Towards a Participant-Oriented Model to Enhance SOA Model with Mobility

Yang Syu and Yong-Yi FanJiang

Abstract—Influenced by newly emerging trends and technologies, in this paper we propose the participant-oriented model, which is an extension and revision of current Service-Oriented Architecture (SOA) model, for modern software and application development. Proposed mode has two primary actors (roles), mobile and cloud participants, and both of actors simultaneously provide and consume services against each other. Furthermore, under such a model, the unit and mechanism of composition as well as the type of register and registry are quite different from conventional SOA. By defining and composing participants, proposed model can be a pattern framework for designers to invent creative and useful applications. Real-time Road Speed Information Application (RRSIA) provided in this paper serve as example to demonstrate the proposed approach. RRSIA has a distinct dynamic binding mechanism, which is a critical concept in SOA, to separate interface and implementation.

Index Terms—Service-oriented architecture, participant-oriented model, mobile service, service composition

I. Introduction

Currently there are many different models for software development and one of them, which is followed and quite popular in both industry and academia, is Service-Oriented Architecture (SOA). Generally a SOA-based system is consisted of plural reusable services with specific process logic. Below we give a brief introduction to SOA as it is the foundation of our proposed model.

Refer to Fig. 1 that is borrowed from Introduction of [1], we use it to explain conceptual SOA model including original triangle model and modified practice model. In original model, it has three main actors: service provider, service consumer, and service registry [1-2]. A basic scenario is as below. First, a service consumer is looking for intended services on a service registry according to interface specifications, and then it composes found and intended services, which are actually supplied by service providers rather than registry, to satisfy user's requirement. On the other hand, in practice most implemented SOA-based systems have dropped service registry for some technical reasons [1]. Thus SOA practice model, shown at bottom half of Fig. 1, only has two actors, service provider and consumer. Nevertheless, many researches criticize this model as it violates SOA's core spirit, i.e. separation between service interface and implementation as well as dynamic binding during runtime. They are most attractive promises and advantages of SOA.

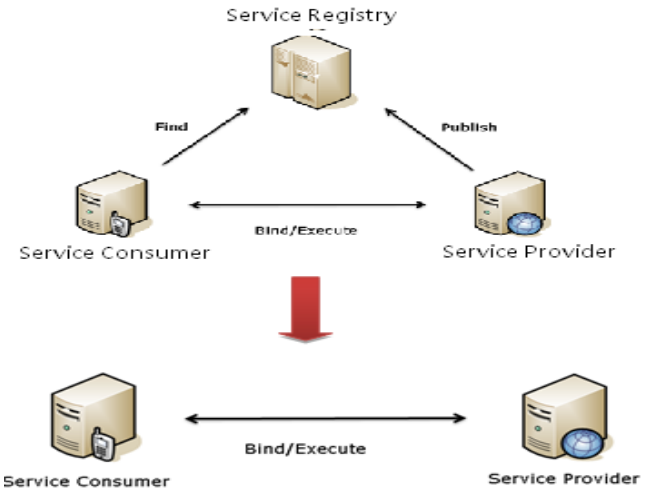

Fig. 1. SOA triangle and practice models

Thus far, there is a simple introduction to SOA for facilitating understanding and it also is a basis of our proposed model. Relatively the proposed model is more like practice SOA model instead of triangle model, but something different is that the proposed model has unusual registry, composition, and dynamic binding mechanism, which are quite different from conventional SOA registry, composition, and dynamic binding. Following we discuss the reasons and the scarcity of present models that motivate us to come up with our proposed model.

First, current SOA model is suitable for developing enterprise systems. They implicitly imply that both service provider and consumer run on fixed enterprise heavyweight server, supplying and requesting computation-intensive or transaction-based services (operations). Yet benefit from constant progress of hardware of mobile devices, today's mobile devices like smartphones and tablets can play the roles both of service consumer [3] and provider [4]. In particular, a service provider on mobile device can offer some types of service that traditional service providers are unable to provide. For example, since every smartphone is a sensor-rich device and always on its user [3], these devices can act as mobile service providers offering ambient context-sensing services for consumers to aware real-time environmental information of specific location. Apparently traditional service providers cannot furnish such type of services.

