International Journal on Digital Learning Technology
2009, Volume 1, Number 3, 245-265

Situational Mashups for Ubiquitous Learning

Angus F.M. Huang
Department of Computer Science and Information Engineering, National Central University

*Stephen J.H. Yang
Department of Computer Science and Information Engineering, National Central University

Abstract

Mashups is an emerging technology on the Web 2.0. Users can propose favorite preferences and compose various widgets in a mashboard. The mashups selection can be performed dynamically identifying the situational set of widgets available at runtime. Students can mash-up the required learning services in ubiquitous learning environment. In this context, there are dynamic changes in user situations, defined as the configuration of accessible widgets depending on user location and schedule. The development of Situational Mashups requires mechanisms to interpret user context and to mash-up optimal widgets. We proposed a system including context representation, situation reasoning, workflow planning, and mashups optimization, called SituMash, capable of supporting the automatic composition of widgets in response to situation changes without user manual selection. Students can interact with the learning services seamlessly. Experimental results demonstrated that the system possess effective capabilities in composition-time, ease-of-use and usefulness.

Keywords: End-user Programming, Mashups, Situation Awareness, Ubiquitous Learning, Web 2.0.

1.Introduction

Situational Computing is an advanced concept in assisting people better proceed with daily work and problem solving. Situations use a series of participant, activity, time, location, equipment to express user's specific state. The situational computing evaluates one user's state and provides him or her with the necessary services. There are several researches devoted to the study of situation detection so far. The core technology is Context Awareness (Dey, Salber and Abowd, 2001). Context awareness is used to detect, represent, and inference the state and behavior of users. It has focused various studies up to now be the academic and industrial applications for human-computer interaction environment. Context awareness supports several applied sciences, such as health-care system (Munoz, 2003), recommendation system (McDonald, 2003), sensor network (Russell, 2004), smart environment (Ballagas et al, 2006), (Wohltorf, Cissee and Rieger, 2005), (Kalle, 2006).

Building situational system is a complex, dynamic and iterative process, especially for a busy businessman or a mobile user facing various activities in a running trigger of events. Manifold context awareness mechanisms are achieved for users. Two complete system development tool kit for building context-aware applications (Salber, Dey, and Abowd, 1999 & Loke, 2006) and the mechanism of analyzing the uncertainty of context information (Ranganathan, Al-Muhtadi, and Campbell, 2004) had been proposed. However, these studies do not provide support for end user programming which are autonomous to select services individually. With the development of software engineering, ubiquitous computing and Web 2.0, supporting end user programming becomes more desirable. The McKinsey Quarterly (McKinsey, 2007) depicted that the most frequently cited investment of enterprise executives is Web services. In addition to the silver bullet of heterogeneous systems integration solution, Web services, the Web 2.0 initiate the trend of civilian programming environment. The rising popularity of user-driven web applications, including Blogs, Wikis, Mashups, RSS, Podcast, Social Networks, P2P networking and Communities, has reflected the aspired initiative of folks. During the Web 2.0 era, these enterprises exploit the relevant technologies not only for sharing daily thought, but also for communicating with business partners, improving customer service, integrating suppliers, and managing knowledge internally. The most important, mashups provide a universal solution for the users with non-IT background to program specific applications. Mashups has two paradigms, one is mash-up with widgets, and another is mash-up with Web services. Paradigm of widgets can operate with a pure program and can operate

with calling other Web services, such as iGoogle (http://www.google.com/ig), MyYahoo (http://my.yahoo.com), Hinet Xuite (http://www.xuite.net), and Windows Live (http://www.live.com) as Figure 1, platforms and portals that can support service selection by users have been developed. And the paradigm of Web services is pure the way of Web services APIs, such as Marmite (Wong, 2007), Yahoo! Pipes (http://pipes.yahoo.com), Microsoft Popfly (http://www.popfly.ms). Although these tools all provide some kind of support for end user programming, none of them are fastened on context awareness and the design of them did not take into consideration supporting situational mashups. Automatic composing the necessary situational mashups is important, when the end user is roaming in a ubiquitous computing environment and executing different tasks, such as ubiquitous learning and real-time business. Jeffrey Wong had mentioned the importance of automatic generation of mashups in (Wong, 2007).

