Information, Knowledge, and Task Web Services Using Generic Service Representative

M. Najafi[1]**, K. Sartipi**[2]**, and N. Archer**[2]
[1]Department of Computing and Software, McMaster University, Hamilton, Ontario, Canada
[2]DeGroote School of Business, McMaster University, Hamilton, Ontario, Canada

Abstract— *In this paper, we propose an extension to the SOA model by adding a new component called a Service Representative. This SOA component, which simulates an enterprise agent, is modeled by a generic software agent that stays at the client site to be employed by different service providers. Moreover, different types of web services are required to model actual services in the business domain. Consequently, we propose to categorize business services into Information, Knowledge, and Task services. While a service client calls an information service to receive processed data, a knowledge service provides enterprise knowledge for either the service client or service representative. However, a service provider can assign a task to the service representative to process the client's data locally. In this paper, we introduce a prototype implementation of the extended SOA model that we used to develop a healthcare enterprise system which offers different types of services.*

Keywords: Information Services; Knowledge Services; Task Services; Generic Software Agent; Service Representative

1. Introduction

Service Oriented Architecture (SOA) proposes service computing as a solution for enterprise organizations where each enterprise business functionality is provided by one or more services. In the real-world business domain, an enterprise organization usually sends its agent or personnel (e.g., trainer, installer, and maintainer) to the client site to deliver services locally. This role has not been modeled efficiently in SOA. Consequently, the services provided by current SOA are limited to services which process the client's data completely at the server side. This limitation introduces severe constraints in the applications of web services, such as:

- *Security and privacy*: confidential client data are processed at the server side.
- *Response time*: processing the client's data at the server-side may increase the service response time.
- *Required bandwidth*: transferring large client data volumes to be processed at the server-side increases the network traffic.
- *Composability*: composition of collaborating web services are not available at the client site.

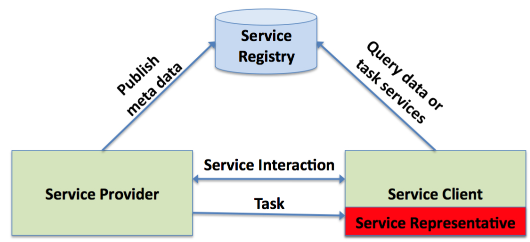


Fig. 1: Proposed extended SOA model. The proposed generic service representative receives tasks from different task services to locally process the client's data.

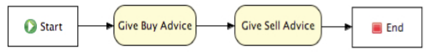
To have client-side web services, we need to simulate enterprise agents efficiently. In this paper, we propose to simulate an enterprise agent by a generic software agent which resides at the client side and is called the *Service Representative*. Then, we extend the SOA model with this new component (Figure 1). Each business service uses enterprise knowledge to process the client's data and provide information (i.e., processed data) for the client. A traditional SOA model offers *Information Services* where all the client's data processes are performed at the server side and the client just receives the resulting information. However, the extended SOA model enables service providers to process the client's data completely or partially at the client side using a novel type of service called *Task Services*. Moreover, the enterprise knowledge can be provided by *Knowledge Services* to be used by service representatives or clients.

The organization of this paper is as follows. Different types of business services are proposed in Section 2. The proposed architecture is discussed in Section 3 and the developed prototype is described in Section 4. A case study of a health-care enterprise system is presented in Section 5, and related work is discussed in Section 6. Finally, future work is discussed in Section 7.

2. Business Services Categories

A business service is a process of applying enterprise knowledge to the client-side and server-side data to provide information for the clients. As opposed to the traditional SOA models that consider business services to be identical, we propose to categorize the business services into information, knowledge, and task services, as follows.

