2007－DPS－133 (2)
2007／11／26

教育的体験の向上を目的とした ユビキタスコンテキスト可視化フレームワーク

叶 献智[†] 荊 雷[†] 程 子学[††]
[†]会津大学大学院コンピュータ理工学研究科
[††]会津大学コンピュータ工学部
E-mail: {[†]m5111225, [†]d8071202, [††]z-cheng}@u-aizu.ac.jp

あらまし 最近コンテキストアウェアネス技術が目覚しい進歩している。コンテキストアウェアネスサービスは、ユーザの状況に基づいて自動的に適切なサービスを提供する。このサービスは、多くの利便性をもたらす。しかしユーザは、最近徐々にユーザとのやりとりを必要としないサービスに対して過度に依存してきている。一方で、サービスのメッセージによって惑わされ、更に危険が発生する場合もある。したがって、ユーザに対して教育を施す機能が必要である。そこで本論文では、教育的体験の充実を目的としたユビキタスコンテキスト可視化フレームワークを提案する。このフレームワークにおいて、ユビキタスペットと呼ぶ仮想的なキャラクターを設計する。このユビキタスペットは、現実世界の生き物と同様の特徴をもっており、ユビキタスサービスの視覚的な表示手段として利用されている。そのユビキタスペットの特徴や行動をユビキタスサービスに対応づけている。ユビキタスペットはユーザに対して、サービスとの関係とユーザの行動の結果を反映して動作する。ユーザが、サービスの正しい利用の仕方を理解することで、教育内容の充実を図る。

キーワード ユビキタスサービス、コンテキストアウェアネス、コンテキスト可視化、ユビキタスペット

A Ubiquitous Context Visualization Framework for Enhancing Educational Experience

Xianzhi Ye[†], Lei Jing[†], Zixue Cheng[††]
[†]The University of Aizu Computer Science and Engineering Department
[††]The University of Aizu Computer Engineering Department
E-mail: {[†]m5111225, [†]d8071202, [††]z-cheng}@u-aizu.ac.jp

Abstract In recent years, the technology of context-awareness services have provided convenience to users. However, users gradually tend to over depend on those services or be miss-led by those services, even result to some dangerous situation. So a real life experience to different situations in ubiquitous environment is desired for user to understand of the right way to use context-aware services. As the first step to solve this problem, we propose a ubiquitous context visualization framework for simulating and visualizing interaction between users and context-awareness services. In this framework, we create a virtual pet referred as ubiquitous pet. The ubiquitous pet has real life attributes. Its attributes and behaviors are linked with the context-awareness services. It grasps user's interaction with services, and the interaction affects its behaviors and attributes. At last it visualizes the effect of current context from context-aware services to users and the long term influence from context-aware services to users, and gives advices to users.

Key words Ubiquitous Computing, Context-aware Services, Context Visualization, Ubiquitous Pet

1 INTRODUCTION

With the number of ubiquitous computing resources is increasing exponentially, it is becoming common to embed little computing resources into homes, offices, and public spaces around people in the form of tiny wireless devices. As a result, the technology of context-awareness has a great growth. Based on different sensors, the context-awareness applications can grasp user's situations, and accordingly provide context-aware service to the user. However, the technology of context-aware services in fact is a double-edged sword. On the one hand, the user enjoys the convenience from the interaction-less of context- aware services, and on the other hand, the user tends to over depend on the context-aware services or may be miss-led by those services. This may even result in some very dangerous situations.

To solve this problem, an educational function is desired for the user, so that the user can experience different situation in ubiquitous environment, and learn the right way to use the context-aware services. In this paper, as the first step, we propose a ubiquitous context visualization framework for simulating and visualizing the interaction between the user and context-awareness services.

In the framework, we create a virtual character referred as ubiquitous pet. The ubiquitous pet is a user interface in the framework and works as an actor to the user. The ubiquitous pet has real life attributes, which link to context-awareness applications. It behaves differently according to the interaction between the user and context-aware services. It acts as a mirror or a stand-in to the user. The framework visualizes the different developing effect of the ubiquitous pet according to the user, and the ubiquitous pet also gives advices to the user.

