EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

A context-aware multimedia service scheduling framework in smart homes

Yunji Liang[1], Xingshe Zhou[1], Zhiwen Yu[1*], Haipeng Wang[1] and Bin Guo[1,2]

Abstract

Many pervasive services are embedded in smart homes. It is a great challenge to schedule these services due to the heterogeneity and the mobility of devices. In this article, we elaborate a context-aware service scheduling framework (CASSF), which discovers available services hosted on those heterogeneous and mobile devices, selects suitable services that meet task requirements (TR) and ultimately enhances user experience with content adaptation. Contributions of this article are in three folds. First, the device ontology is designed to describe device features including built-in services, which benefits the knowledge sharing and facilitates the service discovery; second, a context-aware service selection mechanism is proposed, in which available services and contexts including user situations, TR, device capabilities, etc. are taken into consideration. Furthermore, due to the heterogeneity and the diversity of devices, the content adaptation is introduced in the framework to improve user satisfaction. Based on the CASSF, a multimedia prompting system is implemented to assist the elderly with intelligent services, and the evaluation of the proposed framework is performed in term of the execution time. The experimental results showed the framework is feasible and effective.

Keywords: context-aware, service scheduling, multimedia prompting, smart home

1. Introduction

The population of the elderly is growing at a phenomenal rate. In view of cognitive decline, especially, memory, hearing, and visual impairment for the elderly, the Smart Home (SH) has emerged as a potential approach to assisting old people with lots of intelligent services which make it possible for them to live independently [1]. With the vision of ubiquitous computing becoming reality, people will soon live in environments surrounded by networked computers and mobile devices [2]. Many intelligent services are deployed in those devices. However, those devices differ from each other significantly in terms of computational capabilities, network protocols, and available intelligent services. Furthermore, lots of handheld devices such as smart phones and Personal Digital Assistants (PDAs) serve as terminals. The dynamics and the mobility of those devices have significant effects on the access of built-in services. For example, when the host is ready to go to work with a smart phone, a service is triggered which aims to remind the host to purchase milk and fruit on the way home. This scenario raises the following issues: (1) Where to present the prompt message. As there is no knowledge about what kinds of devices are available in the current situation. (2) Even though the available devices are known, in the aforementioned scenario it's the smart phone, how to access the built-in services in a given device is implicit. The mechanism to match task requirements (TR) and service capabilities is necessitated. (3) Due to the constraints of devices and user preferences, a reasonable way to show the information is crucial. To enhance the user satisfaction, content adaptation is introduced. Thus, in order to overcome all those problems and manage intelligent services in SHs, a context-aware service scheduling framework (CASSF) is required. The CASSF discovers accessible devices, provides semantic service selection to match TRs against service capabilities and allocates suitable services according to current contexts and user preferences.

First, it should be emphasized that a device is a container of services. The device discovery in the CASSF not only explores available devices in the current

* Correspondence: zhiwenyu@nwpu.edu.cn
[1]School of Computer Science, Northwestern Polytechnical University, Xi'an, P. R. China
Full list of author information is available at the end of the article

location, but also presents the descriptions of built-in services and benefits the ensuing service selection. Therefore, a comprehensive description of the device is prerequisite for the device discovery. Although some approaches such as CC/PP [3] and FIPA [4] were proposed, most of them just describe partial properties of devices and have low scalability. Therefore the semantic description of devices, which completely depicts properties of devices including built-in services, is necessitated.

Second, it is a challenge to select a suitable service for tasks due to the differences of available devices. In general, the process of allocating suitable services is time consuming. When contexts are concerned, the decision-making process may filter lots of unrelated resources in the process of service selection. For instance, location is one kind of important contexts in this decision-making process. A limited amount of information covering a person's proximate environments is more important since the interesting part of the world around us is what we can see, hear, and touch [5]. Thus, the CASSF should make full use of user contexts in the process of service selection and allocate suitable service for tasks.

Furthermore, differences among TRs, device constraints, and user preferences make the content adaptation necessary in the CASSF. For instance, a reminding service is ready to show a shopping list for a sight-impaired person. Due to the visual defect, the information should be transformed into voice message and the voice service hosted on the smart phone is invoked, meanwhile the volume of the smart phone is turned up. In this scenario, the content adaptation plays an important role in the process of improving user satisfaction. The content adaptation provides a mechanism to change the content properties according to service capabilities and device constraints.

To address above issues, this article proposes a CASSF for SHs, which discovers available services based on the general device ontology and allocates suitable services according to user contexts. Considering that the allocated services might not exactly meet the needs of user tasks, we introduce the content adaptation to guarantee tasks performed smoothly on specific devices and to enhance user experience.

The rest of this article is organized as follows. Section 2 reviews related work in the field of service scheduling, including device description, service discovery and service selection. Section 3 elaborates the architecture and the workflow of the CASSF. In Sections 4 and 5, we focus on the details of two modules, service discovery, and context-aware service selection. A prototype system based on the CASSF and the evaluation of this framework are presented in Section 6. Finally conclusion and future study are given in Section 7.

2. Related work

In the CASSF, three parts are involved including the device description, service discovery, and service selection. In this section, overviews of existing approaches about the three aspects are presented.

2.1. Device description

In general, there are lots of devices embedded in SHs. Here the term device means the combination of physical components and services hosted on the concerned physical device. All those devices differ from each other significantly ranging from basic construction to built-in services. The heterogeneity of devices makes it difficult to access built-in services. Therefore, the universal and semantic device description of heterogeneous devices is necessary. However, there has been no much work to come up with a framework to semantically describe devices in order to make semantic discovery possible and effective. Previous study has tackled device descriptions for different purposes.

FIPA Device Ontology [4] specifies a frame-based structure to describe heterogeneous devices. However, it merely describes features/properties of physical devices, such as the screen resolution, the memory size, etc. A device is a container of services. To describe a device in detail, services hosted on the concerned device are also necessitated in the description. The CC/PP specification [3] proposed by the W3C is based on the resource definition framework, whose small set of constructs for representing metadata limits the information that can be presented and inferred. In addition, the XML was introduced to describe a device in [6] for content adaptation. This structured-language is not adequate to describe heterogeneous devices due to the weakness of scalability. The Mobile Web Initiative Device Description Working Group focuses on the device description in order to ease and promote the development of web content. The device description structures proposed for content adaptation for the mobile devices just take into account a range of factors such as screen size and image format and do not support the knowledge reasoning from those device descriptions. In this article, we design the semantic device ontology to describe device features including built-in services and physical properties. This device description not only contributes to the service discovery, but also facilitates the content adaptation in the CASSF.

2.2. Service discovery

Over the past decades, many resource management frameworks have been proposed, especially the service discovery protocols including Jini [7], UPnP [8], SLP [9], UDDI [10], and Bluetooth [11]. Many of those protocols are based on the XML schema or the attribute/value

pairs to describe services. The overview of those protocols is presented in Table 1.

One common limitation to them is that they are initially designed based on specific assumption about the underlying network, the user behaviors, or the application needs [12]. Although aforementioned service discovery protocols make use of many methods for service description, most of them are syntax-level and there is not enough semantic information included in the service description [13]. In addition, few contexts are taken into consideration in the design of service discovery framework. We argue that contexts will be very useful in the process of context-aware service selection. For example, in the case of a reminder service, the location of the user and the available terminals in the current location are very useful in the process of determining a suitable terminal to represent the reminder services. In the CASSF, service discovery is based on the semantic device description. By parsing the device description, services in the current situation and states of devices are available.

2.3. Service selection

There are various pervasive services embedded in SHs to benefit our daily life, especially for the elderly. The service selection is necessitated which performs a matching algorithm to select suitable services according to TRs and service capabilities. Currently, many service selection mechanisms have been proposed. All of them are classified into two types: the syntactic service selection and the semantic service selection. The syntactic service selection is keyword-based matching mechanism generally. This mechanism depends on the exact description of service properties, such as interfaces, attributes, or service types (see Table 2).

