Conflict Resolution Model Based on Weight in Situation Aware Collaboration System

Heeseo Chae, TaeYon Kim, Dong-hyun Lee and Hoh Peter In
Department of Computer Science & Engineering, Korea University,
1, 5-ka, Anam-dong, Sungbuk-ku, Seoul, 136-701, Korea
{royalhs, saytykim, tellmehenry, hoh_in}@korea.ac.kr

Abstract

In ubiquitous environments, situation-awareness is a desirable property to support adaptive and personalized services. To describe situation-awareness effectively, a situation-aware interface description language (SA-IDL) which can express a collaboration system with various contexts and actions is researched. However, conflicts can occur when services expressed in SA-IDL collaborate with each other, though each service does not make any errors. This is because each service is described with requirements for each one. So, in order to provide situation-aware services without inconsistency, conflicts among services should be identified and resolved. In this paper, we propose a mathematical modeling of SA-IDL to identify the conflicts among situation-aware services, and suggest conflict resolution graph model to identify the services which occur the conflicts. Also we proposed the resolution algorithm based on weight.

1. Introduction

Ubiquitous computing aims to offer customized and adaptive services for user convenience by integrating and reconfiguring legacy systems according to user situations [1]. At the legacy system, it is annoying to use various devices at the same time because each device has individual interface and does not have inter-operability. But, at the ubiquitous environment, complex operations for user situations can be possible because devices have interoperability. So, adaptive computing environment are possible by arranging the way how to operate collaboratively with each devices for user's situations. Thus the concept of collaboration means providing the service between heterogeneous devices and the situation-aware approach [2] in this collaborative relation is called a situation-aware collaboration system. In order to implement the collaboration system and solve the conflict during execution of the system, the formalized description method that describe the heterogeneous device consistently and capture the conflict between the relations, is essential. The used language in this logical connection is called a Situation-Aware Interface Description Language (SA-IDL) [4]. SA-IDL can describe collaboration of each device and make situation-aware service possible by checking multi context histories and executed actions. SA-IDL can also specify execution time of actions, so this language has an advantage of catching exact situations. Because of this advantage it is used to describe and implement the collaboration system formally. Then now the conflict is redefined by SA-IDL, it can be founded at the case that different executions are operated in the same devices at the same time. To examine these conflicts, not only one service but also other services should be examined at the same time because conflicting commands between services can be executed at the same situation-sensing time even if there are no conflicts in one service.

In this paper, we will define and find the conflict using SA-IDL transformed to a mathematical model and also solve the conflict using tuple as a mathematical set notation, in the collaboration system implemented by SA-IDL. And in order to solve the conflict driven by the formal method, we import the approach of graph theory and express the collaboration system which is composed of nodes and edges (node means service/situation and edge means conflict). Then, the conflict resolution algorithm based on weight is suggested to form the resolution method.

2. Conflict in Collaboration System

2.1. SA-IDL Model

In order to suggest the development method which can describe the situation and make the situation-aware service, SA-IDL is used. And the SA-IDL model such a tuple format is essential to justify the collaborative relation and conflict. There is a reason why the tuple is used, the tuple can accommodate the entire different data format which is transmitted from different

Proceedings of the 11th IEEE International Workshop
on Future Trends of Distributed Computing Systems (FTDCS'07)
0-7695-2810-4/07 $20.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

device[6,7]. Thus SA-IDL model has the component which are Context Tuple, Action Tuple and Derived Context, each component is as follows [2].

Action := (Time, Device, State, StateValue)
Context := (Timestamp, Device, State, StateValue)
A := {x | x is an Action}
C := {x | x is a Context}
DerivedContext := P(C) → {true, false}
Situation := (DerivedContext, P(C), A)

Table 1.SA-IDL model

In this model, the situation means the service which executes the connected command when it is satisfied in a specific condition. To make a model of SA-IDL, the following mathematical notations are used: tuple ( ), function →, set { } and

power set P( ). And each concept of the tuple is in the following.

