Context-Aware Middleware for Pervasive Elderly Homecare

Hung Keng Pung, Tao Gu, Wenwei Xue, Paulito P. Palmes, Jian Zhu, Wen Long Ng, Chee Weng Tang, Nguyen Hoang Chung

***Abstract*—The growing aging population faces a number of challenges, including rising medical cost, inadequate number of medical doctors and healthcare professionals, as well as higher incidence of misdiagnosis. There is an increasing demand for a better healthcare support for the elderly and one promising solution is the development of a context-aware middleware infrastructure for pervasive health/wellness-care. This allows the accurate and timely delivery of health/medical information among the patients, doctors and healthcare workers through a widespread deployment of wireless sensor networks and mobile devices. In this paper, we present our design and implementation of such a Context-Aware Middleware for Pervasive Homecare (CAMPH). The middleware offers several key-enabling system services that consist of P2P-based context query processing, context reasoning for activity recognition and context-aware service management. It can be used to support the development and deployment of various homecare services for the elderly such as patient monitoring, location-based emergency response, anomalous daily activity detection, pervasive access to medical data and social networking. We have developed a prototype of the middleware and demonstrated the concept of providing a continuing-care to an elderly with the collaborative interactions spanning multiple physical spaces: person, home, office and clinic. The results of the prototype show that our middleware approach achieves good efficiency of context query processing and good accuracy of activity recognition.**

***Index Terms*—Context-awareness, pervasive homecare, middleware, query processing, activity recognition, peer-to-peer, service-oriented architecture.**

I. Introduction

WITH the increasing number of the elderly, more people will be suffering from age-related ailments such as memory loss, cardiovascular diseases, respiratory problems, renal disorders, and body coordination problems among others. While there is no substitute for hospitals in treating terminally ill patients, those elderly who are still healthy require help to lead an independent lifestyle. Traditional elderly-care usually involves complex interactions among the caregivers, nurses, physicians, clinics, family members, pharmacists and others in delivering proper healthcare services. These services are usually done on an ad-hoc basis which is likely to contribute to increasing incidence of medical errors or misdiagnosis and the decreasing effectiveness of preventive care. A comprehensive review of the issues and the state of technology in aging services is presented in the report of Alwan and Nobel [1].

The pervasive care[1] of elderly would benefit most from pervasive homecare[2], which is the integration of wireless sensor networks with ubiquitous computing devices in the creation of smart environment for providing unobtrusive monitoring, prompting or reminding of desirable activities/actions, and correcting or assisting the elderly in their daily activities at home and elsewhere.

Pervasive homecare may greatly reduce the risks of traditional elderly-care, lowering the anxiety of caregivers[3], especially the immediate informal ones, due to the better connectivity and networking among the stakeholders [1] as well as reducing the cost and inconvenience of regular visits to clinics/hospitals via remote diagnosis. These merits have motivated us to research into a context-aware middleware for pervasive homecare of the '*young old*' population who can still perform activities for daily living (ADLs).

There are many technical challenges to overcome before pervasive homecare can become a reality. First, most existing solutions focus on developing individual techniques that lead to segmented solutions and poor interoperability [1], [2]. A full-fledged pervasive homecare system requires an appropriate infrastructure that integrates all enabling technologies for information acquisition, management, and processing. Second, context data [3] is the key in such systems which enable pervasive care applications to behave and adapt intelligently. Interpreting and managing such data in an open infrastructure is essential for providing an efficient linkage model among the elderly, their family members, caregivers and professional care workers. Last, a pervasive healthcare system should have a wider usability. It must provide intuitive user interfaces for elderly patients and caregivers, as well as simple programming models for the efficient processing of declarative queries and the convenient orchestration of services from IT-care-service providers.

Aiming to develop enabling key technologies in pervasive homecare and to provide necessary facilities to glue them into

Manuscript received 30 July 2008; revised 9 December 2008.

H.K. Pung, W. Xue, P.P. Palmes, J. Zhu, W.L. Ng, C.W. Tang, and N.H. Chung are with the Department of Computer Science, School of Computing, National University of Singapore, 3 Science Drive 2, Singapore 117543 (e-mail: {dcsphk, dcsxw, dcsppp, g0500652, dcsnwl, dcstcw, chung.nguyen}@nus.edu.dg).

T. Gu is with the Institute for Infocomm Research, 1 Fusionpolis Way, #21-01 Connexis, South Tower, Singapore 138632 (e-mail: tgu@i2r.a-star.edu.sg).

Digital Object Identifier 10.1109/JSAC.2009.090513.

[1]Pervasive care is defined here as IT-assisted continuing care that is deliverable at anytime and anywhere.

[2]Pervasive homecare is defined here as IT-assisted continuing care being delivered to informal places such as homes, informal care centers and the other suitable spaces, excluding formal healthcare institutions.

[3]Caregivers are from hereon referred to as informal caregivers, excluding professional medical and healthcare workers, such as doctors and nurses.

an open, programmable and flexible infrastructure, this paper presents our design and implementation of a **C**ontext-**A**ware **M**iddleware for **P**ervasive **H**omecare (**CAMPH**). CAMPH provides a number of system-level services, such as context data acquisition, context storage, context reasoning, service organization and discovery, to facilitate the development and deployment of various pervasive homecare applications. The middleware infrastructure separates context data and context-aware services in different layers that are accessible by the applications. This scheme, not only decouples the dependency of techniques used in individual layers, but also provides a greater flexibility for the selection and deployment of appropriate techniques in each layer by the system administrator. Similarly, the programming task of context-aware applications would be simplified through the service layer and its programming interfaces.

CAMPH provides a SQL-based query service for context data acquisitions. Unlike existing context-aware systems, our infrastructure assumes autonomous sources of context data. It categorizes and organizes these sources into different context domains in a semi-supervised manner. Hence a unique feature of our system is its ability to support the interactions of services with context data sources in multiple physical spaces. Services are organized and discovered in CAMPH based on a dual-view management approach: local service management and global service management. Also, a hierarchical scheme of context reasoning is carried out in the infrastructure in which low-level domain-specific reasoners can provide input to high-level cross-domain reasoners.

The rest of the paper is organized as follows. Section II gives an overview of our CAMPH middleware, followed by the details of physical space gateways in Section III, context data management in Section IV, and context reasoning in Section V. The service management in CAMPH is briefly described in Section VI, the system prototype of CAMPH and the initial evaluation results of the prototype are presented in Section VII, and related work are discussed in Section VIII. We conclude the paper in Section IX.

II. Overview of Middleware Infrastructure

The overall architecture of CAMPH is illustrated in Fig. 1. It consists of the following four logical layers:

- *Physical Space Layer*. Sources of context data in CAMPH are encapsulated as individual physical spaces. Each space consists of entities such as sensors, actuators and computing devices. A *physical space* is an operating environment that provides context data and mandates all interactions of its entities with the "outside world" through a designated gateway known as *Physical Space Gateway (PSG)*. Moreover, a PSG can be static (e.g. at home or in a clinic) or mobile (e.g. worn by a person or robot). Different kinds of context data can be available from a single physical space, such as the location of a device, the occurrence of an event, the room temperature, and the number of people in a room. We use a simple and pragmatic attribute-value approach to context modeling at the physical spaces and define a *context attribute* as a specific kind of context data. We further define a

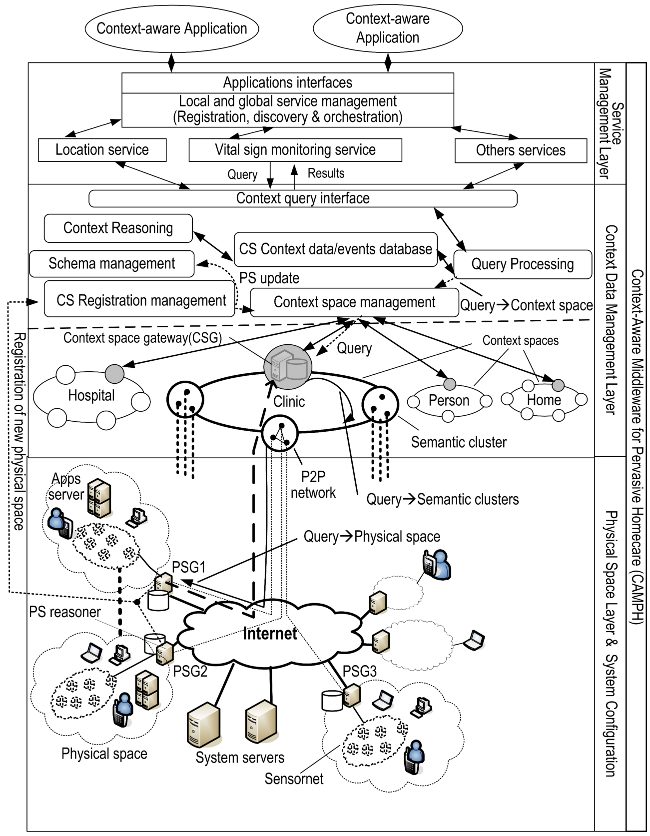


Fig. 1. Architecture and implementation model of CAMPH.

context domain, or *context space*, as a class of physical spaces having similar attributes (Context domain and context space are used interchangeably from hereon.) Examples of context domains in pervasive homecare are HOSPITAL, CLINIC, PERSON and HOME, as depicted in Fig. 1.
- *Context Data Management Layer*. The main objective of this layer is to enable the effective and efficient context data organization, lookup and storage. Its functions are distributed to the system servers and in some cases to the PSGs (see Fig. 1). Briefly, the layer implements a declarative, SQL-based context query interface for services or applications to acquire context data or subscribe to event notifications from multiple physical spaces. It automatically categorizes physical spaces in different context domains by integrating their local context schemas into a global set of domain schemas as part of the system core services at the system servers. The physical spaces in a domain are organized as peers in several clusters of a semantic overlay network [4], over which the context queries for data acquisition are processed. A hierarchical context reasoning scheme is employed in the physical space as well as the context data management layer to deduce the high-level context events from low-level results of activity recognition. Historical context data can also be stored at the system servers or individual PSG.
- *Service Management Layer*. This layer of CAMPH manages context-aware services to enable the orchestration of homecare applications. It implements a dual-view scheme

for local and global service management. The local view of the scheme enables the proximity access of collocated services while the global view allows for a wider access of cross-space services. The scheme queries context data from the data management layer and utilizes the data to enable context-aware service organization and discovery.

