An adaptive user-centric IoT service composition framework

Rawan Sanyour, Manal Abdullah and Salha Abdullah

Information System Department, Faculty of Computing and Information Technology, King Abdul Aziz University, Jeddah, Saudi Arabia

E-mail: Rsanyour0001@stu.kau.edu.sa, Maaabdullah@kau.edu.sa, Salhaabdullah@yahoo.co.uk

Abstract

Users need to utilize several services to efficiently fulfill their requirements. These required services can be integrated to form service compositions by which value-added services can be created to meet the diverse users' requirements. However, in the dynamic IoT environment, creating on demand services driven by end users is a challengeable task. Personalizing service delivery through an adaptive, on- demand integration of available services requires the support of user-centric service composition approaches to spontaneously deliver the required functionalities. This research aspires to personalize IoT service delivery according to the evolve changing of end users' requirements. Motivated by this aspiration, an adaptive user-centric IoT service composition framework is proposed. It allows the user to discover, select and interconnect services on demand at the runtime. The framework provides a flexible and multi-step interaction between the user and the system. It consists of four modules: service discovery, service selection, service composition and service execution. The suggested services list will be filtered based on some selection criteria such as user's profile and Quality of Experience (QoE), Quality of Service (QoS) parameters and environmental context. Filtering services according to these factors can be considered as a Multi objective Optimization Problem MOP with constraints. Optimization techniques such as Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) can be used to find a solution for this type of optimization problems. Several scenarios in different application domains would be conducted to evaluate the applicability and performance of the proposed framework. The applicability will be tested through evaluating the ratio of successful compositions as the context and requirements changing. The performance will be evaluated by measuring the changes in computation time as the number of the candidate services increase. Several simulation environment platforms such as SimpleSoft, NetSim and IoTIFY could be utilized to test and evaluate the performance of the proposed framework.

2010 Mathematics Subject Classification. **62R07**. 68T01, 68M15.
Keywords. IoT, IoT services, service composition, user-centric service composition, adaptive service composition.

Introduction

Service Oriented Computing SOC is a paradigm that packages heterogenous resources as services to be discoverable, accessible and reusable through providing unified services interfaces to ease users' access via communication networks (Chen, 2016). It allows a seamless integration of single function services to create value added applications. Finding the set of relevant services among an extensive number of available services has become a significant issue in the service computing field and has been attempted to be addressed by several research proposals (Cao et al., 2019).

For IoT-enabled services, With the growing number of embedded wireless sensors, IoT evolution has introduced a new level of human-machine interaction which enables to provide a pervasive

Advanced Studies: Euro-Tbilisi Mathematical Journal 16, supplement issue 2 (2023), pp. 135–158.
DOI: 10.32513/asetmj/1932200823211

Tbilisi Centre for Mathematical Sciences.

Received by the editors: 15 November 2022.
Accepted for publication: 15 January 2023.

intelligence. Such intelligence allows the user to select, interact and choose the kind of service to utilize and data to include in the value co-creation process.

Often, services from different applications and domains are required to be merged to meet some complex requirements that cannot be satisfied by a single service. In contrast to the traditional service delivery approaches, wherein service compositions are pre-defined without taking into account the specific characteristics of the end user, in the proposed approach, involving the user as an active participant to personalize service delivery is allowed at the runtime through the so-called user centric or self-acting service composition. this process of personalizing service experience should be performed on demand, on the fly (at the runtime) and based on the demands of a specific service user.

Service users are heterogeneous, they have different background, different knowledge of application domains, different knowledge of the services available, different technical skills to interact with the system and use different device in different context. This diversity in users and their characteristics requires such requirements to be fulfilled in a dynamic adaptive fashion.

However, due to the dynamicity and heterogeneity nature of the IoT environment and its resource constraint devices in terms of energy and processing capabilities, services offered by IoT resources cannot be composed by simply applying existing traditional web service composition approaches. Instead, a shift from software services to real world services, from application centered services to user centric services has become an urgent demand

Combining existing services to provide new ones is a major requirement to gain the full potential of IoT benefits. Service composition is the process of integrating and providing high level functionalities from available atomic services to fulfill users' complex requirements. It can be considered as a middle to fill the gap between the users' high-level specified requirements and the functionalities provided by the different services embedded within the surrounding environment (Yachir et al., 2012). According to (*IoT 2019 in Review: The 10 Most Relevant IoT Developments of the Year*, n.d.), currently, there are about 9.5 billion connected devices and it is predicted to increase exponentially by the end of 2025 to reach 28 billion connected devices. This rapid increasing in the number of these connected devices has led to the current excessive increase in the number of IoT services that we are witnessing today. As a result, this field supports great research opportunities and brings new vision of intelligent services that can be efficiently created "on the fly" and provided smoothly to the targeted users anytime and anywhere.

Due to the dynamicity and the uncertain nature of IoT environment alongside the heterogeneity of the networks, devices, and services, composing services in IoT has become a challenging issue. There are several reasons for which this type of environments is always in a changing status: services could be joining or leaving at the runtime, new services with same functionalities and different qualities are detected, a specific service in use fails, changes in users' needs or context, or loss of services due to mobility issues.

Given this dynamicity nature, Chen et al. (Chen, 2016) have argued that such a composition process faces some significant challenges such as the flexibility issue of composite services since that services are independently provided, deployed and maintained by different hosts, so it is a challenging task for these hosts to collaborate and reduce composition failure. Another significant challenge is to provide adaptable composites that have the ability to adopt to environmental and communication

changes and to avoid service link loss during service execution.

This research aspires to develop a user centric IoT service composition framework that can effectively respond to the various users demands in real time and adaptively changing its services upon the changeable context information. Although several works, that address the issues of composing and integrating services in IoT environment, have been conducted, there is no final satisfactory solution exist that integrates the various standards proposed over the years. Based on the reviewed literature, it is worth noting that even though some requirements and challenges are relatively well tackled, several others such as involving users as a major actor in the service composition process, selecting and composing services on the fly (at the run time) under uncertainty and changing environments, providing monitoring mechanisms to ensure fault tolerance are still in the premilitary stages, thus, there is evidently a substantial room for future work in this area.

Research questions

Based on defined problem, the key research question in this work is: How to provide a user centric IoT service composition framework to accomplish personalized service composition and delivery. This question can be subdivided into the following research questions:

RQ1: How to support different types of users, i.e., users with different characteristics, requirements and technical skills?

RQ2: How to personalize processes such as service discovery, selection and composition to fulfill uses' changing demands?

RQ3: What are the mechanisms that required to provide an adaptive user centric service composition process, so that users with different requirements and environmental context can drive such process at the runtime without reallocating the whole process?

Literature review

With the rapid growth of information and communication technologies, beside the increased usage of external services by entities at multiple scales, the global spanning of service-based business models has changed the way the world work. Since that service innovations are now a key priority for businesses and nations, the requirement of an interdisciplinary science of service that can significantly provide more sustainable and systematic view has become an urgent demand (Maglio & Spohrer, 2013).

The concept of service science was firstly introduced at a seminar hosted by IBM and UC Berkley team in 2002 to study service from the social engineering perspective. In 2004, when this concept has been promulgated in a report published by the American Competition Council, it has been gained a comprehensive attention (Wang, 2010). At this stage, service science started as an idea about studying the various characteristics of services through integrating diverse theories and methods from multiple disciplines such as marketing, engineering, computer science, information system and more to build a scientific foundation for service innovation.