The syntactic service selection is not conducive to the design of transparent services. An application needs to explicitly assign required services or interfaces, which consequently leads to the tight couple between tasks and required services and reduces the scalability of the system. Also, the syntactic service selection leads to low recall and low precision of the retrieved services as query keywords might be syntactically different from the terms in service descriptions [14,15]. Thus, the semantic service selection based on ontology is proposed.

Table 2 Comparison of syntactic service matching methods [19]

Architecture	Service description and matching
Jini	Interface/attribute string comparison
UPnP	Service name string comparison
SLP	Service type/attribute string comparison
Salutation	Service type/attribute string comparison
Bluetooth SDP	Service type/attribute string comparison
UDDI	Keyword string comparison

The semantic service selection uses the semantic description of services rather than specific strings or attributes to match TRs. The semantic service selection includes the functionality-based and process-based service matching methods. Functionality-based service selection determines the semantic similarities between user requirements and service profiles. The service profile includes descriptions of inputs, outputs, preconditions and effects. Similarities among those parameters are calculated to measure correspondences among requirements and services. Different formulas of similarities are elaborated in [16-18]. Process-oriented service selection determines the extent to which the desired operational behavior of a given service in terms of its process control and data flow matches with that of another services [19].

Although both kinds of semantic service selection are not keyword-based comparison, only profiles of services and requirements are taken into account. It is believed that user contexts will benefit the selection of target services. For example, in the case of a reminder service, the user location and available terminals in the current location are very useful in the process of determining a suitable terminal to represent the reminder services. Thus, in the service selection, user contexts should be employed. The CASSF utilizes the Semantic Web Rule Language (SWRL) [20] and Description Logic (DL) [21] to compare TRs with device capabilities, overcomes the weakness of keyword-based matching and employs more contexts to facilitate the service selection.

Table 1 Summarization of service description mechanisms [28]

Service description methods	Service discovery protocols
XML	UPnP [8], SSDS [29], GSDL [30], Konark [31], UDDI [10]
Attribute/Value	Salutation [32], Bluetooth [11], PSDL and PSQL [33]
Formal Grammer	SLP [9], DEAPspace [34]
DAML + OIL	GSD [35]
Ontology	UBIDEV [36], Ronin [37]
RDF	GloServ [38]

Furthermore, content adaptation is essential due to the heterogeneity of device capabilities. The content adaptation makes sure a user task is executed smoothly on the target device, provides high quality of services and enhances user experience. However, to the best of the authors' knowledge, few SSFs mentioned the function of content adaptation. Content adaptation is introduced in the CASSF to ensure the task is executed smoothly on the specific device.

3. Context-aware service scheduling framework

3.1. System requirements

Mobility and heterogeneity are features of devices in SHs. To efficiently manage devices in SHs, the CASSF should offer functions to discover available services in heterogeneous networks, allocate appropriate services to execute user tasks in terms of contexts and provide high quality of services to enhance user satisfaction. To achieve those goals, requirements of the CASSF are depicted as follows.

3.1.1. Device representation

The significant differences among hardware devices make it impossible to handle applications for each special device individually. In order to get rid of the heterogeneity of devices, a unified representation of devices is needed. The unified representation describes not only the composition of devices including hardware components and software components, but details of the external accessible interfaces and other important information such as the location and device type. The unified representation of devices abstracts all information of physical devices in a standard format, which benefits the understanding of devices and development of applications. More importantly, discrepancies among physical devices can be hidden by using the uniformed representation.

3.1.2. Service discovery

The mobility and the heterogeneity of devices in SHs lead to the ever-changes of available services. It is infeasible to allocate services for user tasks in advance due to the changes of user location and that of available devices. Therefore, the service discovery, during which the services hosted on the available devices are discovered based on the unified device description, is necessitated. The service discovery is responsible for getting the accessible services in SHs. Furthermore, more contexts are captured to fully understand the environment and facilitate the ensuing service selection.

3.1.3. Service selection

Allocating suitable services according to user contexts is a challenge. The mobility of devices makes it impractical to allocate a specific service for an application in advance. The SSF should be context-aware, which allocates the suitable devices for tasks according to current contexts. For example, in the case of a reminder service, the location of the user and the available terminals in the current location are very useful in the process of determining a suitable terminal to represent reminder services. The decision-making process takes user tasks, user preferences and contexts as inputs, matches device capabilities against TRs, and allocates suitable services to execute user tasks. The context-awareness makes fully use of the dynamic contexts and shields the mobility of physical devices.

3.1.4. Content adaptation

Due to the dynamics and the mobility of devices in SHs, available devices are not likely to fully match the desired requirements. There are two reasons for those mismatches. First, the needed service does not exist in the SH. For example, a user task needs to play message by voice, but the speech service is not available in the current situation. Therefore, the message needs to be transformed according to those available services; Second, it is difficult to obtain user satisfaction by invoking the services on the target device directly. For instance, a user task needs to represent a picture on the cell phone. Thus, the size of the picture should be rescaled according to the resolution of target device. In such conditions, the content adaptation not only makes the task executed smoothly on the allocated device, but also provides high-quality services for users and enhances the user acceptance.

3.2. CASSF architecture

As shown in Figure 1, the CASSF includes the following components: (1) Device Discovery Manager (DDM) discovers devices within heterogeneous networks and provides the semantic description of accessible devices in the environment. The semantic description not only presents the basic properties of physical devices, but

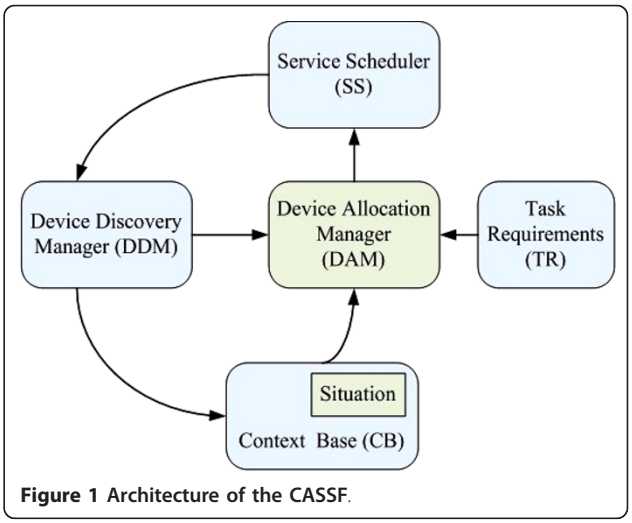

Figure 1 Architecture of the CASSF.

also describes the available services hosted on those devices. (2) Context Base (CB) stores all contexts including user situation (e.g., user location) and preferences. CB not only provides states of entities, but also serves as inputs of the rule-based inference engine. (3) TR abstracts needs of user tasks and applications. The TR includes numerous semantic parameters that specify the desired functionalities and optional device constraints. (4) Device allocation manager (DAM) allocates suitable devices to execute a user task according to three sources of inputs: DDM, CB, and TR. The output of DAM is a set of available devices, which probably includes more than one device to execute a task collaboratively. (5) Service scheduler (SS) invokes services on the allocated devices according to requirements of user task. In addition, the content adaptation is taken into consideration in the SS. SS is responsible for the data preprocessing according to capabilities of allocated devices.

In CASSF, DDM is independent, which dynamically discovers resources in SHs periodically and aggregates the states of devices. When the state of a device changes, the device will post a message to the DDM and the state will be stored in the CB. In addition, the DDM provides query interfaces for DAM to query available devices. When an application or user task is generated, the TR abstracts needs of the task in the form of task ontology and passes the description of the task ontology to the DAM; On receiving the task description, DAM gets the available devices by calling the query interfaces of DDM, then compares TR with capabilities of available devices, and ultimately allocates a suitable device to perform the task. To make sure the task executed smoothly on the target devices, the data preprocessing for content adaptation is optional.