- *Applications Layer*. The pervasive context-aware home-care applications running on top of CAMPH can invoke services in the service management layer as well as acquire data directly from the context data management layer or physical spaces.

III. Physical Space Gateway

CAMPH consists of the system servers and a number of physical space gateways. The core functional components and system services in the top two layers of the middleware are located at the system servers (Fig. 1). In our current prototype of CAMPH, the functionalities of the system servers are decentralized in several desktop PCs.

Each physical space is associated with a physical space gateway. The gateway's function is a logical module that can be deployed at any computer of choice in the physical space. For example, the gateway of a PERSON space can be the person's PDA or laptop, and a HOME space gateway can be a PC at the person's home. The gateway provides a single and uniform interface for accessing the notification of events and acquiring context data from local sensors or other computing devices in the physical space. It contains a set of components that manages and manipulates context data in the space. The components of a PSG are shown in Fig. 2. These components can be divided into three categories:

1) *Service management*. A PSG maintains different kinds of services using its service manager. The core of the services is a set of context data services that encapsulates various operations of context data acquisition in the physical space, similar to the ODBC and JDBC APIs [5]. In addition, a PSG can have system services that monitor and control sensor networks/computing devices, as well as application-oriented services that perform common local processing tasks for the applications. Each of these services can be registered to an HTTP server at the PSG and be exposed as web services for remote invocation via SOAP [6] and WSDL [7]. The plug-in manager allows applications to deploy and share libraries of reusable plug-in modules. A plug-in is designed to be able to work with the web services to alleviate the processing burden of applications.
2) *Query processing*. Applications can issue declarative, SQL-based queries to a PSG. The syntax of the queries is based on a context schema kept by the schema manager which specifies all context attributes that the physical space currently provides. The queries are executed by a query processing engine that the PSG is equipped with. The engine employs a context reasoner to deduce activities in the space. A local context database is used by the engine to store historical context data as well as by the reasoner to store the training datasets and reasoning results.
3) *Data communication*. The connection manager of a PSG has two sub-components: the P2P manager and the mobile neighbor locator (not shown in Fig. 2). The P2P manager links the PSG to a number of other PSGs to form a global Peer-to-Peer (P2P) network, which enables the inter-gateway communication between multiple physical spaces and is managed by the system servers. Furthermore, a PSG can use its mobile neighbor locator to dynamically detect mobile PSGs in the same proximity and establishes connections with them for mutual data exchange. A PSG uses its registration manager to register to the most appropriate context domain at the system servers and then join the corresponding P2P networks.

The technical details of individual components of a PSG can be found in [8].

We have developed a PSG toolkit in our CAMPH prototype that encapsulates the implementation of most components in Fig. 2, as well as the guidelines on how to set up the other components that must be externally implemented and linked. After downloading and installing the toolkit, any networked computer can configure itself as a PSG using the toolkit GUI and register as a member in our infrastructure.

IV. Context Data Management

The role of the context data management layer is to provide the effective organization of physical spaces and the efficient processing of context queries for data acquisition over these spaces.

A. Context Query Interface

CAMPH adopts a declarative, SQL-based query language as the interface for applications to acquire context data from physical spaces. Two classes of context queries are supported in our query language: *data collection queries* and *event subscription queries*. Queries 1 and 2 below exemplify the two query classes. A data collection query returns the values of a list of context attributes from physical spaces that satisfy the query condition. An event subscription query subscribes the issuer of the query to events defined in the physical spaces. The WHERE clause of a query specifies from which physical spaces the data should be acquired, while the FROM clause specifies the context domains these physical spaces belong to.

```
Query 1:
   SELECT   temperature, heartbeat
   FROM     PERSON
   WHERE    name = "Keith"

Query 2:
   SUBSCRIBE   isVacant
   FROM        HOME
   WHERE       location = "10 Steven Road"
```

A CONT keyword can be used to specify a data acquisition query is continuous and push-based, as shown in Query 3. The sampling interval and duration of the continuous data acquisition are specified using the SAMPLE INTERVAL and

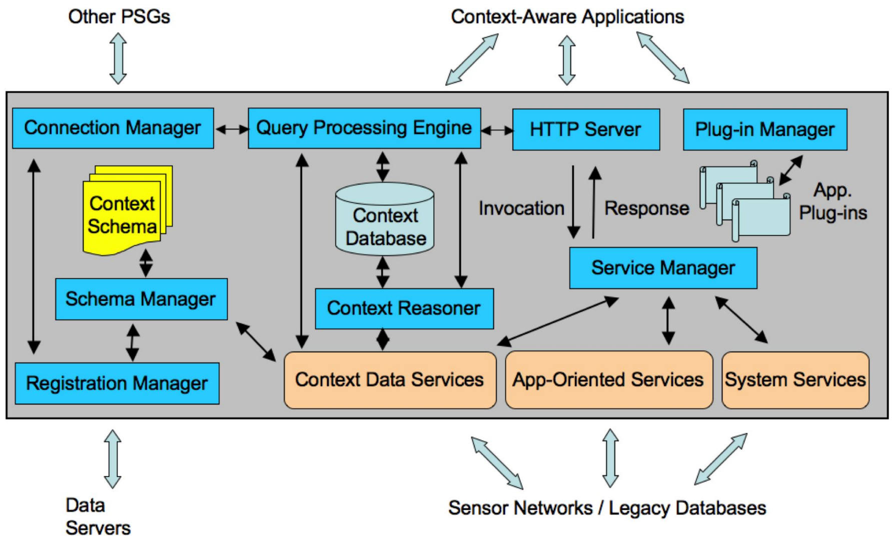


Fig. 2. Physical space gateway's components.

LIFETIME clauses. Without this keyword, a query specifies one-time and pull-based acquisition as in Query 1.

```
Query 3:
   SELECT    CONT   location
   FROM             PERSON
   WHERE            name = "Keith"
   SAMPLE INTERVAL  10 mins
   LIFETIME         2 hours
```

B. Dynamic Schema Matching

Applications issue context queries to the declarative interface based on a global set of context schemas at the system servers. These schemas describe the characteristics of various context domains categorized in the context data management layer of CAMPH, such as HOSPITAL, CLINIC, PERSON and HOME. We call any context schema in this global set a *domain schema*. As an example, "PERSON" in Query 1 is the name of the domain schema that embodies the properties of all physical spaces representing "persons" in the real world.

As depicted in Fig. 3, the global set of domain schemas are dynamically and continuously updated by the schema manager at the system servers. The manager processes the local schemas that are submitted from individual physical spaces upon their registrations and integrate them into different domain schemas. Given the global set of domain schemas, context-aware applications are able to see a unified and abstract view of the underlying heterogeneous physical spaces and access context data from the spaces in a consistent manner. The set of domain schemas evolves when multiple context attributes provided by numerous physical spaces are added and clustered into the corresponding schemas incrementally. Applications can clearly view these correlated attributes and quickly access new attributes when they become available.

CAMPH is aimed to support autonomous physical spaces owned by different organizations. We feel that it is impractical to impose a common name space of context attributes/domains among all spaces for context schema composition. Instead, we

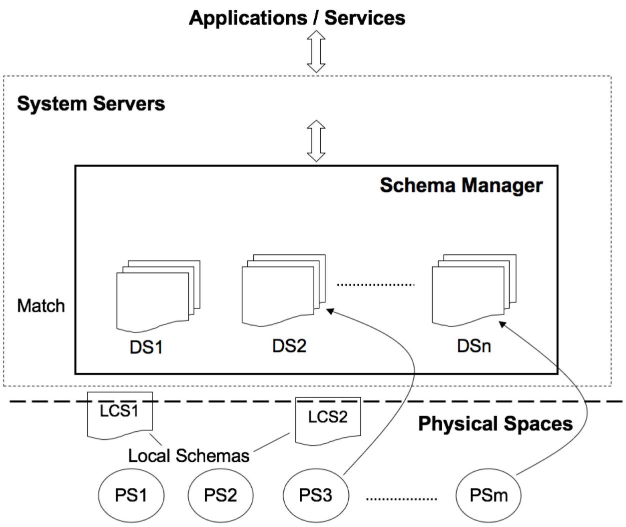


Fig. 3. Matching local context schemas into domain schemas.

allow each space to compose its local schema independently, while we publish the global set of domain schemas at the system servers via a public web interface for reference only.

Two physical spaces can specify different names, e.g. *name* and *personName*, for an attribute with the same semantics. They can also specify different names, e.g. HOME and HOUSE, for the same context domain. To solve this potential problem of context schema mismatching, CAMPH implements a name-based schema matcher [9] at the system servers as a core subcomponent of the schema manager. The matcher has a special lightweight design to ensure the timely matching of continuously arriving local schemas from the physical spaces. It is able to adaptively adjust its performance based on the changing patterns of input schemas in various domains.