Service science can be recognized as the study of service systems, which can be treated as a set of dynamic co-creation configurations of resources including people, technology, organizations and shared information, in order to create a systematic service innovation. It integrates both human and organization understanding with business and technological understanding to demonstrate how

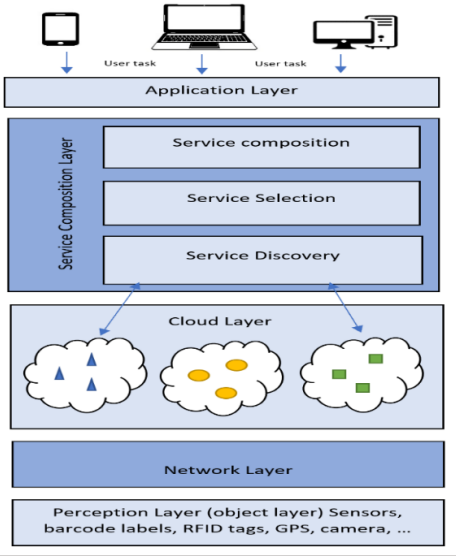
various service systems interact and evolve to create value (Maglio & Spohrer, 2008). In information system, the service science paradigm was introduced as an umbrella for the scientific study of service systems to support and enhance service innovations (Chesbrough & Spohrer, 2006).

SOA-based architecture for IoT systems

The existing Service Oriented Architectures SOAs, that were specifically developed to provide web services, are designed to be utilized by PC devices, thus, they are not suitable to be exploited for IoT resource constraint embedded devices. This implies the urgency of for automated and adaptive discovery of devices and services and their dynamic management (L. Liu et al., 2012).

Asghari et al. (Asghari et al., 2018) and Sosa-Reyna et al. (Sosa-Reyna et al., 2018) have proposed the generic architecture of service composition in IoT. The main requirement of the IoT architecture is that things in the network should be interconnected. In addition, this architecture should play as a bridge between the physical world (things) and the virtual world of IoT. As illustrated in figure1, the architecture consists of five different layers:

- ***The perception layer (object layer):*** in includes sensors, and various smart devices to automatically detect, collect and exchange data within the IoT environment.
- The network layer: the fundamental role of this layer is to process and transfer the data gathered by the perception layer. It consists of the infrastructure that support both wired and wireless connection between things, allows to manage the communication in the environment and transmit messages between the objects and the system. This layer is crucial to any IoT approach due to its essential role in energy management, data processing, security and privacy concerns and QoS functionalities.
- ***The cloud layer:*** provides various subservices available by several private or public cloud.
- ***The service composition layer***: it is responsible for composing several sub services together according to user's functional and non-functional requirements. It receives the user's request, discover the available services based on his demand from one or more clouds in the cloud layer and finally combine the selected services to compose the required composited services to be delivered through the application layer. This layer could be based on middleware technologies which are essential for consuming and executing services in IoT applications. A middleware layer is considered as a suitable solution to guarantee both scalability and interoperability in this kind of systems. it operates as an interface between the object layer and the application layer and responsible for several functions including device and information management, data filtering and aggregating, semantic analysis and information discovery.



- ***The application layer:*** deliver a particular composited service to end users based on their requests.

Figure 1. IoT Service-oriented Architecture (Asghari et al., 2018)

In service-oriented architecture, services are integrated and composed using two different techniques: orchestration through a centralized controller by which a single centralized service controls the execution of other participating services in the composition. It defines an explicit control flow structures to efficiently coordinate the invocation of service operations and it can be centralized or decentralized. A centralized orchestration has a full control over all composed services while a decentralized orchestration identifies sub workflows to collaboratively exchange the control flow over the network. The other composition mechanism is the choreography in which the participating services are allowed to execute partial composite service in a peer-to-peer fashion. Unlike the orchestration, it relies on a protocol that defines a "service conversation" through a decentralized interaction (Arellanes & Lau, 2020)(Dar et al., 2011).

Services Composition in IoT

Since the fact that most of the network-based applications today are utilized as services, combining different services in order to offer value-added services has become one of the most effective and significant ways to response to the users' rapid increasing demands. Composing and reusing existing services and defining a process as a workflow of abstract services to realize different requirements is a powerful mechanism.

According to Beradi et al. (Berardi et al., 2004), service composition is a paradigm in which the functionality of multiple services is combined within a single process in order to answer a complex request that a single service could not satisfy. It provides the ability to accomplish complex activities from available services. these composed services form a new service that can be reused in another composition.

Recently, service composition in IoT and pervasive environment is becoming as one of the active research domains that has received widespread attention in the research community. It aims to provide smooth seamless access to a wide variety of high level and complex functionalities through combining existing services. service composition approaches in IoT can be considered as a "self-motivated" connection between the complex business processes and the suitable atomic service technologies. The convergence of bridging the small islands of connected smart devices with the cyber world of enterprises applications in different domains such as logistics, healthcare, industries, and smart homes have become more and more complicated due to the large number of tiny services offered by the IoT smart devices (Dar et al., 2011). These services cannot be composed using traditional service composition approaches due to the fact that they require a shift from software application centered services to real world, user centric and situation aware services.

In IoT environment, services can be considered as virtual representations of the "behaviour of objects", thus, they can be integrated to perform more complex behaviors, and create complex service oriented IoT systems. The major difference between traditional web services and IoT services is that IoT services are provided by embedded systems and directly related to the real physical world. Each device in IoT provides its functionality as an atomic service. These devices reside in a highly dynamic environment which can significantly affect their performance (Yang et al., 2015). For any given IoT service, it can be either an atomic or composite service. An atomic service is a self-contained part of behavior that cannot be divided to smaller pieces of services. In contrast, a composite service is a complex unit that composed of atomic or other composite services and provides value-added functionalities that cannot be gained by a single atomic service (Arellanes & Lau, 2020)(Bell, 2009).

In IoT environment, several intelligence services that adopted in applications are in the form of service composition. Reviewing the literature, it has been noticed that service composition, as an extensive research area, has carried out on several aspects including domain and service ontologies, objective assessment and quality of service oriented service composition (Ben Mokhtar et al., 2007). Features such as real time sensing, highly concurrent and independent collaboration in addition to issues related to the limited capabilities of IoT resources have raised several challenges regarding service computing. Despite the fact that some of such requirements and challenges are relatively well addressed by several studies in the literature, others are still in the preliminary stages such as adaptive service composition, context and quality of services, and service selection under uncertainty and changes (Yachir et al., 2012).

Combining existing services to provide new ones in the pervasive environment fills in the gap between the high-level users requirements and the functionalities of the numerous available services that embedded in the physical world. It brings new visions of intelligent services by providing on the fly seamless services to users according to their context and requirements anytime and anywhere. However, composing services in such changing environment is a challenging issue due to its dynamicity and uncertainty, as well as the heterogeneity of networks, devices and services. The state of IoT environment is changing frequently for several reasons: services are continuously joining and leaving at the runtime, new devices offering services with better quality are detected, a service in use fails, changes in users' requirements and context, service loss due to the mobility of the user or the device. In IoT, each atomic service is related to a specific device that provide a specific IoT functionality. In the most cases, this single service is not sufficient enough to accomplish

user's complex tasks, thus cannot satisfy his compound requests. In such case, composite service is required.

