Journal of Ambient Intelligence and Humanized Computing
https://doi.org/10.1007/s12652-020-01893-4

ORIGINAL RESEARCH

A novel user interaction middleware component system for ubiquitous soft computing environment by using fuzzy agent computing system

R. Vasanthi[1] · R. Jayavadivel[2] · K. Prasadh[3] · J. Vellingiri[4] · G. Akilarasu[5] · S. Sudhakar[6] · P. M. Balasubramaniam[7]

Received: 4 November 2019 / Accepted: 13 March 2020
© Springer-Verlag GmbH Germany, part of Springer Nature 2020

Abstract
Ubiquitous Computing Environment (UCE) has become increasingly crucial for a component-based Middleware for Ubiquitous Soft Computing Environment by using Fuzzy Agent Computing System (USCE–FACS) that can fully exploit interactive service characteristics to support user-centric services in UCE. It has posed some challenges, such as high Memory Consumption and high Component Building Time to users who are hiring services in a heterogeneous environment. To control the UCE for the user's benefit, FACS works non-rudely in an online deep-rooted learning way to get familiar with the user behavior. The point of incorporation of previously mentioned advances is to make more extensive the connection between individuals and data innovation gear through the use of an undetectable system of UCE devices making dynamic computational-environments equipped for fulfilling the users' prerequisites. Due to a large number of available services without ontology and Metadata Repository (MR), it becomes time-consuming for end-users to find appropriate services to satisfy their various needs. To help end-users obtain their desired services, the USCE–FACS proposed, it reduces MC and CBT in a heterogeneous environment. This paper concentrated on the communication from users to devices to enable a general and quick access to accessible element and administrations gave by the UCE.

Keywords User centric · Middleware · Semantic query · Functional dependency · Ubiquitous computing environment · Soft computing · Fuzzy agent system · Fuzzy logic

1 Introduction

The UE considered as a physical environment saturated with computing and communication capabilities, well-integrated with users (Debashis et. al. 2005). In the future, the day-to-day routine contains a network of more or less specialized computational devices that interact with different users and among themselves. The development of ubiquitous technologies has allowed improvement in service availability in soft computing. These services, offered by information systems, are becoming more ubiquitous, i.e., accessed anytime and anywhere. However, the ubiquitous information

✉ K. Prasadh
prasadhk0123@gmail.com

R. Vasanthi
vasanti_me@yahoo.co.in

R. Jayavadivel
infotech.jaya@gmail.com

J. Vellingiri
vellingirijayagopal@gmail.com

G. Akilarasu
akilgva@gmail.com

S. Sudhakar
sudhasengan@gmail.com

P. M. Balasubramaniam
mebalu3@gmail.com

[1] Department of CSE, Tagore Institute of Engineering and Technology, Attur, Tamil Nadu, India

[2] Department of CSE, Vivekananda College of Engineering for Women, Tiruchengode, Tamil Nadu, India

[3] Vijnan Institute of Science and Engineering, Ernakulam, Kerala, India

[4] School of Information Technology and Engineering, Vellore Institute of Technology, Tamil Nadu, Vellore, India

[5] Department of CSE, Malla Reddy Engineering College for Women, Hyderabad, India

[6] Department of CSE, Sree Sakthi Engineering College, Coimbatore, Tamil Nadu, India

[7] Department of EEE, Sri Shakthi Institute of & Tech, Coimbatore, Tamil Nadu, India

Published online: 21 March 2020

system remains too complicated for the user, who just wants a service satisfying his needs. This complexity requires considerable effort from the user to select the most appropriate service. Identifying those services that deserve the attention of the user becomes increasing a challenging task. User-Centrism requires applications to provide mechanisms to allow users to configure the application according to their personal preferences.

Nowadays, a large number of existing services and diversified needs, it is time-consuming for End-Users to find appropriate services. As the user is the ultimate client of the UE and not the application, the objective of providing USCE–FACS is to provide for each individual user to achieve quality for the end-users. Users are carrying more and more portable devices, including laptops, smart phones, and personal digital assistants. These devices are becoming even more sophisticated, supporting wireless technologies and application platforms. With the emergence of the ubiquitous computing era, it has become increasingly crucial for middleware to support user-centric services. To help end-users obtain their desired services, the proposed USCE–FACS provides a way to search and recommend services based on a User Request (UReq).

As the user is the ultimate client of Ubiquitous Soft Computing Environment and not the application, the objective of providing USCE–FACS Model is to provide for each individual user to achieve quality for the end users. Users are carrying more and more portable devices, including laptop, smart phones, personal digital assistants etc. These devices are becoming even more sophisticated, supporting a wireless technologies and application platforms. With the emergence of the Ubiquitous Soft Computing era, it has become increasingly important for a User Interaction Middleware to support user-centric services. The proposed USCE–FACS designed for reducing CBT and MC using UCE for soft computing. It increases the reusability of components and services in a heterogeneous network. The proposed work is to satisfy the user's need by discovering, detecting, and recommending the most suitable service based on a user-centric approach. It is a more user-friendly manner and recommends the user needed service.

In this administrative work, our commitment is the utilization of another arrangement is FACS. The FACS is a philosophy considered as a connection to the UCE and the conventional control hypothesis. This philosophy is usually applied in the main situations when exactitude isn't of the need or high significance. FACS is a technique for communicating operational laws of a framework in semantic terms rather than numerical conditions. Widespread of the FACS and high adequacy of its applications in an extraordinary broaden is dictated by formalization chances of relevant conduct of a controller as a "fuzzy" portrayal. This portrayal, for the most part, is planned as legitimate (fuzzy) administers under semantic factors of a type " If A then B".The FACS philosophy contains 3 stages: Fuzzification, Rule Engine, and Defuzzyfication.

2 Literature review

Projects just as the scholarly community has propelled loads of ubiquitous activities, in the course of the most recent two decades. SAFE-RD, MARKS, and ETS portray the works completed in the field of universal computing, in which security, flexibility, proficiency, middleware, asset revelation, and self-recuperating principally researched. Berkeley's Endeavor Project additionally give their endeavors to various angles in acknowledging pervasive processing in some specific conditions, fundamentally quickening the development of perceptive spaces.

Several abstract frameworks (Molina et al. 2007) have been proposed to develop a PC that uses context information. Context understood as any information about the user needs and operating environment, which vary dynamically and have an impact on design choices.

An agent-based intelligent middleware (Jang et. al. 2011) that supports user-centric services and proposed a design for middleware architecture that effectively supports several functions, such as context awareness, context management, pattern inference based on user behavior, and dynamic agent generation. Many works have been done on UE middleware (MW), focusing on different aspects and purposes (Raychoudhurya et. al. 2013). The example of user-centric MW is CASS, CoBra, Context Toolkit, CORTEX, Gaia, and SOCAM.

