Intent-Based IoT Service Description and Discovery Postprint

Authors: Liu Xingyu, Jiang Lingyun

Date: 2022-05-10T11:22:58+00:00

Abstract

In the IoT service discovery process, users typically express their requirements through their own intentions, whereas service descriptions constitute explanations of service functionalities; consequently, mismatches between these two elements impact the accuracy of service discovery. Simultaneously, as the diversity of services continues to proliferate, the accuracy of service discovery progressively diminishes. To address these challenges, this paper proposes a methodology that incorporates intent service ontology into IoT service descriptions and extends service context and QoS (Quality of Service) within the intent service ontology. The extended intent service ontology is stored in semantic service description OWL-S (Ontology Web Language for Services) files, enabling the expression of service functionalities in an intent-based manner, thereby enriching the semantics of IoT service descriptions and enhancing service discovery accuracy. Simulation results demonstrate that the proposed service description method and corresponding service discovery algorithm achieve a 6.7% improvement in accuracy relative to traditional service discovery methods.

Full Text

Preamble

Intent-Based IoT Service Description and Discovery

Liu Xingyu[1], Jiang Lingyun[1],[2]†
[1]School of Communication & Information Engineering, [2]Institute of Internet of Things
Nanjing University of Posts & Telecommunications, Nanjing 210003, China

Abstract: In IoT service discovery, users typically express their needs through intentions, while service descriptions merely explain functional capabilities. This mismatch between user intentions and service functionality descriptions

negatively impacts discovery accuracy, a problem exacerbated as service varieties proliferate. To address these challenges, this paper proposes incorporating intentional service ontology into IoT service descriptions, extending it with service context and QoS (Quality of Service). The enhanced intentional service ontology is stored in OWL-S (Ontology Web Language for Services) files, enabling service functions to be expressed through intentions, enriching IoT service description semantics, and improving discovery accuracy. Simulation results demonstrate that the proposed service description method and corresponding discovery algorithm achieve a 6.7% accuracy improvement over traditional approaches.

Keywords: intentional service ontology; context; QoS (Quality of Service); service description; service discovery

0 Introduction

IoT technology has become indispensable across industries, extending the Internet by incorporating "things" to connect everyday objects and devices. As IoT technology advances, the proliferation of service types continuously degrades service discovery accuracy, rendering generic description and discovery mechanisms inadequate. This paper proposes improved service description and discovery algorithms to address these limitations.

Ontology forms the foundation of the semantic web, representing a model of concepts and their relationships within a domain. In service description languages, WSDL (Web Services Description Language) [1] suffers from low accuracy due to its keyword-based matching approach, prompting researchers to develop semantic service description languages. Existing semantic service description languages include OWL-S (Web Ontology Language for Service) [2], WSMO (Web Service Modeling Ontology) [3], and WSDL-S [4]. OWL-S facilitates the transition from syntactic-level (WSDL) to semantic-level (OWL-S) service descriptions through ontology introduction, serving as a standard semantic markup language that enables automated processing by machines. Consequently, this paper proposes enhancing service descriptions within OWL-S.

In IoT services, intention represents both the user' s desired goals/actions and the objectives/actions realized by services [5]. Users typically express needs as intentions, while service providers offer only functional descriptions, creating a mismatch during service discovery. By introducing intentional service ontology into OWL-S, this paper describes service functions through intentions, enabling both user needs and service capabilities to be expressed uniformly and resolving the mismatch problem. The intentional service ontology combines service intentions with other information, defining the relationship between intentions and services as one of satisfaction—intentions satisfy services.

As IoT technology evolves, considering only functional (including intentional)

information reduces both recall and precision in service discovery. When two users request functionally identical services, their context and QoS requirements may differ, preventing discovery from identifying truly suitable services. This paper extends non-functional attributes—context and QoS—within the intentional service ontology, linking service context with device context to ensure discovery accuracy. Related work in this domain is discussed below.

Service-Oriented Architecture (SOA) [6] is widely adopted for service communication, composition, and utilization, with applications in IoT as well. For instance, literature [7] proposed a large-scale IoT SOA ontology supporting interoperability, heterogeneity, flexibility, manageability, scalability, and extensibility; literature [8] applied SOA and IoT concepts to smart home lighting for device control. Literature [9] introduced intention into traditional SOA models, presenting the ISOA (Intentional Service Oriented Architecture) model as a foundational framework for intention-based service description and discovery.

