WORK IN RELATION

Service matching may be accomplished in a number of ways, some of which are logical [2], others are non-logical [3] semantic based techniques, and yet others are hybrid [4] approaches that combine logical and non-logical methods. For instance, Kovacs et al. offer the technical approach and the system architecture necessary to realize a worldwide semantic interoperability solution, which includes combining the FIWARE NGSI and the oneM2M context interfaces. Although this method has theoretical promise, it has yet to be put to the test in either a practical or experimental environment [5]. Additionally, Cassar et al. provide a matching system that combines semantic and probabilistic matching. Using a dataset of 1007 OWL-S descriptions of Web services, the accuracy and Normalized Discounted Cumulative Gain of the proposed approach were calculated.

Simulations have confirmed the concept, and the proposed technique outperforms prior attempts by a significant margin [6]. The method and the TSMatch concept were first presented by us in earlier papers. This research builds on our prior work by verifying TSMatch in real-world contexts after it has been described and implemented [7].

ARCHITECTURE THAT IS APPROPRIATE FOR THE PRODUCT OR SERVICE BEING PROVIDED

TSMatch is a platform that may be used to match IoT data semantically with services. In order to fulfill the needs of the services, TSMatch aims to automatically transmit data between IoT data sources and them. Currently, TSMatch does this by using a matching approach based on semantic similarity. The key participants of TSMatch are shown in Figure 1. The myriad sensors that make up the Internet of Things are represented here in a meaningful way. TSMatch's Engine is a server-based component that may currently be located anywhere, including the network's edge, an IoT gateway, or the cloud. Like a database, the information about Things is consistently recorded in the Thing registry. The TSMatch Client is the software that users like yourself download and install on your computer or mobile device. A person, a company, or even a piece of software might all qualify as users of Internet of Things services.

.

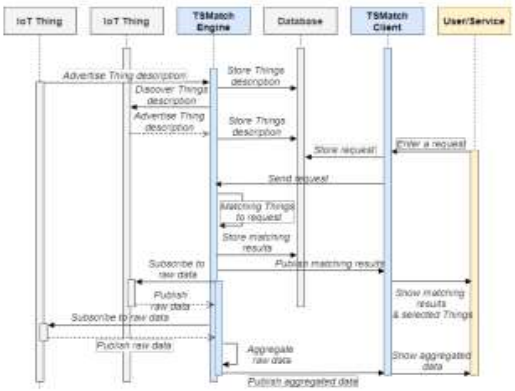

Fig. 1. Sequence diagram of the various actors involved in TSMatch.

In the first stage, shown in Figure 1, the IoT Things activate and then broadcast their descriptions. TSMatch Engine may add items to the Things registry and follow their activity by subscribing to events. In this case, Coaty3 is providing this functionality rather than TSMatch, a semantic matching engine. In order to get information on IoT gadgets, the TSMatch Engine may also contact an outside entity like a broker. That is to say, it's possible to configure it as a subscriber if that's what's required. When a user makes a service request, the request is sent to the TSMatch client, which then forwards it to the TSMatch Engine. This might be used, for instance, to determine the temperature at a specific location. If a service is requested, TSMatch will utilize similarity matching to see which IoT Things in the registry best meet the request and the given descriptions. The information obtained is sent to the TSMatch Client if a match is found.

The TSMATCH Implementation Considerations

In this section, we'll go through how TSMatch's various parts and interfaces are currently being implemented. All of TSMatch's parts are now dockerized4 since it is built on a micro-service design. The TSMatch engine in this implementation is hosted on an Internet of Things (IoT) gateway, while the client is built for Android. Wi-Fi is used for all of the interaction between the TSMatch components and the rest of the IoT architecture. After that, we'll go through each individual part.