An agent-based intelligent Jang et. al. (2011) provides user-centric services in ubiquitous computing environments. The human–computer interface is friendly with MW that supports user-centric services. Ubiquitous user-centric systems (Wirsing et. al. 2010) used a generic component-based methodology. A SODESSONMW (Baumgart et al. 2011) enabled effortless and secure admission to services that received a supply by devices belonging to the user. This communication paradigm of user-centric networking has led to more proficient and protected communication because the indirection introduced by servers has eliminated. A user-centric and QoS-based service session (Soliman et. al. 2011) guaranteed the management of the user-centric session to ensure end-to-end service continuity according to the user's requirements and preferences during the session mobility. The graph-based approach PICO (Kalasapur et al. 2007) proceeds through the construction of a global graph representing all possible service compositions in the environment.

With SODAPOP, Kim et al. (2019) users specify a goal that the system tries to reach out to the available services. SODAPOP manages the environment by automatically

classifying new services according to pieces of information. Then, it tries to combine some of them to reach the user's goal. The SODAPOP translates user interactions and context information into concrete goals. Then, it tries to compose services planning to map users' goals. However, it is dependent on the meta-information that services expose, and users cannot define or customize their scenarios. The SODAPOP functionalities are almost entirely automatic processes, so users can only interact with the system by expressing their needs.

The goal of component-based development is to build and maintain software systems by using existing software components. Reusability achieved by loosely coupled component technology used; that is, the components are focusing on functionality and do not contain any direct operating system or hardware dependencies. Component-Based Software Engineering (CBSE) increases the productivity, reliability, and maintainability of software through reuse (Ao et al. 2005). Reusability further enhanced by the possibility of using configuration parameters to components. A bright, explicit, and well-defined component interface is crucial to enhance the software reusability. Component-based systems are more comfortable to assemble and, therefore, less costly to build than developing such systems from scratch. The importance of component-based development lies in its efficiency. Service composition enables quick development of new application functionality through the reuse of the components for multiple compositions (Al Saiyd et al. 2010). Service composition allows the combination of multiple components into a single composite service, achieved at design-time / run-time.

A product element of software represents an agent that has a few properties of a human, for example, self-rule, thinking, learning, and information level correspondence. The agent-oriented approach, which is exceptionally expected to assume an indispensable job in accomplishing keen space at an elevated level, has just communicated lively impacts on UCE in various applications. Essex's iDorm scheme focuses on understanding the vision of encompassing insight in human services situations by consolidating the utilization of unpretentious sensors and effectors with intelligent embedded-agents (Shetty et al. 2019). Fuzzy frameworks based versatile engineering for specialists installed in ambient intelligent environments, a various leveled fuzzy hereditary multi-operator design for building learning instrument, together with another novel deep entrenched learning approach dependent on intellectual specialists are tended to by Essex's gathering in, correspondingly. Other Contemporary studies inquire about spreading the accompanying regions: utilizing a neural systems specialist based way to deal with perceiving diverse elevated level accomplishments, orderly and supportive philosophy to create operator based framework (Yang et al. 2014), intellectual operators including case-based thinking, (Bhushan et al. 2019).

Semantic interoperability denotes the ability of computer systems to transmit data with unambiguous, shared meaning. Semantic interoperability is the capability of software components to interact without misunderstandings. Semantic interoperability McIlraith et al. (2001) is concerned with ensuring that the exchanged information has the same meaning for both message sender and receiver. Semantic interoperability does not deal just with the packaging of data (syntax), but the simultaneous transmission of the meaning along with the data (semantics). This is made possible by adding data about the data (metadata), linking each data element to a controlled and shared vocabulary. The meaning of the data is transmitted along with the data itself, in one self-describing information package which is thoroughly independent of any information system. It is a shared vocabulary and its associated links to an ontology that provides the foundation and capability of machine interpretation and logic.

Semantic search has been effectively used for information retrieval. A semantic retrieval model which comprised collecting resources, semantic annotation and query analysis and ranking results is proposed (Doctor et al. 2005). For semantic annotation, it used domain ontology and two bilingual dictionaries to extract keywords for annotation. For query analysis, it presented a method that combined lexical relationship and semantic relationship to analyze user's query. For results ranking, they have proposed a modularise strategy that ranks results focused on how predictable a result may be for users, which is a mix of semantic and information-theoretic techniques. It increases the precision and recall rates of webpage searching.

The term Ontology has been used in several disciplines such as knowledge management, intelligent information integration, e-commerce, cooperative information system and database integration where ontology is comprised of concepts, concept properties, relationships between concepts and constraints. Ontologies are defined independently from the actual data and reflect a common understanding of the semantics of the domain of discourse. Ontology is an explicit specification of a representational vocabulary for a domain; definitions of classes, relations, functions and other objects. Ontologies can be used to describe the semantics of information sources and make the content explicit and thus can be used to discover semantic equivalence between information concepts. Ontologies has been seen as a major solution to semantic interoperability because it describes the semantics of information sources and makes its content explicit by providing a shared comprehension of a given domain of interest (Hagras et al. 2003). Protégé tool is used to simplify the creation of Ontologies for the pervasive computing environment. The main uses of Ontologies are interoperability and communication among software systems.

3 Proposed user-centric applications for UCE

The recent propagation of multimedia communication structures, such as Twitter, Skype, BoxBe, is drawing people's attention from all facets of society. They bestow first communication services comprising of instant messaging schemes, video conferencing from a given place, email, and file transfer. The approach to these communication services has raised the need for the user to be more involved in developing user-centric applications. The user-centric approach refers to focusing on the betterment of the user and providing severity to hide the complexity of the underlying technology. User-Centered Design Process Raman et al. (2003) the Specification of the user and organizational requirements to identify business requirements or user goals. Recommender systems Dahl (2007) have the effect of guiding users in a personalized way to new services in an ample space of possible options. With the user-centric nature of service and the user's lack of prior knowledge on the distribution of the raw data, a challenge lies in associating user quality requirements on the clustering results with the algorithmic output properties. With the user-centric nature of service Siano et al. (2013) used the predictable homogeneity degree among entities within a given cluster as the input quality, which necessities specified through the users to drive the data clustering process.

3.1 Research methodology of proposed USCE–FACS

The proposed USCE–FACS needs to be designed to improve the reusability of components and provides user-centric services in a heterogeneous network. The proposed USCE–FACS needs to be designed to improve the Component Building Time (CBT) to reduce Memory Consumption(MC) and Service Delivery Time (SDT) based on user preferences in a heterogeneous environment, and it also guides the user needed service in the form of recommendation table. It improves the operation of searching and retrieving a relative component(s) and services that are matched with UREQ and also their preferences. The SDT and MC reduced by Component Detection and Discovery Phase (CDDP) based on Minimal Cover Functional Dependency Method (MCFDM). The proposed USCE–FACS consumes less time for building components using the clustering technique in User Interaction History Database (UIHDB) and also MCFDM. Collaborative Filtering Recommendation Approach (CFRA) recommends the user with the needed services. It has the effect of controlling users in an adapted way to provide service with possible options. The proposed work is to satisfy the user's need by discovering, detecting, and recommending the most suitable service for UREQ (Santos et al. 2011; Zhao et al. 2011). The Design Concepts of proposed USCE–FACS, as shown in Fig. 1. It consists of four phases namely,

3.1. Accessibility Phase.
3.2. Semantic-based Component Detection Phase.
3.3. Rapid Phase.
3.4. User-Centric Service Delivery Phase.