• **Action tuple**: Action is represented as a tuple (t, device, a_1, a_2, …, a_n), where t is the time when the action is taken, and a_1, a_2, …, a_n are a set of attributes of the action [4]. Action is used in two manners: one is to change attribute values of devices, and the other is to give orders to devices. Ordering to devices can be also represented in changing attribute values of devices, so that Action is unified as changing attribute values. The attribute, denoted as a_n, is a tuple of (attribute, attribute value).

• **Context tuple**: Context tuple, (t, device, c_1, c_2, …, c_n), where t is the time stamp and c_1, c_2, …, c_n are values of the context attributes relevant to the application software and collected through sensors, device and application software directly, such as light intensity, noise level, system time, application invocation rates, etc [4].

• **DerivedContext**: Context tuples and action tuples are discrete samples of raw context and action data [4]. DerivedContext is a mathematical function that determines whether situation requirements are met. Its domain is a power set of context sets, which is mapped to true or false.

• **Situation tuple**: situation is represented by DerivedContext, Context power set and action tuple. This notation implies a situation where if conditions of derived context is satisfied triggers sequence of actions.

2.1. Definition of Conflicts

There is no direct definition about conflict in SA-IDL, so any other formal approach is needed to find it. This formal method is a expressed by a tuple style using SA-IDL model as a (C, action, action). C is a context set that triggers two different actions that are conflicted with each other. The definition of the conflict is denoted by using mathematical set notation.

Conflict := (C, Action, Action)
{ ∃ (c, x, y): Conflict; ∀ a, b: Situation |
c ∈ a.P(C) ∩ b.P(C)
and c ≠ Ø
and a.DerivedContext (c) = true
and b.DerivedContext (c) = true
and x ∈ a.A and y ∈ b.A
and x.Time = y.Time
and x.Device = y.Device
and x.State = y.State
and x.NewState ≠ y.NewState }

Table 2.Definition of Conflicts

There exists a conflict *(c, x, y)* for any two situation *a* and *b*

∃(c, x, y): Conflict; ∀a, b: Situation | *(Table 2)*

Following constraint defines the context set *c* that triggers conflicted action.

The intersection of *P(a.C)* and *P(b.C)* denotes a set of context sets that is shared between two situation a and b. The context set *c* should be an element of this intersected set. As well, it is an input element of the *DerivedContext* functions, and it satisfies both requirements of the two situations. Additionally, the *c* should not be an empty set.

and c ∈ a.P(C) ∩ b.P(C)
and c ≠ Ø *(Table 2)*
and a. DerivedContext (c) = true
and b. DerivedContext (c) = true

Following constraint defines two actions, *x* and *y*, that are conflicted with each other. *X* is an element of *a*'s action set, and *y* is of *b*'s. The two actions change the same state of the same device at the same time with different values.

and x ∈ a.A and y ∈ b.A
and x.Time = y.Time
and x.Device = y.Device *(Table 2)*
and x.State = y.State
and x.NewState ≠ y.NewState

3. Conflict Resolution

The conflict drawn by formalizations is obstacle to disturb the harmonious Collaboration system. In order to solve the conflict occurred in services which are described by SA-IDL in collaboration system,

Proceedings of the 11th IEEE International Workshop
on Future Trends of Distributed Computing Systems (FTDCS'07)
0-7695-2810-4/07 $20.00 © 2007 IEEE
IEEE COMPUTER SOCIETY

approach of graph theory is imported to display situation/service (nodes) and conflict (edge). This approach as graph expression offers an opportunity to make a generalization of the conflict problem through the exact description of conflict situation and make a solution using the abstraction of conflict problem. This clause presents the approach of conflict resolution model and resolution algorithms.

3.1. Conflicts Identification

The conflict expressed by mathematical modeling of SA-IDL can be represented as following graph description [5]. Each situation (the situation means the service which execute the connected command when it is satisfied in a specific condition) is expressed to each node and each conflict among situations is also expressed to each edge.

Below description definitely shows conflicts generated by the logical error among situations. The conflict relations which are occurred by compounding of various situations or are occurred between one situation and the other situation are also expressed in the conflict graph. The conflict identification as shown figure 1 is described in following process.