Unlike the traditional enterprises services systems, IoT services systems require an interaction of billions of services as a response to the rapid growth of the number of connected devices. As a result, scalability has become a crucial concern (Arellanes & Lau, 2020). Scalability is the ability of the system to handle the increasing workload. In terms of IoT, scalability is regarding the number of services composed in the system which is referred to as functional scalability. Existing service composition mechanisms are developed primarily for composing and integrating static business services and not suitable for the real world. For this reason, such mechanisms may not efficiently address the scalability challenges that IoT systems pose.

IoT services composition techniques

During the last decade, although there were several proposed works that are widely investigating service composition in Web services, however, there is a lack of works that tackle the issues regarding service composition in IoT. For this reason, Hamzei and Navimipour (Hamzei & Jafari Navimipour, 2018) explore the available IoT service composition techniques and methods using a systematic matter. In their investigation, they have classified the approaches into different categories. These categories are framework-based approaches, service-oriented architecture approaches, and model-based approaches. According to this classification, various parameters, benefits, and drawbacks of each of these categories were discussed. They systematically presented various IoT service composition techniques, provided recommendations to address some of their issues and determined some potential future research areas.

Framework based approaches

Framework based methods are the methods that utilize a collection of values, theories and practices to discover, select and compose IoT services in order to fulfill users' requests. In their work, Bossi et al. (Bossi et al., 2014) emphasized on the significance of selecting the appropriate services among the tremendous amount of available services. They proposed a framework that measure the reputation of the candidate services to determine their suitability to accomplish a specific task. In addition to the trustworthiness or service reputation, parameters such as availability and functionalities are also considered as well. Despite that their work has been proved as a power approach, it was computationally extensive, thus, it is not suitable for big environment. Furthermore, it didn't consider the dynamicity of the surrounding environment. Khodadadi et al.(Khodadadi et al., 2015) proposed an architecture that consider the role of humans in the process of discovering and exposing IoT objects and services. Their approach was based on facilitating the communication between IoT objects through using standard human/machine readable files by which advertised services are easily be found using standard RESTful web APIs. Their main objective was to provide easy and effective process design, service reusability, openness and security at the same time. The role of humans here is to design the service flows (by domain experts) and then, provide them to be utilized by normal users applications. What distinguishes this work is its two-phase discovery component whereby a syntax-based discovery approach was used to discover entities with specific properties and at the same time having services that match some provided keywords pattern. Performing search processes by matching the data against the used keywords is fast and doesn't require previous knowledge about the underlying data.

Ko et al. (Ko et al., 2016) proposed a user centric IoT-based service framework by which services provided by IoT sources are efficiently integrated in an urban computing environment to accomplish a user task. In their task-oriented framework, tasks are suggested to the user based on the available IoT objects and the contextual data in his surrounding environment. In their proposed framework, Ko et al. used an ontological model to fill the gap between users' goals and IoT services functionalities by representing and analyzing both users goals and contextual data. The final integrated services, that were selected based on a task template, were examined to ensure some predefined QoS criteria. Results indicated that although the proposed framework significantly decreased the computation time and provided high scalability, it required extra efforts to develop the ontologies. Furthermore, it didn't consider the availability and cost as a significant evaluation metrices.

In order to consider the difficulty of selecting services at different IoT nodes due to the issue of information availability, inefficiency and overload, Li et al. (Li et al., 2014) proposed a distributed consensus decision-making method for service discovery, composition, and processing in the IoT. IoT service composition was first tackled by minimizing the multiparameter matching value. Then, a cluster-based distributed algorithm by which a consensus decision for IoT services at the edge nodes was proposed. This work addressed the demand of consensus decision making for IoT services that are provided by resource constraint devices. A three-layers framework (application layer represents the business process components, network layer in which data communication and processing functionalities are provided, and sensing layer that used to record, monitor and collect measurements and observations) was provided to discover, select and compose services in IoT nodes. Based on the application layer requirements, the consensus algorithm is used to select the appropriate services and make a robust universal consensus. Although service clustering significantly enhanced the service discovery process, increased reliability and lower the cost, the proposed approach provided limited performance due to the lack of discovering semantic relations between services with different description models. In the literature, there were some proposed works such as (Baker et al., 2017) that provided energy aware composition strategies through minimizing the number of services utilized to meet users' requirements. Such works has shown significantly low execution time, however, other metrics such as scalability and reliability were not evaluated for this work.

Model based approaches

Model based approaches usually rely on computational models to simulate the required actions. in the context of IoT, these models are used to describe components, interactions, data structure, business models and user requirements. There are several proposed model-based service composition techniques. For example, Lui and Tong (J. Liu & Tong, 2010) have proposed a dynamic service selection model which has the ability to be adaptive to tackle the flexibility and resource-constrained nature of IoT devices. They provided a representation of IoT resources context model by classifying the context into three different models (object-to-Internet-to-human, human-to-Internet- to-object, and object-to-Internet-to-object). Their model context was used in service match and composition processes. Furthermore, the model significantly decreased both cost and resources consumption. Although it had the ability to be adaptive according to resources parameters in the variant IoT context, the proposed work did not evaluate QoS parameters nor simulate the algorithm. Yang et al. (Yang et al., 2014) have exploited the idea of service-oriented composition to develop a decentralized coordination technique to be used in IoT logistic service composition processes. They have considered several QoS parameters such as reliability, availability, and high user satisfaction rat

for evaluating logistics services. However, low scalability is one of the drawbacks of this mechanism. In addition, the model was not evaluated in real world IoT logistics services system.

Park et al. (Park et al., 2016) have utilized the concept of middleware to provide reliable and efficient IoT services. The middleware structure supports integrating information from different kinds of IoT devices and provide an abstract layer that can maintain and process both users' information and devices status. Therefore, a specific level of interoperability is supported. They developed a directed graph composition scheme for efficient trustworthy services in IoT. To connect different services, a small unit service node and directed data graph were used. The middleware constitutes of several unit service nodes which consists of input, processing, and output unit. Each service unit sends its output to the next service unit or to a specific actuator. The six layers proposed architecture was effectively used to lower both computational overload and memory usage. Low execution time and high reliability are considered as significant advantages of this mechanism. However, a real simulation has not been conducted and both scalability and cost were not evaluated.

Since that cloud computing is considered as one of the leverage and innovative tools for managing IoT resources, promoting their availability, and providing on demand services, Liu et al. (L. Liu et al., 2012) have proposed a cloud based IoT service architecture wherein key issues such as semantics of services, context aware service discovery and adaptive service composition have been addressed. The goal of the proposed architecture is to utilize cloud features to provide services from various types of IoT objects. In addition, and to support interoperability, a mechanism for integrating the physical movement and mobile services is proposed as well. Furthermore, they have provided a technique to efficiently creating various logically isolated networks partitions that work over the shared physical resources which allows several virtual networks to simultaneously work over the same infrastructure. Although, their work promised high scalability, availability, and efficiency. However, it has not been tested with a real-world scenario.

SOA -based approaches

Service Oriented Architectures is a software design style that used for creating services architectures. The SOA represents the functional resources through defining their services interfaces, input and output parameters. Advantages such as provisioning and service usage simplicity, seamless integration of heterogeneous applications and reduced expenses are considered some of the major reasons of exploiting SOA method in service provisioning software (Valipour et al., 2009). Representational State Transfer or (REST) is a software architectural style in which a set of coordinated set of constraints are applied on component and data elements when developing web services. It ignores the details of the implementation and protocols syntax of various components, rather, focuses on their functional responsibilities and interpretation of different data elements, thus, provides high scalability and flexibility.