The Accessibility Phase consists of three units, such as the User Interface Unit, Profile Manager, and Query Processing Unit. The jobs of Accessibility Phase are to accept the UREQing query, register a new user to get the personal information and preferences of the user and provide the unique user ID. It also converts the UREQ into SQ and then sends the SQ to Semantic-based Component Detection Phase (CDP).

The SCDP consists of three units, such as Discovery Process Unit (DPU), Component Composition Unit (CCU), and Component Detection Unit (CDU). The SCDP of the UCMWC is to accept the SQ from the Accessibility Phase. By using Discovery Process Matching (DPM) algorithm, it detects whether SQ is available in the User Interaction History database (UIHDB) /Component Repository (CR) / Service Repository (SR) is based on the DPM algorithm, it processes the request and sends it to the User-Centric Service Delivery Phase (UCSDP).

The DPU is responsible for detecting whether the SQ is available in UIHDB or not. If the SQ precisely matched in UIHDB, it gets the Service ID (SID). Using that SID, it can be easily fetched from the SR. If that service is readily available, it is sent to UCSDP and also sent it to RP. If the SQ not precisely matched but related components found in UIHDB and CR, the DPU fetches the related Component ID (CID) and passes it to the CCU. In that unit, it gets the CID from the DPU and fetches components from CR, and then it builds the component using MCFDM. If the SQ is not available in

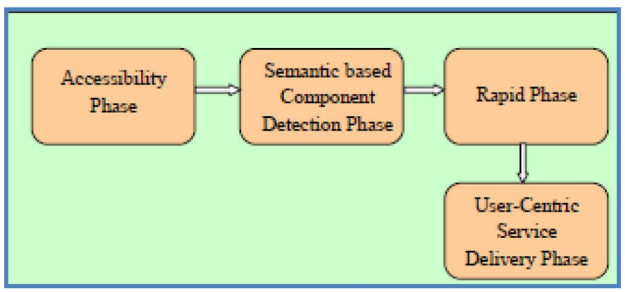


Fig. 1 Design Considerations of USCE–FACS

both UIHDB and CR, the DPU sends its control to the CDU. In the CDU, it searches the related user query in Domain Ontology Repository (DOR) and finds the components from the server. Then it is passed to CCU, where it builds a new service, and the service stored in SR, and its components stored in CR and its related SID, CID is stored in MR. The function of Rapid Phase (RP) is to recommend some related services required by the user based on the preferences and history of the user in UIHDB. Then it sends it to SDP.

3.2 Accessibility phase

The Accessibility Phase consists of three units, namely the User Interface Unit, Profile Manager, and Query Processing Unit. The User Interface Unit accepts the UREQ and checks whether the user already gets registered in the User Information Database. If the user information is already available, its user query sent to the Query Processing Unit. Otherwise, it goes to the Profile Manager.

The Profile Manager is responsible for storing information about the user and devices owned by the user in the User Information Database. It stores user profiles containing necessary user information, such as name, gender, email ID, and address. It also stores user preferences and their corresponding device profiles. It generates a unique user ID for each user. The profile manager allows the user to create, edit, and remove profiles and their preferences.

The Query Processing Unit in Fig. 2 gets UREQ from User Interface Unit if the user already gets registered in the User Information Database. The UREQ parsed by XML parser and converted into separate tokens. After passing through a parser, the user query converted into meaningful tokens with the help of an ontology approach. The SPARQL query language is used to retrieve information through the ontology approach An ontology is a set of related metadata schemas (i.e. classes) and instances (i.e. resources) described by formal semantics. Ontology characteristics are suitable for software component retrieval, as they allow capturing domain semantics and recommending interrelated software components. Thus, ontology-based metadata are incorporated in the proposed model USCE–FACS Model (Hagras et al. 2007). The incorporated metadata are identified from elements belonging to the ontology creation language Web Ontology Language (OWL). The OWL supports basic ontology elements and allows the definition of formal axioms that provide richer semantics to Ontologies. The ontology principles that are considered relevant to storage and retrieval of software components are modeled. Protégé tool is used to simplify the creation of Ontologies for the ubiquitous computing environment. It converts the UReqing query into the semantic query and sent it to the Discovery Process Unit, and also it extracts user preferences based on user ID from the User Information Database and sends it to RP.

3.3 Semantic-based component detection phase

The Semantic-based CDP in Fig. 3 consists of three units, namely Discovery Process Unit, Component Composition Unit, and Component Detection Unit. The DPU gets the input of the SQ of the Query Processing Unit. The DPU performs matching of SQ with a history of UIHDB. After performing operations in Semantic-based CDP, it sends a requested service to UCSDP.

3.3.1 Discovery process unit

The DPU gets SQ from the Query Processing Unit. Then DPU is responsible for detecting whether the SQ is available in UIHDB or not. The DPU searches the SQ with the history of UIHDB using the Discovery Process Matching algorithm. If the SQ exactly matches with UIHDB, it fetches the related SID from UIHDB. Based on that SID, it fetches the service from Service Repository directly and sends it to the UCSDP.

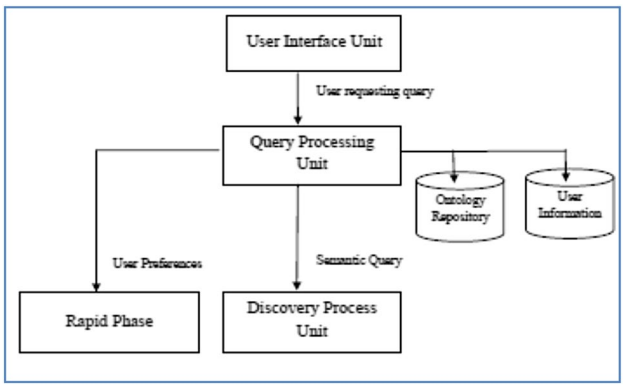


Fig. 2 The Query Processing Unit

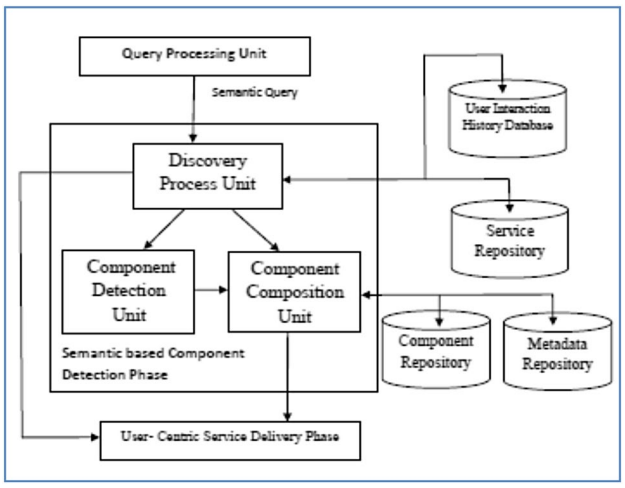


Fig. 3 Proposed Semantic-based CDP

Now the service is readily available in UCSDP, and it provides the service according to the user needed platform. The DPU also sends the selected service to the Rapid Phase to predict the future need of the user.