Related researches show many ways to amplify cognition with context visualization. The smart gate [3] not only checks user's staff whenever the user goes out but also notifies the user by visualizing that information on the door. Focus + context visualization [6] enables users to see an interesting object in detail while having the overview at the same time. A visualization reference model is given in [4], and a refined visualization reference model on mobile device is given in [5]. In addition to visual presentation, also other modalities are proposed. A Smart Schoolbag System [1] for Reminding Pupils of the Forgotten Items, it uses audio-only interface that is installed into user's schoolbag and notifies the user whenever the user forgets bringing books with him. And the notification will vary, according to the time of forgetting; Nomadic Radio [6] is based on several auditory techniques to provide users with notification, passive awareness and control.

The related models or systems above mainly focus on presenting current detail context information to users. Comparing with those researches, the innovate features of our framework is focusing on presenting both the possible effect of the current context of context-aware services and the long term influence from context-aware services to the user. To improve the visualization effect and enhance educational effect, we create the ubiquitous pet. We choose a pet as a character because this character can easily build an emotional connection with the user in psychological layer. The user tends to care the pet and raise the pet, and it is also easily for the user to take advice from pet. The ubiquitous pet is always aware of what the user is doing by grasping the user's interaction through context-awareness applications, and the ubiquitous pet always likes to imitate the user and grows according to the interaction. And the user can find out how the ubiquitous pet grows. The benefit of our framework is that the ubiquitous pet shows the user the behavior of itself instead of giving the direct instruction to the user what to do, and the user can always take advices from the ubiquitous pet.

2 MODEL OF UBIQUITOUS PET

The model consists of a user, real world events, context events, context-awareness applications and a set of ubiquitous pets. In this model, we only consider the situation of a single user. The definition of real world events is catch-able events triggered by user or nature incidences. The context event is defined as the high concise context information produced by specialized context-awareness applications from real world events. For example, the user forgets to close gas, or temperature of the house becomes high. The definition of context-awareness application here is an application that provides context-aware services to user. Context-awareness applications grasp the real world events using sensors, and refine the real world events, and transform them into context events, and then send context events to a ubiquitous pet. There are a set of ubiquitous pets, they are different from appearance. The user can choose a favorite one, and develop it. For simplicity, we will refer the ubiquitous pet as U-pet below. Every U-pet has two types of attributes: *state* and *attribute*. The framework defines basic the aggregate of *state* and *attribute*, *state* consists of a set {happy, unhappy, hungry, hurt, ill, dead}, *attribute* has {current life, max life, tiredness, current stamina, max stamina, hungriness}. In this framework we use two kinds of rule base, internal rules base and external rule base. The external rule base links the *state* and *attribute* to context events through connecting to context-awareness applications. And the internal rule base is used to control the inside interaction of the U-pet, so that U-pet has real life specialties. For example, if there is a gas leak in the house, the current life of the U-pet will be cut down to zero, the U-pet will become dead state. If the U-pet is not feed by user through some real world events for some hours, the hungriness will grow up, and the U-pet will become hungry state, and ask the user to feed it. The character of U-pet is set to always like to imitate user' life pattern, and the age of U-pet is set to be a child and need to be developed up.

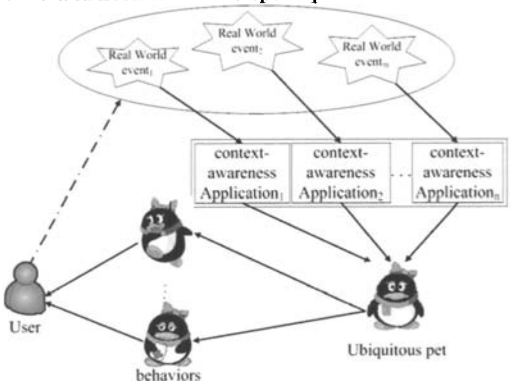

Real World event$_1$
Real World event$_2$
Real World event$_n$
context-awareness Application$_1$
context-awareness Application$_2$
context-awareness Application$_n$
User
behaviors
Ubiquitous pet

Fig.1. A context visualization model of ubiquitous pet

The activities of the user and nature incidences trigger real world events, and those real world events are caught by context-awareness applications through sensors. Context-awareness applications provide context-aware services to the user, and meanwhile context-awareness applications convert the real world events to context events, and then send the context events to the ubiquitous pet. The context events affect the U-pet's behaviors according to rules, which link the U-pet's attributes to context events, and as a result the *state* or *attribute* of the ubiquitous pet changes, and the user can see the behavior of the U-pet.