Task Model



Task Knowledge

Give Buy Advice	Condition I	For each b ε provider. buyItem If (not exist p ε client. portfolio ∧ p.shareName = b.shareName)

	Action I	requiredCash = b.minPercentage * client. holdingValue;
		shareNo = min (requiredCash, client.cash) / b.sharePrice;
		Print " Buy" + shareNo + "of" + b.shareName;

	Condition II	For each b ε provider. buyItem If (exist p ε client. portfolio ∧ p.shareName = b.shareName ∧ p. percentage < b.minPercentage)
	Action II	requiredCash = (b.minPercentage – p.percentage) * client. holdingValue; shareNo = min (requiredCash, client.cash) / b.sharePrice; Print " Buy" + shareNo + "of" + b.shareName;
Give Sell Advice	Condition I	For each s ε provider. sellItem If (exist p ε client. portfolio ∧ p.shareName = s.shareName ∧ p. percentage > s.maxPercentage)
	Action I	shareNo = (p.percentage - s.maxPercentage) * client. holdingValue / s.sharePrice; Print "Sell" + shareNo + "of" + p.shareName;

Task Data
buyItem [shareName, minPercentage, sharePrice], sellItem [shareName, maxPercentage, sharePrice]

Client Data
portfolio [shareName, shareNumber, percentage], holdingValue, cash

Fig. 2: A task service to personalize general financial advice based on the client's personal information at the client site. *Task Model* specifies a task with two steps to customize the general advice (*Task Data*) based on the client's information (*Client Data*). Finally, *Task Knowledge* defines the required customization knowledge in each step of the task model where conditions are expressed in First Order Logic (FOL) and actions are defined using Java statements.

2.1 Information Services

Information services represent the typical web services in the current SOA model where the service provider processes the client's data completely at the provider site. Therefore, a client has to send its data as web service parameters to the service provider. Then, the service provider returns the generated information as its service response which will be consumed directly by the client.

2.2 Knowledge Services

Enterprise knowledge is a valuable asset for enterprises. However, enterprises can offer this knowledge via knowledge services to be applied remotely by the service representatives or other services. Knowledge management provides techniques to represent, store, transfer, and apply different types of knowledge. For example, descriptive and procedural knowledge can be managed by business rules and actions, respectively. A knowledge service receives a knowledge request from a service client, service representative, or service provider. Then, it uses a knowledge representation technique (e.g., PMML for descriptive knowledge or Dynamic-Link Library for procedural knowledge) to encode and send the requested knowledge. It will be the knowledge receiver's responsibility to use the knowledge to process the data and generate information.

2.3 Task Services

Task services enable enterprise systems to process the client's data and resources partially or completely at the client site. In this case, the service provider initially processes the client request and returns a *task* to be performed at the client site. The task will be executed by a service representative and the generated information will be used by the client as the service response. We propose to model a task formally by the following triple.

$$Task =< Model, Knowledge, Data >$$

The *Task Model* specifies a task by an abstract business process model. This model defines the order of applying knowledge to the client-side and server-side data; *Task Knowledge* provides the required knowledge (business rules and actions) to concretize the specified abstract process; and *Task Data* are the server-side data (business objects) that are consumed by the business rules and actions during the business process. A task service performs a task at the client site that implies some of the required data should remain at the client site since they cannot be transmitted efficiently to the provider, such as confidential or large amounts of client data.

A missing part in the current SOA-based models is an enterprise agent that enables a service provider to delegate task services to process client's data and resources locally. To support task services, we propose to extend the major com-

