http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 9 (2): 282-289, 2010
ISSN 1812-5638

Autonomy on Intelligent Home Network

Liguo Liu
MOE Key Lab for Intelligent Networks and Network Security,
Xi'an Jiaotong University, Xi'an 710049, China

Abstract: Autonomy is an effective approach to cracking some hard nuts in an intelligent system. In this study on intelligent home network, the focus is placed on the implementation of autonomy of a device. For this reason, some related concepts such as Virtual Device are introduced in this study. Consisting of a family of function agents and possessing the ability for auto-configuration, auto-announcement, auto-service discovery and so on, a virtual device interacts with users or surrounding devices on behalf of the physical one. Based on the virtual device, the autonomy of a device as well as the entire home network is implemented.

Key words: Intelligent agent, autonomous network, intelligent home network, distributed system

INTRODUCTION

In recent years intelligent household has become popular. More and more researchers commit themselves to it (Chung-Ming *et al*., 2008; Goth, 2008; Jung-Tae *et al*., 2008). One fact to which we should attach more attention is that quite a few end users of home network are non-expert ones. Therefore, the simplicity of installation, maintenance and use becomes a vital factor for a home network while autonomy is an effective approach to confronting this challenge. Inspired by advancements of mobile agent and ubiquitous computation (Charles *et al*., 2008), many results have been achieved in this area. In Jeong-Joon and Dong-Ik (2003), for example, mobile agent is employed to obtain the scalability of interactions between home networks. However, the mobile agent is only taken as a supplementary means to improve the performance of a home network rather than the autonomy of a household appliance. Soldatos *et al*. (2007) have concentrated on the autonomous context-aware ubiquitous computing services. However, the autonomy of home network is obtained by other external auxiliary means rather than household appliances themselves, which may increase the running cost of a home network. In our opinion, the ability of autonomy should be integrated into the household appliance. Consequently, an autonomic home network is a decentralized network and a household appliance should possess the capability for auto-configuration, auto-announcement, auto-service discovery, etc. Much effort has been invested in this aspect in this work, AAHN (Agent-based Autonomic Home Network).

Similar to the concept of plug and play, it is expected that an intelligent home network should be constructed gradually as appliances join it randomly (Brumitt *et al*., 2000; Merabti *et al*., 2008). To achieve this goal, a relevant concept, Virtual Device, is introduced into AAHN. A virtual device, comprising a XML-based device and service description and a family of agents, is designed to assign the ability of autonomy to a physical device. In a virtual device, an agent consists of one or more procedures for a specific function and some mental components (e.g., beliefs, desires, goals, plans, intentions, etc.) (Hindriks *et al*., 1999; Shoham, 1993). Based on these mental components, it could learn the environment where it locates and infer its further operations. Therefore, it provides some built-in autonomy, which lays the cornerstone for implementing the autonomy of a virtual device. As a wrapper of a physical device, the virtual one completes most interactions with the outside world, by which some intelligent operations could be implemented gracefully. From the perspective of a virtual device, a home network may be viewed as a network of various virtual devices. In the next section, a brief description about the architecture of a virtual device is given.

ARCHITECTURE OF VIRTUAL DEVICE

In this study, a virtual device serves three purposes generally: interacting with end users and surrounding devices, decomposing an assigned task into services and scheduling them and wrapping and controlling the physical device. For the first purpose, a virtual device is responsible for registering and authenticating end users, responding their service request and interacting with other devices. For the second one, a virtual device is required to give a list of candidate services for the assigned task (which is called a recipe in AAHN), discover local or remote entities that could complete these services and launch the execution of a