The matcher uses a *Context Attribute Matching* algorithm to match attributes in a local schema with those in the current global set of domain schemas. The algorithm applies multiple linguistic matching criteria that are sorted in a decreasing order of their priorities, such as stemming, substring and

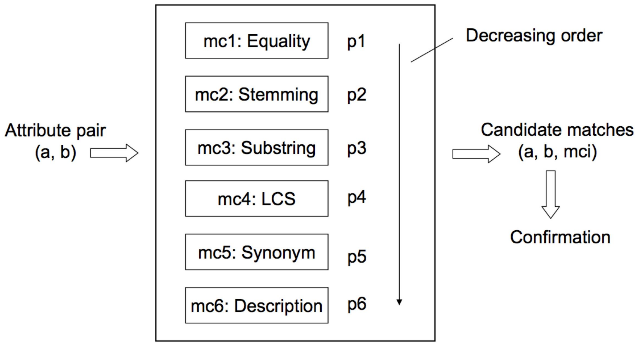


Fig. 4. Matching a pair of context attributes using multiple criteria.

synonym detection. The priorities of the criteria are adjusted dynamically over time based on the recent matching accuracy they achieve to make the algorithm adaptive to the current patterns of schema inputs. In Fig. 4, given a pair of attributes, the criteria in the list are invoked one by one to look for a candidate match between the pair. If a criterion reports a match, its succeeding criteria in the list will not be invoked for the pair any more. In this way, the algorithm always tries to use a single criterion with the highest recent accuracy to offer a candidate match for an attribute pair.

Based on the output of attribute matching, the matcher uses a *Context Schema Matching* algorithm to try to merge a local schema to one of the current domain schemas based on the largest common subset of matched attributes between the two schemas. In order to improve accuracy, the matcher reports all candidate matches of attributes/schemas that it has found to the middleware administrator for human confirmation. Local attributes/schemas that cannot be matched with any existing domain schemas are added as new elements into the global set.

The mappings between a local schema and the set of domain schemas are sent back to and stored at the corresponding PSG. When a context query specified to the declarative interface based on the domain schemas is routed to a PSG, the query syntax is first converted to the local schema using the mapping before the query is evaluated.

In summary, the schema matcher integrates various local schemas from physical spaces into a global set of domain schemas at the system servers, based on which the context queries are syntactically composed and issued to CAMPH. The initial set of the domain schemas can be either pre-defined or empty. Interested readers are invited to read our previous paper [9] for further details.

C. Semantic P2P Overlays

When a domain schema is first generated, the corresponding context domain is mapped to a logical, one-dimensional ring network of semantic clusters. This approach leads to multiple semantic rings in the middleware, in contrast to our previous work [4] that has a single semantic ring containing all physical spaces with no concept of PSG. Our current approach in CAMPH is expected to be more scalable in performance and practical in reasoning (see Section V) due to the more specific scope of context domains.

Each semantic cluster in a ring overlay corresponds to a context attribute in the domain and is implemented as a P2P network in which each peer is a PSG. As a physical space may not provide all current attributes in the corresponding domain schema, its PSG will only join those clusters for the attributes it has. As an initial solution, CAMPH uses Gnutella version 0.4 [10] as the P2P protocol for the semantic clusters due to its wide applicability to both static (e.g. *name* and *gender* for PERSON) and dynamic (e.g. *location* and *temperature* for PERSON) context data.

The rings for all context domains are created and maintained by the context domain (space) manager at the system servers. A *Context Space Gateway (CSG)* is created as a special cluster in each ring to serve as the entrance for routing of context queries (see Fig. 1). The CSG connects to the ring as any other cluster while it is actually a subcomponent of the context space manager rather than a P2P network. It maintains the ring topology and creates new semantic clusters when the context domain evolves.

A PSG can leave the semantic clusters it joins at any time. The leaving is automatically detected by the P2P protocol at the neighboring peers. The detailed operations are similar to the joining/leaving of context sources in our previous work [4]. The PSG must send a new registration request in order to rejoin the infrastructure.

D. Query Evaluation Plans

When the declarative application interface receives a context query, the query string is forwarded to a query processor at the system servers. The processor parses the query and examines its syntax based on the global set of domain schemas. A query with a wrong syntax will not be further processed and an error message is returned to the application.

The query processor generates an evaluation plan [5] for each context query that is successfully parsed. The plan is a tree of operators in which data is streamed from bottom up. A query plan in CAMPH has two main differences when it is compared with a traditional plan in a relational DBMS. First, a scan operator is created for each context domain specified in the FROM clause of the query. The operator encapsulates the lookup process of context data in the domain rather than that from a disk table. Second, the query condition is evaluated at the individual PSGs. This means that the selection operators are lower than the scan operators in the query plan.

The process of context data lookup in a scan operator includes three separate steps of mapping a query to different data structures in the context data management layer. First, the scan operator searches an index of context domains in the context space manager to find the CSG of the domain. Second, the operator establishes connection and sends a lookup request to the CSG. The request is a parsed version of query string that contains the query condition and the attribute list to acquire. It is injected into the ring by the CSG and is processed by the first semantic cluster found that corresponds to one attribute in the projection list. Last, the request is disseminated in the P2P network. Each PSG that satisfies the query condition reports data directly to the server-side scan operator. To cope with the Gnutella flooding protocol, a *Time-To-Live (TTL)* value is set to each query. The value indicates the number of hops the lookup request of a query can reach in the P2P network.

As an example, Query 1 in Subsection A has a simple query plan consisting of two operators: a scan operator in the bottom connects to a merge operator on top. The former performs the aforementioned process of context data lookup while the latter packages all data from the replied PSGs into a specific format and returns the data to the application issuing the query.

V. Context Reasoning

In the design of our context reasoning scheme, a single and comprehensive context reasoner at the server that performs all reasoning tasks centrally is less likely to be a practical solution due to the diversified context domain semantics of physical spaces. Learning or rule-based reasoning techniques require training context datasets or expert-provided rules, and these datasets or rules are typically specific to each physical space and even individual context domain. To provide the reasoning service for a wide-range of context-aware applications, we have to model and infer the context data and application semantics involved in all domains. This leads to an increasing complexity of the centralized reasoner and make it infeasible in a real deployment.

A. Hierarchical Reasoning Scheme

We have designed a hierarchical scheme to support flexible and extendable context reasoning in CAMPH. At the lower layer – the individual PSGs, a single-space context reasoner is employed to deduce the events of interest occurring in the corresponding physical space. At the higher layer – the context data management layer, a high-level context reasoner deduces events related to multiple physical spaces in a single context domain or in different domains. The two-level hierarchy will distribute the context reasoning functionality throughout the whole CAMPH infrastructure and improve the scalability of the system. It will also avoid a single point of failure or overload since the reasoning workload is shared by multiple reasoners within different system components based on their particular domain semantics.

In the hierarchy, the input of a high-level reasoner can be simple context attributes or deduced events from any physical space. The output of a high-level reasoner, if necessary, can be also provided as a kind of feedback input to the low-level reasoners to improve their reasoning quality. We define an *event* in CAMPH as a kind of deduced context attribute output from a reasoner at any of the two levels in our reasoning scheme. For instance, a gas leakage in a house and a heart attack of a person are single-physical-space events. Examples of single-domain events include a conversation between two persons (of PERSON context domain) and Christmas sales of shops (of SHOP domain). An example of cross-domain events can be a shopping trip of an elderly lady (PERSON domain) to a shop (SHOP domain) that involves checking the stock level of her fridge at home (HOME domain).

Figure 5 shows the functional diagram of our reasoning scheme. Each reasoner can model and implement a set of machine learners, probabilistic models and rule-based engines. An event can be tracked and predicted by one of the reasoning

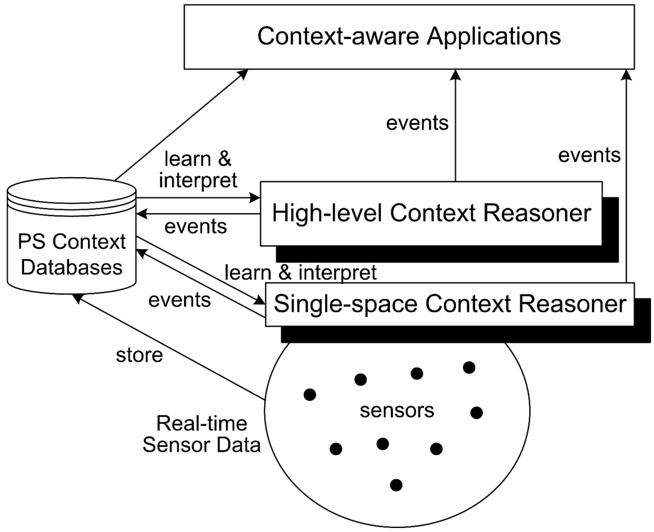


Fig. 5. Hierarchical reasoning scheme in CAMPH.

models or their combination given the application-provided event specification. A new reasoning model can be added into a reasoner any time flexibly without affecting the functionality of existing models. An existing model can also be removed from the reasoner if the model is not used for deducing any application-specified events at the moment.

Real-time context data is collected from each physical space via the context data services at the PSG. A reasoning model of a reasoner may require certain amount of historical data to be stored to the context database for future offline or batch processing. If a training dataset is required for a particular model, the dataset is first sampled and stored in the context database, and then used to initially construct and incrementally update model parameters. The output events of the reasoning models are either directly reported to applications whenever they are detected, or stored in special event result tables in the context database for future probing via queries. Note that the context databases shown in Fig. 5 are usually deployed in a decentralized manner, one for each physical space.