To implement the framework, the following problems must be solved. (1) How to build a connection between the U-pet and context-awareness applications. This problem refers how to transfer real world events to the U-pet. (2) How to design the rule base to link the behaviors of the U-pet to real world events, and make the U-pet become alive, and the U-pet can be developed by the user. (3) How to visualize the U-pet's behaviors, so that the user can feel realistic and interesting, and would like to learn the advices from the U-pet.

3 CONTEXT VISUALIZATION FRAMEWORK

Card et al. [4] has presented a general visualization reference model, which have been applied successfully to different visualizations. Essentially, the model presents phases that are needed in mapping raw data and transform it into visual form. The basic procedure of this reference model includes three parts: *data transformation*, *visual mapping*, and *view transformation*. *Data transformation* is for mapping raw data into data table, *visual mapping* is to transform data table into visual structure based on rules, and *view transformation* is to convert *visual structure* into actual views. This structure is very simple and efficient, it can afford to different extension of visualization. However this reference only transfers data from sensors and only support simple rule mapping, and has not given specific visual structures, so we extend the visualization reference model by Card et al. [4], and made modifications to the basic procedure.

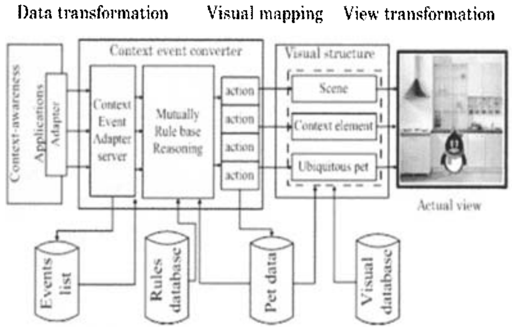


Fig.2. The reference model for context visualization of ubiquitous pet

Fig.1 shows our visualization reference model of our framework. First we made a major modification to *data transformation* by adding a mechanism of context event message transformation, so that the context events can be transferred from context-awareness applications. Another major modification is that we applied a mutually rule base reasoning in *visual mapping* to dynamically adapt the U-pet's behaviors to user's interaction or changes from environment. And context event converter is designed to replace data table to performance mutually rule base reasoning. In *view transformation* we specified three visual structures and converted them to actual view using a visual database.

3.1 Data transformation and Context Event Message Transformation

The traditional architecture of ubiquitous application is built with three layers, the application, switch, and sensor using Roy Campbell's definitions. Switch here means the middleware or interface provided for applications to access sensors. This architecture addresses application-specific, static-sensor deployment to accurately monitor the sensed environment in real time, and applications can easily manage and control the sensors. However, it is difficult for applications to manipulate large number of sensors' data, since transforming of raw data from sensor to meaningful information consume many CPU time. It is a waste of resource, because the information translated from raw sensor data seldom be derived, so it is hard for a high-level application to construct and manage a large ubiquitous environment. To make our framework to be able to manage large context information for the user, a context event message transformation is designed to derive concise context information from context-awareness applications to the context visualization framework.

The context event message transformation not only controls the data transferring from context-awareness applications to the framework, but also gives a standard about context event organization in context-awareness applications. To complete the context event message transformation, three problems must be solved. What data format should context-awareness applications give, how should the context-awareness applications send the data to the framework, and when should the context-awareness applications send the data.

Structure of context event message

To visualize the effect of context to U-pet, it is need for context-awareness applications to give high concise context information about the context, and the framework needs to know the meaningful information about the environment. So we concern mainly four questions about the content of context event message: (1) what event dose happen, (2) when does the event happen, (3) if the event just happened or the event is still lasting, (4) where is user when event happens. Here we give an example for context event message below.

```
<Service Id = "101" Service name = "Gym" time = "2007/9/10 10:30:30"/>
<Event Id = "101001" Event Name = "user is in the gym" User Id = "001" Event type = "persistence" value= "true" Location = "true">
```

We use xml structure to construct the context event message since the xml structure is good in data transferring and data organization.

- **Service Id** and **Service Name** specify the context-awareness service name and identifier.
- **Event Id** and **Event Name** define the context event element name and identifier.
- **Event type** includes two types, persistence and instance. It defines two kinds of facts that the event just happen or the event is still happening. We also refer instance type as event-driven and persistence type as time-driven.