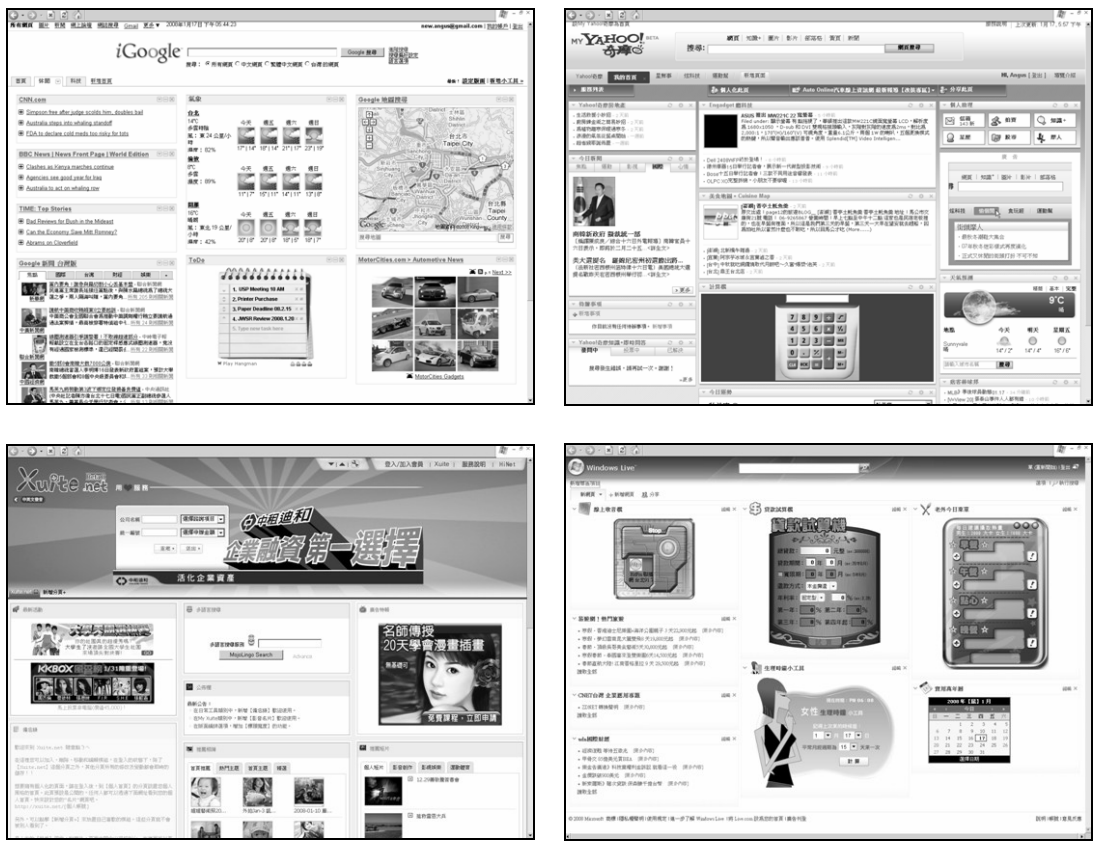


Figure 1 Four popular mashups platforms in Internet, they are iGoogel, MyYahoo, Hinet Xuite, and Windows Live. There are various services and tools supported by these environments, containing sport news, politics news, health-care tool, finance tool, fortune-telling tool, office tool, etc.

Situational mashups has been addressed in other studies. For example, QEDWiki is a wiki-based mashups software published by IBM. QEDWiki is a wiki-based mashups software published by IBM (QEDWiki 2007). However, to the best of our knowledge there are currently no environments that are designed specifically to support context awareness and automatic composition in situational mashups. In order to better achieve context awareness and facilitate situational mashups we first developed a context-aware SOA-based mashups system based on research into mashups in Web 2.0, context awareness, end user programming and our vision of situational mashups activities using this system. Then we built an experiment in order to observe how the system dynamically composes mashups and investigate how the end user gets the benefit from this system. Based on the feedback of the experiment within a campus activity we improved the system further.

Through the experiences of system building and experiment examining we summarize some system evaluation and insight discussion. In this study our main concern was how context-aware mechanism supports automatic situational mashups and how this mechanism can best be improved. To formularize our study, we declare our research questions as follows,

1. how should a context awareness mechanism be developed to facilitate situational mashups;
2. how does such an situational mashups support end user programming?

We organize the primary characteristics of SituMash are:

1. Situation awareness – it can interpret user's context factors and reason users' specific situation,
2. Automatic composition – it can compose user's necessary mashups with its planning,
3. Service orientation – it use service as the software basis that users are need not to build the kernel components,
4. End-user autonomy – it allows users to set the configuration of mashups with their favorites or habit,
5. Single platform – it propose a universal platform to avoid the conflicting of heterogeneous systems and to provide an integrated using experience.

Finally, the implementation of the SityMash system is discussed in more detail, followed by results derived from applications and evaluation which the situational mashups has been shown to perform successfully.

This paper is organized as follows. We commence the related research areas about mashups in Web 2.0, context awareness and end user programming. Then we express our design of this environment, evaluation of the system and discussion of the experiment. Finally, we conclude the paper and suggest some future works.

2.Related Research Areas

2.1 End-user Programming

Who are "end-user" programmers? The border between end-user and programming-user is indistinct. Warren Harrison considered that end-users are individuals who have taught themselves to program in some programming languages, such as C, PHP and Visual Basic (Harrison, 2004). Nevertheless, for the more user-friendly concerns, we think that the end-user even need not have programming skills. We expect that end-users can get aspired services between drag and drop.

End-user programming has various advantages. Capers Jones organized them as that schedules are shorter for small applications, many useful business software features start as end-user prototypes, financial end-user spreadsheet application are common and often useful, end-user development is satisfying and even enjoyable to those who can do it, end-user development augmented by packages serve many small companies, and end-user software can be built anywhere – at home or even on an airplane in (Jones, 1995). The end-user programming will be desirable for personal applications where no commercial or existing product meets a user's requirement. The disadvantages and potential dangers illustrated by Capers Jones (Jones, 1995) can be reduced through the quality control of widget providers and Web services providers in the mashups environment.

2.2 Mashups in Web 2.0

To make a comprehensive survey of Web 1.0 applications, they are mainly database driven. Developers build application logic to manipulate some databases. From the original database schema, web surfers can obtain rich information. However, these applications are centralized and rely on their own relational database, limiting the possibilities for data integration (Oren et al, 2007). Mashups is a rapid growing Web development paradigm and rich use experiences to users. The mechanism can assist in combining different Web applications through supported technologies, such as Ajax, SOAP, REST, screen scraping, Semantic Web, RDF in a succinct program development way.