In our current system prototype, the high-level reasoner is implemented based on a JENA rule-based reasoning engine [11]. This level of reasoning is realized as decision rules that are part of the application logic. As for the single-space reasoners, we currently focus on a specific reasoning task, *activity recognition*, and propose our recognition algorithms, which are described in detail in the next subsection.

B. Activity Recognition and Segmentation

Our proposed activity recognition algorithms leverage on an unsupervised approach, in which we build our activity models by extracting the objects involved in each activity from the web and assigning the corresponding normalized weights to these objects. We obtain the fingerprint of each activity and propose an algorithm to detect activities in the real-time traces collected from RFID tags. We further propose two algorithms, *MaxGap* and *Bigram*, to segment the trace and detect the boundary of any two adjacent activities in the trace.

To recognize an activity being performed by a subject, we first need to obtain the activity model. Given a set of activities

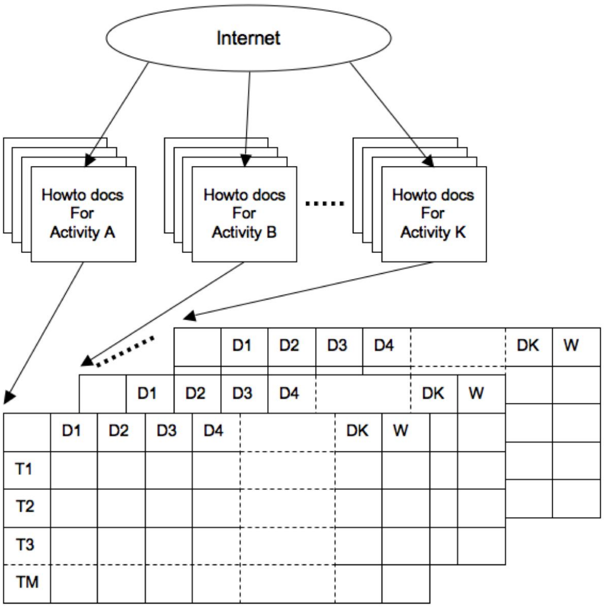


Fig. 6. Term extraction and weighting.

$A = \{a_1, a_2, \ldots, a_p\}$, we build our activity models consisting of a set of terms $T = \{t_1, t_2, \ldots, t_m\}$ used for each activity $a \in A$, together with their associated usage weights W_i.

To automatically build activity models, we mine the *howto* websites, e.g. `www.ehow.com` and `www.wikihow.com`, for web document (DN) containing instructions of activities and extract relevant objects based on their normalized usage frequency. Figure 6 outlines the term extraction and weighting we employ for each activity. We also utilize the stopword removal and object filtering to minimize noise.

The weights are utilized to recognize and segment a sequence of objects used in a number of non-overlapping activities. The discrimination process works under the following principles: (i) each object has relatively varying degrees of relevance in different activities, (ii) certain objects are highly discriminatory and their presence can be used to recognize activities, and (iii) comparing relative weights of nearby objects in two adjacent activities can be used to detect their boundary. This approach is different from most existing activity modeling techniques, such as Hidden Markov Models (HMM) and Dynamic Bayesian Networks (DBN), which rely on *object order* that introduces hard combinatorial and computational challenges.

We determine the weight, W_i, of each term $t \in a \in A$ by computing its **tf-idf** value (term frequency inverse document frequency) using Equation (1).

$$\texttt{tf-idf}_i^a = W_i = |t_i^a| \times \log \frac{|D^a|}{|d^a : t_i^a \in d^a| + 1} \quad (1)$$

where:

$$\begin{aligned} \texttt{tf-idf}_i^a &\rightarrow \texttt{tfidf} \text{ of } i\text{th term in activity } a \in A; \\ |t_i^a| &\rightarrow \text{tf of the } i\text{th term in activity } a \in A; \\ |D^a| &\rightarrow \text{total no. of docs in activity } a \in A; \\ |d^a : t_i^a \in d^a| &\rightarrow \text{no. of docs where the } i\text{th term appears.} \end{aligned}$$

TABLE I
A PARTIAL VIEW OF ADL MODELS MINED

Make Tea		Make Coffee		Make Pasta		Fry Egg	
Object	Weight	Object	Weight	Object	Weight	Object	Weight
tea	1.00	coffee	1.00	pasta	1.00	egg	1.00
water	0.85	water	0.86	flour	0.88	pan	0.99
cup	0.83	cup	0.85	pepper	0.85	oil	0.78
sugar	0.75	pot	0.82	water	0.84	burner	0.76
teapot	0.75	grinder	0.80	sauce	0.83	spatula	0.69
pot	0.74	filter	0.79	tomato	0.81	lid	0.66
bowl	0.72	sugar	0.76	cheese	0.80	water	0.65
lemon	0.70	coffeemaker	0.73	garlic	0.80	bowl	0.60
kettle	0.70	creamer	0.72	oil	0.80	butter	0.60
microwave	0.67	tablespoon	0.72	pot	0.79	dish	0.60

Note: All weights are normalized using the log smoothing function.

Table I lists a partial set of activity models mined from the ehow and wikihow websites with their corresponding top 10 terms ordered by their normalized weights.

Based on the activity models we mined, we observe that the higher the normalized weight is, the more important the corresponding term will be. This suggests that we can make use of a subset of terms, e.g., the top n terms that have higher weights, as *fingerprint*, and then map these fingerprints to the trace for activity recognition. From all the ADL models we mine, we observe that the topmost terms are unique among all the activity models. Hence, in this paper, we use the topmost term (i.e., the *main object*) as the fingerprint to discriminate various activities. The occurrence of one of the main objects in an object sequence signifies the presence of an activity in the neighborhood of these objects.

How to segment the trace and detect the boundary of two adjacent activities is critical since an error in a segmentation process affects the subsequent ones and results in inferior recognition rate. To address the trace segmentation problem, we propose two algorithms – *MaxGap* and *Bigram*.

To detect the boundary between two adjacent activities, A and B, the *MaxGap* algorithm computes the difference between the weight of each object in activity A and its weight in activity B. This difference is defined as the *relative weight* (RW). If the object is more relevant to A than B, RW will be positive whereas the reverse, will be negative. The algorithm computes the difference of each consecutive RW pairs (gap), and the *maximum gap* is the boundary for these two activities.

Algorithm 1 outlines the *MaxGap* algorithm. The input of the algorithm is a sequence of objects, among which are two different main objects (see Fig. 7). The objective is to find the boundary where the object use of one activity ends (i.e. object a) and that of the other activity begins (i.e. object b). The output of the algorithm is the location of an object where we should separate the two activities. The complexity of *MaxGap* is linear with respect to the number of objects in an activity trace.

Another research direction we are investigating to improve the accuracy of recognition is to combine weight information with the pairwise sequence information of objects. We refer to this approach as the *Bigram* algorithm. In this approach, we treat the problem of recognition as a clustering problem in which each cluster is represented by the topmost term (*key*

Algorithm 1: MaxGap Algorithm

Input: Objects: $O = \{o_1, o_2, o_3, \ldots, o_n\}$;
Terms: $T = \{t_1, t_2, t_3, \ldots, t_m\}$;
Activities: $A = \{a_1, a_2, a_3, \ldots, a_p\}$;
where: $a_i = \{t \mid t \in Terms\}, \ i = 1, \ldots, p$;
DetectedActivities $= \{i \mid W_i^a(o_i) = 1\}$ for certain activity a
Result: Boundaries
begin
  foreach $(x, y) \in$ DetectedActivities | o_x, o_y *are adjacent* **do**
    for $ctr = x$ ***to*** y **do**
      $RW_{ctr} = W^x(o_{ctr}) - W^y(o_{ctr})$;
    for $ctr = x$ ***to*** $y - 1$ **do**
      $GAP_{ctr} = RW_{ctr} - RW_{ctr+1}$
    Boundaries.**push**(ctr) such that GAP_{ctr} is maximum ;
  return Boundaries;
end

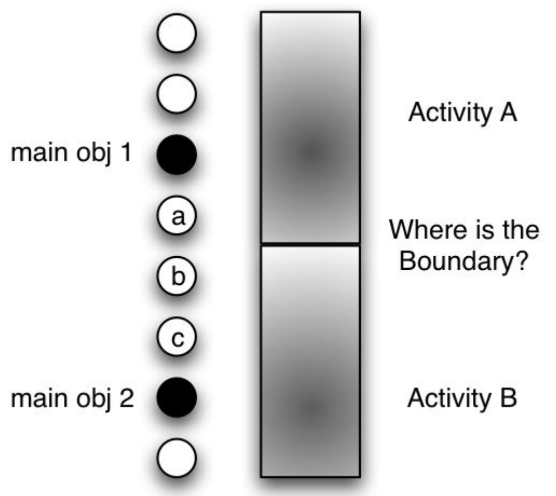


Fig. 7. Boundary detection problem.

object) of each activity. While the weight information can be a good feature for detecting the presence of an activity, using pairwise sequence information of objects may help refine the discrimination process. For instance, activities in the kitchen such as making coffee or tea share many objects having similar weights. In this case, weight information may not be enough to achieve good accuracy in discrimination.

Our *Bigram* algorithm follows certain principles. First, we assume that object-use events are sequential, not simultaneous. This is based on our observation that although two activities may be overlapping, only one object is being manipulated by a person at a certain period of time. For instance, it is not normal to see a person operating the TV remote control button and dialing a phone at the same time. Typically, a person has to either focus his attention in manipulating the remote button or the phone keypad. The implication of this observation is that each object-use event belongs only to one activity cluster. Second, we assume that related objects in an activity usually cluster around their key object (locality criterion). This is based on the observation that the key object is the most important object to complete an activity and most related objects can be found in its vicinity.