Rodríguez-Valenzuela et al. (Rodrígguez et al., 2012) have presented a novel method to manage data acquisition and fusion based on a SOA-middleware service composition mechanism. The proposed method simplifies data and information provisioning and managing in pervasive environment. In their work, they have utilized distributed and dynamic services to exchange context data and perform complex processes. It is a form of home automation system service platform by which dynamic and scalable applications can be effectively developed using lightweight and autonomous services. The distributed data fusion method allows the service model to acquire data from external

services through its composite operations to manage, combine and build new information to be requested and shared with other services when needed through services network. This method makes the service model dynamic and scalable and reduces the cost. However, there are some issues that related to the communication and interaction over the Internet. Dar et al. (Dar et al., 2011) have emphasized the insufficiency of extending traditional web services SOAs approaches to be utilized in handling the dynamicity and heterogeneity nature of the targeted IoT environment. Composing services provided by IoT resources requires an integration of numerous number of physical devices and real world services to offer user centric and context aware service composition processes. They have proposed an architectural approach that supports adaptive composition services through orchestrating the distributed web services in very large scale IoT systems and integrating various web services at different enterprise level. Their investigation has tackled the issue of combining sensors services with traditional information systems by presenting a composition approach at two levels: global choreography and local orchestration by which issues such as dynamicity, scalability and configurability have been addressed. The orchestration is used between devices to allow peer to peer interaction between IoT nodes to reduce network traffic, avoid communication complexity and efficiently utilized the finite resources of IoT devices. On the other hand, choreography method has been found suitable for composing the external services since that it is always provides a kind of global view and allows to be parallelly invoked. Although authors have identified scalability as an issue to be addressed in the large scale IoT context, they didn't provide a method to handle this issue. Furthermore, several service composition Quality of Service QoS parameters were ignored and neither implementation nor evaluation was presented.

Form previously mentioned works, it is worth noting that despite of providing significant results in terms of some QoS parameters, it has been noticed that there was a disparity in considering these quality metrices, some of them such as scalability, execution time, cost, and reliability were efficiently evaluated while others have been totally neglected. This diversity is because that, often, service composition as a paradigm is about providing value added services in order to fulfill users' needs thus, help to reduce time and increase their satisfaction. Also, since that services in IoT environment are provided by distributed devices with limited communication cost, energy consumption and processing capabilities, by considering the cost of this type of services, researchers and service developers pursue to select the most appropriate combination of these services based on the available cost with paying more attention to time complexity and user satisfaction issues. In addition, the performance of some of the proposed works has not been tested in real world applications scenarios, and some others were not suitable for IoT big environment.

The proposed framework

Any given service composite for complex users tasks can be represented as a restrictive data transition case wherein the system data changes from the initial state (input parameters) to the goal state (the requested output data) while fulfilling the requested functionalities and constraints. Any participating service packages its resources to support the data transition process. The composition process starts by searching for the information that represents the request's goals. These goals are resolved backward to reach the request's antecedents through converting these goals into subgoals, then, resolving backward through these subgoals until finding a suitable solution when all the antecedents are reached. In order to effectively perform this process, there are several requirements that need to be considered. These requirements are:

- Transforming the user request to a formal request.
- Finding the required services that would respond to that request.
- Identifying the execution sequence of the selected services to respond to the user's request.
- Identifying the role of the user as a main actor in the service composition process.
- Evaluating whether the composite results match the user's request.

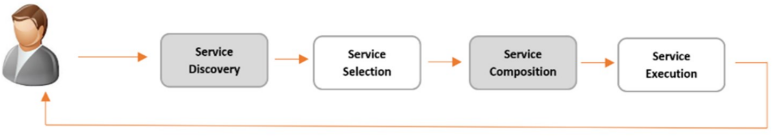


Figure2. Service Composition Life Cycle

Dynamic service composition life cycle

As shown in figure2, the flow starts when the user requires a new service to fulfill some of his/her needs at the runtime. The first phase in the lifecycle is the service request wherein the user specifies his preferences, interests, context and specifications of the desired service. According to request's specification and properties defined by the user, candidate services are discovered through the service registry interface. Services that match the required specifications will be discovered. Since that the retrieved services list may contain numerous amounts of services with the same functionalities and specifications, additional step is required to filter this list. Several techniques can be utilized to filter and select the final services based on some additional information such as user's preferences, interests, context, service QoS parameters, past interaction, frequency of use, etc. Given the filtered list of candidate services, several algorithms can be utilized to build the composition considering the user service request. at this stage, the user role is to guide and refine the composition process according to his requirements and preferences. The final phase is the execution phase in which an executable representation of generated composition is created (Lécué et al., 2007).

Adaptive user-centric IoT service composition framework

Over the past few years, with the Internet becoming ubiquitous, service-based applications are rapidly employed in different domains of people daily lives activities and it is expected to grow in the next few years. The proliferation of the service-oriented systems alongside the tremendous amount of connected IoT objects and nodes have led to an explosion of the number of services in different domains.

Technologies and innovations such as internet of services, cloud computing and Software As Service SAS have pushed new, adaptive, and personalized services applications in which the user plays an active role in the service creation, design and composition processes. This is considered as the main motivation behind what is called Internet of Services.

Often, users have different preferences, interests, and service requests in different context conditions. In addition, due to the lack of technical background knowledge to use advanced services tools,

users require and expect adaptive and dynamic on demand service applications with high level of abstraction in service creation and composition processes.

The role of user in the service composition process

In the traditional service composition life cycle, there are several fixed activities that required to be accomplished in a specific order, these activities are: service discovery > selection> composition> execution. Often, these activities are performed sequentially at the exact same order. However, to provide an adaptive self-acting service composition mechanism, these activities should be accomplished on demand to match the specific requirements of the end users. Users sometimes don't know their specific requirements at a specific moment and what they really want to accomplish when using services. For example, if the user wants to use web services in order to find a restaurant, a hotel and the route for that hotel and restaurant. If the previously mentioned lifecycle was fixed, there will be a problem for several reasons:

- not all users have the same composition order, e.g., the hotel first and then the restaurant.
- sometimes, the user may not have a clear view of what he really wants at a specific moment until he runs a specific service, after that, he would decide what service to be next (e.g., he doesn't need the route service to the hotel until he realizes that he doesn't know the route to that hotel).
- the user may not know the domain in which the required services belong to, thus, he will need some help during the service request. sometimes, even if he knows the domain, he may realize that he needs a specific service after executing the first service.

When required, a user may use the activities in the same fixed order while another user uses the same activities but in different order. although the workflow for the various activities is different here, all of users supposed to run the same commands.

The framework architecture

The main idea is to provide a generic user centric (self-acting) adaptive service composition framework. This could be achieved through allowing the user to create a workflow of IoT services by suggesting the available choices at each step through instant recommendations, thus helping the user to formulate better service compositions. At first, processes such as service discovery, selection and composition require semantic service description. Often, semantic web ontologies that conceptualize the application domains are applied. These ontologies are utilized by all the framework's components and modules. In addition, services in terms of their input, output, preconditions, effects, goals and non-functional requirements are required to be described through these ontologies as well. Several semantic web ontologies such as OWL-S can be used to provide expressive descriptions that are suitable for the automatic service discovery, selection and composition. Exploiting semantic technologies allows to efficiently overcome the syntactic barriers and provide results that approximate the user's request when an exact match does not exist.