The dataflow of the CASSF is illustrated in Figure 2. The logic stream of the CASSF includes four steps. (1) Service discovery obtains available devices in the current environment and employs device descriptions to construct the device ontology. The device ontology not only describes properties and states of devices, but indicates built-in services in the concerned device. The design of the device ontology will be elaborated in Section 4. (2) Service selection performs the match of TRs and available services capabilities. The process of service

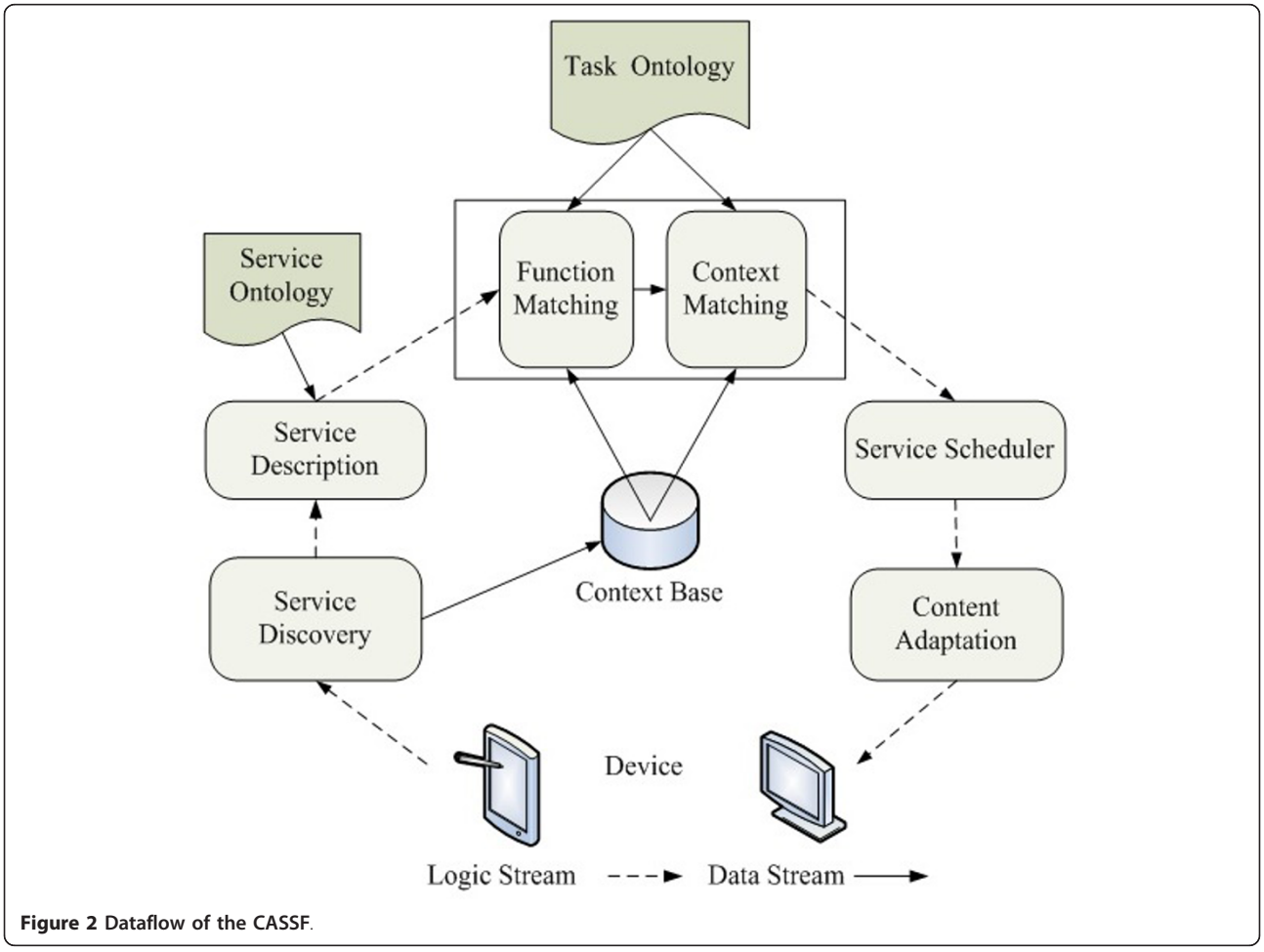

Figure 2 Dataflow of the CASSF.

selection consists of two components: the function matching and context matching. The function matching gets the candidate services that meet requirements of the task; the context matching employs user contexts to get reasonable services from numerous candidate services. Details about the service selection will be embodied in Section 5. (3) SS invokes the allocated service on the specific device. After getting the target service, the SS accesses the service and related parameters are sent to the service. (4) Due to constraints of allocated devices, the mismatch of TR and service capabilities is common. For example, task needs to present an image with resolution of 1024 × 768 on a handheld device. Thus, the content adaptation ensures the service executed smoothly and enhances the user experience.

4. Service discovery in CASSF

4.1. Ontology-based device description

To facilitate the management of devices, the device ontology is introduced to completely describe properties of devices. In distributed environments, the ontology provides a unified representation of device information which benefits the knowledge sharing among applications. Furthermore, the device ontology will be necessary when a service needs the information of a hardware device for service selection. For example, in the case of a reminder service, the screen resolution of the device may be useful in the process of determining suitable devices and the network contexts are used to determine the appropriate presentation method. In addition, it is necessary to harness the inferential benefits of logical reasoning over such description in order to allocate suitable devices for user tasks.

In order to completely describe the details of a device, the device ontology contains four components (see Figure 3): *hardware component*, *software component*, *network component*, and *service component*. In the *hardware component*, it presents the information of hardware, such as CPU, memory, storage and screen resolution, etc. The *software component* describes the information of software installed in the device, e.g., the name, vendor, and version of software. The *network component* shows the network protocols the device supports, which is very useful in heterogeneous networks. The *service component* contains the interfaces provided by the device and functional descriptions about those interfaces. Through the *service component*, external modules can learn about the functions of target device and execute a specific service by invoking the interface. In addition, other free attributes are necessary such as the location and device type. Location will be required when service selection needs to consider the location of the device in choosing the right service [22].

Compared with other device ontologies, our device ontology overcomes the barrier that the term 'device' just represents the physical components. Here, the device is a container of services. This device ontology not only describes features of physical components, but also presents the services, network, and software hosted on the device. Based on this device ontology, we are able to completely describe the capabilities of a givendevice, easily make the decision to allocate suitable devices and conveniently access the services hosted on the target device.

4.2. Sevice discovery workflow

In general, devices in SHs can be classified into stationary devices and mobile devices. Stationary devices have powerful computing capabilities and may support several network protocols, such as TCP/IP, Wi-Fi, and Bluetooth. On contrast, mobile devices are more likely to be resource-constrained devices, which have limited computing resources and only support a certain kind of network protocols. To manage all those resources in an SH, the device discovery mechanism should support multi-protocols and provide query interfaces for external modules.

The DDM can be decomposed into several components as shown in Figure 4. The DDM includes *server*, *adaptor*, *device pool (DP)*, and *devices* distributed in heterogeneous networks. The *server* is in charge of discovering devices and monitoring states of *devices*. *Adaptor*, the bridge of heterogeneous networks, facilitates the transformation of messages. The *device* is the combination of physical devices and the device agent hosted on the concerned physical device. The device agent is responsible for communicating with the *adaptor*. The *device pool* is a persistent storage of description information about available devices. As Figure 4 shows, the *server* supports many network protocols and communicates with *devices* distributed in heterogeneous networks through *adaptors*.