The UIHDB contains information about the past history of service delivery details of users. User Interaction History holds details such as date/time, UserID, SID, ComponentID, Component Description, Service Description, SQ, and User Preferences. The UserID represents the Identification of User. SID is Service Identification, and ComponentID is used to represent what are the components used to provide the services. Whenever a service is delivered, that information is stored in UIHDB to keep a trace of the users. UIHDB is clustered based on similar interests of users.

3.3.2 Pseudo code for clustering technique

```
Begin
Select all the user preferences Up= {Up1, Up2… Upm}
Select all the requesting query Rq= {Rq1, Rq2, …,Rqn}
Cluster User CU= {CU1, CU2… CUk};
For Each
User preference in Up
For Each
Cluster CU
Compute average similarity (Up, CU)
End For
End For
End
```

3.3.3 The steps are followed in the DPM algorithm

```
Step 1: QPU→SQ
Step 2: DPU →SQ → UIHDB
Step 3: For SD=UIHDB
        If SQ = SID → UIHDB && SR→DPU; SDP && RP
        Else If
                SQ = UIHDB
                CID→DPU & UIHDB: CCU → CR
        Then SDP && RP
        Else If (SQ ≠ UIHDB)
        For all MR
        If (SQ = MR)
                MR→CID && CR && CCP
                Where SDP && RP
        Else If
                SQ≠MR) && DPU →SQ →CDU
                CDU→ CCU
Step 4: CCU = CR && MR
        CCU→ SDP &&RP
```

3.3.4 Component composition unit

The CCU gets the result of the Discovery Process Unit. The DPU performs matching of SQ with UIHDB. In the case when the SQ not precisely matched with UIHDB but related components found, then it fetches that Component ID from UIHDB and sends it to the CCU. In this unit, it fetches the components directly from the component repository based on Component ID given by Discovery Process Unit. In another case, when the SQ not found in UIHDB, then it finds the components in Metadata Repository. If the matches found, then the DPU fetches that Component ID and sends it to the Component Composition Unit. Then CCU builds the service using MCFDM and sends it to UCSDP. The CCU sends the selected service to the RP. The CCU builds the service based on the MCFDM. Figure 4 shows the dependency of services and components (Diro et. al. 2018; Xhafa et. al. 2014). Each vertex represents a Service or a Component. The directed edges are used to represent functional dependency.

The overall strategy is to minimize the set of functional dependency to save storage. The components needed to run the current services are loaded while the components which are not disabled by the Minimal Cover method.

Given a component collection ID from the DPU, a dependency graph with a subset of the relevant components from a component repository generated. This subset contains the solutions that meet the request and consists of a set of components connected in sequence — the construction of the graph done by the Depth First Search graph.

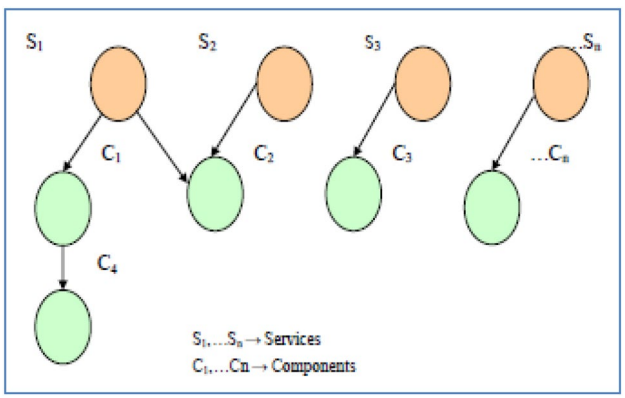


Fig. 4 The dependency of Services and Components

3.3.5 Pseudo code for building a dependency graph is as follows

Input
CC_i=Component Collection ID
URQ = UREQ Query
Output
Dependency graph G= V (Vertex), E (Edge)
Add Service (G)
For all component = MR
If
SQ = MR
Add URQ→V
MR→CID && CC_i→ DPU
DPU→ CC_i
For all CC_i in CR
Add components → to the dependency graph
End For

The Minimal-Cover (MC) Illingworth et. al. (2005) is to detect all components based on the requested Services (Sn), as shown in Fig. 5. A Service is composed of multiple components, each performing a different Kind of task. It is possible to access inter-related components of Service. For example Service S2 component is accessed in S1.

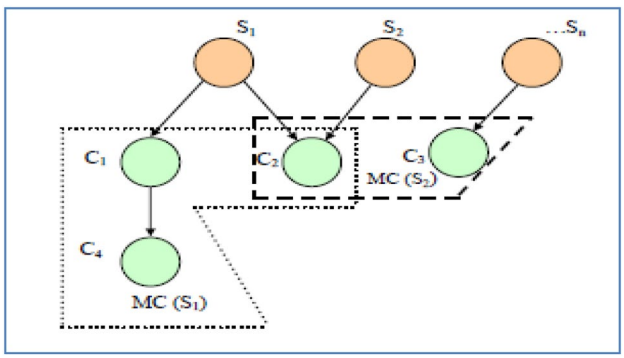


Fig. 5 Minimal Cover method of USCE–FACS

3.3.6 Pseudo code for minimal cover of functional dependency

F is a Functional Dependency
MC is Minimal Cover
MC=F
Initialization
Initially, MC is equal to F
For Each
FD S→ C_1 C_2 ……C_n in MC
Do
Replace S→ C_1 C_2 ……C_n by
S→ C_1
S→ C_2
.....
S→ C_n
Remove unnecessary components of each functional dependency
For Each
FD S→ C_i in MC
Do
For Each
Attribute B belongs to S
Do
Z={S-B)
Tentatively replace S→ C_1 by Z→ C_1
If
The resulting set of FD is equivalent to MC
Replace S→ C_1 by Z→ C_1 in MC
Remove unnecessary functional dependencies
For each remaining FD S→ C_i in MC Do
Tentatively remove S→ C_i from MC
If
Resulting set of FDs is equivalent to MC
Removes S→C_i from MC
End If
End If

3.3.7 Component detection unit

The CDU gets the result of the matching process of DPU. When the SQnot found in both UIHDB and Metadata Repository, it comes to CDU from DPU. This unit searches related User query in the domain ontology repository and finds the components from the server then passes it to the Component Composition Unit.