Literature [10] surveyed recent advances in intent-based technology, suggesting that rapid natural language understanding developments will enhance adaptability. Literature [11] extended service descriptions by adding intentional service descriptors to SAWSDL (Semantic Annotations for WSDL and XML Schema) [12] files, introducing a framework for publishing and matching intentional services. Literature [13] extracted domain knowledge about service functions (i.e., service goals) from textual descriptions, helping requesters query relevant services through intention information using natural language parsing and semantic similarity-based clustering for service recommendation. Literature [14] presented an intention ontology model for storing user intention information and proposed an intention-based service composition architecture.

While these works introduced intention into service description and discovery, accuracy degrades as service varieties increase and similarly-intentioned services proliferate, making intention-only matching insufficient.

Literature [15] proposed an intention-based proactive service method for drinking service robots, considering user context and environmental factors alongside intentions. Literature [16] introduced a context-aware intentional service discovery mechanism based on OWL-S extensions, centering on user requirements. Literature [17] presented a dynamic web service composition method based on user intention and context, employing AI planning techniques with semantic descriptions. Literature [18] proposed a service composition method incorporating context and user intention to correctly respond to user needs. These approaches consider both intention and context but overlook varying user QoS requirements, leading to mismatched services.

Literature [19] introduced an efficient multi-objective automatic service composition method, balancing computational cost and QoS. Literature [20] comprehensively discussed intent-driven management systems, addressing intent definition, modeling, and end-to-end architecture generation considering QoS performance. Literature [21] proposed a requirement-centric approach using keywords

extracted from ISM (Intentional Service Model) specifications to discover and automatically select high-QoS services. These works consider intention and QoS but neglect context impact, which is crucial as service and user contexts evolve.

Literature [22] used user requirement structures concretized by intention graphs, enriched with context information for service selection based on semantic matching, user context, and QoS. Literature [23] constructed composite services from user intention, context, and QoS information, describing service publication and matching. These approaches consider functional and non-functional parameters but lack non-functional attribute models and corresponding discovery mechanisms.

Most existing literature applies intention, context, or QoS to the Internet domain, whereas IoT applications require consideration of device context information. Additionally, none present complete service discovery mechanisms incorporating all three elements. This paper' s main contributions are: (1) proposing an intention-based service discovery framework with complete processes and matching algorithms; (2) extending the intentional service model with context and QoS classes, integrating IoT device context information into service context.

1 Intention-Based Service Discovery Framework

The proposed service discovery framework, illustrated in Fig. 1, involves four entities: service description, service registration, service matching, and language parsing. Service descriptions are stored as OWL-S files. Service registration enrolls provider information in a UDDI repository containing all provider offerings. Service matching identifies services satisfying user needs from the registry. Language parsing converts user requirements into intention, context, and QoS formats using Stanford natural language processing.

The discovery process proceeds as follows: providers supply OWL-S files registered in the UDDI repository. Upon user request, Stanford NLP parses requirements into intention, context, and QoS information for service matching. Matching first performs intention matching (functional attribute matching) against the registry, producing a similarity-ranked service list. Non-functional attribute matching—context and QoS matching—follows. Since service context and QoS are computed in real-time, UDDI database information is updated concurrently. Post-matching, service description files are returned, enabling users to bind with providers and invoke services. This paper details the service matching process and proposes adding intention, context, and QoS information to OWL-S service descriptions.

2 Intention-Based Service Description

Intention-based service description manifests in the service description component of the framework, storing intention, context, and QoS information within OWL-S through intentional service ontology introduction.

2.1 Intentional Service Ontology with Context and QoS

The Intentional Service Model (ISM) [24], originally proposed by R.S. Kaabi, stores service information ontologically with intentions representing service functions. This paper extends ISM with context and QoS, proposing the CQISM (Context and QoS Intentional Service Model) ontology (Fig. 2). Ontology-based representation is chosen because extending service descriptions requires no changes to existing files or parsing tools while standardizing IoT intentional service description conventions.

CQISM design considers three aspects: (1) standardizing service information including basic details (intention, I/O parameters); (2) adding service context and QoS for IoT, storing availability location, status, associated devices, etc.; (3) including a service composition class for computing aggregated service context and QoS. Accordingly, CQISM comprises five classes:

a) **Intention Class**: Represents service intention information parsed from descriptions, consisting of verb, target, and optional parameters.

b) **Service I/O Class**: Comprises pre-condition and post-condition classes and their states, including initial situation, final situation, preconditions, and postconditions.

c) **Service Composition Class**: Indicates service type—atomic or aggregate—where aggregate services combine atomic or other aggregate services.

d) **Service Context Class**: Contains service time, status, location, and associated devices.

e) **Service QoS Class**: Includes availability, security, scalability, etc.