The workflow of DDM is presented as follows. The *server* periodically broadcasts the device discovery command, which is predefined and unique in the system. The device discovery command is interpreted into other forms of broadcast messages that are meaningful in different network protocols by *adaptors*. On receiving the device discovery command, *adaptor* is triggered to launch the broadcast message in the local network. All devices located in the domain of the local network respond to the broadcast message and report the detailed descriptions of device capabilities. On the other hand, the *adaptor* listens to responses from devices and submits all those device information to the *server*. The *server* aggregates all the device information and updates the content of *DP*. *DP* is the storage of device

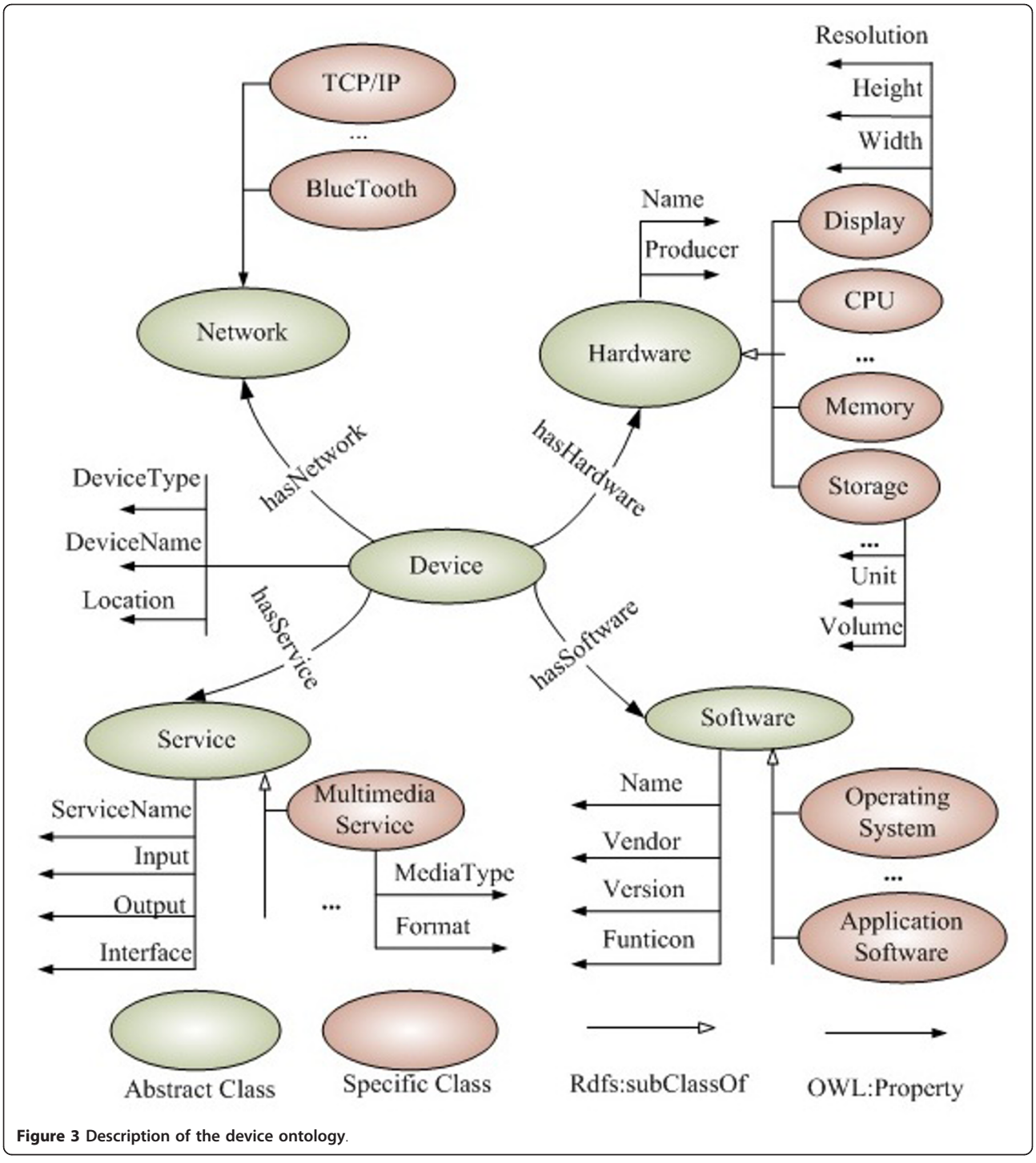

Figure 3 Description of the device ontology.

description, which enumerates all available devices. Meanwhile, descriptions about built-in services of specific device are also stored in the *DP*. By accessing to the *DP*, the CASSF performs the service discovery based on the semantic device ontology. In DDM component, all those components are invisible for external modules except the *DP*. The DDM provides public query interfaces to launch the information of available devices and services for external modules.

4.3. Quantifying the capability index

The device ontology describes capabilities of a device, and extracts features of a device in a unified format. However, the device description is high-level and

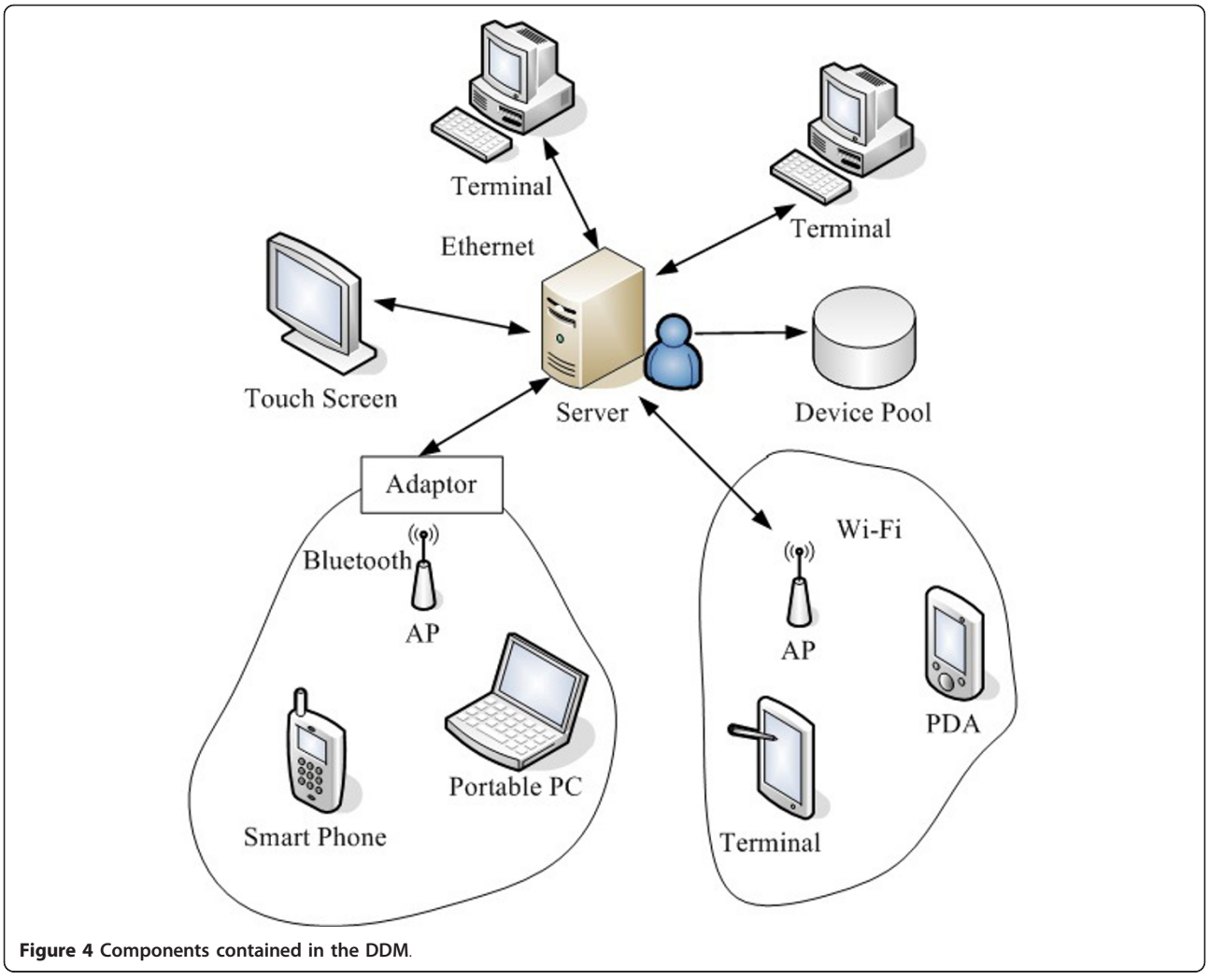

Figure 4 Components contained in the DDM.

semantic-oriented. The quantification of a device is needed to indicate the computing capabilities of a device in the device ontology. In order to quantify capabilities of a device, the *capability index* is introduced. The higher the *capability index* is, the more likely to provide high quality services for users. Therefore, the *capability index* is formalized as follows.