The CCU builds the service based on components provided by CDU and also stores the component in the component repository and also its metadata information in the Metadata Repository. The CCU builds the service and sends it to UCSDP, and the selected service sent to

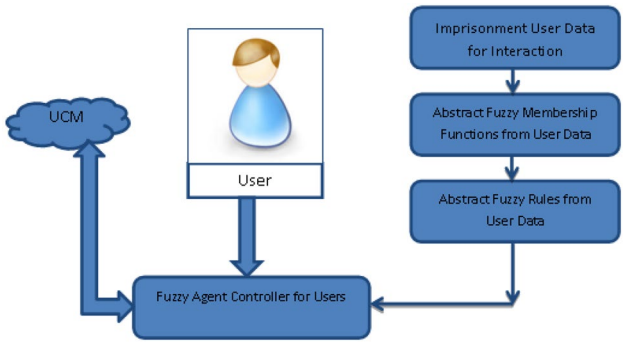


Fig. 6 Flow diagram of Semantic-based USCE–FACS

RP. The flow diagram of Semantic-based CDP is as shown in Fig. 6.

The Semantic-based CDP gets SQ input from Query Processing Unit (QPU) and sends it to the Discovery Process Unit. In the Discovery Process Unit, it compares SQ with UIHDB. It performs the parallel operation; if it exactly matches with UIHDB, it is directly fetched from Service Repository and given to UCSDP. If SQ related component found, but matches are not found exactly with UIHDB, then it is given to CCU. Otherwise, if it does not find any match, it is given to CDU. Then CDU finds the component from the server, sends it to CCU where it builds to service, and sends it to UCSDP and also to RP.

3.4 The rapid phase

The RP maintains the User Interaction History (Pokraev et al. 2006), which contains several user preferences and their related query details (Ma et al. 2010). In this phase, the clustering technique is used to similar group preferences of user interaction history, as a way to reduce the size of the history log. Once the clusters created, predictions for a target user can be made by averaging the opinions of the other users in that cluster. Due to a large number of available services, it gets to be time-consuming for end-users to discover appropriate services to fulfill their various needs.

To help end-users get their desired services, Clustering Collaborative Filtering Recommendation Approach provides a way to automate service discovery and recommendation. Users show regularities in almost everything they do so that it can predict the user action from the user interaction history based on Collaborative Filtering Recommendation approaches (Iroju et al. 2012). Collaborative Filtering relies only on past users' history. Collaborative Filtering Recommendation approach works based on the users' requests and users' preferences, and it recommends a user a service that is similar to those which the user liked in the past in user interaction history.

Collaborative Filtering Recommendation approach is to identify users that have relevant interests and preferences by calculating similarities and dissimilarities between users' profiles. Pearson's correlation used for calculating similarity when predicting the user's preferences. The RP works proactively with the help of the QPU by determining the preference of users.

3.4.1 The pseudo code for collaborative filtering recommendation approach is detailed

Begin
Get user preference from QPU
Get URQ (User Result Query) from Semantic-based CDP
If
User-preference precisely exists in UIHDB
If
There are more actions performed by User ID
RP predicts the user's next interest in UIHDB based on the preferences of the user and also URQ from Semantic-based CDP based on Collaborative Filtering Recommendation approaches.
Store it into recommendation table
Else If
It does not exist in UIHDB
Search semantically if there is a match in UIHDB Based on the Collaborative Filtering Recommendation approach it predicts the preferences of user based on the similar interest of other users and stores it into the recommendation table
End

The Rapid Phase guides the users based on the results from the Semantic-based Component Detection Phase and history of users in UIHDB. It can recommend the service that may be of the user's interest. By using the Collaborative Filtering Recommendation approach, the future need for UReqs determined, and the options in the Service Recommendation Table sent to the UCSDP. Then the user can select the required service and quickly execute it.

3.5 User-centric service delivery phase

The UCSDP receives the service from Semantic-based CDP and also from a service recommendation table with options from RP. It delivers the UReqing the service to the user through java plug-in and also displays a service recommendation table with options of predictions of user needs. If users need the service by selecting the options, the appropriate services downloaded and efficiently executed.

It delivers service to the user according to their platform in the form of plug-in and is ready to be executed as an application.

3.5.1 Pseudo code for UCSDP

Step 1 Get input from Semantic-based CDP and also RP.
Step 2 Send user-requested service to the user according to their heterogeneous platform and also display Service Recommendation Table with options to the user.

4 Proposed UCMCM–UESF algorithm

The steps of USCE–FACS Algorithm in Fig. 7 carried out as follows:

- UReqs query to User Interface Unit of Accessibility Phase
- The User Interface Unit accepts UReqing queries from heterogeneous platforms. It checks whether the user already gets registered his information in the user information database.
- If the User is a new user, the request sent to the Profile Manager, where the new user registers his information with his preferences. The Profile Manager stores the information about the user and provides a unique UserID to the user
- The Profile Manager sends the preferences of the user interest and UserID to Query Processing Unit
- The User Interface Unit sends the UReqing query to Query Processing Unit (QPU) if the User already gets registered.
- In the QPU
- It semantically converts the UReqing query with the help of ontology and sends it to DPU.
- It also extracts the preferences of the User interest from User Information Database based on UserID and also sends the preferences and UserID to RP

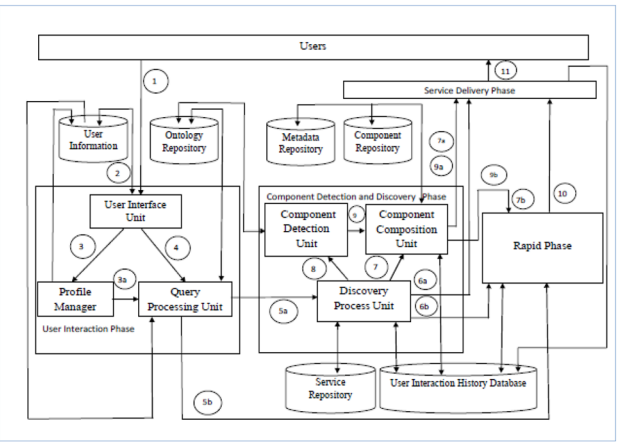


Fig. 7 Flow diagram of proposed USCE–FACS

- The DPU searches the SQ with the history of UIHDB. If the SQ exactly matches with UIHDB, it fetches the related SID from UIHDB. Based on that SID, it fetches the service from the Service Repository.
- Now the service is readily available and is given to UCSDP. Service Delivery Phase provides the service according to the user platform.
- The DPU sends the selected service to the RP.
- In the DPU, the SQ not precisely matched with UIHDB, but related components found, then it fetched that component from UIHDB and sent it to the CCU.
- In the CCU, it finds the components in the Metadata Repository; if found, then it fetches the ComponentID from Metadata Repository and builds service using MCFDM.
- The CCU builds the service with the selected component and sent to UCSDP.
- The CCU sends the selected service to the RP.
- In the DPU, if the SQ not found in both UIHDB and Metadata Repository, it goes to CDU. In that unit, it searches related user query in the ontology repository and finds the components from the server then passes it to the Component Composition Unit.
- The CCU builds the service based on components provided by CDU, stores the component in the component repository, and also its metadata information in the Metadata Repository.
- The CCU builds the service and sent to UCSDP.
- The selected service also sends to RP to predict the preference of the user.
- The RP guides the users based on the results from the Semantic-based Component Detection Phase and history of the user stored in a UIHDB. From that information, the RP can anticipate the user's future needs. Proactively, it can recommend the user, a service that may be of interest. The Semantic-based Component Detection Phase sends the selected service to the RP. By using Clustering Collaborative Filtering Recommendation Approach, the RP determine the future need of the user. As a result, a service predicted by the RP presented in the form of Service Recommendation Table and sent to the user.
- The Response of UReqs and the RP recommended service sent to UCSDP. Then the user can select the required service from the recommended service and execute it. After each successful service discovery process, the related information of UserID, SID, ComponentID, User Interest, and Device ID stored in UIHDB.