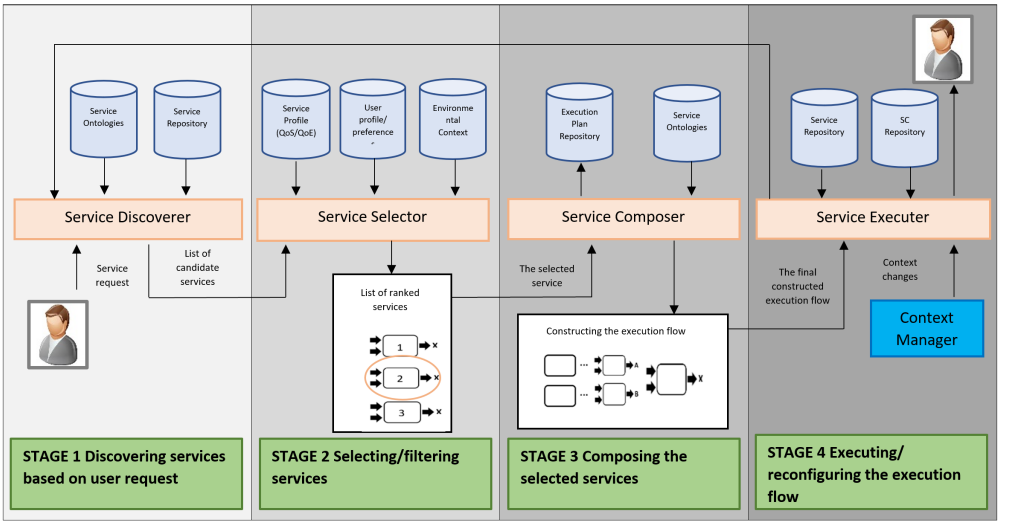


Figure 3. The proposed framework

As depicted in fugure3, complex users' tasks can be modeled as a service request that contains a unique identifier for this request, a set of functional requirements, a set of constraints such as execution time constraints that determine the success or failure of the composite service process. the composite service request also contains semantically annotated request parameters including a set of inputs (initial parameters) and a set of outputs (goal parameters), preconditions, effects and some non-functional requirements. The interface that provided to the user to specify the request may vary according to the targeted users and application domain field. Technologies such as natural language processing would be helpful here to provide the user the choice to request the service in a simple smooth way. The role of the request handler is to extract the properties of the requested services from the user request. by using a parser, properties such as input, output, preconditions, effects and ontologies could be extracted from a specific request. the extracted annotations then will be matched using a semantic analyzer against the elements defined in ontologies described in OWL-S. the role of this analyzer is to utilize domain ontologies to disambiguate and interpret users queries to construct a formal request. The purpose of the visualizer is to provide the end user with a visual description of the whole composition process along with its alternatives available.

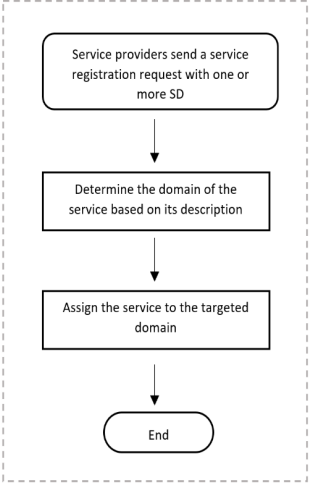
When a formal service request is defined, the service discovery module is triggered and queries the services repositories for a service that matches the service request, if no matches are found, a composite service will be created to fulfill that request. services are registered in repositories using different techniques according to their type. Web Services providers publish service descriptions and respond to different requests SOA interfaces. Wireless Sensor Networks WSN services providers also publish services descriptions and exchange sensors data using the publish/subscribe mechanism. The Autonomous Service Providers ASP listen to the discovery messages asking for the available services using the publish/subscribe mechanism as well. Figure4 illustrates this service registration process. the aim form classifying the services according to their domain is to offer fast service discovery and composition.

Matching services functional properties (Service discovery)

Given the semantic description of services, they are matched based on the semantic similarity in terms of their functionalities, categories and requirements using a matchmaking function. The role of this function is to find some levels of semantic compatibility among services based on some predefined relationships.

In the functionality matching process, services can be semantically matched using the service description five aspects: input, output, preconditions, and effects. In addition, services can be categorized according to their different service categories. Apart from the required functionalities, an essential determinant of the suitability of a selected service is the similarity and the degree of matching between the context that the service has been developed in and the context in which this specific service is intended to be used. Often, the suitability of a specific service to be used in the composite request is determined by the matching between the service's functionality and the request's goal. This can be ranked through different matching levels: usable if the output of the service is totally fulfilling the requested goal, part usable if the output can partially fulfill the goal and unusable otherwise (Chen, 2016).

Service selection (Filtering the matched services)



As the number of IoT service is exponentiality growth during the last few years due to the increasing amount of connected device and nodes, the number of displayed services in the matching list can be extremely high in several scenarios. data such as QoS non-functional attributes, devices characteristics, user and service context, user's past interactions and preferences could be effectively used to filter the list of matched services.

The main objective of this phase is to find IoT services that fulfill users' functional and nonfunctional requirements. Therefore, data such as users' interests and preferences according to the usage history can be considered as an effective source to be exploited to recommend services (Usage history could be stored as records constitutes the service used and the corresponding specified QoS parameter value). Moreover, extracting users' QoS parameters preferences from the previous records to be considered during the selection process allows to recommend suitable services that can both meet the required functionalities preferred QoS values.

At this stage as depicted in figure5, services are going to be filtered based on other selection criteria such as user's profile, usage history, QoS parameters and context. Filtering services according to these factors is significantly improve the service recommendation process.

- ***User profile*** maintains the user profile information including for example user's contact information, personal information, schedule, device characteristics, etc. The user also is allowed to provide his personal preferences such as temporal constraints (time), spatial constraints (location), data supply preferences (providing the input for a specific service from the user's profile), and delivery preferences (the way the user prefers to receive the output).
- ***Context:*** despite the significance of the context role in processes such as service selection and recommendation, there is no clear and precise definition and notion of context and what can be included as items in this concept. Context may include almost everything and can be differ in several ways according to researchers' perspectives. The usage of the context can differ from one application to another as well. LIU et al. (L. Liu et al., 2013) have presented the context as the set of conditions wherein the user intends to use a specific service and it is independent from the service definition. According to LIU et al., there is an implicit relationship between users, services and context, since that users' selections of services are highly dependent and change under different contextual situations. They have identified seven different context parameters including: time of use, execution platform, intended use, usage frequency, type of use and service delivery.

context is the information, by which the situations of an entity (person, thing, service, etc.) and the interactions between them, can be described. Three different types of contexts could be considered: service context, device context and user context. Service context includes parameters related to the service such as the location of the specific service, the category of such service (either a context acquisition or analysis service) and the energy level of the device that hosts the service. User context can be specified by several characteristics that are related to the user such as location of the user at a specific moment, user's preferences including preferences for some QoS criteria such as the level of security, calendar, etc. finally, the device context that includes the hardware and software characteristics of the user's device.