Devices Connected to the Internet of Things

The OGC Sensor Thing API [8] has been used to model the IoT Things. Information on the sensors, including their attributes and whereabouts, is included in the description of the Thing. Coaty, an application-layer communication platform, is used to spread the word. Coaty allows Internet of Things devices to broadcast their semantic descriptions via multicast, and it alerts subscribers when the device goes offline by broadcasting a "deadvertized" event. In order to find out what's out there, the TSMatch Engine sends out a "discover" event and receives a response with the thing's description. Semantic representations of cyber physical systems using heterogeneous hardware, such as a sensor, a single-board computer (SBC), or a programmable logic controller

(PLC) coupled with sensors via its Input/output interface, relate to IoT Things. The sensor driver and a Coati agent are the software components that make up the IoT cyber-physical system and are responsible for handling tasks like publishing information about an IoT Thing.

TSMatch System

TSMatch is a server-side engine used to perform semantic matching between IoT Thing descriptions and service descriptions. The engine deals with service requests issued by the TSMatch Client. Using queries submitted to the TSMatch Thing registry, it also controls the process of matching the features and attributes of Things descriptions with the semantic description of services. To measure the degree of semantic similarity between two texts, researchers use the Sorensen-dice coefficient and a word frequency-inverse document frequency. Once a matching set is found, all of the available Things inside it are merged into a single object, and the updated version of the object is stored in the database. Whether or not a match is detected, the TSMatch Client will receive the matching result and provide it to the user. If a match is found, the engine will initiate a sequence of events that, depending on the selected location, will either subscribe to data from the selected sensors or calculate an average value for the selected sensors. The changes are then reflected in the TSMatch Client. If a request is deleted, TSMatch will stop subscribing the TSMatch client to the broker so that it may get updates on the observations made by IoT Things. Node.js The TSMatch Engine is a product of the computer languages JavaScript and Typescript5. We've implemented Srensendice similarity using npm's string-similarity and natural packages.

Consumer of TSMatch

The TSMatch Client takes care of the service's semantic description, either by constructing one in response to a user's request (such as "monitor temperature") or by retrieving one from a distant location. With the help of the React Native JavaScript framework8, we have created a mobile application called the TSMatch Client. In its present state, the client only supports Android 4.1 and later. TSMatch subscribes to a MQTT broker during launch. Users may browse a catalogue of available Things along with detailed descriptions, articulate their needs, and watch as their data is updated in real time.

Information Management, Search, and Sharing Concerning Things

Coati v2.0 allows IoT Things to register, declare themselves, be discovered, and communicate with the end user (the subscriber). An MQTT broker built on top of Mosquito v2.0.11 facilitates the conversation. The IoT descriptions are kept in a database that uses the JSONB data type for binary storage and retrieval of JSON objects. This database is built on PostgreSQL 13.2. Docker images9 built from the official PostgreSQL source code have been utilised.

Using TSMatch on EFPF Applications

European Connected Factory Platform for Agile Manufacturing (EFPF) ecosystem includes elements such as Industry 4.0, IoT, AI, big data, and digital manufacturing. An open, standardized "Data Spine" serves as the backbone of EFPF, facilitating the smooth integration of many different types of systems, platforms, tools, and services. separate businesses provide separate EFPF components, which work together through a central data spine. The EFPF ecosystem is, thus, constructed in a service-oriented architecture. Connected smart factories' unique requirements are met by the EFPF's ecosystem, which includes the Data Spine, the EFPF Web-based platform (which provides unified access to various tools and services through a Web-based portal), the base digital platforms (four base platforms funded by the European Commission's Horizon 2020 programme), and the external platforms. TSMatch is now included into EFPF. Its integration with the IoT automation platform Symphony Factory Edition from EFPF partner Nextworks10 (one of the External Platforms inside the EFPF federation) has been proven in real-world applications. Symphony is an end-to-end IoT platform that can accommodate a wide range of heterogeneous hardware, including sensors and actuators from a wide range of IoT vendors. Integrating the IoT Symphony platform with TSMatch through the EFPF Data Spine enables provisioning and data interchange with the IoT data already present in operational systems. EFPF has integrated and implemented TSMatch in three aircraft manufacturing use-cases (IAI), with help from Walter Otto Muller & Co. KG11 (WOM) and Innovint Aircraft Interior GmbH12. One use case is keeping a factory at a steady temperature and humidity to prevent components from failing due to improper storage conditions (WOM); another is keeping tabs on raw materials in a freezer to avoid waste due to excessive heat (IAI); and yet another is keeping tabs on a vacuum former from a distance to take corrective action as soon as pressure values deviate from acceptable ranges (IAI).