- **Event value** defines the content for context event. The value's type contains Boolean, integer, and double type.
- User Id defines the user who is related with the event
- Location shows if the event contains user's position information. Only some context-aware services related with the location information, and can provide this information, for example GPS service, RFID card security service on door.

Context event message transferring method

To lighten the burden of Context-awareness application, an adapter is installed in context-awareness applications. And once context-awareness applications give context events to the adapter, the adapter will send the message to message adapter server in the framework shown in Fig .2. C/S structure is designed between the adapter and the message adapter server, and the transferring is built above the TCP/UDP protocol.

Context event message updating policy

This policy is designed for context-awareness applications to decide the timing to send data. Whether the U-pet can correctly synchronize its behaviors with environment is depended on this policy. According to the types of facts, we have to consider both the conditions of persistence context event and instance context event.

- **Instance type**: those events are sent to message adapter server when the event happened.
- **Persistence type**: Those events are sent to message adapter server every unit time, considering balance between CPU time and importance of event, the unit time is different for events. For example event likes current weather is sent every hour, while the gas degree in kitchen should be sent to message adapter server every second.

3.2 Visual mapping and mutually rule base reasoning

Visual mapping is the core control of the visualization framework. Its purpose is to adapt the behaviors of the U-pet to the context events using rules base, and transfer actions to visual structure. However the question is that the rules are static, while the ubiquitous pet is alive and it need not only show the current reaction to the interaction of user, but also show the long term influence of user's custom or habit. So this means that firstly the rules base must be dynamical and soft, former rule reasoning result can become a factor in the next rule reasoning process. Secondly, the time- related should be considered because most of real world objects change by time. Here we propose a Mutual Rules Base Reasoning to solve this question.

We first applied the conditional sentence reasoning in the context event converter. The advantage of this structure is that the syntax of logical inference is very clear and understandable, so user can define some complex logical facts. However, the disadvantage is that it is not dynamical and time-related. Then we tried to add variables in the conditional sentence reasoning, though this time this structure can be dynamical and soft, using variables in prediction and action can affect next prediction. However the time-related still is the problem. We also considered that listing all the case of time could be a solution to time-related problem, but there is too much rules need to write. Then we considered about using associate rule mapping. The advantage is that the quantity of rules needs to write reduce greatly, and former events can be new facts as factor in deducing new results, and the associate rule mining very real time system because it is always in deducing and finding new facts. The disadvantage is that the rules is not very understandable for every user, and can not reach totally time-related, because there is no precise time cycle.

Mutually Rule Base Reasoning

A double rule base structure referred as Mutually Rule Base Reasoning is designed for solving this problem. The reason to design Mutual Rules Reasoning is that, all the living organisms have two environments, the internal environment inside them, and the external environment outside them. The external interaction with living organisms is by events, and the internal interaction is controlled by a biologic clock, so it is time-related.

The rule mapping has two types of different rule bases to correspond to the different environment. Since many real world objects automatically changes mainly with time, and the time should be considered as an important factor, so we also introduce an internal cycle into the Mutually Rules Base Reasoning to solve the time-related problem.

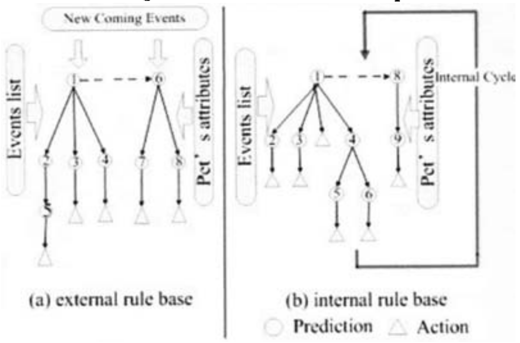


Fig.3. The procedure of Mutually Rule base Reasoning

The structure of Mutually Rules Base Reasoning is in Fig.3. One of the rule bases is called external rule base shown in Fig.3 (a), this rule base handles all the new coming events from context-awareness applications, the handling policy applies conditional sentences reasoning using variables. This procedure is only done, when an event is fired and it adapts to all the external physical information. The other rule base is called internal rule base shown in Fig.3 (b), this rule base handles all the events in the events list in which the events are persistent event. And internal rule base has an element called internal cycle, which controls how long to recheck the internal rules and the data of ubiquitous pet again, using internal cycle the ubiquitous pet can simulate the long term change of user's interaction. The internal rules also apply the case-base reasoning using variables, but the content of rule is different to external rule. This procedure is done in every internal time cycle.