According to Hussein et al. (Hussein et al., 2017), contextual data can be classified into objective and subjective context. The objective context is related to the physical aspects of the user's surrounding environment including location, time, device tatus, etc. while the subjective context represents the human factors including user's goals, preferences, experience, trusted services, etc. Combining these two types of context data can effectively enhance the ability to accomplish situation awareness in smart environment and allow for adaptive dynamic service discovery which has a significant impact on composing the appropriate set of services thus, meet user's complex requirements. Hong et al. (Hong et al., 2009) have emphasized the significance of considering internal context, i.e., subjective context in order to provide personalized services and satisfy users' needs.

- ***Quality of Service QoS*** is used to describe the non-functional characteristics of web services. QoS parameters can be classified into user independent properties which have identical values for all different users such as price, availability, popularity, etc. they are always provides by service provider or by some third parties registries. Some other properties are users dependent thus, have different values for different users such as response time, invocation failure rate, etc. (Zheng et al., 2011).

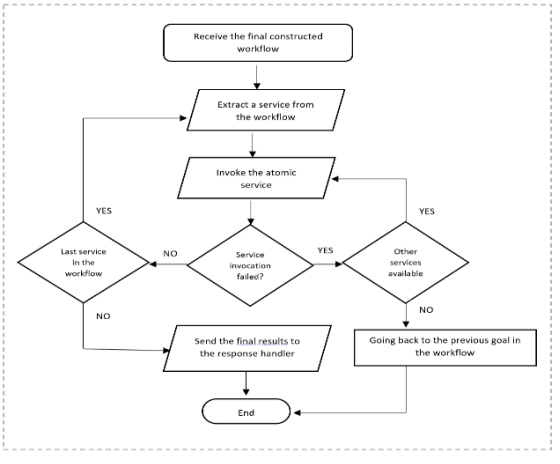
After filtering the matched services, a bunch of selected services will be instantly recommended to the user to refine their partially composed services and complete the composition task. These services will be ranked according to their Quality of Experience QoE and usage frequency. Through reviewing the literature, it has been noticed that previously proposed studies are less aware of the importance of the frequently used IoT services and the potential value in the service usage and QoE data in the service selection process. information such as users' experience have higher probability to meet new users' requirements and have been proved to be reliable and robust, thus, increases the correctness of the composed services. Through analyzing this historical data, similar frequently used IoT services could be suggested to the user as candidates for service selection. Here, when the user selects a specific service, a default rating for this service will be stored in both the service log and in the user's profile to be used in the user's future context settings. In addition, service numerical rating could be extracted from users' service reviews and integrated to the service representation.

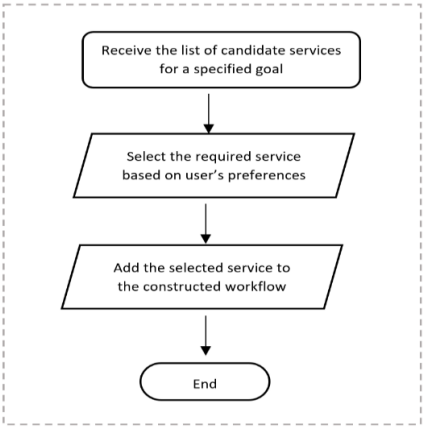
The processes of service discovery and selection will be incrementally repeated until all the subgoals are met. Here, finding and selecting services can be considered as goal driven and open-ended problems and will be performed hope by hope based on service data dependencies.

Constructing the service composition workflow

When the user selects the required service for a specific goal, as shown in figure6, the service composition module is responsible for creating the service compositions workflows by finding the

semantic similarity and dependency between the selected services. Services are said to be similar when they have similar functionalities or require similar inputs and generate similar outputs with similar types. Two services are semantically dependent if the output for the first one has the same semantic type of the second service's input. Matchmaking models such as Causal Link Matrix CLM+ (Lécué et al., 2007) and MatchAP (Chen & Clarke, 2014) can be used to explore semantic and dependency between services interfaces and descriptions. At this stage, semantic connections between IoT services are constructed and stored in the Causal Link Matrix CLM+. The CLM is used to compute and construct the set of "semantic graph based" compositions that matching the user's request.





When the semantic link established between services, they become semantic neighbours to each other. According to (Chen & Clarke, 2014), these semantic links can be ranked and classified based on their match types: i) same: means that service A and B have the same functionalities with the same set of inputs, ii) reqI: service A provides the same functionalities with B but asks for additional input data, iii) share: A and B have shared functionalities, iv) dep: when service A depends on B's execution results, v) in: service A provides input data to B. For example, a navigation service relies on both location and map data as inputs to generate navigation information between locations. The semantic link between the navigation service and getLocation service can be created with the type dep if the second service can provide any of inputs for the first one. In addition to these relations, there are some other relationships that represent different degrees of semantic relations in the typical service composition literature. These relations are: i) exact: it means that the two services are the same, thus can be used interchangeably during the composition. ii) plugin and subsume: they represent an approximate relationship between two services, thus, can be used to approximately match inputs to outputs. This type of relations is useful when there is no exact match is found. These relations are constructed by applying semantic reasoning mechanisms over the ontologies that are defining the services descriptions (Tzortzis, 2016). It is worth noting that semantic neighbours services are logically neighboring and not necessarily physically. In addition, it is also possible that services which have semantic relations are published by the same service provider (peers).

Service execution

As shown in figure7, after the service composition workflow is constructed, it will be sent to the execution module. At this stage, the execution module starts to execute the composition by invoking each individual service and exchange the data between these services based on the workflow that constructed by the user.

Composition failure could be occurred even with the availability of the targeted service due to several reasons such as the emptiness of the retrieved services matching list for one of the inputs of an already previously selected service, to avoid this and instead of stopping the composition process, the user would be prompted to select another service to replace the selected one. In the case that there are no alternative services are available, it would be allowed to revert to an earlier step of the composition process to replace the selected service at this step and resume the process further. Other possible reasons that require to adapt

The composition workflow at the run time and replace specific selected service including lack of trust to its provider, security issues, cost, time constraints, service availability etc.

In order to successfully achieve this goal, a set of execution guideposts that are responsible for controlling the elements involved in the service workflow could be identified. Each of these guideposts is maintained by one of composite participants and used to keep the direction of the execution path. Moreover, it may contain the post-condition services that can be selected as the next hope execution. Furthermore, it can be utilized to resolve a data parallel task This could be useful in performing a rollback during the reselection process to recover the system in failure cases. At this stage, it would be useful to calculate the non-functional properties of service composition each time a new service is added to the composition to evaluate the performance of the composite services. the overall non-functional properties of the resulting composition can be computed by aggregating the non-functional properties of the atomic selected services. In addition, user satisfaction degree could be utilized to measure the performance during the service composition process. The user satisfaction degree could have a specific value in the range of [0,1], the higher its value, higher is the degree of user satisfaction. The two QoS parameters response time and throughput would be used to measure the overall user satisfaction degree as well. This measurement can enable effective services ranking in the future.

These previously constructed compositions are supposed to be available to the system as named services which can be filtered and used as well. Moreover, they can be used and discovered by other services to perform a complex service composition. Each of them has a description, a profile with the added nonfunctional properties.

Testing and evaluating the proposed work

Several scenarios would been conducted to evaluate the performance of the proposed framework. In such scenarios, it will be assumed that there is a smart environment in which thousands of heterogeneous objects such as mobile devices, wireless sensors, smart home and smart building devices are abstracted as services to be accessible and interconnected.