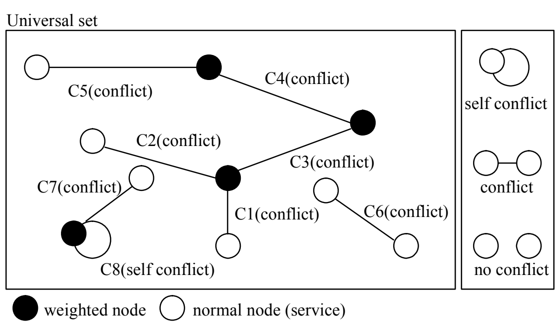


Fig1. Conflict Graph

- **Mapping a situation to a node**

The method of expressing the activating situation in main device is registering a new service as normal node and displaying the node in universal set which means a main frame.

- **Description the conflict among situations**

Logical conflicts which is against the SA-IDL grammar are detected by exhausted searching method above two services, and nodes having the conflicts are connected by edge which means the conflicts between the situation.

- **Distinction of nodes**

In order to discriminate the complexity of conflicts and solve the conflict problem, the node which can cause more than two conflicts in the node perimeter is called a weighted node. This node is distinguished with normal node which means no or one conflict and it assists to solve the conflict problem effectively.

The conflict identification helps the suggestion of resolution models and resolution algorithms theoretically to solve conflicts. Especially, the situation conflict model expressed by graph supplies the base of research which can find a solution applying graph theory, and it gives the way to generalize the conflict.

3.2. Conflict Resolution Model

According to probable situations, there are various ways how to find some conflicts and which solution is offered to solve the conflicts. Those probable situations are considered to remove the weighted node having heavy weight. As a previous stage of conflict resolution, it is exceedingly important to classify situations and to suggest the baseline and models in accordance with situations.

Through the figure 1, the conflict resolution model can be constructed and represented in respective solution approach.

- **Assumption**

In order to develop the conflict resolution model, some of assumptions are required as follows:

Assumption	Contents
Purpose	Reducing the minimum number of service that user wants.
	Providing a specified service to satisfy user's requirements.
Presentation	Node represents the service in accordance with situation.
	Edge represents the conflict in accordance with definition of conflicts (2.2)

Table 3.Assumption of Conflict Resolution Model

In order to gratify the purpose in assumption, the conflict model and the resolution based on weight are contrived. The validity of these things is discussed in 4.3

- **Weighted Node Model**

It is a model that solves the conflict problem using conflicts identification graph described by edges and nodes. It is an approach of this model to find a node which has the largest number of edges in the graph and the purpose of this approach is to remove the least number of nodes but the largest number of edges.