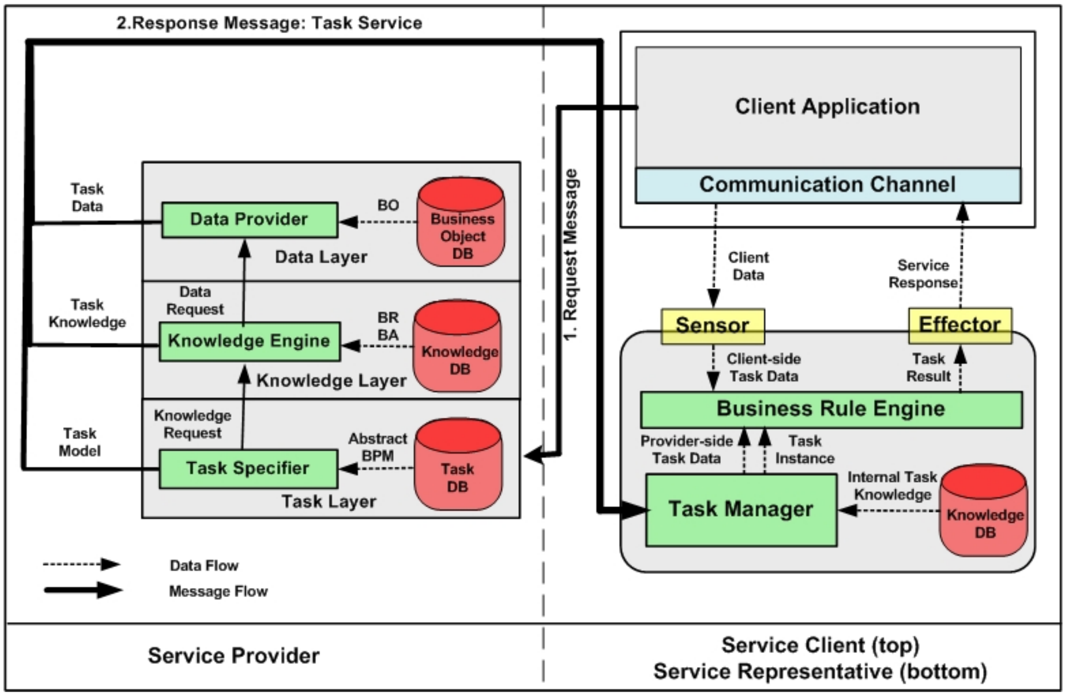


Fig. 3: Proposed extended SOA architecture with service representative and the notion of task services. Based on the client's request message, the provider generates a 3-segment response message (task message) to customize a generic agent as its service representative to perform the assigned task.

ponents of the SOA model (service provider, service client, and service registry) with the generic service representative, as it is shown in Figure 1. In this context, the service providers send *task components* to the service representative to customize it for the processing of the client's data. The service representative executes at the client site with access to the client's local resources, which can potentially violate the client's security and privacy. Therefore, the client controls the required computer resources (e.g., CPU time, storage, and memory) for the service representative.

Figure 2 illustrates different components of a task generated by a task service that provides personalized financial advice without asking the client to send its personal information. This service receives the client's general preferences such as: category of investment (stock, option, or mutual fund); term duration (short term or long term); and risk level (low, medium, or high), while the client keeps the sensitive information local and private, such as the client's financial information (portfolio, saving, debt, and salary). Then, this service generates a set of general financial advice (stock buy and sell advice) according to the client's preferences. A stock buy (or sell) advice recommends the client to have minimum (or maximum) percentage of a specific share in their portfolio. Moreover, it assigns the defined task to the service representative to personalize the generated general advice based on the local client's financial information.

3. Architecture

In this section, we extend the typical architecture of SOA implementations to enable service providers to provide information, knowledge, and task services. The proposed architecture in Figure 3 consists of three main components as follows.

3.1 Service Client

Each service client (or simply client) consists of a client application and a communication channel as follows.

3.1.1 Client Application

This is a traditional client application which can directly call and use information and knowledge services. In case of a task service, the client application puts the required client data and resources into the communication channel based on the schema received from the service registry. This schema describes the type and order of client resources that must be made accessible through the communication channel ports for the service representative. Then it sends a task request message to the service provider and waits to receive the task results through the communication channel.

3.1.2 Communication Channel

This consists of a number of ports that are connection links to the internal resources of the client application, as