Device ontology, denoted with Ω, is a set of contexts. Each context, symbolized with c_i, is an element of the universal set Ω. The *capability index* is expressed according to Equation (1), where Q represents the *capability index*.

$$Q = F(c_1, c_2, \ldots, c_n) \tag{1}$$

To quantify the *capability index* of a specific device, there is no need to take all contexts into account. Several critical contexts are selected from the device ontology as parameters. Therefore, the *capability index* is calculated according to Equation (2), where the constant M represents the number of contexts and the variable β_i indicates the weight of the context c_i. In Equation (2), the variable β_i ranges from 0 to 1 and Equation (3) is satisfied.

$$Q = \sum_{i=1}^{M} \beta_i C_i \tag{2}$$

The symbol C_i, which ranges from 0 to 1, represents the normalization variable of the ith context c_i. For the normalization of the ith context, the C_i is defined as formula (4). Here, q_i is the attribute value of context c_i, $q_{\max}$ is the maximal value of the context c_i; correspondingly, $q_{\min}$ is the minimal value of the context c_i.

$$\sum_{i=1}^{M} \beta_i = 1 \tag{3}$$

$$C_i = \begin{cases} (q_i - q_{\min})/(q_{\max} - q_{\min}) & q_{\max} - q_{\min} \neq 0 \\ 1 & q_{\max} - q_{\min} = 0 \end{cases} \quad (4)$$

The *Capability Index* quantifies the capabilities of devices and provides a numerical presentation of devices properties. The *Capability Index* takes the physical properties and the load, etc., into account, and indicates the availability of the device.

5. Service selection in CASSF

Due to the dynamic changes of contexts in SHs, context-awareness is needed for the SSF. Many service selection methods were proposed in the previous study [16,17,19]. However, all those methods have shortcomings, such as the constraints of keyword-based selection and the over-reliance on the service descriptions in the semantic selection. Thus, we propose a two-step service selection method which takes user contexts into account and allocates reasonable services for tasks. The two steps in the context-aware service selection are named function matching and context matching, respectively.

5.1. Function matching

The function matching focuses on the degrees of match between TR and service capabilities. This study incorporates the semantic matching method introduced by Paolucci et al. [23]. They claimed that selection of web services should be based on the semantic match between a declarative description of the service being sought, and a description of the service being offered. And they calculated the similarities among the input/output of the requirement and that of service. The degree of match is determined by the minimal distance between concepts in the taxonomy tree and four degrees of matching are defined. More details about the function matching are elaborated in [23].

The function matching calculates the matching degrees of TR and service capabilities toward input and output descriptions. Through the function matching process, the candidate services are obtained and the size of target services is reduced. The function matching makes use of similarities between TR and service capabilities to determine the match degree. Services with higher match degree are selected as candidate services. After this process, many unrelated services are excluded from the candidate services which server as inputs for posterior context matching.

5.2. Context matching

The context matching allocates reasonable services for user tasks according to user contexts, service capabilities and TRs. As shown in Figure 5, the context matching is

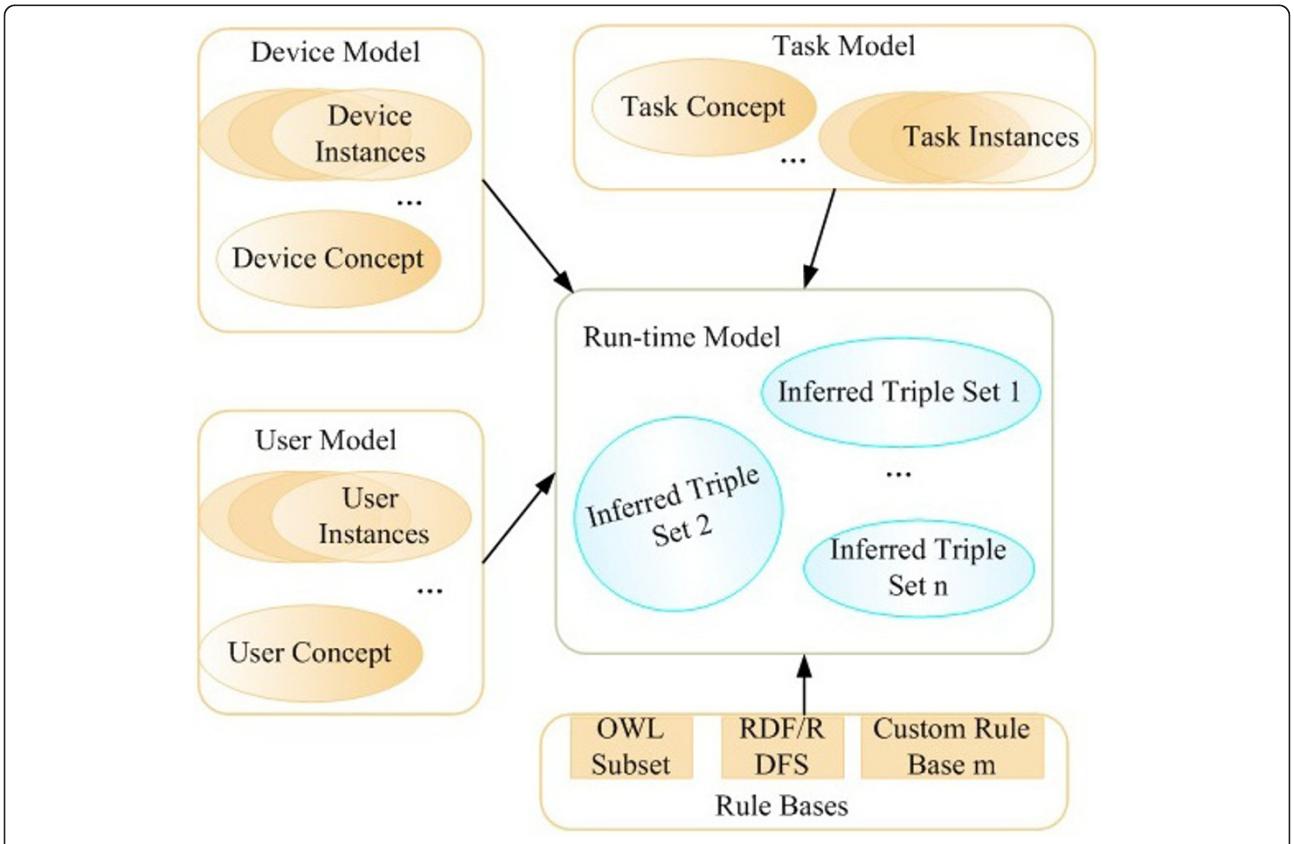

Figure 5 Compositions of the context-aware service matching.

composed of device model, task model, user model, run-time model, and rule bases.