Second, traditionally the relationships between elements of a SOA system (providers and consumers) are unidirectional. Implicitly an entity acting as provider to supply services did not act as consumer requesting services.

Manuscript received July 23, 2012; revised August 16, 2012.

Y. Syu is with the Department of Computer Science and Information Engineering, National Taipei University of Technology, Taipei, Taiwan (e-mail: a29066049@gmail.com).

Y.-Y. FanJiang is with the Department of Computer Science and Information Engineering, Fu Jen Catholic University, Taipei, Taiwan (e-mail: yyfanj2@gmail.com).

We consider that this is obsolete and should be re-defined for meeting future tendency. On the other hand, it is predictable that future software context should be a mixture blending mobile and cloud elements. Under such a context, each entity comprised in an application could be fixed or movable, and the relationships between entities should be bidirectional and mutually beneficial (i.e., mutualism). In proposed model, an entity is to act as service provider and consumer at the same time. Inversely, an entity consumes services when it provides services. We name such an entity as *participant* and, in our model, there are two types of participant, *mobile participant* and *cloud participant*.

Furthermore besides providing and consuming services, in terms of participant there still must have registry listing and storing all available resource interfaces to realize separation between interface and implementation as well as dynamic binding. In our model, registry could be on mobile participant or cloud participant and the types of registry, dynamic binding, and composition are largely distinct from tradition because of the mobility and special service types involved in proposed model.

The rest of this paper is organized as follows. Section II defines and explains proposed participant-oriented model. Section III presents RRSIA and its prototype. Section IV discusses the dynamic binding and composition issues used by RRSIA. Finally, conclusion and future work are provided in section V.

II. Participant-oriented model

In this section, we define and explain proposed participant-oriented model, which is an extension and revision of two obsolete models. Fig. 2 shows the concept of proposed model. Comparing with two models in Fig. 1, the biggest distinction is that in participant-oriented model has two model roles simultaneously serving as service provider and consumer.

Another distinction is how model distinguishes disparate roles. In two SOA models, originally roles were divided by the relative relationship to service (i.e., providing, consuming, or registering service). However, in participant-oriented model each model element (participant) acts as service provider and consumer in the meantime, thus we use another manner to distinguish different participants. The manner adopted is to differentiate via the mobility (movable or not) and computation type (computation-intensive and storage-intensive, or not) of model role, rather than the relative relationship to service. The two roles in the model are “mobile participant” and “cloud participant”.

Presently a common pattern of mobile application is as follows. To acquire a result of service invocation that better meets user’s current need, a lightweight application running on mobile device (consumer) can collect environmental context information via the device’s sensors and then sending the information to a context-aware service running on heavyweight server (provider) as part of input to the service [3]. In this way, the relationship between two entities is unidirectional. Nevertheless, a more flexible pattern could consist of service participants they bi-directionally provide and consume various types of service, like context-sensing, human-provided [5], or computational services, to each other. In addition, classical centralized paradigm of SOA (there usually are plural service consumers and single service provider) could be broken by mobile participants. Within the range of a mobile ad-hoc network formed by a group of adjacent mobile participants, they can directly provide/consume services for each other, as shown in left part of Fig. 2. A description of possible scenario of ad-hoc mobile service network is in Introduction of [6]. Without an awkward center (one service provider), the topology can be decentralized, localized, and more flexible.

Following we separately define mobile participant as well as cloud participant in detail. And then there is a discussion regarding two possibilities of this model: registry on mobile participant and registry on cloud participant.

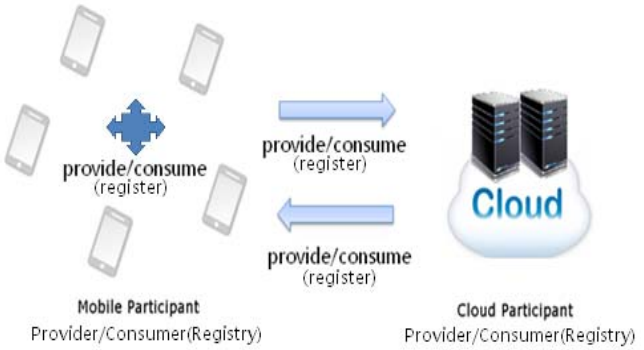

Fig. 2. Participant-oriented services computing architecture.