Duane Merrill gave a brief survey of the prominent mashup genres in (Merrill, 2006). These are mapping mashups, video and photo mashups, searching and shopping mashups and news mashups at present. For example, one of the big catalysts for the appearance of mashups was

Google's project, the Google Maps API. Another popular application, news mashups, has been so for quite some time. Syndication feed mashups (Lindahl & Blount, 2003) can aggregate a user's feeds and present them over the Web, creating a personalized newspaper that plays up to user's interests or domain.

MashMaker assists non-expert users in creating their own mashups easily based on data and queries proposed by other users and by remote sites (Ennals & Garofalakis, 2007). They encourage users to find information by exploring, rather than by writing queries. Users can share data, widgets, and widget suggestion collaboratively especially. Although they applied the social network to improve the mashups selection, users have to explore those candidates by themselves eventually.

2.3 Context Awareness

Microsoft Research proposed a new class of applications that relies on real-time sensor data and its mash-up with the geocentric Web to provide instantaneous environmental visibility and timely decision supports (Nath, Liu, and Zhao, 2007). They challenged the data publishing, scalable data management, data visualization and sensor discovery issues to make users can take advantage of the portal and tools to make queries over live data sources. In order to extend the potential of SensorMap, they also provide the mashup APIs. Users can mash up the online live data sources with other applications. It is powerful. Nevertheless, it is not easy for end-users to compose these applications. A context-interactive application (Tan, Liu, and Chang, 2007), based on radio frequency identification (RFID), Internet, ubiquitous computing, and embedded system, has been developed. This system provides a mobile-based interactive learning environment (MOBILE) server for teachers and mobile-tools (m-Tools) for students. Teachers can achieve their outdoor teaching pedagogy effectively by providing the context-aware expert guidance and outdoor learning tools. Students can observe outdoor conditions and materials attentively; further, get the useful knowledge with these technologies.

3.Design and Implementation

3.1 Situational Mashups Environment

The SituMash system supports the situational mashups composition according to the user context. The logical abstraction is that select ubiquitous widgets which are suitable for user's

situation when they move to a specific location or play a specific role.

The situational mashups environment is illustrated as Figure 2. End-users will handle some computing device and roam at some physical environment. There are many Web services providing various functionalities provided by service providers. These Web services also can be invoked by the widgets; therefore, the more convenient software paradigm is achieved. End-users can arrange the mashups for their requirement. In order to achieve the more powerful applications and reduce the interruption to users, we will build the SituMash system plugging in the mash hub. The SituMash system support facilities compose widgets according to user's situation automatically. Users can move to different locations and thereby play the specific roles. The specific context constraints can lead to users in a specific situation and the situation will trigger the system to recompose the necessary widgets for users.

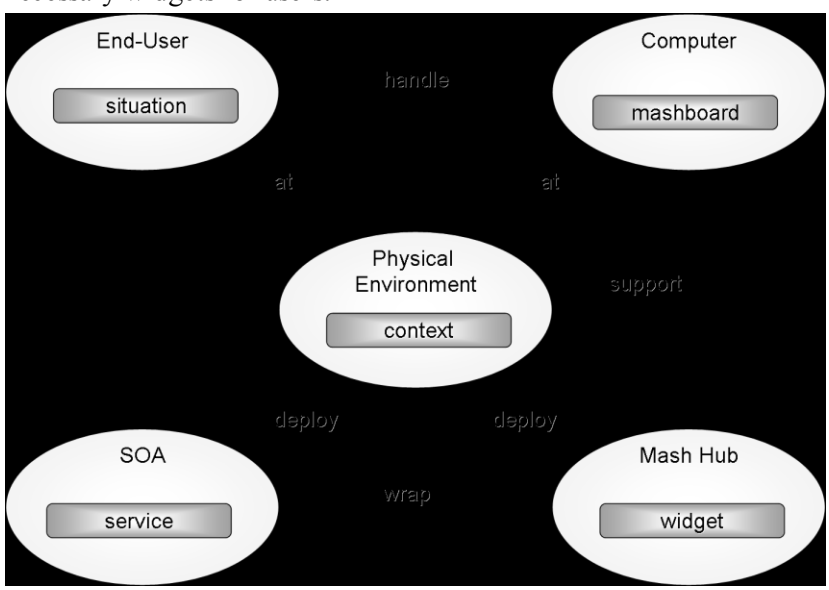
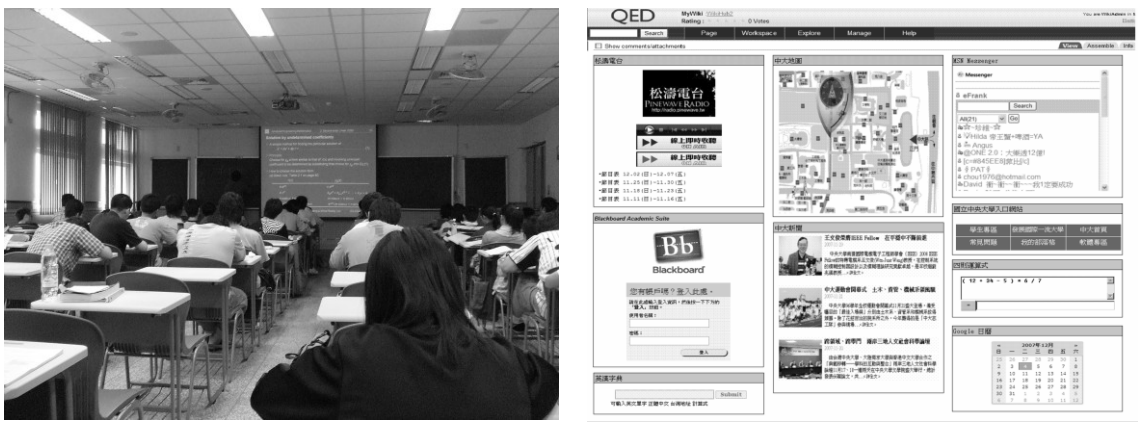


Figure 2 Situational mashups environment. There are four spirit roles in such environment, end-user, computer, SOA, mash hub and physical environment.