The device model enumerates available devices in SHs and dynamical information according to the changes of device states. The device model consists of concepts and instances of devices. The concept of device model describes the knowledge of device and the relationships among entities. To represent a physical device and make the representation understandable for machines, we use the instantiated device ontology to describe a physical device. The instance of device model is a set of instantiated device ontology, which shows the current states of devices and lists capabilities of each device. The device model represents the information of the available devices in the current environment. Task model is the set of user tasks, and represents the ongoing to be executed tasks. Similarly to the device model, the task model includes the concepts and instances of tasks. The concept in task model represents definitions of user tasks, and the instance in task model details requirements and attributes of the task. When a new task is generated, a new instance of task will be added into the task model. The user model describes the contexts of the user, such as the location, preferences, etc. The user model fully describes the properties of users. All descriptions about users make it possible to construct personalized services for individuals.

The run-time model is the core of the architecture, which is responsible for matching TRs against device capabilities and allocating suitable services for user tasks. Allocating appropriate services for user tasks not only makes sure the quality of service, but enhances user acceptance for the service. However, how to select appropriate devices for tasks is still a challenge due to the complexity of contexts and TRs. The ontology-based reasoning is adopted for this purpose. We use the OWL for ontological model and representation. OWL is based on the well-developed knowledge representation formalism for DL. More importantly, DL supports the reasoning of concepts and infers the relationships between concepts.

Rule bases are definitions of relationships amongst concepts and provide criterion for new knowledge. In general, the rule bases not only contain the predefined rules in the SWRL domain, but also support the rule customization. Through rules defined in the rule bases, new knowledge could be inferred from existing knowledge. To facilitate the process of matching, the rule customization is necessary. Therefore, we defined a few rules as follows.

It is believed that a limited amount of information covering a person's proximate environments is more important for pervasive computing since the interesting part of the world around us is what we can see, hear, and touch [5]. Thus, the user location in the CASSF is regarded as a filter, by which all available devices are classified. Based on this principle Rule 1 is proposed and represented in the SWRL (see Table 3).

Table 3 Rule 1 and formalization in SWRL

	Definition of Rule 1
Semantic	The allocated resource should be close to users, which convinces users' interactions with devices.
Expression in SWRL	P*:Person(?x) ^ D*: Device(?z) ^ T*: Task(?q) ^ D:has Device Name(?z, ?t) ^ D: has Location(?z, ?s) ^ P: has Location(?x, ?y) ^ SWRLB: equal(?y, ?s) → T: has Device(?q, ?t)

*, "*P*", "*D*", and "*T*" are three namespaces, which are defined in the original ontology.

In Rule 1, the location is used as a filter. The property *hasLocation* of class *Person* and that of class *Device* are used to select devices from all the available devices in the SH. Only devices satisfying the Rule 1 are likely to be allocated for the tasks. Based on Rule 1, the CASSF maintains a list of available devices in current situation. All devices in the device list serve as candidates for user tasks.

In order to hide differences of interfaces, all user tasks are described in task ontology files, which indicate the requirements of tasks. For tasks, they do not need to care about the specific interfaces. They are just responsible for describing TRs. To match capabilities of devices and requirements of user tasks, Rule 2 is proposed and presented in SWRL (see Table 4).

To match requirements of user task and capabilities of devices, we use the similarity of class to make the decision. In the device ontology, the class *service* is defined, which describes features of a specific service. The *service* classes in device ontology are structured in a hierarchical tree with sub-class inheriting all properties from their super-classes. For example, the class *multiMedia-Service*, a sub-class of class *service*, illustrates features of multi-media service on the concerned device. According to Rule 2, the property *hasRequirement* of class *task* and the property *hasServiceName* of class *service* are compared with each other. If they are consistent, the concerned device will be added into the set of target

Table 4 Rule 2 and formalization in SWRL

	Definition of Rule 2
Semantic	The services with the similar or same name are consistent for user tasks and devices.
Expression in SWRL	T*: Task(?x) ^ T: has Requirement(?x, ?y) ^ D*: Device(?z) ^ D: Service(?s) ^ D: has Service(?z. ?s) ^ D: has Device Name(?z. ?t) ^ D: has Service Name(?s. ?a) ^ SWRLB: contains Ignore Case(?y, ?s) → T: has Device(?x, ?t) ^ T: all cate Service(?x, ?a)

*. "*T*" and "*D*" are namespaces defined in the original ontology.

Table 5 Definition of Rule 3

	Definition of Rule 3
Rule 3	The higher capability index, the better quality of service to perform the user task.

services, which stores all usable services to execute the user task. There might be more than one service in the set of target devices to execute the task. In this case, Rule 3 as shown below is proposed to select the service with higher capability index in order to provide better quality of service (Table 5).

In addition, to make sure the task executed smoothly on the specific device and enhance the user satisfaction, content adaptation is introduced. The content adaptation determinates the appropriate presentation method according to user preferences, user contexts and device constraints. We employ a generic and flexible $N \times M$-dimensional recommendation model elaborated in our previous study [24] to implement the content adaptation according to the user contexts and device capabilities.

6. Implementation and evaluation

We have built a multimedia prompting system based on the CASSF for the elderly. In this multimedia prompting system, the reminder task determines a suitable device according to device capabilities and the remainder information is presented in different forms such as image, voice, and text according to capabilities of available devices. Furthermore, user preferences are incorporated in the design of the prompting system. The multimedia prompting system is deployed in the Intelligent Assistive Technology and Elderly Care Lab (IATECL)–an SH in Northwestern Polytechnical University (NPU), China. The IATECL (Figure 6) aims to expand the time to live independently for the elderly by intelligent assistive technology.

The multimedia prompting system is implemented in the Java language. We adopted Open Services Gateway initiative (OSGi) [25] to construct the system architecture. All components in the system are encapsulated in the form of OSGi bundles and the communication among bundles is implemented by the event mechanism. We use Protégé [26] to construct the ontology files, including device ontology, task ontology and person ontology. In the processing of matching the capabilities against needs, we employ the Jena [27] API to implement the rule-based reasoning.

In general, the efficiency is the most sensitive problem of the rule-based reasoning system. Therefore, we evaluate the efficiency of the proposed framework based on the time consumption of the CASSF. The process of the CASSF is composed of three phases: (1) to load ontology files; (2) to load predefined rules; and (3) to match device capabilities against TRs and allocate suitable services to execute user tasks. The three phases are symbolized by P_1, P_2, and P_3, respectively. The execution of the CASSF is influenced by the size of instances and

Figure 6 View of the intelligent assistant technology and elderly care lab in NPU.

Table 6 Execution time of the CASSF

Number of instances	Number of rules	T_{P1}	T_{P2}	T_{P3}
50	40	3171	31	187
	80	3138	32	203
	120	3251	47	312
100	40	3422	47	250
	80	3406	31	296
	120	3508	46	313
150	40	3718	33	328
	80	3906	47	359
	120	3782	46	375

that of rule files. Here, we got the following test data by increasing the size of instances and that of rule files (see Table 6). The unit of those test data is millisecond (ms).

According to Table 6, we draw Figures 7 and 8 to illustrate the tendency of execution time with the changes of instances and rule files. Figure 7 is draw based on the original samples of the consumption time in three phases. Deviations of original samples are calculated and Figure 8 illustrates the tendency of execution time with the increase of ontology instances and rule files.

As shown in Figure 7, we draw some conclusions directly. (1) The time consumption of the Phase 1 (T_{P1}), during which the initialization of the pervasive system is done including the load of the device ontology and the construction of the context database, increases with the growth of instances. (2) The execution time of the Phase 2 (T_{P2}) seems to grow with the increase of rule files. In Phase 2, the process of loading the predefined rule files is finished. The more rules are predefined in the rule files, the more time is consumed. (3) The consumption of the Phase 3 (T_{P3}) is determined by the instances and rule files directly. With the enrichment of the instances and rule files, more time is consumed in the phase 3.