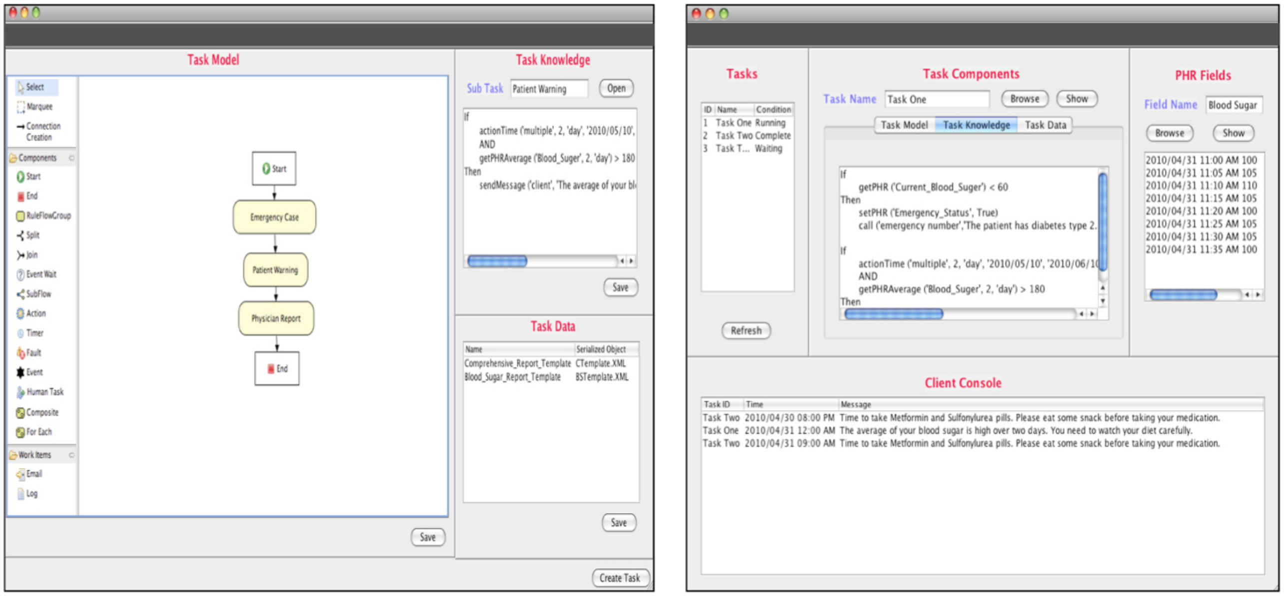


Fig. 4: (Left): defining a task service using *TaskService* package. (Right): client application GUI, provided by *ServiceRep package*, to call and execute a task service.

well as the means for the client application to receive the result of the requested task through the service representative. The client grants permission to the service representative to read/write a number of its resources through these ports where ports can be input, output, or bi-directional (from the client's point of view).

3.2 Service Provider

Information and knowledge services can be provided efficiently in the traditional SOA model where service responses have only one segment (i.e., either information or knowledge). However, a three layer service provider is required to provide three-segment response messages of task services. These layers are described as follows.

Task Layer. The *task specifier* receives a task service request from a client and retrieves the task model from the *task database*. The task model, represented by an abstract business process model (BPM), describes the process steps to achieve a specific business goal according to abstractly defined business rules and actions. Then, the task specifier sends a request to the knowledge layer to provide the corresponding business rules and actions. The rationale for having a separate task layer is to mimic the situation in real-world enterprise organizations, where the agents are assumed knowledgeable when they are assigned tasks and the required knowledge to perform the task is acquired from other sources (e.g., during the training phase).

Knowledge Layer. Knowledge layer provides the required task knowledge of a task service, as follows. The *knowledge engine* extracts the requested task knowledge from the *knowledge database* and fills out the task knowledge segment of a task service response message. Task knowledge includes business rules and actions that are abstractly defined in the task layer. A business rule or action may require server-side data that are requested from the data layer.

Data Layer. This layer provides the required server-side data of a task service, as follows. The *data provider* extracts the requested task data from the *business objects database*. Then, the objects will be serialized to be sent by messages and form the task data segment of the task service response message.

3.3 Service Representative

The service representative is modeled by a generic software agent which has a number of *sensors* to receive the client's data through a communication channel and *effectors* to send task results to the communication channel. Other components are introduced below.

- The *Task Manager* supports the entire life cycle of a task instance (i.e., from creation to termination) that is divided into three phases as follows.
  1) *Customization phase*: sets up the agent configuration (including sensors and effectors) based on the task service description published on the service registry and creates an abstract business process based on the task model segment of a task message.
  2) *Training phase*: instantiates a task instance from the abstract process using internal and external task knowledge.