For the illustration of the situational mashups scenario, we consider the case of a campus with various activities, e.g. course, conference, meeting, experiment, and entertainment. RFID equipments provide connectivity. In detail, each building or field has the RFID Reader that provides infrared ray communication within the reasonable area. Student PDA, NB, or UMPC are equipped with RFID Tag and running the personal-assisting mashups application. One campus server hosts all the widgets, mash hub, and mashboard. Figure 3 illustrates the situational mashups system. Students can use the platform to execute specific widgets for corresponding situation ubiquitously and

dynamically. The SituMash system will suggest the suitable mashups composition and students can modify them if need. In Figure 3 (a), when the system aware the student is in school, it will infer his situation, go to class, and then compose the corresponding widgets such as campus radio station, Blackboard learning system, dictionary, campus map, campus news, etc. In Figure 3 (b), when the system aware the student is in conference, it will compose the corresponding widgets such as conference agenda, presentation video stream, presentation slides, conference map, personal schedule, conference peer instant message, etc.

(a)



(b)

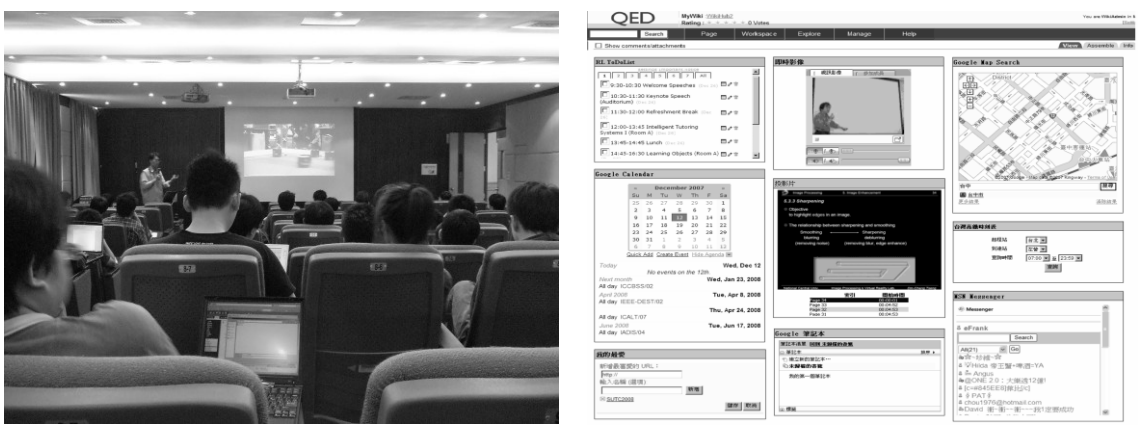


Figure 3 (a) School Situation: when aware student is in School, (b) Conference Situation: when aware student is in Conference.

4.Identifying Requirements

In order to provide users with the situational services when they roam in different situations we designated the following basic concerns that should be provided by the SituMash system:

- When to detect user's context?
- How to reason user's situation?
- How to determine the necessary widgets for specific situation?
- How to select the optimal widgets from numerous candidate widgets?
- How to compose the selected widgets dynamically?

The situation reasoning includes many inference rules. Considering the complexity of these rules, the users may not be aware of them when the system reason users' situations. By some means or other, the SituMash system should infer the specific situational widgets by users' context model quietly. One approach could be to design the system to determine the necessary widgets autonomously and compose them to users automatically. The disadvantage of this approach is that users will not agree to them completely and have there own opinions. Another approach could be to allow them to modify the partial widgets even whole. We anticipate that the situation reasoning mechanism will reason users' situation appropriately and the service inference mechanism will infer the necessary widgets suitable for the situation.

For more maturity, we suggested the SituMash should take some issues into consideration:

- Context Modeling, to build a knowledge model to represent user's context factors;
- Situation Reasoning, to reason user's situation through the context model with situation inference rules;
- Service Inference, to infer the situational widgets through user's situation with service inference rules;
- Mashups Optimization, to compose the final widgets in an optimal way for the minimal cost and maximal benefit;

5.Design and Development

We explain the system architecture as a data flow diagram in the Figure 4. This SituMash system has three main components, i.e. situation reasoning, tasks planning, mashups optimization.