A. Mobile Participant

In recent years, computing paradigm has been changed hugely. In the past, computation was always done on fixed machines such as enterprise servers, workstations, and personal computers. A mutual characteristic of them is that they are not movable and they primarily process computation-intensive tasks consuming a large part of CPU resources. However, this paradigm has been broken by the emergence and popularization of mobile devices like earlier PDAs and recent smartphones and tablets. Now each modern mobile device is a powerful computing platform for software and applications running on it. Computing paradigm no longer is restricted by non-movable fixed and centralized pattern (e.g., client-server pattern). It is moving toward distributed, ubiquitous [7], and context-aware [3] computing, because every modern mobile device is small, easy-to-carry, and sensor-rich equipment. That means that computation can occur in everywhere with environmental information sensed as input.

With these conditions, modern powerful mobile devices are fully eligible to participate in service-based system to provide and consume services [8]. Thus in participant-oriented model, we call each of them (smart mobile devices) *mobile participants*. Each mobile participant is mobile equipment carried by its holder moving constantly and it can provide and consume services concurrently. Especially, in terms of offering service, the type of service provided by mobile participant would be quite different from the type of service provided by traditional service provider because, comparing with traditional provider, mobile participant have some specialties.

Mobile participants may consume any kind of services (e.g., computation-based service, human-provided service, or context-sensing service), but they should not provide computation-intensive and storage-intensive services. Mobile participants may provide context-sensing or human-provided services but they should not provide the type of service requiring enormous computation resource or

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

storage space as these types of service are, according to the concept implied in participant-oriented model, the responsibility and forte of cloud participant. A service provided by mobile participant has one or plural of the following characteristics.

- Non computation-intensive
- Non storage-intensive
- Constantly moving
- Context-aware
- Location-based
- Human-provided

Despite modern mobile devices relatively have much better hardware and capability, they still are inferior devices. Hence their main responsibility and purpose should not be massive computing and storing. Mobile participants can do something that fixed entities are entirely unable to do. Thus the goal is to make the best use of their specialties, exploiting their special features and advantages such as that "mobile device's holder always carries about it" and that "mobile devices are sensor-rich equipments", as much as possible. For instance, with the holder's movement a smart mobile device can constantly collect instant environmental information, being a mobile participant offering context-sensing service. Otherwise, a mobile device also can act as an intermediary for expert to contribute his/her expertise and professional skill (i.e. as a mobile participant providing human-provided service [5]).

B. Cloud Participant

A cloud participant is a mixture of service provider and consumer running on and profiting from cloud platform. Thus far there are many different definitions regarding what cloud computing is and where it comes from is not so clear [3]. However, in this paper we do not concern with these issues. Here the focus is on the advantage and benefit of cloud. In terms of organizations confronting exponentially increased big data, cloud computing is a best way to efficiently handle these data, while it also largely reduces costs associated with the establishment, management, and maintenance of software and hardware. Basically, cloud is a centralized pattern shifting computation and storage to network (cloud) [3]. Usually a cloud is consisted of numerous machines, dynamically supplying sufficient computing and storing resources for customers. By adopting cloud solution, the number of CPUs and the amount of storage space needed can be reallocated dynamically and economically, saving budget and ensuring the continuity and quality of business (system).

In the past, computation-intensive and storage-intensive services were always on a server of providing organization. As we just explained, current computing paradigm is shifting toward cloud computing in order to deal with growing big data and computation requirement as well as save cost. In the near future, when the price of smart device is to be lower and lower, we expect that each person will have at least one smart device on him/her. Thus under such a circumstance every person is fully eligible to become a mobile participant. Therefore consider such a situation, it is foreseeable that we will have numerous mobile participants concurrently asking computation-based and storage-based services because, as we already mentioned, a mobile participant (device) is not an appropriated place to do heavy calculation or storing. Doubtlessly cloud participant is most suited place to provide such services. For example, mobile participants can provide context-sensing services and a cloud participant is responsible to consume these services, quickly processing and integrating large amount of real-time data acquired from them in order to achieve the best use of these services. It is unwise to satisfy such kind of huge computation and storage requirements through traditional manner, therefore we need cloud to participate in the model.