The *Bigram* algorithm works as follows. For each activity during training, we compute the bigram probability of each object pair and their corresponding weights. Then, we identify the key object for each activity based on the highest weight. For the discrimination process in a given activity trace, we employ a two-stage clustering process to assign the object-use events to different clusters. For the preliminary stage, each object-use event in an activity trace is assigned to those clusters in which its bigram probability is non-zero. To refine the discrimination process, we employ the locality criterion in the second stage. This stage makes sure that any object-use event shared by different clusters is exclusively assigned to the cluster with its topmost term closest to the object.

VI. Service Management

We briefly describe the service management in CAMPH in this section; further technical details can be found in [12].

The services in CAMPH are managed according to two perspectives: local and global. Services that are collocated within a predefined geographical region by the management of the services are known as *local services* in that region in contrast to services categorized otherwise, which are known as *global (non-localized) services*. For example, a clinic-X service within a commercial mall-Y is regarded as a local service with respect to its proximity with mall-Y. Hence, locating a local clinic in mall-Y will always return clinic-X, whereas, locating a clinic globally may return clinic-X together with the other clinics outside the proximity of mall-Y. It is often the case that service consumers in real world tend to utilize more often the collocated services. Other applications requiring cross-space services, however, may take advantage of 'global' services discovery for their respective purposes. It is therefore logical to organize services into dual views to support both needs: the Local Service Management View and the Global Service Management View.

Local service management further organizes services in a hierarchical manner (e.g. by area or region). Different areas of services are linked though a predefined hierarchy template. For a specific local area (the lowest of the hierarchy), the services are organized using one of the following approaches: (i) a centralized local service manager, or (ii) a local p2p service management using Chord [13].

Global service management organizes services according to their domain classification, e.g. ODP [14]. Each domain has a corresponding Domain Service Manager as part of our system servers that performs global service management and domain-specific context reasoning for global service discovery. In addition, there is a Root Domain Service Manager that performs domain hierarchy maintenance and handles service registration. Unlike the local service management distributing service information to respective local areas, the global view stores all the information at the system servers.

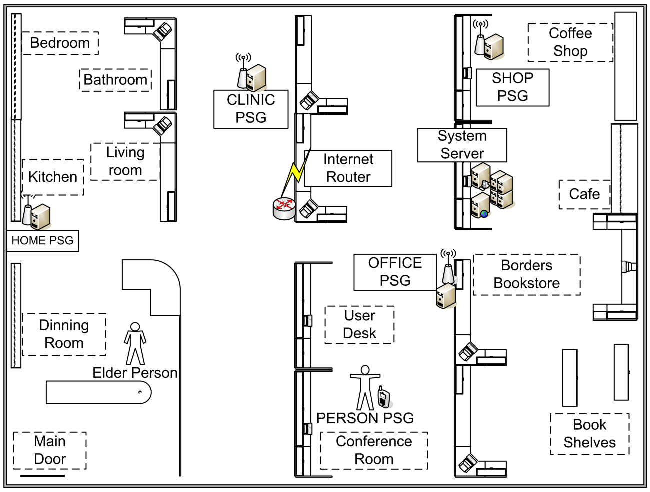


Fig. 8. Testbed deployment of CAMPH prototype.

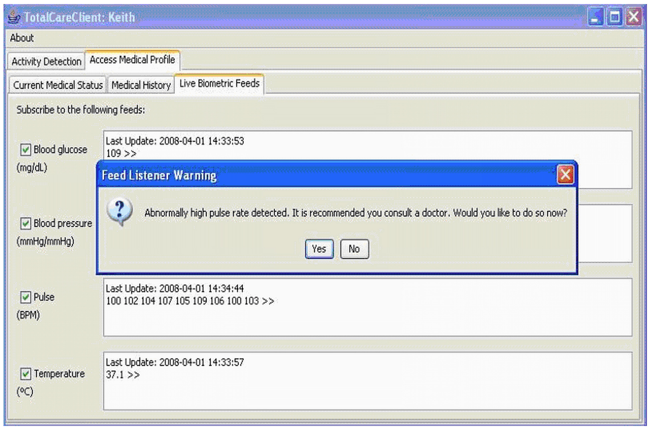


Fig. 9. Personalized homecare application.

VII. System Prototype and Evaluation

CAMPH is one of the eight projects under the UWB (ultra-wide-band) Sentient Computing research program [15] in Singapore that aims to provide the middleware infrastructure for the program. The other projects in the program explore related research areas such as UWB wireless communications, sensor networking, localization methods and audio/video signal processing.

We are making continuous research progress on the development and improvement of various components in CAMPH. These components are progressively integrated into a system prototype for performance evaluation and functional demonstration of the middleware infrastructure. In our current prototype, we used desktop PCs for the system servers and four PCs in a LAN to virtualize all PSGs registered to the middleware. Each PC has an Intel Pentium IV 2GHz CPU and 1GB main memory running Windows XP.

A. System Testbed

We have set up a testbed deployment of the prototype in our research laboratory. The testbed initially included five real physical spaces in different context domains: a smart home, office, shop, clinic and person. Each physical space was equipped with a separate desktop computer as its gateway except for the person space that had a laptop PSG to enable mobility. Each space contained a number of Crossbow MICAz motes [16] and RFID tags to provide sensory context data. To enrich demonstration scenarios, the testbed further included many other physical spaces in various context domains with simulated sensor data, together with a remote home space at the Institute for Infocomm Research that tracked and reported the eating activities of a patient in real-time [17].

Figure 8 illustrates the deployment layout of the CAMPH testbed excluding the remote home space. Dashed rectangles in the figure indicate the names of several sub-areas within a physical space. The names are used to symbolically represent the person's location when the person is moving within the spaces in our experiments.

B. Application demonstrator

We have developed a case study context-aware application to evaluate and demonstrate the functionalities of CAMPH. The scenario begins with Madam Tan, the elderly mother of Keith, who had recently been hospitalized due to a heart attack. Although she had been discharged, the doctor still needed to monitor her condition for a few days. Upon her return to her smart home (a HOME space) with body sensors installed and a person-PSG set up (her PERSON space), her son, Keith, (another PERSON space) will act as her principal caregiver. Fig. 9 shows a GUI screenshot of this *personalized homecare application*. It runs at the PDA of Keith, where his PERSON space PSG also runs. Through the GUI, Keith can subscribe to various services packaged in the PDA application.

The application allows Keith to track the daily activities of his elderly mother staying at home alone when he goes to work in his office (an OFFICE space). It alerts or recommends him different actions depending on the seriousness of the events happening to his Mum with respect to his personal context and activities, such as where he is and what he is doing. Various real-time bio-signals monitoring services are linked in the application to monitor Madam Tan's current medical profile, viz. blood glucose, blood pressure, pulse, body temperature, etc. Simulated body sensors on Madam Tan update their data to Keith periodically. The application is also able to link to services that monitor the daily activities of Madam Tan such as eating, as well as events such as falling down, so that any abnormal activities of her can be detected by Keith in a reasonable period of time.

We anticipate that similar applications can be extended to formal caregivers such as family doctors. Activities can be detected by the physical space reasoner in real time and stored in an activity report at Madam Tan's PSG as requested by the applications. This report can be accessed by Keith later for a historical review or be pushed to him whenever it is updated. When the application deduces through a cross-space reasoner that Madam Tan does not feel well and Keith should bring her to a clinic (a CLINIC space), the clinic PSG automatically detects and obtains from Madam Tan's PSG the most recent medical data for the doctor to view.

TABLE II
TIME BREAKDOWN OF PHYSICAL SPACE REGISTRATION

Item	Time (Milliseconds)
Registration Request	109
Schema Matching	23
Return List of Semantic Clusters	954
P2P Connection Setup	125
Total	1211

TABLE III
TIME BREAKDOWN OF A CONTEXT QUERY

Item	Time (Milliseconds)
Query Parsing	16
Context Domain Lookup	15
Semantic Cluster Lookup	85
P2P Physical Space (PSG) Lookup	903
Sensor Data Acquisition	442
Result Reply	16
Total	1477

In summary, the following technologies in CAMPH are demonstrated in the case study application[4] : (i) the application is composed of web services each of which may invoke the other web services, (ii) interactions take place over multiple physical spaces – person, home, office and clinic, (iii) new physical spaces can be added or removed from the system, and be detected with the domain schema updated, (iv) the system responses are prompted depending on the tasks, e.g., detection of falling down is almost instantly based on human perception.

We present a number of performance evaluation results we obtained from our current CAMPH prototype in the following subsections.

C. Timing of Physical Space Registration

Table II shows the time breakdown upon the registration of a physical space. Each value in the table is the average of tens of independent experimental runs with different parameters in the prototype system varied, including the number of context domains, number of semantic clusters per domain and number of peers per semantic cluster. In a run of the experiment, one context domain and a few semantic clusters were randomly selected for the physical space to register.

In the table, "registration request" is the time lapsed from the time the PSG sends a request to the system servers to the time the servers finish parsing the request; "schema matching" is the time spent in the schema manager to incorporate the local PSG schema to the global set of domain schemas; "return list of semantic cluster" is the time taken for the context space manager to identify all semantic clusters the PSG needs to join and return this list to the PSG, including the time to create new semantic overlays or clusters if necessary; "P2P connection setup" is the time that the PSG has taken to connect to its P2P neighbors.

The total PSG registration time in the table is approximately equal to 1.2 seconds, and is dominated by the time spent on getting the return list of semantic clusters. This is due to the considerable computation overhead for searching and creating the semantic data structures for the joining PSG. The time for the other items are relatively small.