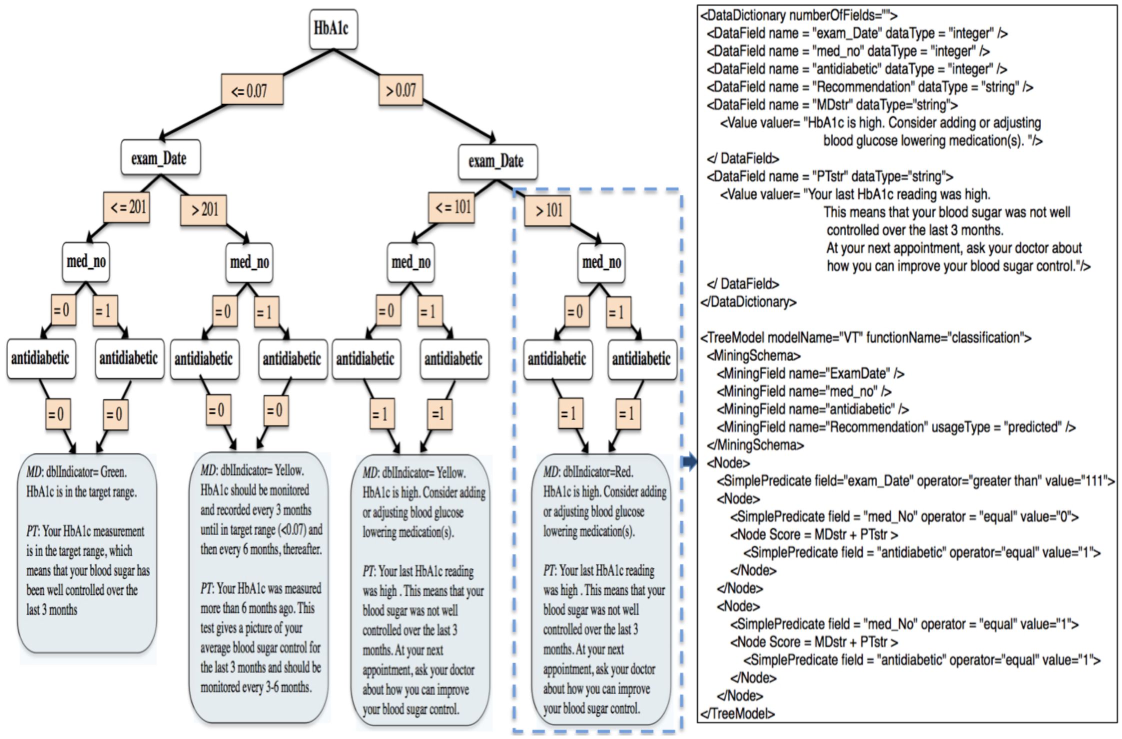


Fig. 5: Left: decision tree corresponding to a Vascular Tracker (VT) medical guideline, consisting of three types of nodes: decision, split, and action (leaf boxes). This tree gives recommendations to both patient and physician about the result of a blood test (i.e., Hb1Ac) by considering four patient PHR fields. Right: a part of the generated service response (encoded in PMML V3) by *getGuidelineWS*.

3) *Execution phase*: passes the task instance with the relevant task and client data to the business rule engine to be executed. Finally, the task results are given to the effectors to be written into the communication channel.

- The *Business Rule Engine* executes a task instance by applying business rules and performing business actions in a defined order.
- The *Knowledge Base* stores domain-based business rules and actions to relieve the service provider from sending them each time.

4. Prototype Tool

We implemented a prototype of the proposed architecture including the service representative, three-layer service provider, and client application. This prototype, namely *Enterprise Representative version 1.0* (EntRep v1.0), is implemented based on J2EE 1.5 technologies. We also used *Drools* APIs [1] to design and execute task services. Drools is a Java-based and open source rule engine that works based on the forward chaining strategy. In addition to the rule engine, Drools has a process engine that allows integration of processes and rules using a process model called *ruleflow*. The EntRep v1.0 uses Drool's rule flow as its task model and Drool's business rules and actions as its task knowledge. Moreover, EntRep v1.0 can receive and understand knowledge sentences that are compatible with PMML (Predictive Model Markup Language) V3. EntRep v1.0 is provided as two Java packages: *TaskService* and *ServiceRep* that can be imported into any service provider and client applications. While a task service developer uses the *TaskService* package to develop a task service graphically, a client application developer uses the *ServiceRep* package to generate one instance of the service representative and communication channel. Figure 4 shows snapshots of two applications developed by these two packages.

5. Case Study: Healthcare Enterprise System

Using the implemented prototype tool (EntRep v1.0), we developed a healthcare enterprise system that offers a variety of information, knowledge, and task services. This system acts as a Clinical Decision Support System (CDSS) to assist physicians and patients with decision-making tasks. Moreover, it preserves the patient's privacy, increases the system availability, and simulates healthcare resources.