4.1 User interaction phase

The UIP consists of three units, such as the UIP unit, Profile Manager, and QPU. The jobs of UIP are to accept the

UREQing query, register a new user to get the personal information and preferences of the user, and provide the unique UserID. It also converts the UREQing query into SQ and then sends the SQ to CDDP. It also converts the UReqing query into SQ. The following steps are carried out in converting UREQ to SQ.

Step 1 Get an UREQing query in the heterogeneous platform.
Step 2 Parse the UREQ into separate tokens using an XML parser.
Step 3 Retrieve the semantic meaning of UREQ from ontology by using SPARQL query language.
Step 4 Send the SQ to CDDP.

Ontologies can be used to describe the semantics of information sources and make the content explicit and thus can be used to discover semantic equivalence between information concepts. Protégé tool is used to simplify the creation of Ontologies for the ubiquitous computing environment.

4.2 The CDDP framework

The CDDP consists of three units, such as DPU, Component Composition Unit (CCU), and Component Detection Unit (CDU). The CDDP of the USCE–FACS model is to accept the Semantic Query (SQ) from UIP. By using Discovery Process Matching (DPM) algorithm, it detects whether SQ is available in the UIHDB /Component Repository (CR) / Service Repository (SR). Based on the DPM algorithm, it processes the request and sends it to the SDP. The flowchart of CDDP framework is shown in Fig. 8.

4.3 Composition of fuzzy agent procedure

The creation of the Fuzzy Set Values (FSV) and their related principles into the level of participation for every one of the yield sets finished by taking the MAX fuzzy value related to every one of the yield sets determined for every one of the information sources. This profits a variety of qualities for every one of the yield sets that can be utilized to decide the result value in the following part, defuzzification.

4.3.1 Pseudo code for composition process

The composition procedure is straightforward, just placing values of fuzzy into an array demonstrating to the yield sets if the worth is more prominent than the one as of now in there. This procedure can be practiced by the following Algorithm 1 and 2.

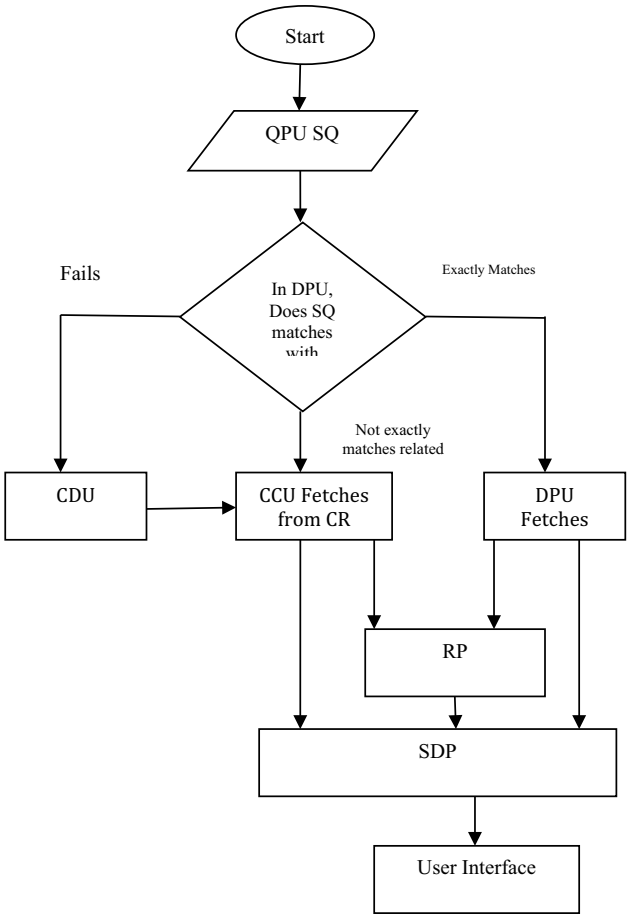


Fig. 8 Flowchart of CDDP framework

- In the fuzzification process, FSV is an array containing the FSV and rules calculated where [0] to [3] are the fuzzy values and rules for an input.
- An array output that contains the MAX values for each of the result sets where [0] is set 1, [1] is set 2, and so forth.

In turn, this process for each of the inputs done; after all the inputs have processed, finally getting the MAX values for each of the result sets.

4.3.1.1 Algorithm of pseudo code for fuzzification process

```
For Fuzzy Sets (FS) = Max FSV
   If FSV [FS + 1] > −1
      If Output [FSV (FS + 1)] <FSV [FS]
         Output [FSV [FS + 1]] = FSV [FS]
      End IF
         Increment FS by 2
End For
```

4.3.2 Defuzzification process

The defuzzification process takes the array generated by composition and returns the final crisp value that is used to determine the rating of the action. It uses the mean of the maximum method of defuzzification to calculate the single output value, based on the max values of each of the output sets and the FSV for that set.

4.3.2.1 Pseudo code for defuzzification process The defuzzification process uses the output, and the stored data for the output sets to calculate the output value.

4.3.2.2 Algorithm of pseudo code for defuzzification process

For Fuzzy Sets (FS)
Max FSV = Output [FS]* Output [FS]
Min FSV = Output [FS]
End For
Mean value of FSV = Max FSV / Min FSV
Formulas for calculation trapezoidal fuzzy memberships

$$\begin{cases} m_A(x) = 0 \text{ if } x < a_1, \\ m_A(x) = \frac{x-a_1}{a_2-a_1} 0 \text{ if } a_1 \le x \le a_2, \\ m_A(x) = 1 \text{ if } a_2 \le x \le a_3, \\ m_A(x) = \frac{a_4-x}{a_4-a_3} \text{ if } a_3 \le x \le a_4, \\ m_A(x) = 0 \text{ if } x > a_4 \end{cases}$$

5 Results and discussion

The work is demonstrated by using Vicon Physical Action (3000 instances and 27 attributes) and Iris Plant (150 instances and 4 attributes) is chosen from UCI repository Datasets. The performance evaluated by the following metrics.