Actions

When the event matches the prediction in the rules base, then the action is executed. There are mainly two types of actions. They affect the *attribute* and *state* of the U-pet. According to the attributes of U-pet, the actions are defined into two kinds. To define the content of an action we add a value field into the action, the value field contain the content of action result.

- Changing the U-pet's *state*
  The value of action is a simple sentence. And action will change the current *state* of the U-pet.
- Adjusting the U-pet's *attribute*
  The value of action is a simple syntax sentence. In the sentence includes some operators like "+,-, =,*, /, " etc. the *attribute* of the U-pet is changed, according to the simple syntax sentence.

3.3 View Transformation and Visual Structures

The last phase of our visualization reference model is view transformation. The view transformation converts the visual structures into actual view. Traditionally, a view in real world consists of sense, objects, and character, they define where the event happens, and who is there, and what objects are in the sense, and have relations with character. Therefore the visual structures are divided into three layers: sense, context object and the U-pet. The difference from common view transformation ways like location probes, viewpoint controls, is that we focus on presenting the different behavior of U-pet according to context events, further to display the relation between the user and context objects.

To better probe the actual view, the visual structures are connected to a visual database, the sense and context object image are stored in the database, and the different expression of the U-pet are also stored in the visual database as animation data. If the context events contain the location information, the sense layer will probe the visual database for the location information, the situation of indoor or outdoor will be different in the presenting in sense. In the outdoor situation, a digital map with GPS information is presented, while in the indoor situation the picture of room is presented, for example some dangerous spot in factory. The context object layer matches a new coming context event in visual database, and present the image or value of context event above the sense layer. Finally the ubiquitous pet layer probes the proper expression in the database based on the state and attribute of U-pet, and display using animation above the sense layer and context object layer.

4 CASE STUDIES

This case study is built in a personal gym. In the gym the user uses gymnastic apparatus to do exercise. Early research [8] of our lab has built an application, which provides guiding service for helping the user to do exercise. Some sensors are installed in the gymnastic apparatus for the application to get the context of gymnastic apparatus.

The purpose of this experiment is to use the framework to build a system, which can connect the early application to the system, and link exercise contexts to virtual attributes of pet, so that user can workout together with the U-pet, and the U-pet also grows together with user.

Two context-awareness services, a security door and application in [8] are connected to the system. And the context events from those applications are mapped to U-pet. If the user exercises for long time, the U-pet will tell the user that she is tired, and need to have a rest. After some enough time of rest, the U-pet can do exercise again. If user insists to do exercise, the ubiquitous pet will become ill and finally dies. As the time of exercise increasing the U-pet will grow stronger and can do longer exercise with the user. On the other hand if user does not exercise for a long time, the U-pet will become weak. The U-pet will tell user should do exercise more.

(a) The procedure of the exercise

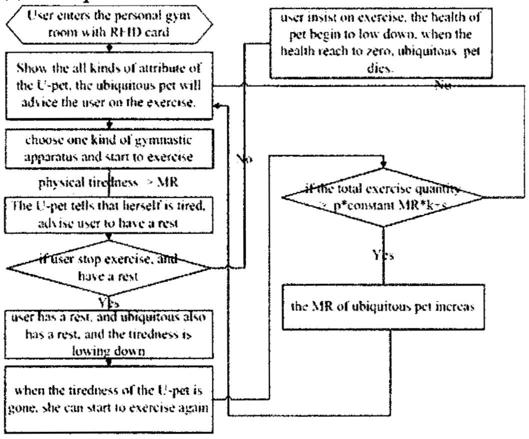

User enters the personal gym room with RFID card
Show the all kinds of attribute of the U-pet, the ubiquitous pet will advice the user on the exercise.
choose one kind of gymnastic apparatus and start to exercise
physical tiredness > MR
The U-pet tells that herself is tired, advise user to have a rest
if user stop exercise, and have a rest
Yes
No
user has a rest, and ubiquitous also has a rest, and the tiredness is lowing down
when the tiredness of the U-pet is gone, she can start to exercise again
user insist on exercise, the health of pet begin to low down, when the health reach to zero, ubiquitous pet dies.
No
if the total exercise quantity > p*constant MR*k-s
Yes
the MR of ubiquitous pet increas