As depicted in Figure 8, the phase 1 consumes lots of time, which seems to be unacceptable for users and services. Fortunately, the phase 1 is executed only once when the system is launched. During this process, the initialization of the system is completed including loading the device ontology, the construction of context base and the maintenance of a run-time model of contexts. After the initialization, the run-time model makes it possible to access some contexts in the memory

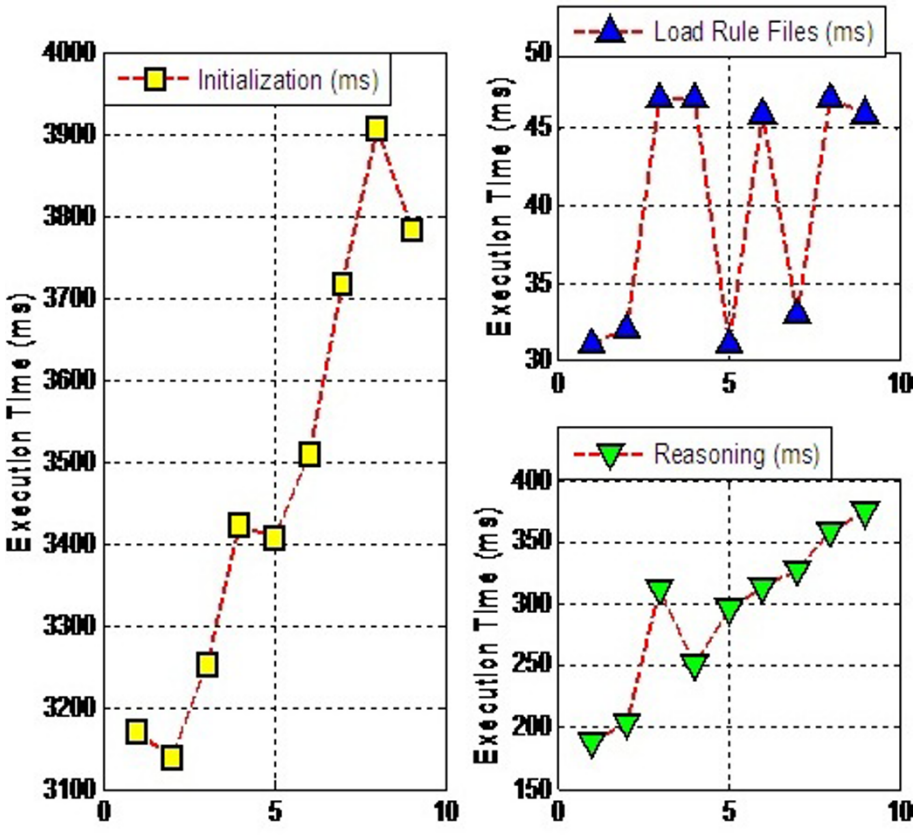

Figure 7 Execution time of the semantic service matching, in which the unit of execution time is millisecond (ms).

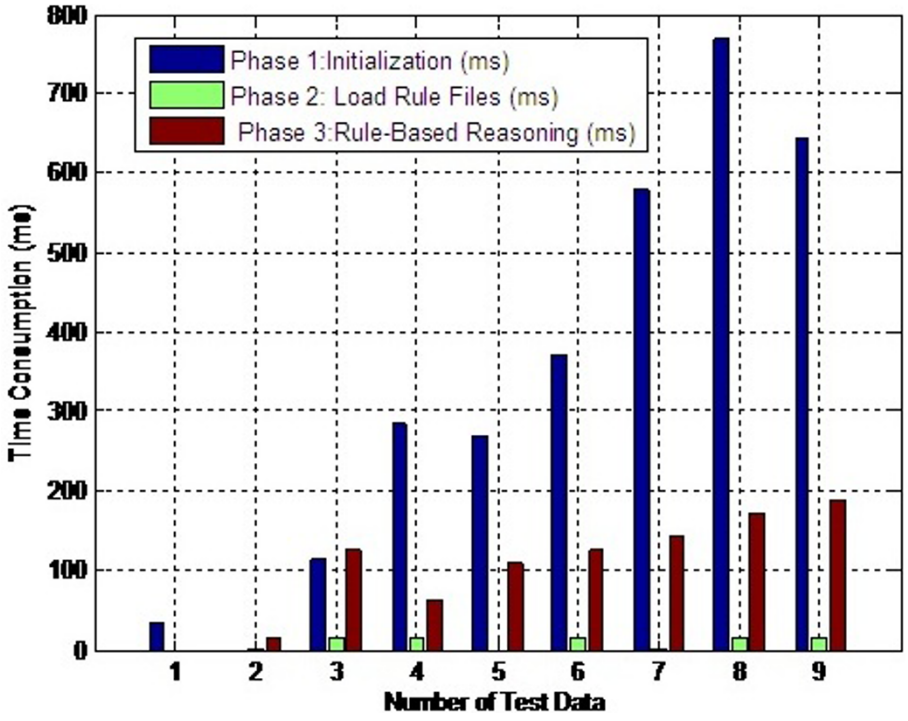

Figure 8 Deviation of the execution time, in which the unit of execution time is millisecond (ms).

directly, which accelerates the process of data processing and facilitates the subsequent phases.

In addition, the test data of Phase 2 seems to be disordered and it is likely to be unreasonable to reach the conclusion 2. However, two factors should be taken into consideration. First, the size of rule files may be a little small, by which it is difficult to discriminate the differences of the Phase 2, as the total time is only tens of milliseconds. On the other hand, the changes of computational load may have a significant influence on the execution time. As the round-robin algorithm is adopted to schedule a process in the operating system, the waiting time contributes to the jetty of the samples. When numerous processes are running in the current situation, the target application cannot occupy the computational resources immediately and has to share resources with others periodically. Thus the round-robin algorithm is the crucial reason for the jitter of the execution time. Nevertheless, the conclusion 2 is reached by comparing the data which was got under the same size of rule files.

Finally, we designed a scenario to demonstrate the feasibility of this system. An old man is watching TV in the living room, and the telephone is ringing. When the telephone state is detected, a task reminding the user to pick up the hook is generated. According to the device located in different rooms, the CASSF will allocate a suitable service to execute the task. If the user is located in the living room, devices in living room are the candidates. Because the TV has higher capability index than other monitors, the prompting information will be presented on TV as shown in Figure 9a. When the elderly is reading newspaper in the study, a medication reminder will be displayed in voice-based presentation as shown in Figure 9b.

7. Conclusion and future study

This article focuses on a CASSF, which aims to overcome challenges caused by the heterogeneity and the mobility of devices in SHs. The framework outperforms traditional service scheduling systems in three aspects: First, it introduces a service discovery mechanism based on the universal device ontology to manage services in heterogeneous network. Second, rule-based reasoning is employed in the process of matching requirements of tasks and capabilities of devices. Third, to make sure the task executed smoothly, the content adaptation takes charge of shielding differences of interfaces and enhancing user satisfaction.

In the current prototype system, we conducted the experiment to execute atomic services. Our future study will focus on complex services, which are decomposed

(a)

(b)

Figure 9 Prototype of the context-aware multimedia service scheduling framework. where **(a)** presents a phone reminder on the TV; **(b)** indicates a medication reminder by the voice-based service on the mobile phone.

into plenty of atomic services, and make sure the atomic services of the complex service are executed in the accurate sequence.

Abbreviations

CASSF: context-aware service scheduling framework; CB: context base; DAM: device allocation manager; DL: description logic; DP: device pool; IATECL: Intelligent Assistant Technology and Elderly Care Lab; NPU: Northwestern Polytechnical University; OSGI: open services gateway initiative; PDA: personal digital assistants; SH: smart home; SS: service scheduler; SWRL: semantic web rule language; TR: task requirement.

Acknowledgements

This study was partially supported by the National Basic Research Program of China (973 Program) (No. 2012CB316400), the National Natural Science Foundation of China (No. 61103063, 60903125), the Program for New Century Excellent Talents in University (No. NCET-09-0079), the Science and Technology Research Plan in Shaanxi Province of China (No. 2011KJXX38), and the Doctorate Foundation of Northwestern Polytechnical University..