In IoT, CQISM stores service intentions while considering non-functional attributes, improving discovery accuracy through intention, context, and QoS matching. For example, in UAV inspection systems, services include video/image transmission and path control, but the UAV' s environment and status constantly change. Thus, discovery must consider both functional and non-functional attributes including service context, device context, and QoS.

The following sections detail CQISM classes.

2.1.1 Intention Class Intentions comprise words or phrases maximally expressing service functions or user needs. The intention model (Fig. 3), originally from N. Prat [25], includes: verb (dominant action), target (most relevant noun), and optional parameters (direction, beneficiary, time, manner, quality, quantity, location, etc.).

In UAV inspection systems, the service "TransmitImageByWIFI" transmits image information via WIFI. Input is null, output includes image information and capture time. The intention is expressed as verb (Transmit), noun (Image), and parameter (By WIFI), where the parameter belongs to the manner category.

Verb, target, and parameter extraction uses Stanford CoreNLP' s English dependency parsing to obtain word relationships and generate intention expressions.

2.1.2 Service Composition Class Service composition divides into atomic and aggregate services. Atomic services are indivisible minimum units stored as Service_{name} in the Atomic class. Aggregate services combine atomic or other aggregate services, with expressions stored as Service_{expression} in the Aggregate class. Expression generation requires MAP graphs and composition operators.

1) MAP Graphs: A MAP graph is a labeled directed graph whose minimal unit (Fig. 4) contains Start (source intention), Stop (target intention), and Strategy (edge). MAP graphs represent all atomic or aggregate services, where aggregate services decompose into sub-aggregate services until fully resolved into atomic services. Aggregate service expressions are derived from all paths in the MAP graph.

In UAV inspection systems, the aggregate service "PerformInspection" executes inspection tasks. Its MAP graph (Fig. 5) shows that performing inspection requires first checking remaining battery level via a voltmeter, then collecting information through either non-designated or designated position strategies.

For the aggregate service "CollectInformationByDesignatedPosition" (collecting information at a given position), the MAP graph (Fig. 6) shows the UAV must first fly to the position, then transmit position and image information. Position transmission uses absolute or relative positioning; image transmission uses OFDM or WIFI.

2) Service Composition Operators: Services combine through operators to generate new aggregate services, classified as composite or variant services.

Composite services execute sequentially or in parallel. Sequential composite services (operator "•") require ordered execution, as shown in Eq. (1) for Fig. 6. Parallel composite services (operator " ") execute simultaneously without order constraints, as shown in Eq. (2).

Variant services include three types: selective, alternative, and multi-path. Selective variant services (operator " ") offer non-exclusive strategies—e.g., Fig. 6' s position transmission allows choosing absolute, relative, or both positions (Eq. 3). Alternative variant services (operator " ") have mutually exclusive strategies—e.g., Fig. 5' s information collection must choose either non-designated or designated position (Eq. 4). Multi-path variant services (operator " ") achieve the same intention through multiple paths, simplifying context and QoS computation for aggregate services.

3) Aggregate Service Expression Generation: OWL-S Process Model contains all process information for extracting MAP graphs, which completely represent complex aggregate services. Expressions are generated from all MAP

graph paths. Since IoT services are predominantly aggregate and their context/QoS computation requires sub-service and device information, expressions enable context/QoS calculation. Tables 1 and 2 show expressions for Figs. 5 and 6 (asterisk "*" indicates repeated service invocation), listing atomic services and aggregate service expressions.

2.1.3 Context and QoS Classes Traditional OWL-S-based semantic descriptions and intention-based descriptions match services by functionality. To improve IoT discovery accuracy, this paper additionally considers non-functional attributes: context and QoS classes.

1) Context Class: Unlike traditional web services, IoT services involve sensors, actuators, and other devices with tight service-device coupling. Thus, the IoT service context class adds the SC_{attachedDevice} field (Fig. 7). Atomic service context and single device context are provider-supplied; aggregate service context includes sub-service context and associated devices.