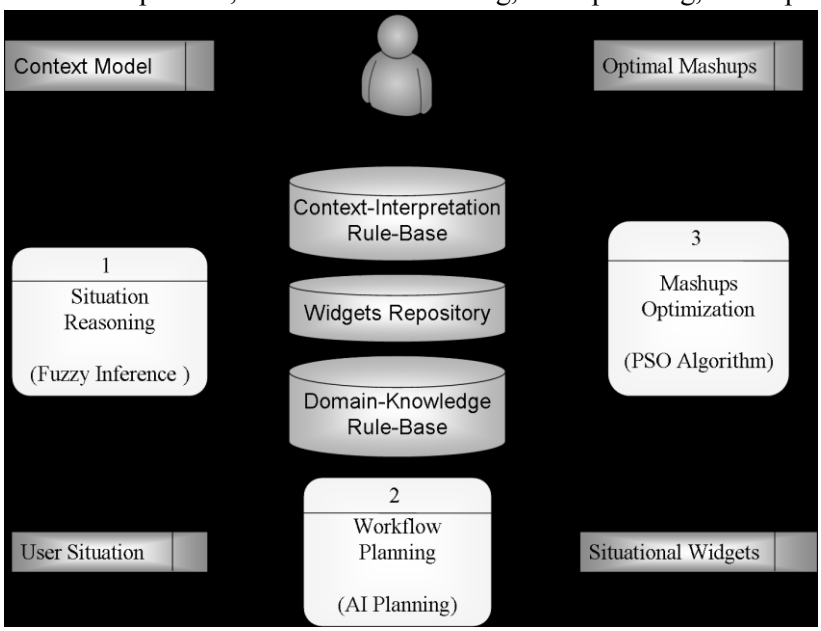


Figure 4 DFD of SituMash System Architecture

5.1 Context Representation.

In order to provide situational mashups in designing support to use in a ubiquitous computing environment we go deep into the study of context representation. A common definition of context is that Context is a five tuple {people, activity, time, location, object}. However we argued that the definition is too vague and widespread. The characteristics of such model may let it be driven to futility. Therefore, we refine the context definition as that, Context Factor {Social, Event, Schedule, Location, Equipment}, to reflect the usage of mobile and ubiquitous computing environment. This model will be clearer and without a hitch to represent users' context. And it will make the situation reasoning mechanism go smoothly in reasoning users' situation from context factor. The social factor means the people interacting with users; it will assist in reasoning users' situation through the interactive people's professional title. The event factor means the activity executed by user. The schedule means the users' personal schedule; it will assist in understanding users' assignments in

specific duration. The location factor is a popular context factor in this research area; it can indicate users' working state through corresponding location information. The equipment factor means the devices surrounding the users; it may limit and shape the usage of information acquisition.

5.2 Situation Reasoning.

Some improved researches about service provision can take user profiles to promote the satisfaction of use experience. Nevertheless, these concerns were still stiff. They had not considered the changing and dynamic user situation. They can not infer and predict user's specification situation, therefore they can not propose the applicable services to users. When using context awareness, the goal of the SituMash system is to create a seamless, intelligent Web experience for the end-user by composing necessary widgets with the workflow planning rather than select the widgets manually.

This study based on the rule-based system, JESS, to construct a situation reasoning mechanism. A context-interpretation rule base had been proposed. According to the knowledge of context interpretation, the system can predict users' situations such as to attend meeting, to attend class, to operate a vehicle, to join outdoor learning, to visit museum, and so on. Therefore, the SituMash system can use this situation information to plan the suitable and necessary tasks for users' goals. The automatic mechanism also can avoid too many users' manual inputs for added user requirement. It is an important using experience in ubiquitous environment.

5.3 Workflow Planning.

Situational-mashups systems for ubiquitous-computing environments are an emerging trend in service science. But now to the dramatic problem, the situations in ubiquitous-computing environments are changeable states. From the characteristics of user mobility, device variety, time validity, location dependence, users require special solutions that address with different mashups guidelines. The set of potentially available mashups widgets varies because of the above-mentioned characteristics, so it can not be forecasted and pre-planed. Situational Mashups should be designed to let users with minimally manual inputs of context achieve the situational tasks in automatic widgets composition.

We take the widgets configuration as workflow planning (Thompson, Li, and Xiao, 2007). The user request is as a set of inputs, user context is as constraint, and user's goal is as a set of desired outputs. The widgets are similarly treated as operators. Based on the foundation, we can use

planning skills, a special search that use local knowledge of action operators and constraints, to find a set of widgets that will satisfy the inputs of user request and outputs of desired goal. In the SituMash system, the Partial Order Planning (POP) algorithm was achieved automatically to find necessary widgets as Table 1.

Table 1 The POP algorithm for the necessary widgets planning

```
Function POP(request, goal, widgets) return mashups_plan
mashups_plan←MAKE-MINIMAL-PLAN(request, goal)
Loop do
        If SOLUTION?(mashups_plan) then return mashups_plan
        Wneed, t←SELECT-SUBGOAL(mashups_plan)
        CHOOSE-OPERTOR(mashups_plan, widgets, Wneed, t)
        RESOLVE-THREATS(mashups_plan)
End
---------------------------------------------------------------------------------------------------
Function SELECT-SUBGOAL(mashups_plan) return Wneed, t
  pick a mashups_plan step Wneed from STEPS(mashups_plan) with a precondition t
  that has not been achieved
return Wneed
---------------------------------------------------------------------------------------------------
Procedure CHOOSE-OPERSTOR(mashups_plan, widgets, Wadd, t)
choose a step Wadd from widgets or STEPS(mashups_plan) that has t as an effect
    if there is no such step then fail
    add the causal link Wadd→Wneed to LINKS(mashups_plan)
    add the ordering constraint Wadd < Wneed to ORDERINGS(mashups_plan)
   if Wadd is a newly added step from widgets then
     add Wadd to STEPS(mashups_plan)
    add Start < Wadd < Finish to ORDERINGS(mashups_plan)
---------------------------------------------------------------------------------------------------
Procedure Resolve-Threats(mashups_plan)
  for each Wadd that threatens a link Wi-> Wj in LINKS(mashups_plan) do
    choose either
        Promotion: Add Wthreat < Wi to ORDERINGS(mashups_plan)
        Demotion: Add Wj < Wthreat
    if not CONSITENT(mashups_plan) then fail
end
```