Author details

[1]School of Computer Science, Northwestern Polytechnical University, Xi'an, P. R. China [2]Department of Networking and Service,Institut TELECOM SudParis, 9 Rue Charles Fourier 91000 Evry, France

Competing interests

The authors declare that they have no competing interests.

Received: 17 June 2011 Accepted: 27 February 2012
Published: 27 February 2012

References

1. M Chan, D Estève, C Escriba, E Campo, A review of smart homes present state and future challenges. Comput Methods Programs Biomed. **91**(1), 55–81 (2008). doi:10.1016/j.cmpb.2008.02.001
2. Z Yu, Y Nakamura, D Zhang, S Kajita, K Mase, Content provisioning for ubiquitous learning. IEEE Pervasive Comput. **7**(4), 62–70 (2008)
3. The W3C http://www.w3.org/Mobile/CCPP/. Accessed June 12, 2011
4. FIPA Device Ontology Specification, http://www.fipa.org/specs/fipa00091/PC00091A.html. Accessed June 12, 2011
5. B Schilit, N Adams, R Want, Context-aware computing applications, in *Proc of 1st International Workshop on Mobile Computing Systems and Applications (WMCSA'94), Santa Cruz*, pp. 80–85 (December 8-9 1994)
6. R Eisinger, MG Manzato, R Goularte, Devices descriptions for context-based content adaptation, in *Proc of the Third Latin American Web Congress (LA-WEB'05), Buenos Aires, 31*, pp. 121–129 (31 October-2 November 2005)
7. Sun Microsystems. Jini Architectural Overview: Technical White Paper (1999)
8. Microsoft Corporation, Understanding Universal Plug and Play: a white paper (June 2000)
9. E Guttman, Service location protocol. IEEE Internet Comput. **3**(4), 71–80 (1999). doi:10.1109/4236.780963
10. UDDI technical white paper http://www.uddi.org/pubs/. Accessed June 12, 2011
11. Bluetooth SIG, Bluetooth protocol architecture version 1.0 Bluetooth white paper (September 1999)
12. P Bellavista, A Corradi, R Montanari, A Toninelli, Context-aware semantic discovery for next generation mobile systems. IEEE Commun Mag. **44**, 62–71 (2006)
13. T Dong, Q Li, K Zhang, L Cui, An extended matching method for semantic web service in collaboration environment, in *Proc of the 11th international Conference on Computer Supported Cooperative Work in Design (CSCWD'07), Melbourne, Australia*, pp. 508–513 (April 26-28, 2007)
14. E Sirin, B Parsia, J Hendlers, Filtering and selecting semantic web services with interactive composition techniques. IEEE Intell Syst. **19**(4), 42–49 (2004). doi:10.1109/MIS.2004.27
15. J Lindenberg, W Pasman, K Kranenborg, J Stegeman, MA Neerincx, Improving service matching and selection in ubiquitous computing environments: a user study. Person Ubiquitous Comput. **11**(1), 59–68 (2007)
16. SS Yau, J Liu, Functionality-based service matchmaking for service-oriented architecture, in *Proc of 8th International Symposium on Autonomous Decentralized Systems (ISADS), Sedona, Arizona, USA*, pp. 147–154 (2007)
17. O Choi, S Han, Ubiquitous computing services discovery and execution using a novel intelligent web services algorithm. Sensors. **7**(7), 1287–1305 (2007). doi:10.3390/s7071287
18. A Fernandez, A Polleres, S Ossowski, Towards fine-grained service matchmaking by using concept similarity, in *Proc of the SMR2 2007 Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2 2007), Busan, Korea*, pp. 31–45 (November 2007)
19. Q Lv, J Zhou, Q Cao, Service matching mechanisms in pervasive computing environments, in *Proc of International Workshop on Intelligent Systems and Application, Wuhan, China*, pp. 1–4 (May 23-24, 2009)
20. I Horrocks, PF Patel-Schneider, H Boley, S Tabet, B Grosof, M Dean, Swrl: a semancit web rule language combing owl and ruleml. W3C Member submission, Technical Report (2004)
21. F Badder, D Calvanese, D McGuinness, D Nardi, P Patel-Schneider, in *The Description Logic Handbook: Theory, Implementation, and Applications*, ed. by Badder F (Cambridge University Press, New York, 2003)
22. A Bandara, T Payne, D Roure, G Clemo, An ontological framework for semantic description of devices, in *Proc of the 3rd International Semantic Web Conference (ISWC'04), Hiroshima, Japan*, pp. 2–3 (November 2004)
23. M Paolucci, T Kawamura, T Payne, K Sycara, Semantic matching of web service capabilities, in *Proc of the International Semantic Web Conference (ISWC'02), Sardinia, Italy*, pp. 333–347 (June 2002)
24. Z Yu, X Zhou, D Zhang, CY Chin, X Wang, J Men, Supporting context-aware media recommendations for smart phones. IEEE Pervasive Comput. **5**(3), 68–75 (2006). doi:10.1109/MPRV.2006.61
25. OSGi Alliance, OSGi service platform core specification release 4 http://www.osgi.org. Accessed June 12, 2011
26. The webpage introducing Protégé http://protege.stanford.edu. Accessed June 12, 2011
27. HP Labs, http://jena.sourceforge.net/. Accessed June 12, 2011
28. W Xu, Y Xin, G Lu, Q Chen, A new service description, matching and selection mechanism for pervasive computing, in *Proc of the 6th International Conference on Fuzzy System and Knowledge Discovery (FSKD'09), Tianjin*, pp. 509–514 (2009)
29. SE Czerwinski, BY Zhao, TD Hodes, AD Joseph, RH Katz, An architecture for a secure service discovery service, in *Proc of the Fifth Annual International Conference on Mobile Computing and Networks (MobiCom'99), Seattle*, pp. 24–35 (August 15-20, 1999)
30. C Campo, M Muñoz, JC Perea, A Marín, C García-Rubio, PDP and GSDL: a new service discovery middleware to support spontaneous interactions in pervasive systems, in *Proc of IEEE Middleware Support for Pervasive Computing (PerWare 2005) at the 3rd IEEE Conference on Pervasive Computing (PerCom 2005), Kauai Hawai*, pp. 178–182 (March 18-12, 2005)
31. S Helal, N Desai, V Verma, C Lee, Konark, SF Midkiff, Service description for pervasive service discovery, in *Proc of the 25th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'05), Columbus, USA*, pp. 273–279 (2005)
34. R Hermann, D Husemann, M Moser, M Nidd, C Rohner, A Schade, DEAPspace–transient ad hoc networking of pervasive devices. Comput Netw. **35**(4), 411–428 (2001). doi:10.1016/S1389-1286(00)00184-5
35. D Chakraborty, A Joshi, T Finin, Y Yesha, GSD: a novel group-based service discovery protocol for MANETS, in *Proc of the 4th IEEE Conference on Mobile and Wireless Communications Networks (MWCN 2002), Stockholm, Sweden*, pp. 140–144 (September 2002)
36. S Maffioletti, S Kouadri Mostefaoui, B Hirsbrunner, Automatic resource and service management for ubiquitous computing environments, in *Proc of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW'04), Orlando, Florida, USA*, pp. 219–226 (March 2004)
37. H Chen, A Joshi, T Finin, Dynamic service discovery for mobile computing: intelligent agents meet Jini in the Aether. Baltzer Sci J Cluster Comput. **4**(4), 343–354 (2001)
38. K Arabshian, H Schulzrinne, GloServ: global service discovery architecture, in *Proc of first Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04), Boston*, pp. 319–325 (August 2004)

doi:10.1186/1687-1499-2012-67
Cite this article as: Liang *et al.*: **A context-aware multimedia service scheduling framework in smart homes.** *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:67.