Fig. 7 shows the IoT service context class with: SC_{ID} (unique identifier), SC_{type} (atomic/aggregate), SC_{status} (availability), SC_{time} (invocable timeframes), SC_{position} (invocable locations), and SC_{attachedDevice} (associated devices). During context matching, device context (stored in device ontologies, Fig. 8) updates service context in real-time. The device ontology model includes static attributes (fixed information) and dynamic attributes (energy, status, location, time). Capabilities include communication, control, events, and presentation. Dynamic attributes affect service context—e.g., service status (SC_{Status}) is available only when associated devices have sufficient energy and are operational.

2) QoS Class: Considering diverse user QoS requirements (e.g., high security priority), the CQISM extends ServiceQos class (Fig. 9, based on IoT QoS metrics from [26]). QoS includes availability, performance, security, scalability, reliability, interoperability, and accessibility, qualitatively rated as "1" (very low), "2" (low), "3" (high), "4" (very high) to unify provider-user representations. Each metric's level is provider-defined. Atomic service QoS is provider-supplied; aggregate service QoS derives from sub-service QoS.

2.2 OWL-S Service Description File Extension

OWL-S files comprise three main parts: Service Profile (what the service does—name, description, QoS, provider), Process Model (how to use the service and its results), and Grounding (how to access the service—protocols, message formats, ports) [27].

Extension involves adding CQISM ontology to base semantic descriptions, as shown in Fig. 10. CQISM is added to the ServiceProfile section. Since OWL-S stores atomic and aggregate service information in ProcessModel, the composition relationships are extracted to generate MAP graphs and service composition expressions, stored as ServiceComposition class in CQISM. The extended

description enriches OWL-S semantics, resolves concept mismatches between provider descriptions and user needs, and improves discovery accuracy by considering non-functional attributes.

3 Intention-Based Service Matching

Intention-based service matching operates in the service matching component of Fig. 1, retrieving services from the UDDI registry through functional and non-functional attribute matching.

Before matching, CQISM ontology is extracted from OWL-S files to parse intention, context, and QoS information into the service registry. Upon user input, Stanford NLP first parses user intentions for intention matching (Algorithm 1). Post-matching, services are similarity-ranked, and the top candidates undergo context and QoS matching.

Services include atomic and aggregate types with various IoT devices, requiring context computation and matching (Algorithm 2). For atomic services, context correlates with device context. Device context computation follows: (a) character attributes (status, location, time) use intersection ( )—for j associated devices (Device_1···Device_j) with attribute q values vj, the atomic service' s q value is v_1 v_2 ··· v ; (b) numeric attributes (energy, security) use minimum values—v = min(v_1, v_2, ···, v). For example, two devices at positions NJSH and NJ yield atomic service position NJ.

For aggregate services, context derives from sub-service contexts per the aggregate expression. Composite services (operators " • " or " ") use intersection and minimum calculations. Variant services (operators " " , " " , or " ") use union and maximum calculations. When expressions involve nested aggregate services, rules (a) and (b) apply recursively until all sub-services resolve to atomic services.

Real-time context computation involves only similarity-ranked services (top 100) to balance accuracy and performance. Algorithm 2' s pseudocode (lines 14-20) selects services matching user context requirements.

QoS matching precedes similarly. Atomic service QoS is provider-supplied; aggregate QoS computation mirrors context computation but excludes associated devices.

4 Experiments

4.1 Development Environment

The implementation uses Python (cross-platform, running on Windows, Mac, and Linux/Unix) with Anaconda' s Jupyter notebook. Ontology construction employs Protégé-5.5.0 for CQISM development. Stanford NLP handles intention extraction. The test dataset is OWLS-TC_{v2}.2_{revision}3, a collection spanning healthcare, education, food, and industry domains.

4.2 Service Description File Extension

Using a UAV image transmission service via WIFI as an example, Table 3 shows partial OWL-S content after CQISM extension. `<profile:CQISMPath>` stores the CQISM ontology path; `<profile:CQISMIRI>` stores the IRI (Internationalized Resource Identifier). Both uniquely identify the intentional service ontology for Python-based OWL-S parsing and CQISM extraction.

Parsing CQISM yields intention, context, aggregate expressions, and QoS for matching. Table 4 shows sample CQISM data: lines 3-5 contain intention information (verb, target, parameters); lines 9-10 show context (availability status "1" = available, associated device); line 14 shows QoS (reliability level 4).

IoT context matching requires device context. Table 5 shows a camera device (Camera_1) ontology excerpt, with line 7 indicating 80% remaining energy.