D. Performance of Context Query Processing

Table III shows the time breakdown of a selection-projection context query in our prototype, such as Queries 1-2 in Section IV. In each run of this experiment, we registered 1000 PSGs to a randomly selected semantic cluster in a certain context domain, which we called the *query cluster*. We then randomly selected a fraction of the PSGs in the query cluster containing the required results for the query used in this run. All PSGs in the cluster were deployed to the four PCs in our prototype uniformly. Each value in the table is the average of tens of independent experimental runs using different queries.

In the table,"query parsing" is the average time spent on analyzing the query string at both system servers and PSGs; "context domain lookup", "semantic cluster lookup" and "P2P physical space lookup" represent the three steps of context data lookup in a scan operator described in Section V; "sensor data acquisition" is the time taken by a PSG in fetching data from the sensor nodes in the physical space; "result reply" is the time these PSGs took to package and send back the query result to the scan operator at the server.

We observe from the table that the total response time of a context query is about 1.5 seconds. However, the delays of P2P query routing and sensor data acquisition dominate the latency of the query response time. It is worth mentioning that if the context data requested by a query is not live sensory but legacy data in databases or main memories, the corresponding context data acquisition time would be in the range of a few milliseconds. In this case, the response time of a context query should be reduced to about one second in the prototype.

The large P2P PSG lookup time in Table III is mainly due to the less efficient Gnutella flooding protocol used in our prototype. This has prompted us to research for a new and efficient P2P search technique as most existing P2P protocols based on distributed hash tables (DHTs) [13] are not suitable for indexing of dynamic context data.

We further studied the query response time with respect to the number of PSG in each semantic cluster. We found that the response time increases gradually and then flat-off in a sub-linear fashion with the increasing cluster sizes (from 200 PSG to 1000 PSG in a step size of 200). We also increased the number of PCs used for the virtualization of PSG beyond the current four in the prototype and found that such change in the setup had very little impact on the query response time.

In addition, we have evaluated the performance of our context schema matcher using the schemas for 30 real-world websites in three context domains. The results showed that the matcher can have as high as 95% precision and 85% recall of context attribute matching with human confirmation.

[4]We have also developed a personalized shopping application running over the same testbed, providing advises/recommendations to a person by exploring contexts at home, office, person and shop.

Fig. 10. (a) RFID glove readers, (b) RFID tagged objects, and a subject is performing the *make tea* activity, and (c) another subject is performing the *make coffee* activity.

TABLE IV
ADLS PERFORMED

1	make coffee	9	make orange juice
2	make tea	10	take pills
3	make pasta	11	read books
4	make oatmeal	12	clean dining table
5	fry eggs	13	play PC games
6	make phone call	14	watch TV
7	brush teeth	15	put on make-up
8	wash clothes	16	use toilet

The accuracy achieved by the matcher for context schema matching is 100%. The detailed results are available in [9].

E. Accuracy of Activity Recognition

To evaluate our low-level context reasoner with respect to the task of activity recognition, we set up our sensor platform as illustrated in Fig. 10.

Trace collection was done in a smart home environment. We tagged over 100 objects using RFID tags. We had four volunteers wearing RFID reader gloves on their dominant hands. The data were collected over a period of two weeks. Each day, each of the volunteers selected 12 to 16 ADLs from a possible set of 45, and performed his choices in any order as he liked. Table IV summarizes all 16 activities they performed. The subjects were instructed to consecutively perform their activities in order to avoid overlapping activities. There was only one subject performing activities at any given time to reduce annotation efforts. A sequence of traces (i.e. RFID tag readings) was logged each day at a server with a Crossbow Mica2Dot wireless sensor interface board [16] connected to it. We annotated the traces by hand to establish the ground truth for our experiments. These manual annotations were only used for evaluating our algorithms.

Given a trace, the activity recognition algorithm computes the number of activities recognized and identifies the name of each activity. We calculated the following two performance metrics, *precision* and *recall*, and compared the results to the ground truth:

$$precision = \frac{tp}{tp + fp} \quad (2)$$

$$recall = \frac{tp}{tp + fn} \quad (3)$$

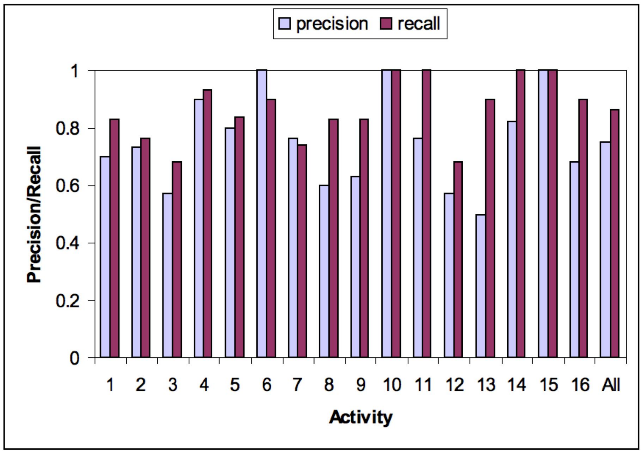


Fig. 11. Per-activity and overall precision/recall of the *MaxGap* algorithm. Numbers in x-axis correspond to the sequence numbers in Table IV.

True Positive (TP) refers to the number of instances where an activity recognition algorithm correctly identifies an activity. *False Positive (FP)* refers to the number of instances where the algorithm spuriously detects an activity that does not actually occur. *False Negative (FN)* denotes the number of times that an activity that actually occurred is not detected.

Figure 11 plots the results of both per-activity precision and recall of the *MaxGap* algorithm. As shown in the rightmost pair of bars, the algorithm achieved the overall precision and recall of 75.1% and 86.4%, respectively. From these results, we observe a limitation that a number of missed detection occurred when RFID readers failed to report the tag IDs or received corrupted tag IDs. This kind of errors is a common problem among all sensor-based activity recognition systems. Efforts to improve the reliability of sensor deployment and acquisition process require further investigation.

Our initial evaluation for the *Bigram* algorithm involved most activities from Table IV except activities 12, 15, and 16. In this evaluation, we prioritized kitchen activities because of their inherent difficulty due to the presence of many common objects in their workflows. In order to check for the robustness of our algorithm, we had to control the degree of overlapping using simulated data. The data for training and generating the activity traces were generated by extracting the sequence of objects in the activities mined from the web.

Figure 12 shows the precision/recall of the *Bigram* algorithm for non-overlapping activity traces. The algorithm performed well in both metrics (98%–100%). As expected, the worst performance happened among kitchen activities while its best performance was in activities that did not share common objects such as “making phone call”, “read books”, and “play PC games”.

In order to test the performance of the *Bigram* Algorithm for overlapping activity traces, we generated different degrees of overlapping. The degree of overlapping was controlled by the size of the skip-block used. A *skip-block* is a block of related objects in one activity inserted between two related objects of another activity. Figure 13 shows an example of generating an activity trace containing skip-blocks size of 2. It uses similar principle of merging in the merge-sort algorithm where the

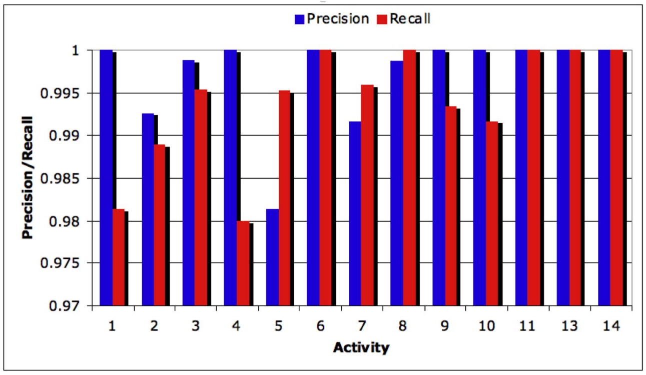


Fig. 12. Overall precision/recall performance of the *Bigram* algorithm for non-overlapping activity traces.

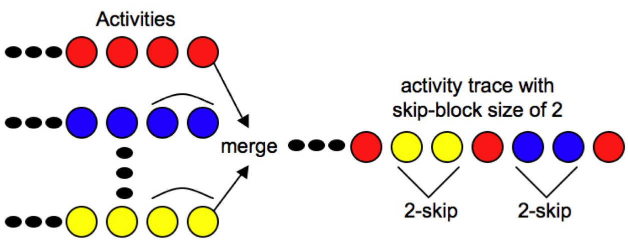


Fig. 13. Generating activity trace containing skip-blocks of size 2.

relative order of objects in each activity is preserved in the final output. The current evaluation used skip-block of sizes ranging from 1 to 6. The smaller is the range of the skip-block size used, the higher is the degree of overlapping. Since the average size of each activity is around 10 objects, the degree of difficulty we have chosen for this experiment is of medium quality.

Figure 14 shows the precision/recall of the *Bigram* algorithm for overlapping activity traces. It can be seen from the plot that the performance of the *Bigram* algorithm was robust from the presence of overlapping activities. Similar to the trend of the preceding problem, its worst performance happened in kitchen activities. For the other activities with less common objects, the algorithm performed almost as good as with the preceding problem. For overlapping activities that do not share common objects, the first stage of clustering is enough to discriminate the objects. However, if the overlapping activities involve common objects, our experiments indicated that the locality criterion is an effective mechanism to resolve the conflict of assignment among competing clusters.

While the finding of this initial study is very encouraging, there is still much to be done to improve further our algorithm. In the future, we would like to study the effect of changing the size of skip-block on the overall performance of the algorithm. Also, more experiments will be conducted by our group to improve its performance using other features (e.g. location, frequency, time) in combination with other weighting criteria.