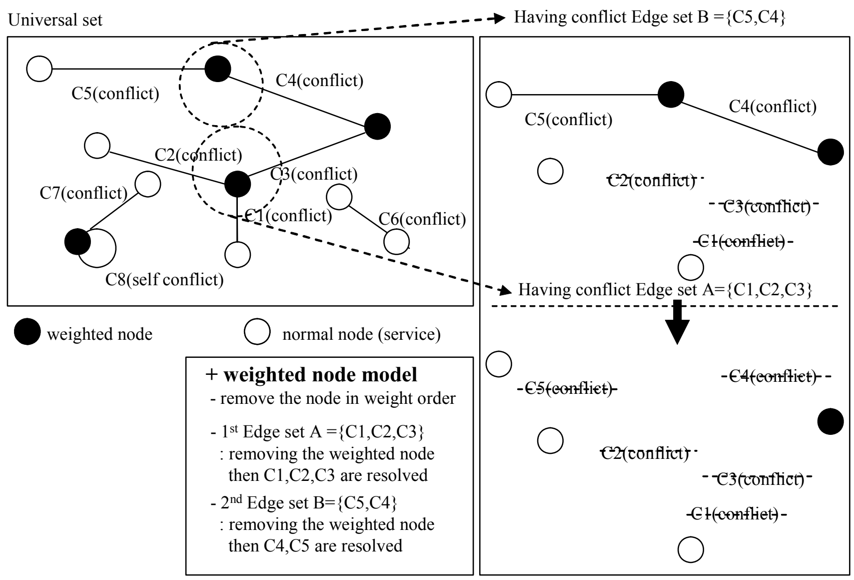


Fig2. Weighted Node Model

Distinct subset is formed center around the particular weighted node in universal set. The edges can be eliminated together according to elimination of the weighted node in this subset by applying the resolution algorithm which will be explained next chapter. And then the subset which has the next weight node will be applied the algorithm. In this way, nodes and edges are eliminated by turns.

3.3. Conflict Resolution Algorithm

The generalized conflict resolution can be suggested on the basis of relation between conflicts and situations modeled by graph expression. Fundamentally, correcting the conflict according to elimination of the node including edge is major approach in this algorithm.

- **Resolution Algorithm based on Weight**

The purpose of this algorithm is finding a node which largely connected with conflict expressed by edge and getting rid of edge by means of removing that node. Also, the core of this algorithm is comparing the summation of nodes' weights in their subset to others and eliminating the root node of smaller one. This approach can leave the largest number of nodes but the least number of edges as possible. The procedure of the Resolution Algorithm based on Weight is described by step as follows (fig.3):

Setp1. **The weight setting of a node**

The weight value of each node is determined by the number of connected edge. The node which is - connected with the largest number of edges is called weighted node to distinguish from other normal nodes.

Step2. **The constitution of weight subset**

Adjacent nodes which have one hop relation with weighted node are made up subset as a small unit center around the weighted node. Each subset can be contained more than one weighted node and the overlapping of nodes among subsets is permitted.

Step3. **The comparison of weight summation among the subsets**

The weight values of whole nodes in the subset are totalized. If the summation value of subset is the smallest one among subsets then the root node as the weighted node of the subset is eliminated. Using arbitrary subset A and B, it is simply presented as follows:

$$\sum weight_{subset\ A} > \sum weight_{subset\ B} \rightarrow B(subset) - R(R \in B, root) \qquad (Fig3)$$

Step4. Regenerate the weight value and the repetition of the comparison

Calculate again the changed value due to the node elimination in the previous step and repeat the step3 until all of the related edges are eliminated.

- **Proving the Resolution Algorithm based on Weight**

Basically, the more eliminating is progressed, the sooner reminding nodes are declined. Thus in order to leave the maximum number of nodes, the frequency of

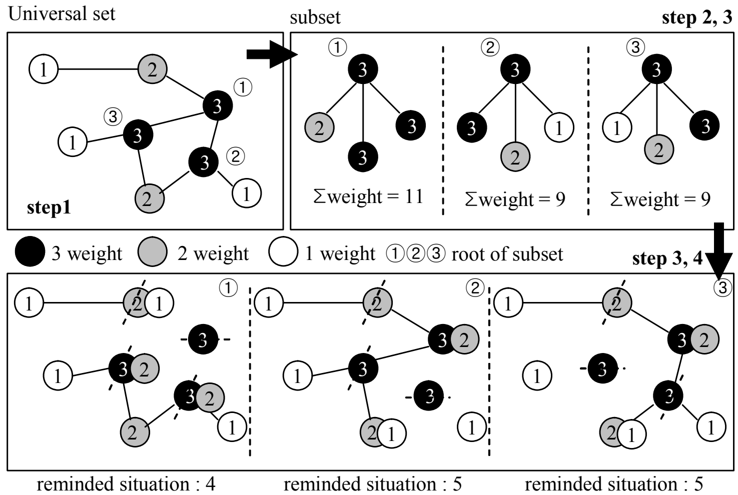


Fig3. Resolution Algorithm based on Weight

elimination has to be reduced. By the way, the fact which summation of node's weight is small means that the number of weight nodes is also small because of one hop relation. It can be proved by Chromatic Function Polynomial [5]. This significance comprises that depth of elimination becomes low state in case of small summation.