(a) Number of Components (NC)-NC required for UREQ
(b) CBT-Time is taken to build the components required by the user
(c) Users Load Demand Variance (ULDV)-MC of number of applications required by the user
(d) Number of Required Services (NRS)- Number of Services required by the user

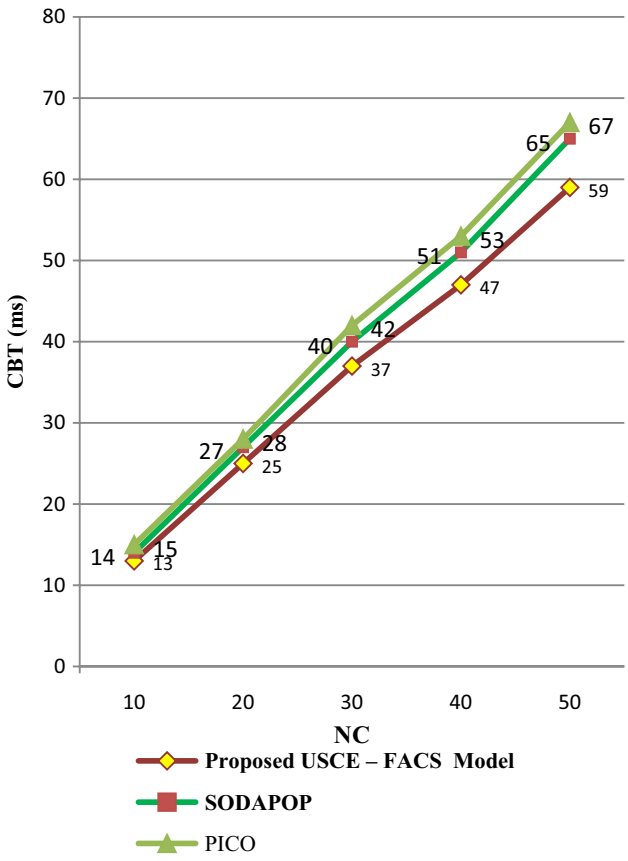


Fig. 9 Performance of CBT

Table 1 Performance of ULDV

Service required count	ULDV(in KB)		
	Proposed USCE–FACS model	SODAPOP model	PICO model
2	950	980	1020
4	1600	1650	1735
6	2150	2250	2354
8	2590	2760	2860
10	2975	3207	3284
12	3890	4212	4320

5.1 Performance of component building time

The CBT based on the NC provided using the proposed USCE–FACS and the comparison made with SODAPOP shown in Fig. 9. An increase in NC increases CBT.

For the purpose of the evaluation of Component Building Time the following measures are defined: CT_Begin is the time at the beginning of the invocation of device and CT_Search is the time to search a specific component in a component repository. Using those measures the component

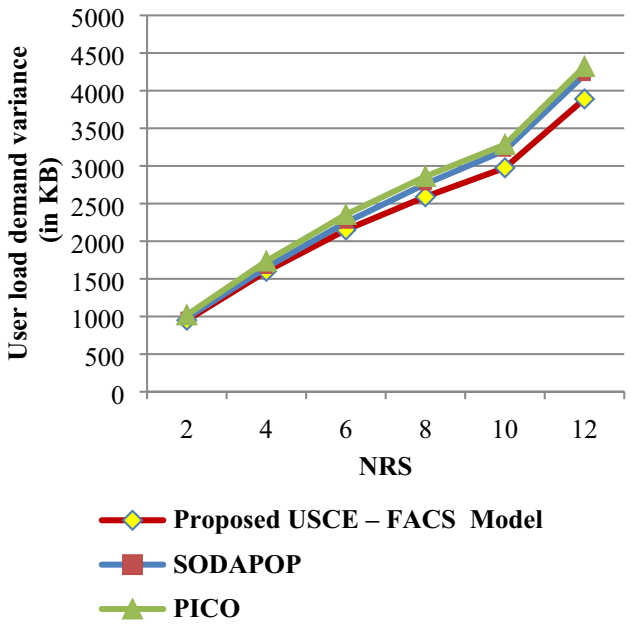


Fig. 10 Performance of ULDV

Table 2 Performance of service delivery time

Number of required services	Number of components	Service delivery time (in ms)		
		Proposed USCE–FACS Model	SODA-POP Model	PICO Model
2	20	24	25	26
4	40	50	53	54
6	60	75	80	82
8	80	100	108	110
10	100	125	136	139
12	120	150	165	168

building time can be defined as CT_Build = CT_Search–CT_Begin. The CBT for the USCE–FACSis comparatively lower when compared to the existing model. It provides a user-centric component based on MCFDM, and also it searches UIHDB based on clustering, which subsequently reduces the CBT.

5.2 Performance of ULDV

ULDV per NRS by the user illustrated in Table 1 and Fig. 10.

The MC of NRS is comparatively less in the proposed USCE–FACS than the existing model of SODAPOP

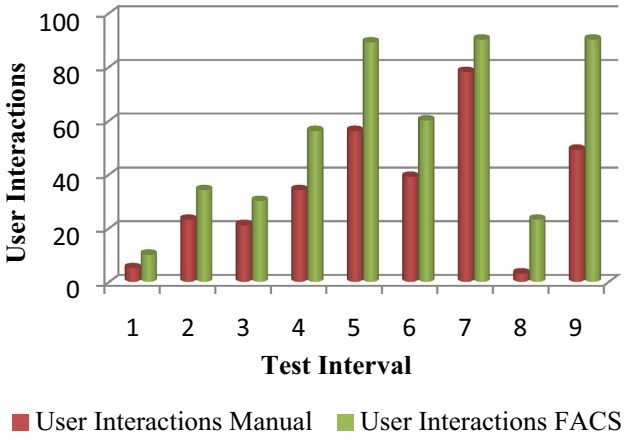


Fig. 11 The received packets against experimental time

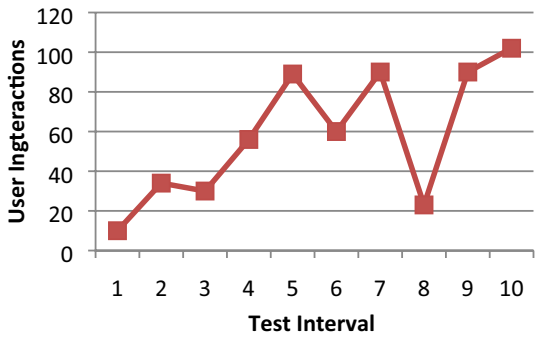


Fig. 12 The number of manual interaction with the proposed model

because of the USCE–FACS uses MCFDM for service composition to reduce the MC of the service requested by the user. The MC of NRS by the proposed model is 3–7% less than that of SODAPOP.

5.3 Performance of service delivery time

Table 2 shows the performance of Service Delivery Time for the Number of Required Services contains Number of components and also the comparison of Service Delivery Time of the proposed UCMWC, existing SODAPOP and PICO model.

5.4 FACS performance analysis

Every day in this trial period, we evaluated framework exhibition two times every day, which utilized our specialist based arrangement by contributing 60 reproduced information, and afterward distinguished whether the yields are right by a human. In the interim, to demonstrate the accessibility of the learning instrument utilizing hybrid AI

methods. Additionally, we built up the arrangement with no fuzzy module under a similar circumstance, and afterward, we contrasted our framework and that by including the overall received packets in every 24 h. Figure 11 explains the consequence of this examination.