VIII. Related Work

Existing context-aware middleware infrastructures hardly consider multiple physical spaces of different context domains. Most systems only involve a single physical space (e.g. a

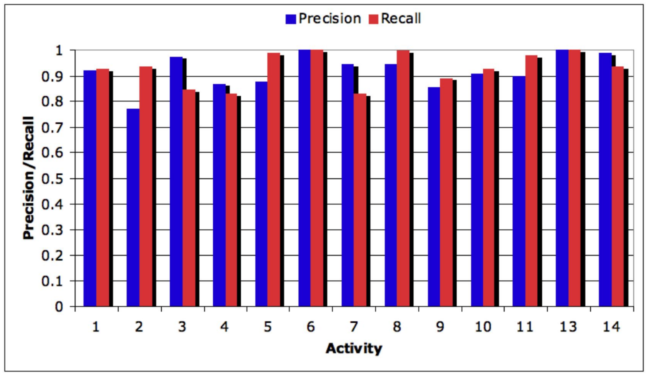


Fig. 14. Overall precision/recall performance of the *Bigram* algorithm for overlapping activity traces.

smart room), or multiple spaces (e.g. smart rooms) in a single domain (e.g. a smart home). There is also very little study on effective and efficient mechanisms for organizing and searching context data over various physical spaces as we have proposed in CAMPH.

Henricksen and Indulska [18] proposed a set of conceptual models and designed a corresponding infrastructure to support the software engineering process of context-aware applications. While this conceptual framework abstracts the functionalities of a context-aware infrastructure, the design challenges for various system components remains unsolved and the programmability of their system needs to be further investigated. In comparison, we have completed the first-hand designs and implementations for all system components in our CAMPH middleware, as we presented in this paper. CAMPH provides simple SQL-based or keyword queries to help applications conveniently access context data or services from the infrastructure.

Hu et al. [19] studied model-based automatic context data management in their system prototype called ACoMS. The authors focused their work on providing fault-tolerant context data acquisition for applications by dynamic self-configuration of context sources as well as runtime replacement of faulty sources. The context sources in the work are individual sensors rather than the high-level abstraction of physical spaces in our study. We make the underlying details of context acquisition in a physical space transparent to the applications via the context data services at the physical space gateway, and use system services at the gateway to monitor and configure local sensor network in the physical space.

Chen et al. [20] have proposed the CoBrA infrastructure for context representation, knowledge sharing and user's privacy control. CoBrA provides a centralized model where context data is shared by all devices, services and agents in a smart space. A set of ontologies written in OWL has been developed in CoBrA for an intelligent meeting room application. The scalability of such ontology context data model is a concern. In our CAMPH, the context data is distributed to multiple physical spaces in different context domains and we apply a simple but pragmatic attribute-value approach for context modeling.

Judd and Steenkiste proposed the Contextual Information Service (CIS) [21] that defines physical spaces as one type of context source while the other types are people, devices and networks. In comparison, our definition of physical spaces in CAMPH includes all context sources of various types such as persons, homes, clinics and hospitals. The places in the CoolTown [22] project bear a similarity to our physical spaces. However, all places in their system are managed as a whole without considering high-level semantics of context domains for classifying the large number of individual places as in our middleware.

A physical space in CAMPH is similar to an active or smart space in Gaia [23]. Gaia provides many services to manage and program a physical space and abstracts the data lookup in the space into a context service. Context data can be acquired from a context provider in Gaia by issuing a query or by subscribing to the event channel published by the provider. In CAMPH we organize multiple physical spaces in different context domains as peers in many semantic overlay networks, and map context data lookup from these spaces to P2P search in these networks.

Huang [24] defined a global virtual context domain used in all applications supported by a middleware. Each application processes a particular sub-domain projected from the global domain and those sub-domains of different applications may overlap with each other. Our CAMPH allows sub-domains for individual context domains without imposing a pre-defined super-domain over all domains. The context schema matcher in our middleware dynamically creates and updates a global set of domain schemas that describe various context domains and automatically categorizes the physical spaces into these domains based on their local schemas.

The authors of Semantic Space [25] and SOCAM [26] have studied the context data management over physical spaces in different domains. Different from CAMPH, these systems have not used the semantics of 'context domain' to organize the physical spaces for accelerating context data lookup. In these systems, each physical space separately develops and registers its own wrappers for every context provider in the space. The details of these wrappers are not provided.

Castelli et al. [27] used a simple quadruple model – (who, what, where, when) to represent context data from the physical world and to enable user queries upon the data. The authors further designed and implemented a middleware infrastructure for context-aware applications to effectively browse and access the modeled data using GIS tools on mobile user devices such as laptop and PDA. The authors have developed application examples to demonstrate the functionalities of their system, but a detailed performance evaluation is missing. The work also assumes a common name space for the tuple fields in the model among all context sources. In comparison, we have investigated schema matching across heterogeneous physical spaces in our CAMPH and evaluated in detail the performance of individual components in our current system prototype.

The problem of schema matching has been widely studied in database community [28]. Compared to traditional database scenarios, we have considered a few novel features of physical spaces in context-aware computing when designing our light-weight and adaptive context schema matcher in CAMPH. These features include no instance-level or constraint-based matching as well as the autonomous, dynamically-joining physical spaces [9].

Most existing papers [29]–[44] on human activity recognition have focused on using wearable sensors to collect the activity traces and applying supervised learning techniques to label the training data manually. Bao et al. [29] placed accelerometers on a human body and applied pre-trained classifiers to recognize activities from the collected data. Tapia et al. [30] proposed an activity recognition system based on a set of small and simple sensors, and used the Naive Bayesian classifier to predict the activity labels.

Philipose et al. [31] proposed to attach RFID tags on objects of interest, and represented each activity as a probabilistic order of the objects that the human uses when performing the activity. These activity models are converted into the Dynamic Bayesian Networks (DBNs) to compute the probabilities of the activities.

Vail et al. [40] proposed to use Conditional Random Field (CRF) to infer activity labels given a set of observations. Kasteren et al. [41] evaluated and compared CRF and HMM in a smart home setting. The variants of CRF such as Skip Chain CRF [42] can be used to recognize overlapping activities and Factorial CRF [43] can be used to recognize concurrent activities.

Compared to all these previous techniques, our proposed activity recognition algorithms in CAMPH are based on an unsupervised approach. Moreover, these previous techniques typically require the classifiers to be pre-trained by a set of manually labeled data. Hand labeling is time-consuming in a real-world condition and requires domain expertise to label the training data correctly.

Perkowitz et al. [44] proposed an interesting work on mining the generic models for many day-to-day activities on the web. They represent activities using HMM, a special type of DBN, based on the order of objects used with probabilistic distributions. Their models rely on manually segmenting the traces, whereas in CAMPH we develop algorithms to segment the traces automatically. Furthermore, the use of order of objects may fail to capture the idiosyncrasies of any particular activity in real-world deployments. In comparison, our activity recognition algorithms rely on the object weights rather than orders for discrimination.

IX. Conclusion

We present in this paper our design and implementation of a context-aware middleware for pervasive homecare, CAMPH, which is used to support various pervasive elderly-homecare applications. The main strength of CAMPH is its ability to support context-awareness to services operating over multiple physical spaces. It delivers applications to end-users through existing Internet infrastructure. This setup enables elderly to interact or be in-touched with care-giving stakeholders 'living' at different geographical spaces, regardless of whether the elderly are at home or moving from one physical smart-space to the other. In this way, a continuous care of the elderly at anywhere and at anytime can be delivered. The working principles of CAMPH have been demonstrated via an example personalized homecare application mashed from web services running in our system testbed with multiple physical spaces in different context domains.

We have considered several key challenges when designing CAMPH as an infrastructure for delivery of context-aware services over Internet: (i) how to manage the massive amount of dynamic context data generated by huge number of sensors from various spaces, (ii) how to manage potentially large number of smart physical spaces, some of which can be mobile, (iii) how to provide the intelligence for reasoning over multiple physical and context domains, (iv) how to manage services and support orchestration of context-aware services. These considerations have led us to the design approach of modeling the physical world as different but limited classes of physical spaces so that we can better manage their resources and services in term of locating, processing, and provisioning of context-aware intelligence.

Our present prototype is still far from usable. However, the usability issue cannot be assessed until we have developed the prototype to a stage that is suitable for a large scale field trial involving substantial number of elderly and caregivers. There are also some outstanding technical issues to be solved. We are working on a more efficient method to locate dynamic context data in a P2P network, semi-or unsupervised methods of recognition in a specific domain for overlapping activities, reasoning techniques in cross-domain environment, accurate service matching methods for service lookup and orchestration, as well as context-aware workflow specification and support for service composition.

ACKNOWLEDGMENT

The work reported in this paper has been supported by the Science and Engineering Research Council (SERC) of the Agency for Science, Technology and Research of Singapore (A*Star) under the research grant SERC 0521210083. The authors are grateful for their support.

REFERENCES

[1] M. Alwan and J. Nobel, "State of technology in aging services," Center for Aging Services Technologies, Tech. Rep., Nov. 2007.

[2] ——, "State of technology in aging services according to field experts and thought leader," Center for Aging Services Technologies, Tech. Rep., Feb. 2008.

[3] A. Dey, "Understanding and using context," *Personal Ubiquitous Computing*, vol. 5, no. 1, pp. 4–7, Feb. 2001.

[4] T. Gu, H. Pung, and D. Zhang, "Information retrieval in schema-based p2p systems using one-dimensional semantic space," *Computer Networks, Special Issue on Innovations in Web Communications Infrastructure*, vol. 51, no. 16, pp. 4543–4560, Nov. 2007.

[5] R. Ramakrishnan and J. Gehrke, *Database Management Systems*. McGraw-Hill, 2002.