Datasets

- ***IoT service description***

An IoT services dataset released by Gary White et al (White et al., 2018) and publicly available[1] could be used to test the proposed framework. It has been collected from services using real sensors.

- ***Web services description***

A composite service test collection called OWLS-TC4 can be used. It is a service dataset that consists of 1083 semantic web services described in OWL-S and covering nine application domains including communication, economy, education, food, geography, medical, simulation, travel and weapon. In addition, 42 queries are included and associated with relevance sets to allow for performance evaluation experiments (Urbieta et al., 2017).

In addition to this dataset, a dataset released by Zheng et al. (Zheng et al., 2014) could be used as well. It consists of a matrix of the response time and throughput of 339 users from 30 countries for 5,825 real-world web services from 73 counties. Users are represented through a number of distributed computers that are not co-located with the services.

The evaluation process mainly focuses on measuring the suitability of the proposed framework to support various application domains and users. This could be achieved through two types of evaluation: applicability evaluation and performance evaluation (Goncalves da Silva, 2011) .

- applicability in order to evaluate the applicability of the proposed framework in different conditions, success ratio will be measured, i.e., its capacity to successfully responding to the heterogeneous users changing requests.

- performance evaluation the framework is particularly designed to be suitable for runtime service composition situations to create personalized services. To accomplish this, a kind of real time responses to the various actions of the composition process should be provided. This could be evaluated by measuring the changes in computation time as the number of the candidate services increase and capturing the response time of the different commands through the whole process, from the moment that the user issues a service request until he receives the results. Several simulation environment platforms such as SimpleSoft (SimpleSoft's IoT Simulator for CoAP, MQTT, MQTT-SN, HTTP/REST Sensors and Gateways., n.d.), NetSim (NetSim-Network Simulator & Emulator | NetSim Professional, n.d.) and IoTIFY (IoT Simulator | IoTIFY - Cloud Based IoT Simulator and IoT Testing Platform, n.d.) could be utilized to test and evaluate the performance of the proposed approach.

Discussion and conclusion

Despite the considerable efforts to study and resolve service composition issues, most of these works have been done by focusing on design-time service composition in which the composition process is accomplished at the design time. Developing IoT service compositions at the run-time wherein the composite services are created and executed when they are required on the fly and involving users in such process is still limited. Broadly investigating the literature, several shortcomings concerning the related works were encountered and open the way for further exploration in this topic. For example, most of the reviewed works emphasized on automating the process of integrating the selected services without any consideration of the heterogeneity users' characteristics and the diversity of their requirements. In addition, the vast majority of the investigated works did not

[1] https://www:scss:tcd:ie/~whiteg5/data/sensor data:zip

consider the adaptation behaviour by which adaptation mechanisms are utilized to cope changes in the environmental context and users requirements and automatically select another candidate service out from the available pool of services. Runtime workflow validation is another issue that has been neglected. Due to the dynamicity nature in which the composed services are resided in, checking the validity to avoid invalid service sequences has become an urgent demand.

This research aspires to personalize IoT service delivery according to the evolve changing of users' requirements by proposing an adaptive user-centric IoT service composition framework. It allows the user to discover, select and interconnect services on demand at the runtime. The framework provides a flexible and multi-step interaction between user and system. It allowed the user to specify his preferences, interests, and specifications of the required service. In addition, data such as QoS non-functional attributes, devices characteristics, user and service context, user's past interactions (usage history) and preferences is effectively used to filter the list of matched services. Technologies such as natural language processing, request handler and parsers are used to provide the user the choice to request the service in a simple smooth way and extract the properties such as input, output, preconditions, and effects from that request. The adaptivity and self-acting composition mechanism provide an on-demand and flexible service composition lifecycle to adopt users' different characteristics, requirements and technical skills (**research question1**).

The visualizing tool is helpful to provide the end user with a visual description of the whole composition process along with its alternatives available. When a formal service request is defined, the service discovery module is triggered and queries the service repository for a service that matches the service request, if no matches are found, a composite service will be created to fulfill that request. processes such as service discovery and selection will be incrementally repeated until all the subgoals are met. Here, finding and selecting services is considered as goal driven and open-ended problems and performed hope by hope based on service data dependencies (**research question 2**).

When the user selects the required service for a specific goal, the service composition module is responsible for creating the service compositions workflows by finding the semantic similarity and dependency between the selected services using Matchmaking models such as Causal Link Matrix CLM+. If a failure occurred, instead of stopping the composition process, the user would be prompted to select another service to replace the selected one. In the case that there are no alternative services are available, it would be allowed to revert to an earlier step of the composition process to replace the selected service at this step and resume the process further. A set of execution guideposts are responsible for controlling the elements involved in the service workflow by performing a rollback during the reselection process to recover the system in failure cases (**research question 3**).

References

[1] Arellanes, D., & Lau, K.-K. (2020). Evaluating IoT Service Composition Mechanisms for the Scalability of IoT Systems. *Signal Processing: Image Communication*, 115780. https://doi.org/10.1016/j.image.2020.115780

[2] Asghari, P., Rahmani, A. M., & Javadi, H. H. S. (2018). Service composition approaches in IoT: A systematic review. *Journal of Network and Computer Applications*, *120*, 61–77. https://doi.org/10.1016/j.jnca.2018.07.013

[3] Baker, T., Asim, M., Tawfik, H., Aldawsari, B., & Buyya, R. (2017). An energy-aware service composition algorithm for multiple cloud-based IoT applications. *Journal of Network and Computer Applications*, *89*, 96–108. https://doi.org/10.1016/j.jnca.2017.03.008

[4] Bell, J. (2009). *SOA MODELING PATTERNS FOR SERVICE -ORIENTED.*

[5] Ben Mokhtar, S., Georgantas, N., & Issarny, V. (2007). COCOA: COnversation-based service COmposition in pervAsive computing environments with QoS support. *Journal of Systems and Software*, *80*(12 SPEC. ISS.), 1941–1955. https://doi.org/10.1016/j.jss.2007.03.002

[6] Berardi, D., Calvanese, D., De Giacomo, G., Lenzerini, M., & Mecella, M. (2004). A foundational vision of e-Services. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3095*(May), 28–40. https://doi.org/10.1007/978-3-540-25982-4_4

[7] Bossi, L., Braghin, S., & Trombetta, A. (2014). Multidimensional reputation network for service composition in the internet of things. *Proceedings - 2014 IEEE International Conference on Services Computing, SCC 2014*, 685–692. https://doi.org/10.1109/SCC.2014.95

[8] Cao, Z., Qiao, X., Jiang, S., & Zhang, X. (2019). *SS symmetry An Efficient Knowledge-Graph-Based Web Service.* https://doi.org/10.3390/sym11030392

[9] Chen, N. (2016). *Goal-Driven Service Composition in Mobile and Pervasive Computing. 1374*(c), 1–14. https://doi.org/10.1109/TSC.2016.2533348

[10] Chen, N., & Clarke, S. (2014). A dynamic service composition model for adaptive systems in mobile computing environments. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8831*, 93–107.