The non-fuzzy arrangement's over-burden is a lot higher than that of our useful framework. Also, during our trial time, the two arrangements perform steadily in each predefined interval. Besides, the quantity of manual association with our proposed framework was recorded by us every day. Inferable from the presence of the foreseeing and choice module, the collaborations were diminished evidently, as per Fig. 12.

6 Conclusion and future work

The proposed USCE–FACS designed for reducing CBT and MC using UCE for soft computing. It increases the reusability of components and services in a heterogeneous network. It is a more user-friendly manner because of RP. It recommends the user needed service. The proposed work is to satisfy the user's need by discovering, detecting, and recommending the most suitable service based on a user-centric approach. We completed special tests in which a user remained in the USCE –FACS. The disconnected and online exhibition of the FACS indicated that learning the practices of a user and adjusting and tuning its guidelines online to meet the user's inclinations when the natural conditions had altogether changed. The FACS FLC had additionally utilized less number of rules than the common sort fuzzy logic control. The FACS was subsequently ready to display and limit the impacts of vulnerabilities to create a superior in the general execution of the framework.

The limitation of proposed work is no integrated architecture for the various services like scalability, fault tolerance, flexibility, dynamicity, and reusability nor a unified global framework for addressing various needs of users. The future work focused on the design of such a full-fledged architecture. We proposed an automated, designed process for future work for generating FACS sets directly from data.

References

Al Saiyd AN, Said IA, Takrori AAH (2010) Semantic-based retrieving model of reuse software component. Int J Comput Sci Netw Secur 10(7):154–161

Ao EJ, Kirste T (2005) Ambient intelligence- towards smart appliance ensembles. In: Hemmje M, Niederee C, Risse T (eds) From integrated publication and information systems to information and knowledge environments. Springer, Berlin, pp 261–270

Baumgart I, Hartmann F (2011) Towards secure user-centric networking service-oriented and decentralized social networks, self-adaptive and self-organizing systems. In: IEEE 2011 conference on self-adaptive and self-organizing systems, pp 3–8

Bhushan K, Gupta BB (2019) Distributed denial of service (DDoS) attack mitigation in software defined network (SDN)-based cloud computing environment. J Ambient Intell Human Comput 10:1985–1997. https://doi.org/10.1007/s12652-018-0800-9

Debashis S (2005) Pervasive computing: a vision to realize. vol 64, Elsevier, Amsterdam

Diro AA, Reda HT, Chilamkurti N (2018) Differential flow space allocation scheme in SDN based fog computing for IoT applications. J Ambient Intell Human Comput. https://doi.org/10.1007/s12652-017-0677-z

Doctor F, Hagras H, Callaghan V (2005) A fuzzy em-bedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments. IEEE Trans Syst Man Cybern Part A 35(1):55–65

Hagras H et al (2003) A hierarchical fuzzy genetic multi-agent architecture for intelligent buildings learning, adaptation and control. Int J Inf Sci 150:33–54

Hagras H, Doctor F, Callaghan V, Lopez A (2007) An incremental adaptive lifelong learning approach for type-2 fuzzy embedded agent in ambient intelligent environments. IEEE Trans Fuzzy Syst 15(1):41–55

Illingworth FR, Callaghan V, Hagras H (2005) A neural network agent based approach to activity detection in AmI environments. In: Proceedings of IEEE international workshop on intelligent environments (IE05)

Iroju O, Soriyan A, Gambo I (2012) Ntology matching: an ultimate solution for semantic interoperability in healthcare. Int J Comput Appl 51(21):7–14

Jang H, Song C-H, Kim MJ, Eom YI (2011a) Agent-based intelligent middleware for user-centric services in ubiquitous computing environments. J Inf Sci Eng 27:1729–1746

Jang H, Song CH, Kim MJ, Random YI (2011b) Agent-based intelligent middleware for user-centric services in ubiquitous computing environments. J Inf Sci Eng 27(5):1729–1746

Kalasapur S, Kumar M, Shirazi BA (2007) Dynamic service composition in pervasive computing. IEEE Trans Parallel Distrib Syst 18(7):907–918

Kim JA, Park DG, Jeong J (2019) Design and performance evaluation of cost-effective function-distributed mobility management scheme for software-defined smart factory networking. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-019-01356-5

Ma B, Yang Y, Zhou X, Zhou J (2010) An ontology based semantic retrieval model for uyghur search engine. In: the proceedings of IEEE symposium on web society, pp 191–196

Machado C, Silva E, Batista T, Leite J, Nakagawa EY (2013) Architectural elements of ubiquitous systems: a systematic review. In: The eighth international conference on software engineering advances

McIlraith S, Son T, Zeng H (2001) Semantic web services. IEEE Intell Syst 16(2):46–53

Molina AJ, Koo HM, Ko IY (2007) Template-based mechanism for dynamic service composition based on context prediction in ubicomp applications. In: IWBT 2007 in conjunction with the 19th IEEE international conference on tools with artificial intelligence, pp 1–8

Pokraev S, Quartel D, Maarten WA, Steen MR (2006) Requirements and Method for Assessment of Service Interoperability. Springer, Berlin, vol 4294, pp 1–14

Raman B, Katz RH (2003) An architecture for highly available wide-area service composition. In: Computer communications journal, special issue on recent advances in communication networking

Raychoudhurya V, Cao J, Kumar M, Zhang D (2013) Middleware for pervasive computing a survey, pervasive and mobile computing. Mob Comput J 9(2):177–200

Santos OC, Boticario JG (2011) Requirements for semantic educational recommender systems in formal E-learning scenarios. Algorithms 4:131–154

Shetty SM, Shetty S (2019) Analysis of load balancing in cloud data centers. J Ambient Intell Hum Comput 10:1–9

Siano P, Graditi G, Atrigna M, Piccolo A (2013) Designing and testing decision support and energy management systems for smart homes. J Ambient Intell Hum Comput 4(6):651–661

Soliman HA, Coude P, Simoni N (2011) User-centric and QoS based service session. In: IEEE 2011 Asia-Pacific services computing conference, pp 267–274

Wirsing M, Sebastian SB, Schroeder A (2010) Modeling and analyzing adaptive user-centric systems in real-time mode. In: RTRTS 2010 first international workshop on rewriting techniques for real-time systems, pp 1–25

Xhafa F, Wang J, Chen X, Liu J, Li J, Krause P (2014) An efficient PHR service system supporting fuzzy keyword search and fine-grained access control. Soft Comput 18(9):1795–1803. https://doi.org/10.1007/s00500-013-1202-8

Yang M, Li Y, Jin D, Yuan J, You I, Zeng L (2014) Opportunistic sharing scheme for spectrum allocation in wireless virtualization. Soft Comput 18(9):1685–1697. https://doi.org/10.1007/s00500-014-1267-z

Yngve D (2007) Ubiquitous computing at point of care in hospitals: a user-centered approach, Ph.D. thesis, Norwegian University of Science and Technology, Trondheim

Zhao YW, Chi C-H, Ding C (2011) User-centric homogeneity-based clustering approach for intelligence computation. Commun Comput Inf Sci 136:364–372

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.