All three cases have been focused on safeguarding industrial operations' reliability and high quality by monitoring vital process variables and issuing warnings if thresholds are exceeded. TSMatch communicates with the external platform Symphony to transmit collected data on the appropriate environmental variables from IoT Things. In order to request services for IoT devices, the Symphony Factory Connector13 uses the TSMatch Client, as shown in Figure 2. Through the EFPF Data Spine, which offers services with compatible security capabilities, TS Match's data is ingested by a Cloud instance of the Symphony IoT automation platform. For the Things to have visual monitoring, sensor data and event storage, signal analysis, and alarm systems, the Symphony HAL (a software module that primarily abstracts the low-level details of various heterogeneous fieldbus technologies and provides a common interface to its users), Symphony Data Storage, and Symphony Visualization are used. After combining data from multiple sources and data brokers (e.g., TSMatch) (through stack light, which offers visual and aural signals), the Symphony Event Reactor determines actions such as control actions on field-level devices, alerts (emails, SMS), and alarms. Real-time data and the current condition of thresholds and alerts may be seen and handled in the Cloud deployment of the platform using the Symphony graphical user interface.

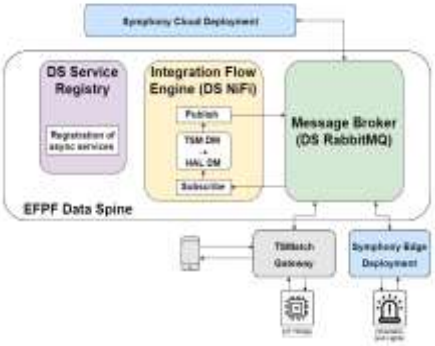

Fig. 2. EFPF interconnected components on the manufacturing use-cases.

As illustrated in Figure 3, a successful deployment has been done on production sites. The EFPF components, including TSMatch, have therefore been validated based on the defined requirements, usability aspects, as well as ease of installation/configuration. After 6 months of deployment, traceability and detection of abnormalities has been also achieved, which increased the reliability of the manufacturing process, reduced delivery delays, and minimized rejects/waste occurrences. Fig. 3. Installation of the use-cases.

PERFORMANCE EVALUATION AND RESULTS

TSMatch Testbed

Fig. 3. The TSMatch demonstrator at the forties IIoT Lab.

We have established a TSMatch testbed on the forties IIoT Lab14 based on the operational requirements generated from the TSMatch integration on real-world industrial use-cases, as shown in Fig. 4. As shown in Figure 3, the testbed consists of the following parts:

Two real Internet of Things (IoT) devices and ten simulated IoT devices are available, with the former sporting a total of five sensors (for measuring things like temperature, humidity, sound, air quality, and particles in the air). According to section IV, each virtual IoT thing is linked to a virtual IoT sensor, each of which is housed in its own Docker container.

• TSMatch Engine, Message Broker, and Database: TS Match Engine is containerized, and the Mosquito message bus and PostgreSQL database are also deployed as containers in the forties IoT gateway.

Using a MQTT client15, the IoT service request simulator may mimic third-party IoT services. TSMatch Client was abandoned in favour of the service request, which allowed for precise regulation of the inter-request time gap.