Let subset S be a simple graph, and let S-e and S/e be the graphs obtained from S by *deleting and contracting an edge e*. Then,

$$P_S(k) = P_{S\text{-}e}(k) - P_{S/e}(k) \qquad \text{(Theorem 1)}$$

Strictly speaking, however, this subset is tree in collaboration system.

So the theorem 1 is edited as follow:

$$P_S(k) = k(k-1)^{n-1} \qquad \text{(Theorem 2)}$$

This procedure terminates when no edges remain - in other words, when each remaining graph is null graph. Since the chromatic function of a null graph is a polynomial (=k^r, where r is the number of nodes), it follows by repeated application of (Theorem 2) that chromatic function of graph S must be sum of polynomials, and so must itself be a polynomial. Thus it is clear that

$$\text{if } k < \quad (s), \text{ then } P_S(k)=0, \text{ and that if } k \geq \quad (s), \text{ then } P_G(k) > 0 \qquad \text{(Theorem 3)}$$

The summation of weight value is quoted as $P_S(k)$ and the number of nodes is quoted as *(s)* ,and then the proposition is proved in case of one hop relation.

In conclusion, eliminating a subset that the summation of node's weight is small provides a large number of nodes which can remind in final.

This procedure satisfy the purpose of ubiquitous computing, which can supply various service amicably.

4. A Case-study: Situation-Aware Collaboration System

Situation-aware collaboration system among ubiquitous devices consists of each situation-aware system which is being developed by SA-IDL.

SA System	Description
SmartView[4]	It offers situation-aware service according to the scale of resources.
Light Manager	It controls the intensity of light by sensing entrance and exit.
EEM	It controls brightness in a room and audio volume when watching movies or video clips. [EEM: Entertainment environment manager]

Table 4. Situation- Aware Systems which compose the collaboration system

Each situation-aware system described with SA-IDL works well in a stand-alone operation, but can make conflicts in a collaborative operation with other systems. To develop the system, some of assumptions are required as follows:

Assumption	Contents
Relation among services	The relation among nodes is set up as one hop. This assumption mitigates the complexity of services.
	The number of conflict between one services and another is just one.
Adapting of the weight	The weight based model is displayed as not a human centered model but a device centered model outwardly. However it can explain user optimized

Proceedings of the 11th IEEE International Workshop
on Future Trends of Distributed Computing Systems (FTDCS'07)
0-7695-2810-4/07 $20.00 © 2007 IEEE

service and it will be described in 4.3

Table 5. Assumption of Experiment

This section gives a description of the definition of these conflicts in a collaboration system with mathematically modeled SA-IDL, and shows resolutions.

4.1. Experimental Setting

The detail operation of three systems and relations are described as follows:

```
Action a1 = (0, Light, Power, On)
Action a2 = (0, Light, Power, Off)
Context c1 = (t, Host, InHouse)
Context c2 = (t, Light, Power)
DerivedContext d1 = (c1.InHouse == True and
                     c2.Power  == Off)
DerivedContext d2 = (c1.InHouse == False and
                     c2.Power == On)
Situation LightOn = (d1, {c1, c2}, {a1})
Situation LightOff = (d2, {c1, c2}, {a2})
```

Situation aware system 1 : Light-Manger

```
Action a1 = (0, Light, Power, Off)
Action a2 = (0, Speaker, Power, On)
Context c1 = (t, TV, Play)
Context c2 = (t, Light, Power)
Context c3 = (t, Speaker, Power)
DerivedContext d1 = (c1.Play == On and
                     c2.Power == Off)
DerivedContext d2 = (c1.Play == On and
                     c3.Power == On)
Situation LightOff = (d1, {c1, c2}, {a1})
Situation SpeakerOn = (d2, {c1, c3}, {a2})
```