With aforementioned characteristics, the main responsibility of cloud participant is to be a basis for the large amount of computation and the persistence of big data. Similar with mobile participant, a cloud participant may consume any kind of service, but the key difference is in service type provided. Obvious, cloud participant is immovable and it is unable to provide services like mobile context-sensing service that can be provided by mobile participant. Hence the type of service provided by cloud participant should be computation-intensive and storage-intensive due to the nature and constraint of cloud.

C. Registry

In proposed participant-oriented model, registry could be on mobile participant or on cloud participant, depending on specific application's design and requirement.

In original concept of SOA, registry is a cornerstone for realizing concept "separation between interface and implementation" as well as its extension "dynamic binding". One thing that must be noticed is that the elements stored in registry are service interfaces, instead of real executable service implementations (instances). Usually the multiplicity (cardinality) between interface and corresponding implementation is one-to-many, which means that there usually are multiple providers offering services they are identical in terms of functionality interface (e.g. input and output of service). Hence with plural choices dynamic binding means that which service instance of bound interface should be executed is determined during runtime according to indicated criteria, and the binding is dynamically changeable to avoid failure or quality violation of bound instance.

Traditionally, looking for intended services among a conventional fixed registry containing all available service specifications (interfaces) relies on criteria such as intended functionality or quality. However, with unusual type of provided and consumed service like mobile context-sensing service, the design of registry and the searching criteria adopted are quite different from convention. Below we discuss two possibilities of proposed participant-oriented model, i.e. registry on mobile participant and registry on cloud participant.

In participant-oriented model, mobile participants can scatter in decentralized pattern and move constantly. They can directly consume or provide service to each other without any telecommunication center (e.g., a wireless mobile ad-hoc network formed by a group of mobile participants). For reference, a similar scenario is mentioned in Introduction of [6]. To application design that registry is on mobile participant, each mobile participant has a registry maintaining list about available mobile participants as well as services provided from them. And it must dynamically and frequently update the list because, influenced by the nature of

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

mobile device, each mobile participant constantly moves and thus continuously reconstructing the network of available participants. Available cloud participants should be static and are always accessible via infrastructure network like 3G; they do not need to be updated periodically. There must be an efficient algorithm or mechanism to periodically and quickly refresh the content of mobile registry. A possible design can refer to event-based middleware proposed in [9]. In this case mobile registry must dynamically maintain information and is much more complex than conventional static registry.

Another possibility is that registry is on cloud participant. In this case a cloud participant acts as center storing information about all available participants as well as services of them. Since cloud participant is static and fixed, a register on it is reasonably akin to conventional registry. The difference is in binding (matchmaking) criteria due to the possible service type provided by mobile participants (e.g., mobile context-sensing services). In proposed RRSIA its registry belongs to this type and the criteria of binding (matchmaking) include location and functionality (capability). For example, there is a request to look for mobile participant at specified location having mobile context-sensing capability.

III. Real-time Road Speed Information Application

This section comprises two subsections. In first subsection we propose the requirements of real-time road speed information application (RRSIA), which is an instance of proposed model, to prove the possibility and capability of participant-oriented model being an application pattern framework. In second subsection, we illustrate an initiative prototype of RRSIA by screenshots, including a brief description about implementation detail.

A. Requirements

The main purpose of RRSIA is to collect, calculate, integrate, and supply real-time traffic speed information. Nowadays, benefit from web maps and their path planning services (e.g. Google map service), it is very easy to get precise map information and optimal path guidance. Nevertheless currently a mutual fatal drawback of these map services is that all of the information supplied from them is graphical static data, lacking an important data dimension. For example, a user looking at a static web map can easily understand the geographical distribution of roads (road network), but via the map it is all unlikely to know immediate road traffic situation (e.g., the average speed of vehicles running on a road).

On the other hand, there are uncountable vehicles on roads today and the number of them is still increased. These reasons cause unendurable traffic jams frequently in everywhere and anytime. Thus far static web maps are unable to provide real-time dynamic information, which probably are what the users of web maps really want to know in today (e.g., road speed and traffic jam reporting). Proposed RRSIA tries to enhance and improve this mutual drawback of static web maps.