5.4 Mashups Optimization.

It was possible that there are many functional-matching widgets planned by the workflow planning mechanism in the widgets repository. Selecting one optimum combination from the tasks set is an important and difficult problem. Each widget has the own characteristics, such as cost, response time, reliability, etc. from software view, and device computing capability, memory, operating style, mobility, etc. from hardware view, and authorization, learning age, course dependency, etc. from learning methodology view, among software engineering and ubiquitous computing concerns. To optimize the widgets composition according to the limitation of software and hardware capabilities and users' requirements is the goal of the mashups optimization mechanism. This classical software requirement optimization also has been investigating for a long time.

There were various optimization solutions have been proposed, such as mathematics programming, genetic algorithm (Hopgood and Hirst, 2007), ant colony optimization (Bland, 2001), cultural algorithm (Coello & Becerra, 2004), and particle swarm optimization (Liang, Qin, Suganthan and Baskar, 2006). In the study area of search-based software engineering (SBSE) research area (Harman & Jones, 2001), there are many contributions for combinational optimization based on the meta-heuristic mechanisms. PSO algorithm is a meta-heuristic methodology which was designed from the natural observation of birds seeking the foods. This algorithm requires only original mathematical operators and profits form less computation memory and quick computing speed (Kennedy & Eberhart, 1995). The sharing of information among participants offers an evolutionary advantage. This insight was the essential of particle swarm optimization. This study based on the PSO algorithm to compute and customize the optimal mashups composition through the parameters and fitness function designated by users. Finally, end-users will obtain the situational and optimal mashups for their goals in the mashboard from the situation reasoning to workflow planning to mashups optimization.

6.Formative Evaluation

We made a series of experiments of SituMash to demonstrate the efficiency of this system and to explore some phenomenon of situational computing. We conduct this experiment with 320

students from two different characteristics user groups: users with programming skills, and users with no programming skills. Students were asked to perform two scenarios with two systems, QEDWiki (with no situational computing) and SituMash (with situational computing). The experimental designs are illustrated as the Table 2. Three experiments were conducted to estimate the composition time of final mashups completeness, ease of use, and usefulness. The User-P was substituted for the end-users with programming skills, and the User-noP was substituted for end-users with no programming skills. Among the treatments, SituMash represented the situational mashups, and the QEDWiki represented the traditional mashups.

We used the Technology Acceptance Model (TAM) (Davis, 1986) to measure how users come to accept the situational computing and use the SituMash. The ease-of-use means the degree to which the user expects that the use of the system is free of effort. The usefulness means the subjective probability that the use of a system increases a user's performance in a specific activity.

Table 2 Three experimental designs for the assessment of system efficiency.

Experiment	Group	Number of Students	Treatment	Assessment
E1	User-P	70	SituMash	Composition-Time, Ease-of-Use, Usefulness
	User-noP	70	SituMash	
E2	User-P	70	QEDWiki	
	User-noP	70	QEDWiki	
E3	User-noP	90	SituMash	
	User-noP	90	QEDWiki	

Test data were collected using a combination of qualitative and quantitative methods. During the overall activities we visited the participants to observe them using these mashups systems and collocated the questionnaires responded by the students. We find some interesting phenomenon. Students convinced that the SituMash system can effectively economize the configuration time, because they need not to select the widgets manually. And it can automatically arrange the suitable composition of mashups for the specific situation that can make them proceeding with the tasks seamlessly. Even they felt that the tasks planning mechanism can redeem the ignorance that what tools they should possess with.

Although the supply of software is rich, however, students informed that if the test can equip with other devices they would felt more convenient, such as PDA, smart phone, widget player, or thinner Tablet PC. It is worthy to think deeply, more students with programming background

preferred to modify the mashups configuration made by SituMash. Nevertheless, students with no programming background preferred to resign themselves to the SituMash's configuration. Therefore, they indicated that the using experience of pure QEDWiki system is inconvenient and troublesome. From this point, we can demonstrate that the situation reasoning and tasks planning mechanisms is much contributed to situational and pervasive computing domain.

7.Applicable Pedagogies and Applications

The situational mashups provides the foundation to mobile and ubiquitous learning environments. The appropriate learning contents and learning services can be acquired through various wireless, mobile, pervasive and ubiquitous devices. Such mobility materializes some learning methods and excites the learning activities in the ubiquitous learning environment. We suggest some applicable pedagogy with the situational mashups for mobile and ubiquitous learning paradigm. In these contexts, the content adaptation will achieve the pedagogies in this innovative environment and richen students' experience. In the other side, we suggest some applicable applications from SituMash system. In these activities, the situational mashups will improve the traditional computing and richen the user experience.

7.1 Applicable Pedagogy 1: Situational Mashups for the Silent Way

Caleb Gattegno (1972) presented The Silent Way for language teaching designed to train students to become independent, autonomous and responsible learners. This pedagogy advocates students to be experimental learners that will help them to construct the conceptual models profoundly. The learning materials usually associated with Silent Way are in fact a set of tools which allow teachers to apply Gattegno's pedagogical theory. The traditional learning materials are such as sound chart, color chart, Fidel, Cuisenaire rods and word charts, etc. are applied to carry out the learning activities. Nevertheless, the teacher will propose different situations and various digital learning contents for the students to respond to in the ubiquitous learning environment. Students need to operate the learning services and recognize the situations voluntarily. They must confront the diversification of dynamic learning activities.