4.3.1 Service Discovery Accuracy

First, we validate intention-only vs. function-only impact on accuracy. Using identical requests across varying service counts (functional information only), accuracy is computed via Eq. (5). Results (Fig. 11) show intention-based descriptions consistently outperform function-based descriptions, with function-based accuracy declining sharply as service count increases. Averaging across 200, 400, and 800 services yields 77.5% accuracy for function-based vs. 88.3% for intention-based—a 10.8% improvement, confirming intention extraction' s feasibility and effectiveness.

In IoT discovery, considering only intentions or functions is insufficient. Traditional matching (e.g., OWLS-MX [28]) combines function with I/O information. Intention-based matching adds context and QoS (e.g., [16] intention+context, [21] intention+QoS). Fig. 12 compares OWLS-MX, [16], [21], and our approach (intention+context+QoS). Averaged across 200, 400, and 800 services, accuracies are: intention+context+QoS (84%), OWLS-MX (77.3%), intention+context (80%), intention+QoS (73%). Simultaneously considering functional and non-functional attributes improves discovery accuracy.

4.3.2 Service Discovery Time

Table 6 compares discovery times for intention-based vs. OWLS-MX matching at 200, 400, and 800 services (10 requests shown). Time units are seconds (R_1 = first request). Average discovery time is computed via Eq. (6). Fig. 13 shows both methods' times increase with service count, but intention-based matching requires more time due to more parameters and the extra intention extraction step. Averaged across 200, 400, and 800 services, OWLS-MX takes 7.0951s vs. 8.8865s for intention-based—an average overhead of 1.7914s.

After intention matching, a similarity-ranked list is generated. Using the top 100 services for context and QoS matching, Table 7 shows query times for 20

requests. Averaged context and QoS times are 0.268015s and 0.06888s respectively—negligible overhead relative to intention matching time (Fig. 13) while significantly improving accuracy.

5 Conclusion

This paper proposes an intention, context, and QoS-based service description method and matching algorithm. CQISM ontology stores service intention, context, and QoS information within OWL-S files for matching. Intention extraction resolves description-need mismatches; context and QoS incorporation further improves accuracy. Using ontology standardizes intentional service description without altering existing OWL-S parsing tools. Comparative experiments with [16] and [21] demonstrate that simultaneous context and QoS consideration improves accuracy within acceptable timeframes.

Future work will focus on improving matching algorithms, standardizing discovery mechanisms, reducing intention-based discovery time, and enhancing IoT applications.

References

[1] Chinnici R, Moreau J J, Ryman A, et al. Web Services Description Language (WSDL) version 2.0 part 1: core language [EB/OL]. (2007) [2022-01-15]. http://www.w3.org/TR/2007/REC-wsdl20-20070626.

[2] Martin D, Burstein M, Hobbs J, et al. OWL-S: semantic markup for web services [J/OL]. W3C member submission, 2004, 22(4). (2004-11-01) [2022-01-15]. http://www.daml.org/services/owl-s/1.1/overview/.

[3] De Bruijn J, Bussler C, Domingue J, et al. Web Service Modeling Ontology (WSMO) [J]. Interface, 2005, 5(1): 50.

[4] Akkiraju R, Farell J, Miller J A, et al. Web service semantics-WSDL-S [EB/OL]. (2005-11-07) [2022-01-15]. https://www.w3.org/Submission/2005/SUBM-WSDL-S-20051107/.

[5] Ponce V, Abdulrazak B. Intention as a context: an activity intention model for adaptable development of applications in the Internet of Things [J]. IEEE Access, 2021(9): 151167-151185.

[6] Perrey R, Lycett M. Service-oriented architecture [C]// 2003 Symposium on Applications and the Internet Workshops. Piscataway, NJ: IEEE Press, 2003: 116-119.

[7] Mishra S K, Sarkar A. Service-oriented architecture for internet of things: a semantic approach [J/OL]. Journal of King Saud University-Computer and Information Sciences, 2021, 2021(24): 1-12. (2021) [2022-01-15]. https://doi.org/10.1016/j.jksuci.2021.09.024.

[8] Irawan Y, Linarta A, Febriani A, et al. Smart home light based service oriented architecture and IoT [J]. Journal of Physics: Conference Series, 2021, 1845(1): 012070.

[9] Rolland C, Kirsch-Pinheiro M, Souveyet C. An intentional approach to service engineering [J]. IEEE Transactions on Services Computing, 2010, 3(4): 292-305.