A way to gather dynamic road traffic speed data is to use camera catching real-time road video and then analyzing the moving speed of each object in the video. This manner is quite common today, but with this manner speed probes are restricted and limited to only at a few locations, such as some representative and critical positions along a freeway. In RRSIA, the main goal is to use mobile participants as speed probe (i.e., mobile participant provides context-sensing or speed-sensing service). In this way, the number of speed probes is enough to gather massive and complete real-time road traffic speed data and construct dynamic web map. The idea about RRSIA's architecture and components is illustrated in Fig. 3. In RRSIA, the cardinality (multiplicity) between mobile participants and cloud participant is many to one. More specifically, the application has plural mobile participants and only one cloud participant. The relationship between mobile participants and cloud participant of RRSIA is that they profit and profit from each other simultaneously (i.e., mutualism). Each of mobile participants provides context-sensing service for cloud participant and requests real-time road traffic speed information service offered by cloud participant. On the other hand, cloud participant exploits mobile context-sensing services provided by mobile participants and offers real-time road traffic speed information service for mobile participants. Two types of participant rely on each other and they cannot without each other. Following is a primary scenario of RRSIA.

When a driver is driving on road, he/she opens RRSIA's mobile participant side application and becomes a mobile participant. He/she wants to know several candidate roads' real-time speed information in order to decide a most fluent and fastest path to reach his/her destination. Subsequently the driver inquires real-time road speed information service provided by cloud participant. At meantime, as the driver's mobile participant side application is running, cloud participant uses context-sensing service offered by mobile participants for real-time speed and GPS location. Cloud participant constantly consumes all available mobile context-sensing services. Collecting and integrating large amount of speed and GPS location data acquired from context-sensing services are inevitable basis of road speed information service and dynamic map service provided by cloud participant.

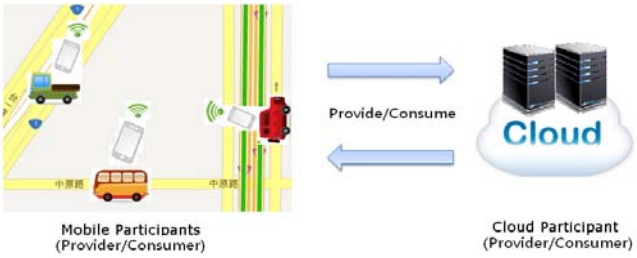

Fig. 3. Real-time road speed information application

B. Prototype

With the aid of application screenshots, in this subsection we illustrate a preliminary prototype of RRSIA. The implementation of the prototype is based on Web architecture, including service and user interface. Both the services provided and consumed by two types of participant are implemented via Restful Web Service (RWS) technology because it has been proved that, in mobile environment, RWS is doubtlessly more appropriate and efficient than conventional SOAP Web Service [4, 8, 10] . The prototype of mobile participant side has been implemented in forms of HTML 5 web application. Through HTML 5 API, it is quite simple to access Restful Web Service and instruct mobile device's sensors to collect contextual information. Otherwise, in terms of RRSIA user, by using Web browser to download and open web application written in HTML 5 a device immediately becomes a mobile participant without any

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

application installation and, in terms of application developer, reworks for re-developing the application for running on diverse mobile platforms.

Fig. 4. A mobile participant collects real-time context data

Fig. 4 is a screenshot of mobile device downloading RRSIA mobile participant side application and becoming a mobile participant. Instruct by HTML 5 code, the mobile participant provides context-sensing service, constantly gathering real-time traffic speed and GPS location data. And then the service is consumed by cloud participant. In Fig. 5, the holder of mobile participant tries to query an interested point of road for its current traffic speed. By touch the point, behind the scene this mobile participant sends a HTTP request including the point's GPS location to Restful service running on cloud participant (invoking real-time road speed information service). After the cloud participant receives the request, it calculates the speed of the point according to real-time data acquired by using context-sensing services nearing the point, and consequently sending the resulting speed (45.0 in this case) to the mobile participant as a result of the service invocation.