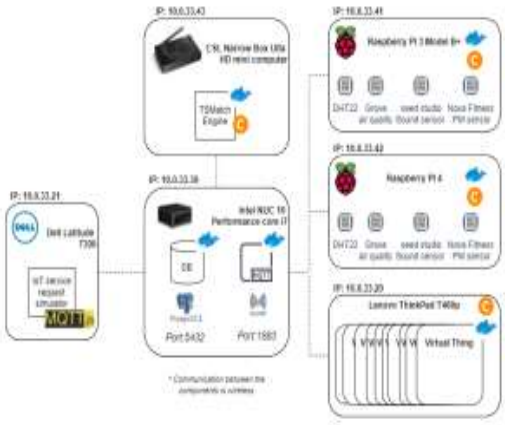

Fig. 4. TSMatch components and experimental interconnections.

The hardware specifications for each of the devices that is part of the demonstrator are given in Table I.

conclusion

TSMatch is a new approach described in this study for semantically matching IoT data sources and service descriptions. We outline the TSMatch software architecture and discuss its practical applications in the industrial.

TABLE I MEAN PT AND TTC OF SEQUENTIAL AND SIMULTANEOUS SCENARIOS.

	Sequential Scenario						Simulta-neous Sce-nario
	Fixed TI			Variable TI			
	Low Range	*Medium Range*	*High Range*	*Low Range*	*Medium Range*	*High Range*	
Mean PT(ms)	151.88	32.76	33.72	55.8	53.76	72.84	74.8
Mean TTC(ms)	321.56	170	111.08	127.8	198.88	260.96	237.04

contexts of the European EFPF project settings. The report also includes a first performance assessment of TSMatch and details the existing open-source TSMatch implementation's processing time and time to completion of requests. In order to identify the "most" appropriate collection of IoT Things whose aggregated data may satisfy a certain semantic request, future work will concentrate on enhancing the semantic matching engine by including a more intelligent parsing and also learning.

REFERENCES

[1] A. Verma, and S. Kaushal," Cloud computing security issues and challenges: a survey," In International Conference on Advances in Computing and Communications Springer, Berlin, pp. 445-454, July. 2011.

[2] A. Segev and E. Toch, "Context-Based Matching and Ranking of Web Services for Composition," in IEEE Transactions on Services Computing, vol. 2, no. 3, pp. 210-222, July-Sept. 2009.

[3] H. Fethallah, A. Chikh , and A. Belabed. "Automated discovery of web services: an interface matching approach based on similarity measure." In Proceedings of the 1st International Conference on Intelligent Semantic Web-Services and Applications, pp. 1-4, 2010.

[4] M. Klusch and K. Patrick, "isem: Approximated reasoning for adaptive hybrid selection of semantic services," In Extended Semantic Web Conference, Springer, Berlin, Heidelberg pp. 30-44, 2010 .

[5] E. Kovacs, M. Bauer, J. Kim, J. Yun, F. Le Gall and M. Zhao, "Standards-Based Worldwide Semantic Interoperability for IoT," in IEEE Communications Magazine, vol. 54, no. 12, pp. 40-46, December 2016.

[6] G. Cassar, P. Barnaghi, W. Wang and K. Moessner, "A Hybrid Semantic Matchmaker for IoT Services," 2012 IEEE International Conference on Green Computing and Communications, pp. 210-216, 2012.

[7] N. Bnouhanna, R. C. Sofia, and A. Pretschner,, "IoT Thing To Service Semantic Matching," 2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), pp. 418-419, March 2021.

[8] S. Liang, C.Y. Huang, and T. Khalafbeigi. "OGC SensorThings API Part 1: Sensing, Version 1.0.", 2016.

[9] I. Martens, D9.1 - Implementation and Validation through Pilot-1. [Deliverable] https://www.efpf.org/deliverables, 2021.

[10] I. Martens, D9.2 - Implementation and Validation through Pilot-2. [Deliverable] https://www.efpf.org/deliverables, 2021.

[11] I. Martens, D9.3 - Implementation and Validation through Pilot-3. [Deliverable] https://www.efpf.org/deliverables, 2021.