[11] Chesbrough, H., & Spohrer, J. (2006). A research manifesto for services science. In *Communications of the ACM* (Vol. 49, Issue 7, p. 35). https://doi.org/10.1145/1139922.1139945

[12] Dar, K., Taherkordi, A., Rouvoy, R., & Eliassen, F. (2011). Adaptable service composition for very-large-scale internet of things systems. *Proceedings of the 8th Middleware Doctoral Symposium, MDS'11 of the 12th ACM/IFIP/USENIX International Middleware Conference*, 3–4. https://doi.org/10.1145/2093190.2093192

[13] Goncalves da Silva, M. E. (2011). *User-centric Service Composition - Towards Personalised Service Composition and Delivery* (Issue 11). https://doi.org/1381-3617

[14] Hamzei, M., & Jafari Navimipour, N. (2018). Toward Efficient Service Composition Techniques in the Internet of Things. *IEEE Internet of Things Journal*, *5*(5), 3774–3787. https://doi.org/10.1109/JIOT.2018.2861742

[15] Hong, J. yi, Suh, E. ho, & Kim, S. J. (2009). Context-aware systems: A literature review and classification. *Expert Systems with Applications*, *36*(4), 8509–8522. https://doi.org/10.1016/j.eswa.2008.10.071

[16] Hussein, D., Han, S. N., Lee, G. M., Crespi, N., & Bertin, E. (2017). Towards a dynamic discovery of smart services in the social internet of things. *Computers and Electrical Engineering*, *58*, 429–443. https://doi.org/10.1016/j.compeleceng.2016.12.008

[17] *IoT 2019 in Review: The 10 Most Relevant IoT Developments of the Year*. (n.d.). Retrieved May 3, 2020, from https://iot-analytics.com/iot-2019-in-review/

[18] *IoT Simulator | IoTIFY - cloud based IoT simulator and IoT testing platform*. (n.d.). Retrieved November 21, 2020, from https://iotify.io/iot-network-simulator/

[19] Khodadadi, F., Dastjerdi, A. V., & Buyya, R. (2015). Simurgh: A framework for effective discovery, programming, and integration of services exposed in IoT. *2015 International Conference on Recent Advances in Internet of Things, RIoT 2015*, *April*, 7–9. https://doi.org/10.1109/RIOT.2015.7104910

[20] Ko, I. Y., Ko, H. G., Molina, A. J., & Kwon, J. H. (2016). SoIoT: Toward a user-centric ioT-based service framework. *ACM Transactions on Internet Technology*, *16*(2), 1–21. https://doi.org/10.1145/2835492

[21] Lécué, F., Silva, E., & Pires, L. F. (2007). A framework for dynamic Web services composition. *CEUR Workshop Proceedings*, *313*, 59–75. https://doi.org/10.1007/978-3-7643-8864-5_5

[22] Li, S., Oikonomou, G., Tryfonas, T., Chen, T. M., & Xu, L. Da. (2014). A distributed consensus algorithm for decision making in service-oriented internet of things. *IEEE Transactions on Industrial Informatics*, *10*(2), 1461–1468. https://doi.org/10.1109/TII.2014.2306331

[23] Liu, J., & Tong, W. (2010). Dynamic services model based on context resources in the internet of things. *2010 6th International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2010*. https://doi.org/10.1109/WICOM.2010.5601423

[24] Liu, L., Lecue, F., & Mehandjiev, N. (2013). *Semantic Content-Based Recommendation*. *7*(3).

[25] Liu, L., Liu, X., & Li, X. (2012). Cloud-based service composition architecture for internet of things. *Communications in Computer and Information Science*, *312 CCIS*, 559–564. https://doi.org/10.1007/978-3-642-32427-7_80

[26] Maglio, P. P., & Spohrer, J. (2008). Fundamentals of service science. *Journal of the Academy of Marketing Science*, *36*(1), 18–20. https://doi.org/10.1007/s11747-007-0058-9

[27] Maglio, P. P., & Spohrer, J. (2013). A service science perspective on business model innovation. *Industrial Marketing Management*, *42*(5), 665–670. https://doi.org/10.1016/j.indmarman.2013.05.007

[28] *NetSim-Network Simulator & Emulator | NetSim Professional.* (n.d.). Retrieved November 21, 2020, from https://www.tetcos.com/netsim-pro.html

[29] Park, J. H., Zhe, L. Q., & Kim, S. D. (2016). Reliable services composition method for the internet of thing using directed service-object graph deployment scheme. *2015 IEEE 2nd International Conference on InformationScience and Security, ICISS 2015*, 5–8. https://doi.org/10.1109/ICISSEC.2015.7371029

[30] Rodrígguez, S., Valenzuela, J. A. H., Terriza, J. L. M., & Cobos, J. M. G. (2012). Data fusion mechanism based on a service composition model for the internet of things. *Actas de Las III Jornadas de Computación Empotrada*, *January*, 19–21.

[31] *SimpleSoft's IoT Simulator for CoAP, MQTT, MQTT-SN, HTTP/REST sensors and gateways.* (n.d.). Retrieved November 21, 2020, from http://www.smplsft.com/SimpleIoTSimulator.html

[32] Sosa-Reyna, C. M., Tello-Leal, E., & Lara-Alabazares, D. (2018). Methodology for the model-driven development of service oriented IoT applications. *Journal of Systems Architecture*, *90*(August), 15–22. https://doi.org/10.1016/j.sysarc.2018.08.008

[33] Tzortzis, G. (2016). *A Semi-Automatic Approach for Semantic IoT Service Composition.*

[34] Urbieta, A., González-Beltrán, A., Ben Mokhtar, S., Anwar Hossain, M., & Capra, L. (2017). Adaptive and context-aware service composition for IoT-based smart cities. *Future Generation Computer Systems*, *76*, 262–274. https://doi.org/10.1016/j.future.2016.12.038

[35] Valipour, M. H., Amirzafari, B., Maleki, K. N., & Daneshpour, N. (2009). A brief survey of software architecture concepts and service oriented architecture. *Proceedings - 2009 2nd IEEE International Conference on Computer Science and Information Technology, ICCSIT 2009*, 34–38. https://doi.org/10.1109/ICCSIT.2009.5235004

[36] Wang, X. (2010). *2010 International Conference on Service Sciences A Brief Study on the Research of Service Science Theory. 2008.* https://doi.org/10.1109/ICSS.2010.31

[37] White, G., Palade, A., & Clarke, S. (2018). QoS Prediction for Reliable Service Composition in IoT. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10797 LNCS*, 149–160. https://doi.org/10.1007/978-3-319-91764-1_12

[38] Yachir, A., Amirat, Y., & Chibani, A. (2012). *Towards an event-aware approach for ubiquitous computing based on automatic service composition and selection.* 341–353. https://doi.org/10.1007/s12243-012-0306-y

[39] Yang, R., Li, B., & Cheng, C. (2014). Adaptable service composition for intelligent logistics: A middleware approach. *Proceedings - 2014 International Conference on Cloud Computing and Big Data, CCBD 2014*, 75–82. https://doi.org/10.1109/CCBD.2014.10

[40] Yang, R., Li, B., & Cheng, C. (2015). A petri net-based approach to service composition and monitoring in the IOT. *Proceedings - 2014 Asia-Pacific Services Computing Conference, APSCC 2014*, 16–22. https://doi.org/10.1109/APSCC.2014.11

[41] Zheng, Z., Member, S., Ma, H., Lyu, M. R., King, I., & Member, S. (2011). *QoS-Aware Web Service Recommendation by Collaborative Filtering. 4*(2), 140–152.

[42] Zheng, Z., Zhang, Y., & Lyu, M. R. (2014). Investigating QoS of real-world web services. *IEEE Transactions on Services Computing*, *7*(1), 32–39. https://doi.org/10.1109/TSC.2012.34