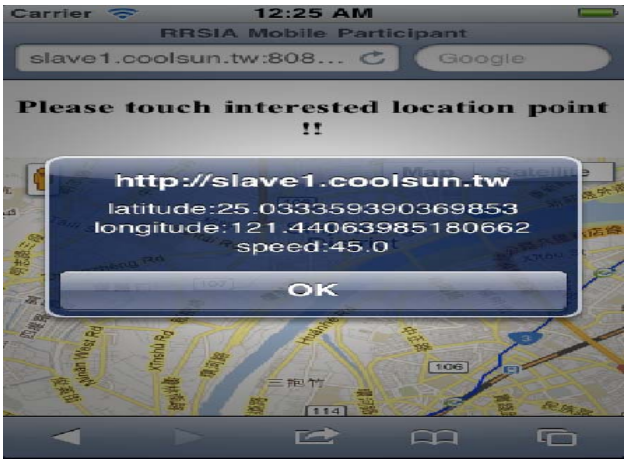

Fig. 5. Mobile participant queries real-time speed through RRISA

Fig. 6. A simple dynamic web map service

Fig. 6 is prototype screenshot of a simple dynamic web map displaying current real-time road speed information. With the real-time data received from mobile participants' context-sensing services and the sufficient computation power and storage space sustained by cloud participant, it is possible to enhance static web map as dynamic. A dynamic map service is very valuable to map users and path planning (guiding) researches such as [11]. In [11], when doing path planning, its planning algorithm considers traffic situation and road distance. In theory, such algorithm is better than traditional poor shortest distance planning algorithms. However, as the data used is rough statistical traffic data (time profile) rather than instant real-time information of roads, probably the path planed is not an optimal path due to imprecise statistical data. This fatal drawback can be improved by dynamic map service and its real-time traffic information.

IV. Dynamic Binding and Service Composition

Following the description of RRSIA, in this section we discuss RRSIA's service composition and dynamic binding mechanisms that are quite different from general concept of traditional SOA.

In most case, it is unlikely to satisfy a user requirement by a single service, thus normally there must be a service composition to create a value-added composite service for satisfying the requirement [12]. Conventionally, a composite service is an ordered set of services provided by different providers and there is a workflow of these services to indicate the execution process (process logic). Fig. 7 is a simple graphical presentation of a workflow-based composite service. In workflow-based composite service, each component service runs on a server (provider) and implicitly there is a workflow to specify how they are to be started. To traditional workflow-based composition, constructing a workflow by functionality matchmaking between services (e.g., comparison between a service's output and another service's input) is key point and composition basis. Nevertheless service composition within RRSIA is thoroughly distinct from traditional workflow-based composition. To know accurate real-time road speed of a specific location, it also is inappropriate and inaccurate to fulfill this requirement by only a single context-sensing service. Thus a composition is required to compose context-sensing services nearing the specified location to satisfy the requirement (integrating the information acquired from these context-sensing services). In such kind of service composition, it does not imply a workflow for generated composite service and the composing criterion for selecting proper services for composition is location of service (mobile participant) rather than functionality.

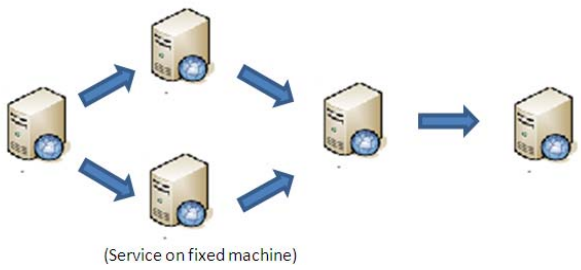

Fig. 7. A workflow-based composite service

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

In tradition, dynamic binding means that, during runtime, a service instance (could be a single service instance or a composite service instance) can be replaced by another instance having identical functionality because, for example, the original instance is failed to be executed or it violates quality restriction. In terms of a service user, it is to invoke a service interface instead of service instance and, during runtime, there is certain mechanism to bind service interface and appropriate service instance. In convention, the standard to judge that a service instance is eligible or illegal to be bound to an interface is functionality (e.g., input, output, pre-condition, and effect of service). But in RRSIA, as a cloud participant is to request a context-sensing service interface of specific location, the criterion to select service instance for the interface is based on location. On the other hand, because each mobile participant providing context-sensing service could move constantly, it can act as service instance for numerous different sensing service interfaces at different locations. Binding between interface and instance is deconstructed and re-bound repeatedly and frequently when each mobile participant moves out and moves in continuously. Conclusively, location-based specialty and mobility cause these differences.