Task Model

Task Knowledge

Category	Business Rules & Actions
Blood Pressure	*If* SystolicBP >= 120 *or* DiastolicBP >= 80 *Then* Report.Text:=Report.Text+"High BP"
Blood Pressure	*If* SystolicBP < 90 *or* DiastolicBP < 60 *Then* Report.Text:=Report.Text+"Low BP"
Blood Pressure	*If* max(SystolicBP) - min(SystolicBP) >= 30 *or* max(DiastolicBP) - min(DiastolicBP) >=30 *Then* Report.Text:=Report.Text+"Variable BP"
Body Temperature	*If* BT >= 36.5 *and* BT <=37.5 *Then* Report.Text:=Report.Text+"Normal BT"
Body Temperature	*If* BT >= 37.5 *and* BT <=39.5 *Then* Report.Text:=Report.Text+"Fever"
Body Temperature	*If* BT >= 39.5 *Then* Report.Text:=Report.Text+"Hyperthermia"
Symptom	*If* Symptoms ! = Empty *Then* Report.Text:=Report.Text+Symptoms

Task Data

Report Template

Client Data

BP, BT, Symptoms

Fig. 6: Virtual Remote Nursing task service. The task model guides the service representative to check the patient's blood pressure, body temperature, and recently observed symptoms, respectively. The task knowledge component provides the required guidelines for each step and the task data provides a report format defined by the patient's physician. The service representative applies this task to the PHR information specified in the client data section.

5.1 Enterprise Specification

The developed healthcare enterprise system incorporates the three-layer architecture presented in Section 3. The data layer has databases containing drug and disease information. Moreover, the knowledge and task layers have databases containing medical guidelines and procedures that belong to Vascular Tracker (VT) [2]. Medical guidelines represent descriptive knowledge in healthcare where each guideline is applied through the patient's Electronic Medical Records (EMR) to generate recommendations for both physician and patient. Similarly, a medical workflow includes some medical guidelines in a specific order. VT is a decision support system that assists physicians to observe and control patient's different risk factors within the domain of cardiovascular diseases. The VT guidelines are categorized into diabetes, hypertension, coronary artery disease, cerebrovascular disease, peripheral vascular disease, and healthy.

5.2 Information Services

The developed system provides several traditional information services. For example, *getDrugInfoWS (drug name)* and *getDiseaseInfoWS (disease name)* return drug and disease information requested by a client. Moreover, *getRecommendationsWS (guideline name, EMR fields)* applies the requested guideline to the received EMR fields and returns the resulting recommendations and diagnosis.

5.3 Knowledge Services

Each VT medical guideline is converted into a decision tree format to be stored in the enterprise knowledge base. Then, the VT medical guidelines can be provided via web services as knowledge services. For example, *getGuidelineWS (guideline name)* returns the requested guideline where it is encoded in PMML standard. The service client can apply this guideline to the patient's EMR data using any PMML engines. Consequently, the service caller always has access to the up-to-date and trusted version of the guidelines. Moreover, in a cloud environment, dynamic CDSSs can be developed using knowledge services from different sources. Figure 5 represents this service response for a VT guideline.

5.4 Task Services

The proposed service representative enables CDSSs to control and process a patient's EMR information remotely at the client's site. This feature introduces several novel services for healthcare systems such as the *Virtual Remote Nursing (VRN)* service. In this case, the service representative is customized by the CDSS to act as a private nurse for a patient (i.e., service client). For this purpose, the client feeds his/her health information (e.g., blood pressure and body temperature) regularly to a PHR system (e.g., Google Health in this case). Moreover, the client enables the service representative to access their health profile through the communication channel. Then, the client's physicians can

Table 1: Comparing different types of web services.