Situation aware system 2 : Entertainment Environment Manager

```
Action a1 = (0, TV.MediaURL = PDA.MediaURL)
Action a2 = (1, PDA.Play, Off)
Action a3 = (2, TV.Play, On)
Action a4 = (2, Light, Power, off)
Context c1 = (t, TV, MediaURL)
Context c3 = (t, PDA, MediaURL)
Context c4 = (t, PDA, Play)
Context c5 = (t, TV, Loc)
Context c6 = (t, PDA, Loc)
Context c7 = (t, Light, Power)
DerivedContext d1 = ( c5.Loc == c6.Loc
        and c4.Play == On and c2.Play == Off
        and c7.Power == Off )
Situation SmartView_Work
= ( d1, {c2, c4, c5,    c6,c7},  <a1, a2, a3,a4> )
```

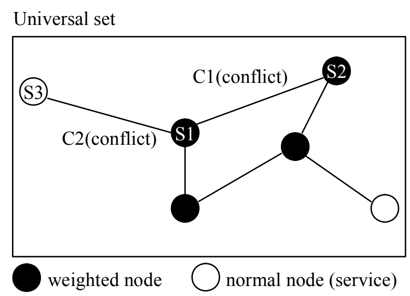
Situation aware system 3 : SmartView & Operation

Table6. SA-IDL Description of SA systems

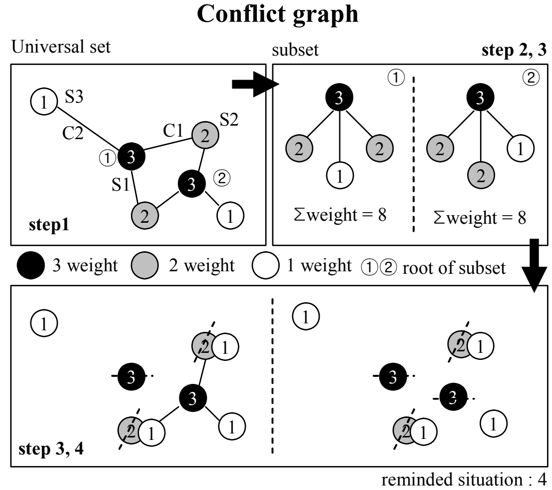
Our test bed consists of three situation-aware systems, light management system, SmartView, and entertainment environment management system. Each system gives services with pre-described situations, and each situation consists of Contexts and DerivedContexts, then Actions are executed according to each situation.

4.2. Experimental Results

A conflict is found among 'Situation LightOn' from light management system, 'Situation LightOff' from EEM, and 'Situation SmartView_Work' from SmartView system. The common Context c2 (in light manager and EEM) and Context c7 (in smartview) = (t, Light, Power) triggers two actions that can not happen at the same time: (0, Light, Power, On) and (0, Light, Power, Off). This conflict can be modeled as the following graph, and 'Situation LightOn', 'Situation LightOff' and 'Situation SmartView_Work' are denoted node S1, S2 and S3 respectively.



(a) Situation-Aware Collaboration System Conflict graph



(b) Situation-Aware Collaboration System Conflict Resolution

Figure 4. Experimental Results of Conflict Resolution

IEEE COMPUTER SOCIETY

S1 (Situation LightOn) is the weighted node which makes conflict with S2 (Situation LightOff) and S3 (Situation SmartView_Work). In Fig 4-(a), conflict C1 is occurred between S1 and S2, and C2 is occurred between S1 and S3. Other nodes are extra situation-aware systems and not considered here. These extra nodes may have relations with S1, S2 and S3. Above result depicted in figure 4-(b) can be obtained through the resolution algorithm based on weight.

Comparing with subsets, summations of weights are same. Thus four nodes remains by eliminating root nodes arbitrary, then S2 and S3 which corresponds the service of EEM and SmartView system are remained among final four nodes. Eventually, S1 is eliminated through conflict resolution algorithm, and this means that the logical error caused by light management system should be corrected for solve the conflict in the situation-aware collaboration system

4.3. Discussion

The conflict of situation-aware collaboration system can be solved with graph theory approach.

Table 7 show resolution results of Resolution Algorithm based on Weight.