[10] Zeydan E, Turk Y. Recent advances in intent-based networking: a survey [C]// 2020 IEEE the 91st Vehicular Technology Conference. Piscataway, NJ: IEEE Press, 2020: 1-5.

[11] Aljoumaa K, Assar S, Souveyet C. Publishing intentional services using extended semantic annotation [C]// 2011 the 5th International Conference on Research Challenges in Information Science. Piscataway, NJ: IEEE Press, 2011: 1-9.

[12] Kopecky J, Vitvar T, Bournez C, et al. Sawsdl: semantic annotations for wsdl and xml schema [J]. IEEE Internet Computing, 2007, 11(6): 60-67.

[13] Zhang Neng, Wang Jian, Ma Yutao, et al. Web service discovery based on goal-oriented query expansion [J]. The Journal of Systems and Software, 2018, 142: 73-91.

[14] Daosabah A, Guermah H, Nassar M. PDDL planning and ontologies, a tool for automatic composition of intentional-contextual web services [M]// Ouaissa and Mariya, Computational Intelligence in Recent Communication Networks. Switzerland: Springer, Cham, 2022: 163-190.

[15] Hao Man, Cao Weihua, Wu Min, et al. An initiative service method based on fuzzy analytical hierarchy process and context intention inference for drinking service robot [J]. IEEE Transactions on Cognitive and Developmental Systems, 2018, 11(2): 221-233.

[16] Najar S, Pinheiro M K, Souveyet C, et al. Service discovery mechanism for an intentional pervasive information system [C]// 2012 IEEE the 19th International Conference on Web Services. Piscataway, NJ: IEEE Press, 2012: 520-527.

[17] Daosabah A, Guermah H, Choukri I, et al. Integrating context and intention for optimal semantic web service composition using AI planning [C]// 2021 the 4th International Conference on Advanced Communication Technologies and Networking (CommNet). Piscataway, NJ: IEEE Press, 2021: 1-9.

[18] Daosabah A, Guermah H, Nassar M. Dynamic composition of services: an overview of approaches led by the context and intent of the user [C]// Proceedings of the 4th International Conference on Big Data and Internet of Things. New York: ACM Press, 2019: 1-8.

[19] Wang Zhaoning, Cheng Bo, Zhang Wenkai, et al. Many-objective automatic

service composition based on temporal goal decomposition [J]. IEEE Transactions on Network and Service Management, 2021, 18(3): 2765-2779.

[20] Mwanje S S, Banerjee A, Goerge J, et al. Intent-driven network and service management: definitions, modeling and implementation [J/OL]. TechRxiv, 2021, 2021(1): Article ID 17075450. v1. (2021-12-06) [2022-01-15]. https://doi.org/10.36227/techrxiv.17075450.v1.

[21] Driss M, Moha N, Jamoussi Y, et al. A requirement-centric approach to web service modeling, discovery, and selection [C]// International conference on service-oriented computing. Berlin, Heidelberg: Springer, 2010: 258-272.

[22] Fki E, Tazi S, Drira K. Automated and flexible composition based on abstract services for a better adaptation to user intentions [J]. Future Generation Computer Systems, 2017, 2017(68): 376-390.

[23] Khanfir E, Djmeaa R B, Amous I. Automated publish, discovery and composition of intentional web services adaptable to both quality and context [C]// 2016 IEEE the 18th International Conference on High Performance Computing and Communications. Piscataway, NJ: IEEE Press, 2016: 639-646.

[24] Kaabi R S. A methodological approach for modeling and operationalizing intentional services [D]. Paris: Universite of Paris 1 La Sorbonne, 2007.

[25] Prat N. Goal formalisation and classification for requirements engineering [C]// Requirements Engineering: Foundation for Software Quality. Paris: archive ouverte HAL, 1997: 145-156.

[26] Singh M, Baranwal G. Quality of service (qos) in internet of things [C]// 2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU). Piscataway, NJ: IEEE Press, 2018: 1-6.

[27] GAO Yixiang. Research and design of QoS simulation analysis tool for composite web services based on OWL-S [D]. Harbin: Harbin Institute of Technology, 2016.

[28] Klusch M, Fries B, Sycara K. OWLS-MX: A hybrid semantic web service matchmaker for OWL-S services [J]. Journal of Web Semantics, 2009, 7(2): 121-133.

Note: Figure translations are in progress. See original paper for figures.