Type	Applications	Pros	Cons
Information Services	Server-side processing of the non-sensitive client's data	- Integrated service logic - Easy to asses the service reliability - Maintaining the enterprise privacy	- Violating the client's privacy - Increasing the network traffic - Increasing the service response time
Knowledge Services	Service-to-Service (S2S) and Service-to-Representative (S2R) transactions	- Maintaining the client's privacy - Reducing the network traffic - Improving the service response time	- Revealing the enterprise knowledge - Client's responsibility to apply the knowledge
Task Services	Client-side processing of the confidential, large volume, or real time client's data.	- Maintaining the client's privacy - Reducing the network traffic - Improving the service response time	- Distributed service logic - Difficult to assess the service reliability

assign different tasks to the service representative by defining the required task components. These tasks include, but are not limited to:

- Make a report of the client's PHR fields over a determined period of time to represent the client's health condition. These tasks guide the service representative about the required PHR fields as well as how and under which condition they should be reported. Figure 6 illustrates an example of these tasks.
- Give proper recommendations and necessary warnings to a patient in specific situations. These tasks include one or more situation specifications as well as the corresponding recommendations and warnings.

Thanks to mobile application platforms, a smart phone can take advantage of this service to guarantee that the client's health condition is always under control. A prototype version of VRN was developed and described completely in [3].

6. Related Work

The proposed work in this paper covers different topics related to web services, software agents, and business processes. A software agent is a piece of software that acts on behalf of an agency to serve a user. Software agents may have the authority to decide which action is appropriate to take. They have been integrated into web services and similarly, web services have been modeled by software agents [4]. Mobile agents can physically travel across a network and perform tasks on different nodes. Therefore, they can be considered as an alternative to the proposed resident generic agents for modelling service representatives. There are several security and privacy issues to be considered in mobile agent computing. Mobile agent architectures also suffer from low efficiency as they need to send the entire computer program or process. These issues motivated us to employ resident agents at the client site as opposed to sending them to the client.

Each business service can be modeled by a business process [5]. The proposed task services divide the service business process into server-side and client-side processing. Then, the client-side portion can be assigned to the service representative agent to be performed at the client-side. Software agent technology has been applied to model and execute business processes. The idea of the agent-based process management system is to split business processes into parts and assign the control over such part to individual software agents [6]. Integration of agents and services is proposed to model the business aspects of enterprise systems, where each role in an enterprise system is considered as an agent [7]. Finally, the proposed architecture extends our previous work in [8] where it introduces a generic software agent that resides at the client side and customizes service responses based on the client's context.

7. Conclusions and Future Work

In this paper, we extended the SOA model to support three types of services. In addition to traditional information services, we introduced knowledge and task services to efficiently model business functionalities of enterprises. However, these web services are complementary and have their own applications and features (Table 1). For future work, we plan to enable service representatives to receive their required knowledge via distributed knowledge services. Finally, collaborating web services will be able to employ the service representative to perform composite tasks at the client's site.

References

[1] M. Proctor, M. Neale, P.Lin, and M.Frandsen, "Drools documentation," JBoss.org, Tech. Rep., 2008.

[2] *Compete III Vascular Tracker website*, http://www.competestudy.com/.

[3] M.Najafi, S.Aghtar, K.Sartipi, and N.Archer, "Virtual Remote Nursing System," in *The 1st IEEE Conference on Consumer eHealth Platforms*, USA, 2011, pp. 101–105.

[4] Y. Cheng, A. Leon-Garcia, and I.Foster, "Toward an Autonomic Service Management Framework: A Holistic Vision of SOA, AON, and Autonomic Computing," *IEEE Communications Magazine*, vol. 46, no. 5, pp. 138–146, 2008.

[5] R. Lu and S.Sadiq, *Lecture Notes in Computer Science*. Springer Berlin/ Heidelberg, 2007, ch. A Survey of Comparative Business Process Modeling Approaches, pp. 82–94.

[6] R.Miller, U.Greiner, and E. Rahm, "Aw: a workflow system supporting rule-based workflow adaptation," *Data and Knowledge Engineering*, vol. 51, no. 2, pp. 223–256, 2004.

[7] L. Xiang, "A Multi-Agent-Based Service-Oriented Architecture for Inter-Enterprise Cooperation System," in *Second Conference on Digital Telecommunications*, USA, 2007, pp. 22–32.

[8] M. Najafi and K.Sartipi, "A Framework for Context-Aware Services Using Service Customizer," in *The IEEE Conference On Advanced Communication Technology.*, Korea, 2010, pp. 1339–1344.