Fig.4. The exercise procedure

As shown in Fig .4, when the user enters the personal gym using RFID card, the system will received context events from door security application, and the context event will tell the system the user is in the gym and begin the exercise procedure. If the physical tiredness is bigger than MR (we use MR to define the max tiredness), the U-pet will tell the user that she is tired and need to have a rest. If user insists to do exercise, the health attribute of ubiquitous pet will begin to low down, when it reaches to zero, the U-pet dies. If user has a rest, the physical tiredness of the U-pet also lows down, and then the U-pet finally recovers. The system always checks if the total exercise quantity is bigger than p*constant^{MR*k+s}, if it does, then the MR of ubiquitous pet will increase, this means that the user can exercise more time in every set of exercise. The U-pet can also show the user the physical attributes of itself.

(b) System design

Context Event Messages definition

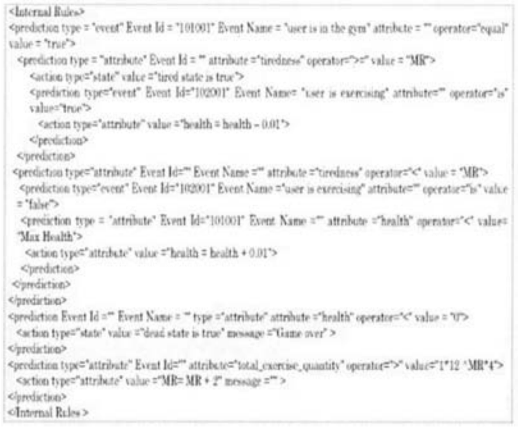
The context events are defined for context-awareness applications.

(1)When user enters the gym room, the following context event is fired. <Service Id = "101" Service name = "Door Security" time = "2007/9/10 10:30:30"/><Event Id = "101001" Event Name = "user is in the gym" User Id = "001" Event type = "persistence" value= "true" location ="true">.

(2)When user leaves the gym room, the following context event is fired. <Event Id = "101001" Event Name = "user is in the gym" User Id = "001" Event type = "persistence" value= "false" location ="false">

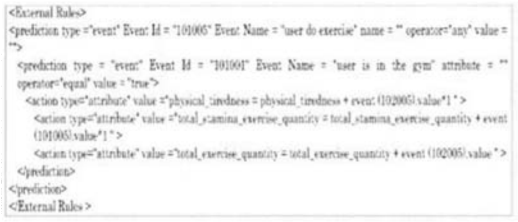
(3)When user does exercise, the following event is fired. The value defines the quantity of exercise user do. <Service Id = "102" Service name = "gymnastic apparatus application" time = "2007/9/10 10:30:30"/><Event Id = "102005" Event Name = "user do exercise" Event type = "instance" User Id = "" value= "0.4" location ="false">(4)User is exercising. <Event Id = "102001" Event Name = "user is exercising" Event type = "persistence" User Id = "101001" value= "true" location ="false">

Internal rules



```
<Internal Rules>
<prediction type = "event" Event Id = "101001" Event Name = "user is in the gym" attribute = "" operator="equal"
value = "true">
  <prediction type = "attribute" Event Id = "" attribute ="tiredness" operator=">=" value = "MR">
     <action type="state" value ="tired state is true">
     <prediction type="event" Event Id="102001" Event Name= "user is exercising" attribute="" operator="is"
     value="true">
       <action type="attribute" value ="health = health - 0.01">
     </prediction>
  </prediction>
 <prediction type="attribute" Event Id="" Event Name ="" attribute ="tiredness" operator="<" value = "MR">
  <prediction type="event" Event Id="102001" Event Name ="user is exercising" attribute="" operator="is" value
  = "false">
  <prediction type = "attribute" Event Id="101001" Event Name ="" attribute ="health" operator="<" value=
  "Max Health">
   <action type="attribute" value ="health = health + 0.01">
  </prediction>
 </prediction>
</prediction>
<prediction Event Id ="" Event Name = "" type ="attribute" attribute ="health" operator="<" value = "0">
  <action type="state" value ="dead state is true" message ="Game over" >
</prediction>
<prediction type="attribute" Event Id="" attribute="total_exercise_quantity" operator=">" value="1*12 ^MR*4">
  <action type="attribute" value ="MR= MR + 2" message ="" >
</prediction>
<Internal Rules >
```