V. CONCLUSION AND FUTURE WORK

In this paper, we have proposed and introduced a new participant-oriented model to replace obsolete SOA models and a useful application that is based on the model. They both are inspired and motivated by emerging technology trends, trying to benefit from and comply with these trends. We also have discussed the distinctions between proposed model and obsolete SOA models as well as the reasons causing these distinctions. For future application development, we think that the proposed model is more flexible and better than out-of-date SOA models.

Our future work focuses on two parts, the design and implementation of architecture and algorithms for the cloud as well as mobile participant sides of RRSIA. To cloud side, although every cloud platform is an extremely powerful foundation for massive computation and storage, it still is needed to have an efficient mechanism to promptly store, structure, update, and query real-time big data produced from mobile side services. As to mobile participant side, we try to adopt intelligent software agent technology to do dynamically local mobile sensor services composition and to collaboratively and cooperatively judge and detect more precise and rich traffic information. Also, by this way the heavy load of cloud will be mitigated largely.

ACKNOWLEDGEMENT

This work is partially sponsored by the National Science Council (Taiwan) under grant NSC 101-2221-E-030-022 and by Fu Jen Catholic University (Taiwan) under grant FJU 410031044044.

REFERENCES

[1] A. Michlmayr, F. Rosenberg, P. Leitner, and S. Dustdar, "End-to-End Support for QoS-Aware Service Selection, Binding, and Mediation in VRESCo," *Services Computing, IEEE Transactions on*, vol. 3, pp. 193-205, 2010.

[2] D. Zhovtobryukh, "A Petri Net-based Approach for Automated Goal-Driven Web Service Composition," *Simulation*, vol. 83, pp. 33-63, 2007.

[3] W. Qian and R. Deters, "SOA's Last Mile-Connecting Smartphones to the Service Cloud," in *Cloud Computing, 2009. CLOUD '09. IEEE International Conference on*, 2009, pp. 80-87.

[4] A. Ennai and S. Bose, "MobileSOA: A Service Oriented Web 2.0 Framework for Context-Aware, Lightweight and Flexible Mobile Applications," in *Enterprise Distributed Object Computing Conference Workshops, 2008 12th*, 2008, pp. 345-352.

[5] R. G. Daniel Schall, Christoph Dorn, and Schahram Dustdar, "Human Interactions in Dynamic Environments through Mobile Web Services," presented at the IEEE International Conference on Web Services (ICWS 2007), , 2007.

[6] W. Jianping, "Exploiting Mobility Prediction for Dependable Service Composition in Wireless Mobile Ad Hoc Networks," *Services Computing, IEEE Transactions on*, vol. 4, pp. 44-55, 2011.

[7] R. Tergujeff, J. Haajanen, J. Leppanen, and S. Toivonen, "Mobile SOA: Service Orientation on Lightweight Mobile Devices," in *Web Services, 2007. ICWS 2007. IEEE International Conference on*, 2007, pp. 1224-1225.

[8] R. Mizouni, M. A. Serhani, R. Dssouli, A. Benharref, and I. Taleb, "On the Performance of Hosting Web Services on Mobile Devices," in *Services Computing (SCC), 2011 IEEE International Conference on*, 2011, pp. 763-764.

[9] R. Meier and V. Cahill, "On Event-Based Middleware for Location-Aware Mobile Applications," *Software Engineering, IEEE Transactions on*, vol. 36, pp. 409-430, 2010.

[10] F. AlShahwan and K. Moessner, "Providing SOAP Web Services and RESTful Web Services from Mobile Hosts," in *Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on*, 2010, pp. 174-179.

[11] K. Xuan, D. Taniar, M. Safar, and B. Srinivasan, "Time constrained range search queries over moving objects in road networks," presented at the Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia, Paris, France, 2010.

[12] S. Yang, F. Yong-Yi, K. Jong-Yih, and M. Shang-Pin, "Towards a Genetic Algorithm Approach to Automating Workflow Composition for Web Services with Transactional and QoS-Awareness," in *2011 IEEE World Congress on Services*, Washington, DC USA 2011, pp. 295-302.