One principle of this pedagogy is that the ideology development starts with the attention. However, the learning services with single and fixed version are not easy to display to students in

the diversified environment. There are various devices supporting for different learning activities. Students need to move the vertical and horizontal bar to acquire the desired information. These burdensome procedures will shift students' attention. Therefore, column-wise allows people to view content from top to bottom as far as possible by reducing the movement of a horizontal bar e.g. using PDA, smart phone or iPod to learn. This automatic mechanism of layout awareness will simplify the procedures from seeking the desired information and promote students to pay more attention on the necessary information.

7.2 Applicable Pedagogy 2: Situational Mashups for Suggestopedia

Georgi Lozanov (1971) proposed the teaching methods, Suggestopedia, based on the study of Suggestology. The purpose of Suggestopedia is to improve learning by reducing the affective filter of learners. Physical surroundings and atmosphere in classroom are the influential factors to make sure that the students feel comfortable and confident. The six important factors of this pedagogy are authority, infantization, double-planedness, intonation, rhythm, and concert pseudo-passiveness.

The double-planedness emphasizes that the personality of teacher, learning style, and even the decoration of classroom hardware, background music, and presentation of learning services will influence the effect of students' learning. Through the situational mashups, the original semantics of the learning flow meditated by teachers can retain in different layout. This contribution will cater for the requirements of double-planedness aspects. Further, the mechanism can achieve this pedagogy with copious physical surroundings in the mobile and ubiquitous learning environment.

7.3 Applicable Pedagogy 3: Situational Mashups for Situational Language Teaching

Some linguists of the British proposed the Situational Language Teaching between the 1930s to the 1960s. This pedagogy encouraged students to induce and understand the correct meaning and usage of learning objectives through the situation. Some principles of this teaching method are that learning is habit-formation, analogy is a better foundation for learning than analysis, and the meanings of learning objective can be learned only in a linguistic and cultural context. The procedures of situational learning move from controlled to freer practice of structures, or move from oral use of sentence patterns to their automatic use in speech, reading and writing. However, the diversification of supported equipments, learning contents, and learning activities will appear in various situations. Hence the situational mashups can make the learning procedures implement more seamlessly.

7.4 Applicable Applications 1: Intelligent Personal Learning Portal

In this scenario, the SituMash system may supply users with the time-aware or location-aware widgets by their school timetable or daily schedule. Toward the future convenient lifestyle, the intelligent ubiquitous computing will play an important role. Students can get the abundant supports according to their learning requirement anytime anywhere.

7.5 Applicable Applications 2: Interactive Academic Conference

In this scenario, the SituMash system actively detects users' workshops or sections, and composes the specific widgets in their own devices. Then, the situational contents will deliver to them to different activities, such as different slides for different sections, famous product introduction for coffee breaks, and travel guidance for conference surroundings.

7.6 Applicable Applications 3: Group-based Collaborative Learning

In this scenario, the SituMash system can base on the tasks and characteristics of each group to suggest the suitable widgets. From this situational assistance, students can go through and collaborate in proper sequence to complete the final goal.

8.Conclusion and Discussion

The appearance of human-computer interaction is changing gradually. Users need not be restrained from sitting front the desktop with the development of mobile computing and ubiquitous computing techniques. The interactions between human and computer can be more humanized because the computing units had fused into the physical environment. It means that everyone can benefit the convenience from computers anytime and anywhere in daily life. Every person plays a different role in different situation. Context constraints, such as partner, event, time, location, and equipment, situate people in a special identity. Lying in different situations will confront different tasks, and thereupon users will need some hybrid applications system to support them. The application of situational computing is referred to as a situational mashups. With the contribution of this study, situational mashups can be aware of user’s situation, thereby accommodating user’s activities in ubiquitous environment, as well as augmenting the capability of Web 2.0 for ubiquitous computing.

We find that the colleagues or participants around one user using the situational mashups will influence the widgets configuration. We conjecture that the social networks of users will inference their social-based learning activities, regardless of positively or negatively. Therefore, we will take more social network factors into situation reasoning concerns.

Following the growth of mashups, the more convenient composition mechanisms will appear and the more powerful platforms will be proposed. The usability and validity of mashups configuration will be an important problem. We will devote to the verification of mashups to make sure the end-users' artifacts are safe and usable.

Acknowledgement

This project was primarily funded by National Science Council, Taiwan under grants NSC96-2628-S008-008-MY3. The authors would also like to thank the anonymous reviewers for their constructive comments that were valuable for improving this study and revising the paper.

References

Ballagas, R., Borchers, J., Rohs, M., Sheridan, J.G. (2006). The smart phone: a ubiquitous input device. *IEEE Pervasive Computing*, 5:1, 70 – 77.

Bland, J.A. (2001). Optimal structural design by ant colony optimization, *Engineering Optimization*, 33:4, 425 – 443

Coello, C.A. and Becerra, R.L. (2004). Efficient evolutionary optimization through the use of a cultural algorithm. *Engineering Optimization*. 36:2, 219 – 236.

Davis, F.D., Bagozzi, R.P., and Warshaw, P.R. (1989). User acceptance of computer technology: A comparison of two theoretical models. *Management Science*, 35, 982 – 1003.

Dey, A.K., Salber, D. and Abowd, G.D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16:2, 97-166.