[6] SOAP Specifications, http://www.w3.org/TR/soap/.

[7] Web Service Definition Language, http://www.w3.org/TR/wsdl.

[8] W. Xue, H. Pung, W. Ng, C. Tang, and T. Gu, "Gateways of physical spaces in context-aware computing," in *Proc. 4th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP 08)*, 2008, pp. 441–446.

[9] W. Xue, H. Pung, P. Palmes, and T. Gu, "Schema matching for context-aware computing," in *Proc. 10th International Conference on Ubiquitous Computing (Ubicomp 08)*, 2008, pp. 292–301.

[10] Gnutella, http://rfc-gnutella.sourceforge.net/.

[11] Jena Semantic Web Framework, http://jena.sourceforge.net/.

[12] J. Zhu and H. Pung, "A practical framework for context-aware service organization and discovery," Network Systems and Services Lab, Department of Computer Science, National University of Singapore, Tech. Rep., Sep. 2008.

[13] I. Stoical, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proc. ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM 01)*, 2001, pp. 149–160.

[14] Open Directory Project, http://www.dmoz.org/.

[15] Ultra Wide Band-enabled Sentient Computing (UWB-SC), http://cnds.ece.nus.edu.sg/uwb-sc/.

[16] Crossbow, Inc., http://www.xbow.com/.

[17] A. Tolstikov, J. Biswas, C. Tham, and P. Yap, "Eating activity primitives detection – a step towards adl recognition," in *Proc. 10th International Conference on e-Health Networking, Applications and Services (Healthcom 08)*, 2008, pp. 35–41.

[18] K. Henricksen and J. Indulska, "Developing context-aware pervasive computing applications: Models and approach," *Pervasive and Mobile Computing*, vol. 2, no. 1, pp. 37–64, Feb. 2006.

[19] P. Hu, J. Indulska, and R. Robinson, "An autonomic context management system for pervasive computing," in *Proc. 6th International Conference on Pervasive Computing and Communications (PerCom 08)*, 2008, pp. 213–223.

[20] H. Chen, T. Finin, and A. Joshi, "Semantic web in the context broker architecture," in *Proc. 2nd International Conference on Pervasive Computing and Communications (Percom 04)*, 2004, pp. 277–286.

[21] G. Judd and P. Steenkiste, "Providing contextual information to pervasive computing applications," in *Proc. 1st International Conference on Pervasive Computing and Communications (PerCom 03)*, 2003, pp. 133–142.

[22] T. Kindberg and J. Barton, "A web-based nomadic computing system," *Computer Networks*, vol. 35, no. 4, pp. 443–456, Mar. 2001.

[23] M. Romn, C. Hess, R. Cerqueira, A. Ranganat, R. Campbell, and K. Nahrstedt, "A middleware infrastructure for active spaces," *IEEE Pervasive Computing*, vol. 1, no. 4, pp. 74–83, Oct.–Dec. 2002.

[24] Q. Huang, "Supporting context-aware computing in ad hoc mobile environments," Department of Computer Science and Engineering, Washington University, Tech. Rep., 2002.

[25] X. Wang, J. Dong, C. Chin, S. Hettiarachchi, and D. Zhang, "Semantic space: An infrastructure for smart spaces," *IEEE Pervasive Computing*, vol. 3, no. 3, pp. 32–39, July–Sep. 2004.

[26] T. Gu, H. Pung, and D. Zhang, "A service-oriented middleware for building context-aware services," *Journal of Network and Computer Applications*, vol. 28, no. 1, pp. 1–18, Jan. 2005.

[27] G. Castelli, A. Rosi, M. Mamei, and F. Zambonelli, "A simple model and infrastructure for context-aware browsing of the world," in *Proc. 5th International Conference on Pervasive Computing and Communications (PerCom 07)*, 2007, pp. 229–238.

[28] E. Rahm and P. Bernstein, "A survey of approaches to automatic schema matching," *The VLDB Journal*, vol. 10, no. 4, pp. 334–350, Dec. 2001.

[29] L. Bao and S. Intille, "Activity recognition from user-annotated acceleration data," in *Proc. 2nd International Conference on Pervasive Computing (Pervasive 04)*, 2004, pp. 1–17.

[30] E. Tapia, S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in *Proc. 2nd International Conference on Pervasive Computing (Pervasive 04)*, 2004, pp. 158–175.

[31] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Fox, H. Kautz, and D. Hhnel, "Inferring activities from interactions with objects," *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 50–57, Oct. 2004.

[32] B. Logan, J. Healey, M. Philipose, E. Munguia-Tapia, and S. Intille, "A long-term evaluation of sensing modalities for activity recognition," in *Proc. 9th International Conference on Ubiquitous Computing (Ubicomp 07)*, 2007, pp. 483–500.

[33] C. Lombriser, N. Bharatula, D. Roggen, and G. Trster, "On-body activity recognition in a dynamic sensor network," in *Proc. 2nd International Conference on Body Area Networks (BodyNets 07)*, 2007.

[34] D. Wilson and C. Atkeson, "Simultaneous tracking and activity recognition (star) using many anonymous, binary sensors," in *Proc. 3rd International Conference on Pervasive Computing (Pervasive 05)*, 2005, pp. 62–79.

[35] J. Lester, T. Choudhury, and G. Borriello, "A practical approach to recognizing physical activities," in *Proc. 4th International Conference on Pervasive Computing (Pervasive 06)*, 2006, pp. 1–16.

[36] J. Ward, P. Lukowicz, G. Trster, and T. Starner, "Activity recognition of assembly tasks using body-worn microphones and accelerometers," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 28, no. 10, pp. 1553–1567, Oct. 2006.

[37] D. Patterson, D. Fox, H. Kautz, and M. Philipose, "Fine-grained activity recognition by aggregating abstract object usage," in *Proc. 9th International Symposium on Wearable Computers (ISWC 05)*, 2005, pp. 44–51.

[38] D. Wyatt, M. Philipose, and T. Choudhury, "Unsupervised activity recognition using automatically mined common sense," in *Proc. 20th AAAI Conference on Artificial Intelligence (AAAI 05)*, 2005, pp. 21–27.

[39] S. Wang, W. Pentney, A. Popescu, T. Choudhury, and M. Philipose, "Common sense based joint training of human activity recognizers," in *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI 07)*, 2007, pp. 2237–2242.

[40] D. Vail, M. Veloso, and J. Lafferty, "Conditional random fields for activity recognition," in *Proc. 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 07)*, 2007, p. 235.

[41] T. Kasteren, A. Noulas, G. Englebienne, and B. Krse, "Accurate activity recognition in a home setting," in *Proc. 10th International Conference on Ubiquitous Computing (Ubicomp 08)*, 2008, pp. 1–9.

[42] D. Hu and Q. Yang, "Cigar: Concurrent and interleaving goal and activity recognition," in *Proc. 23rd AAAI Conference on Artificial Intelligence (AAAI 08)*, 2008, pp. 1363–1368.

[43] T. Wu, C. Lian, and J. Hsu, "Joint recognition of multiple concurrent activities using factorial conditional random fields," in *Proc. AAAI Workshop on Plan, Activity, and Intent Recognition*, 2007.

[44] M. Perkowitz, M. Philipose, D. Patterson, and K. Fishkin, "Mining models of human activities from the web," in *Proc. 13th International World Wide Web Conference (WWW 04)*, 2004, pp. 573–582.

Hung Keng Pung is an associate professor in the Department of Computer Science at the National University of Singapore. He heads the Network Systems and Services Laboratory as well as a Research Scientist at the Institute for Infocomm Research (I2R). His areas of research are context-aware systems, service-oriented computing, quality of service management, protocols design and networking. He received his Ph.D. in 1986 from the University of Kent at Canterbury, UK.

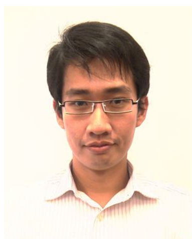
Tao Gu is currently a Scientist at Institute for Infocomm Research in Singapore. He received his Ph.D. in Computer Science from National University of Singapore. He led and co-led several research projects related to context-aware infrastructure, context reasoning and sensor-based activity recognition. His main research interests are ubiquitous and pervasive computing, including context-aware systems and wireless sensor networks.

Wenwei Xue is currently a postdoctoral research fellow in the Department of Computer Science, National University of Singapore. He received his Ph.D. in Computer Science from the Hong Kong University of Science and Technology in 2007. His current research interests are data management for pervasive computing, peer-to-peer networks, sensor query processing and context-aware middleware infrastructure.

Paulito Palmes finished his Doctor of Engineering degree from the Toyohashi University of Technology, Japan. He is currently a postdoctoral research fellow in the Department of Computer Science, National University of Singapore. His research interests are in the areas of pattern recognition, data mining, machine learning, and evolutionary computing.

Jian Zhu received his B.Sc. degree from National University of Singapore in 2005. He is currently a Ph.D. candidate in the Department of Computer Science, NUS. His research interests are context-aware service organization, discovery and matching.

Wen Long Ng received his B.Eng. degree from National University of Singapore in 2007. He is currently a research assistant in the Network Systems and Services Lab, School of Computing, NUS. His area of interest is data management in context-aware computing.



Chee Weng Tang received his B.Sc. degree from National University of Singapore in 2007. He joined the Networks Systems and Services Lab in School of Computing, NUS as a research assistant in 2008. His area of interest is mobility and personalization of physical spaces in context-aware computing.

Nguyen Hoang Chung received his M.Sc. degree from Nanyang Technological University, Singapore under ASEAN postgraduate scholarship. In 2007, he joined the Network Systems and Services Lab in School of Computing, NUS as a research assistant. His area of interest is context-aware services and workflow management.