System	Conflict Description	weight	Result
Light Manager	SmartView and EEM → conflict	3	Solve of Conflict
SmartView	Light Manager → conflict	1	Preservation of Situation
EEM	Light Manager → conflict	2	Preservation of Situation

Table7. Result of Resolution Algorithm based on Weight

Conflicts were solved by eliminating the Situation LightOn, and then this collaborative system provides the stable service for user.

However, we can find the difference between two services, which can show not only a device centered operation but also a user optimized service like this:

Resolution	Average conflict weight	Next Conflict Occurrence Ratio
SmartView	1.3	23/100
Light Manager	4.4	88/100

Table8. The Difference between two services

According to the result of above experiment, the heavy weight of service conflict means that the service, having a heavy conflict, can occurs more conflict than any other services and its function overlaps with that of other services. Thus other services can replace the service. In other words, the service which has a heavy conflict is regarded as the system having not a particular function but a general function. It is also regarded as a system which has no indigenous role by previous experiment.

Therefore constraint of a service which has a heavy conflict assists to provide the environment that can offer the specified service to user. This conclusion is a verification of the assumptions in chapter 3.2 and 4, and it is also the keynote of Situation-aware system providing a user customized service.

5. Conclusion

In ubiquitous environments, almost all electronic devices can be accessed by using computing system, and they can be programmed to collaborate with each other to provide adaptive services based on user's preference and circumstances. However, even though each device does not have logical errors, conflict problems are expected in the collaboration of them. And in the case of this conflict is not solved, the collaboration system provide not convenience for user but inconvenience. Thus in order to construct the reliable ubiquitous environment, we have to concern the conflict problem between services. In this paper, so we have formalized the conflict concept by using mathematical notations and proposed conflict resolution model. Additionally, the resolution algorithms have been proposed: one is *Resolution Algorithm based on Weight.*

The conflict concept that is formalized in this paper and the conflict resolution model becomes a important issue in the ubiquitous environments, where collaboration is necessary for providing adaptive services. And this research area will be essential base for any other researcher who wants to operate the collaboration system. Thus in order to confirm the contribution and complement the research, the future work is needed as follow: the first thing is to develop an effective conflict searching algorithms. In this paper, exhausted searching method is used to find out conflicts in situation-aware services. However, in large ubiquitous environments, the exhausted searching method would not be practical. Additionally, In order to import more effective resolution, comparing method by priority in the exceptive occasion, the summation of weight is same. Finally, resolution algorithms should be devised to incorporate with other requirements such as QoS, and authorization requirements.

6. Acknowledge

This work was supported by the 2nd Brain Korea 21 Project.

7. References

[1] Dongwon Jeong, Heeseo Chae, and Hoh Peter In, "The Performance Estimation of the Situation Awareness RFID System from Ubiquitous Environment Scenario" International Conference EUC 2005, Nagasaki, Japan, December 6-9, 2005. Proceedings p. 987 - 995.

[2] Yau, S.S.; Yu Wang; Dazhi Huang; In, H.P. "Situation-aware contract specification language for middleware for ubiquitous computting..Distributed Computing Systems" FTDCS 2003. Proceedings. The Ninth IEEE Workshop on Future Trends of 28-30 May 2003 Page(s):93 - 99.

[3]M. Weiser, "Some Computer Science Issues in Ubiquitous Computing", *Comm. ACM*, Vol. 36, No. 7, July 1993, pp. 75-84.

[4] S. Yau, Y. Wang, and F. Karim, "Developing Situation-Awareness in Middleware for Ubicomp Environments," Proc. 26th Int'l Computer Software and Applications Conference 2002' 233-238

[5] Robin J.Wilson, "Introduction to Graph Theory" Prentice Hall (1996)

[6]G. Bollella, S. Graham, and T. Lehman, "Real-time TSpaces", IEEE 25th Industrial Electronics Society (IECON 1999), pp. 837-842, 1999.

[7] P. Wyckoff, S. McLaughry, T.Lehman, and D. Ford, "T Spaces", IBM Systems Journal, Vol 37, Num 3. pp. 454-474, August 1998

0-7695-2810-4/07 $20.00 © 2007 IEEE

IEEE COMPUTER SOCIETY