Ennals, R.J., and Garofalakis, M.N. (2007). MashMaker: mashups for the masses. *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. 1116 – 1118.

Gattegno, C. (1972). Teaching Foreign Languages in Schools – The Silent Way. New York: Educational Solutions Inc. (ISBN 0878250468).

Harman, M. and Jones, B.F. (2001). Search-based software engineering. *Information and Software Technology*, 43, 833 – 839.

Harrison, W. (2004). The dangers of end-user programming. *IEEE Software*, 21:4, 5 – 7.

Hopgood, A.A. and Hirst, A.J. (2007). Keeping a distance-education course current through eLearning and contextual assessment. *IEEE Transactions on Education*, 50:1, 85 – 96.

Jones, C. (1995). End-user programming, *IEEE Computer*, 28, 68 – 70.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. *IEEE International Conference on Neural Networks*, 4:27, 1942 – 1948.

Liang, J.J., Qin, A.K., Suganthan, P.N. and Baskar, S. (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Transactions on Evolutionary Computation, 10:3, 281-295.

Lindahl, C., Blount, E. (2003). Weblogs: simplifying web publishing. *IEEE Computer*, 36, 114 – 116.

Loke, S.W. (2006). Context-aware artifacts: two development approaches. *IEEE Pervasive Computing*, 5:2, 48 – 53.

Lozanov, G. (1971). Suggestology and Outlines of Suggestopedy, New York; Gordon & Breach 1978 (Translation of: Nauka I Iskustive, Sofia 1971).

McDonald, D.W. (2003). Ubiquitous recommendation systems. *IEEE Computer*, 36:10, 111 – 112.

McKinsey Quarterly, "How Businesses Are Using Web 2.0: A McKinsey Global Survey", June 2007, http://www.mckinseyquarterly.com/links/26068

Merrill, D. (2006). Mashups: The New Breed of Web App. <ibm.com/developerworks/library/x-mashups.html>

Munoz, M.A., Rodriguez, M., Favela, J., Martinez-Garcia, A.I., Gonzalez, V.M. (2003). Context-aware mobile communication in hospitals. *IEEE Computer*, 36:9, 38 – 46.

Nath, S., Liu, J., and Zhao, F. (2007). SensorMap for wide-area sensor webs. *IEEE Computer*, 40, 90 – 73.

Oren, E., Haller,A., Hauswirth,M., Heitmann,B., Decker, S., and Mesnage, C. (2007). A flexible integration framework for semantic Web 2.0 applications. *IEEE Software*, 24, 64 – 71.

QEDWiki (2007). http://services.alphaworks.ibm.com/qedwiki/

Ranganathan, A., Al-Muhtadi, J. and Campbell, R.H. (2004). Reasoning about uncertain contexts in pervasive computing environments. *IEEE Pervasive Computing*, 3:2, 62 – 70. [Russell 2003] Russell, D.M. (2003). UbiComp 2003: Sensors in Seattle. *IEEE Pervasive Computing*, 3:2, 76 – 80.

Salber, D., Dey, A.K. and Abowd, G.D. (1999). The context toolkit: aiding the development of context-enabled applications. *In Proceedings of ACM CHI'99*, 434 – 441.

Tan, T.H., Liu, T.Y., and Chang, C.C. (2007). Development and evaluation of an RFID-based ubiquitous learning environment for outdoor learning. *Interactive Learning Environment*, 15:3, 253 – 269.

Thompson, C.W., Li, W.N., and Xiao, Z. (2007). Workflow planning on a grid. IEEE Internet Computing, 74 – 77.

Wohltorf, J., Cissee, R. and Rieger, A. (2005) BerlinTainment: an agent-based context-aware entertainment planning system. *IEEE Communications Magazine*, 43:6, 102 – 109.

Wong, J. (2007). Marmite: towards end-user programming for the Web. IEEE Symposium on Visual Languages and Human-Centric Computingm, 23:27, 270 – 271.

Authors

黃福銘，中央大學資訊工程學系，博士生

Angus F.M. Huang is a candidate for doctor's degree of the Department of Computer Science and Information Engineering, National Central University, Tao-yuan, Taiwan.

E-mail: fmhuang@csie.ncu.edu.tw

楊鎮華，中央大學資訊工程學系，教授

Stephen J.H. Yang is a professor of the Department of Computer Science and Information Engineering, National Central University, Tao-yuan, Taiwan.

E-mail: jhyang@csie.ncu.edu.tw

收稿日期：2009.04.15

接受日期：2009.05.15

數位學習科技期刊
民國 98 年，第一卷第三期，245-265

利用情境混搭服務進行無所不在的學習

黃福銘
中央大學
資訊工程學系

*楊鎮華
中央大學
資訊工程學系

摘　要

混搭服務機制是網路 2.0 中新興的技術，使用者可以在混搭儀表板中部署自己喜愛的小工具。學生可以在無所不在的學習環境中組合所需的學習服務，其中，混搭服務的選用通常會配合使用情境的動態變化而有所不同。當學生的情境隨著位置或行事曆的內容而改變時，學習服務勢必也需要動態的改變以因應情境的變化。本研究提出了一個系統(SituMash)，整合了情境描述、情境推論、工作流程規劃與混搭服務最佳化等機制來達到自動組合混搭服務的成效，以減少人工編排的負擔並實現無間斷的使用流程經驗。實驗結果驗證本研究提出的系統能有效的降低服務組合的時間並達到容易使用的目的。

關鍵詞：終端使用者程式規畫、混搭、情境感知、無所不在的學習、網路 2.0