External Rules



```
<External Rules>
<prediction type ="event" Event Id = "101005" Event Name = "user do exercise" name = "" operator="any" value =
"">
  <prediction type = "event" Event Id = "101001" Event Name = "user is in the gym" attribute = ""
  operator="equal" value = "true">
    <action type="attribute" value ="physical_tiredness = physical_tiredness + event (102005) value*1 " >
      <action type="attribute" value ="total_stamina_exercise_quantity = total_stamina_exercise_quantity + event
      (101005) value*1 " >
      <action type="attribute" value ="total_exercise_quantity = total_exercise_quantity + event (102005) value " >
  </prediction>
</prediction>
</External Rules >
```

Example of the system user interface is shown in Fig. 5. In Fig. 5(a) ubiquitous pet is doing exercise with user in the screen. In Fig. 5(b) when user exercises for a long time, the ubiquitous pet will be tired, and she will ask the user to have a rest shown

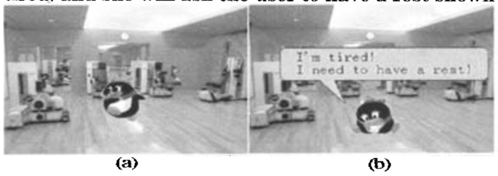


Fig.5. Snapshots of system user interface

5 CONCLUSION

In this paper we described a context visualization framework. This framework can be used to design kinds of experimental environment based on real world, user could develop the U-pet through daily activities and experience different effects related with context-aware services. In the future, log layer in visual structure will be considered, so that the user can not only know the effects, but also know what causes the effect. This would be a promise of the better educational experience for user to understand the context-aware services in real life, and use them in right way.

REFERENCES

[1] Lei Jing, Noriko Yamamoto, Zixue Cheng, Hui-Huang Hsu, and Tongjun Huang, "A Smart Schoolbag System for Reminding Pupils of the Forgotten Items," Ubiquitous Intelligence and Computing, publisher Springer Berlin, Volume 4159, pp. 44-50, 2006

[2] Kawanishi Nao, Kida Nobuo, Morikawa Hiroyuki, and Aoyama Tomonori, "Applying Sensor Networks to Environmental Information Adaptive Computer Entertainment," IPSJ SIG technical reports IPSJ SIG technical reports, Vol.2006, No.24(20060313) pp. 17-24, 2006-EC-3-(3)

[3] Sung Woo Kim, Min Chul Kim, Sang Hyun Park, Young Kyu Jin, and Woo Sik Choi, "Gate reminder: a design case of a smart reminder," Proceedings of the 2004 conference on Designing interactive systems: processes, practices, methods, and techniques, pp. 81-90, 2004

[4] Stuart K. Card , Jock D. Mackinlay , and Ben Shneiderman, "Information visualization, Readings in information visualization: using vision to think," Morgan Kaufmann Publishers Inc., San Francisco, CA, 1999

[5] Antti Aaltonen, and Juha Lehikoinen, "Refining visualization reference model for context information," Perional and Ubiquitous Computing, Volume 9, Issue 6, pp. 381-394, 2005

[6] Staffan Björk, and Johan Redström, "Redefining the focus and context of focus+context visualization," Proceedings of the IEEE Symposium on Information Vizualization 2000, pp. 85-99, 2000

[7] Sawhney N, and Schmandt, "Nomadic radio: speed and audio interaction for contextual messaging in nomadic environments, " ACM trans Comp hum Interact (TOCHI), Volume 7, Issue 3, pp. 353–383, 2000

[8]Shengguo Sun, Zixue Cheng, Lei Jing, Tongjun Huang, and Hui-Huang Hsu, "A Ubiquitous Learning System for Improving Quality of Learner's Behaviors Based on Shaping Principle," Proceedings of the Sixth IEEE International Conference on Computer and Information Technology (CIT'06) - Volume 0, pp. 216-221, 2006