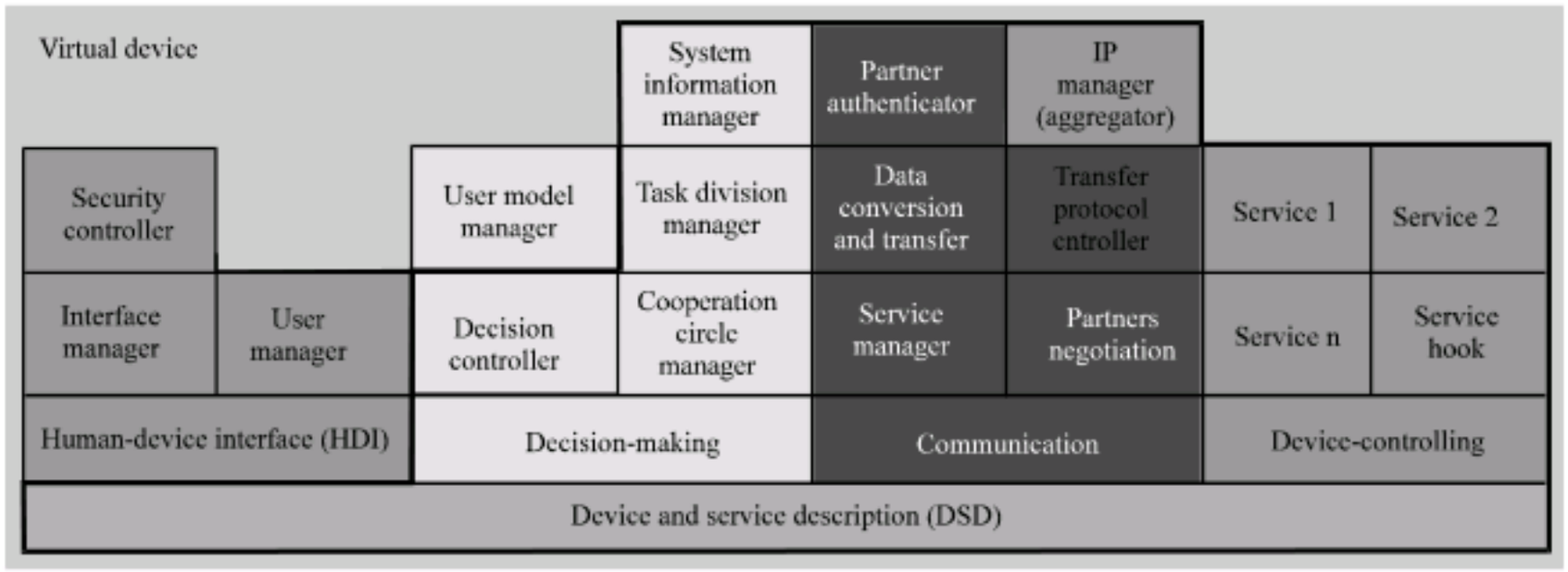

Fig. 1: Architecture of a virtual device

recipe. Another task is to model personal preferences of end users. As a series of software components, the role of a virtual device is to wrap, monitor and invoke the physical device to complete the specific function, which is the meaning of the third purpose.

As shown in Fig. 1, a general virtual device is divided into five modules: Device and Service Description (DSD), Human-Device Interaction (HDI), decision-making, communication and device-controlling. Apart from the DSD, each component here is implemented as one or more agents. Analogous to Gerald *et al*. (2004), Michael and Robin (2005), these agents are responsible for their own internal and external behaviors, such as adjusting their own mental states, managing the resources and forming and maintaining relationship with other agents. Note that only those that surrounded by the bold black lines concerns the autonomy of a device and others are about the human-device interaction. We concentrate on the former in this study.

IMPLEMENTATION OF AUTONOMY OF VIRTUAL DEVICE

Device and service description: In order to facilitate the interaction between virtual devices and enable the control of physical devices, the device and service description is introduced. In AAHN, the description is compiled in XML. Each appliance is described as a collection of parameters, which denote the global environment where it operates and a list of services corresponding to its functions. A service description also consists of some contextual parameters, which specify prerequisites and/or results for a service and a series of actions, which represent various operations a service supplies. By exchanging device and service descriptions, virtual devices could discover services they expect, negotiate parameter assignments and finally complete the cooperation. Inside a virtual device, it controls the physical device in accordance with this description.

Auto-configuration

Normal IP configuration: Given the current situation of home network, dynamic IP configuration is employed in AAHN. DHCP is a popular approach to implementing dynamic IP management. Though the IP of a device can be obtained dynamically, the DHCP server has to be set up in advance and runs on the network continuously. On the one hand, the setup of DHCP server is a daunting job for some end users and on the other hand, the dedicated DHCP server increases the cost of running a home network. Therefore, it is reasonable that both IP configuration and IP server are implemented dynamically. Unfortunately, DHCP can not meet this requirement and other approach should be adopted to address this issue. As a result, the concept, IP aggregator, is introduced into AAHN. Each appliance, if it has the proper processing capacity and is willing to act as an IP aggregator, it would be. An aggregator holds a part of or the whole IP space of a home network and delivers IPs to new joining devices. Note that the default IP space is 192.168.0.*/24 and it could be set as a system parameter in AAHN.

When a device is ready to enter into the network, it requests its IP by broadcast on the network. As shown in the Fig. 2a, assuming that there is no aggregator existing on the network and it is competent for IP management, it becomes the first aggregator on the network by choosing the IP space it manages, by default which is the whole IP space used in AAHN. In addition to delivering IP, an IP aggregator also inquires whether the new joining device could assume the duty of an aggregator. If the device accepts this request, the aggregator divides its own IP space equally and transfers a half to the new joining device. Consequently, a new aggregator emerges on the network. This processing is shown in Fig. 2b.

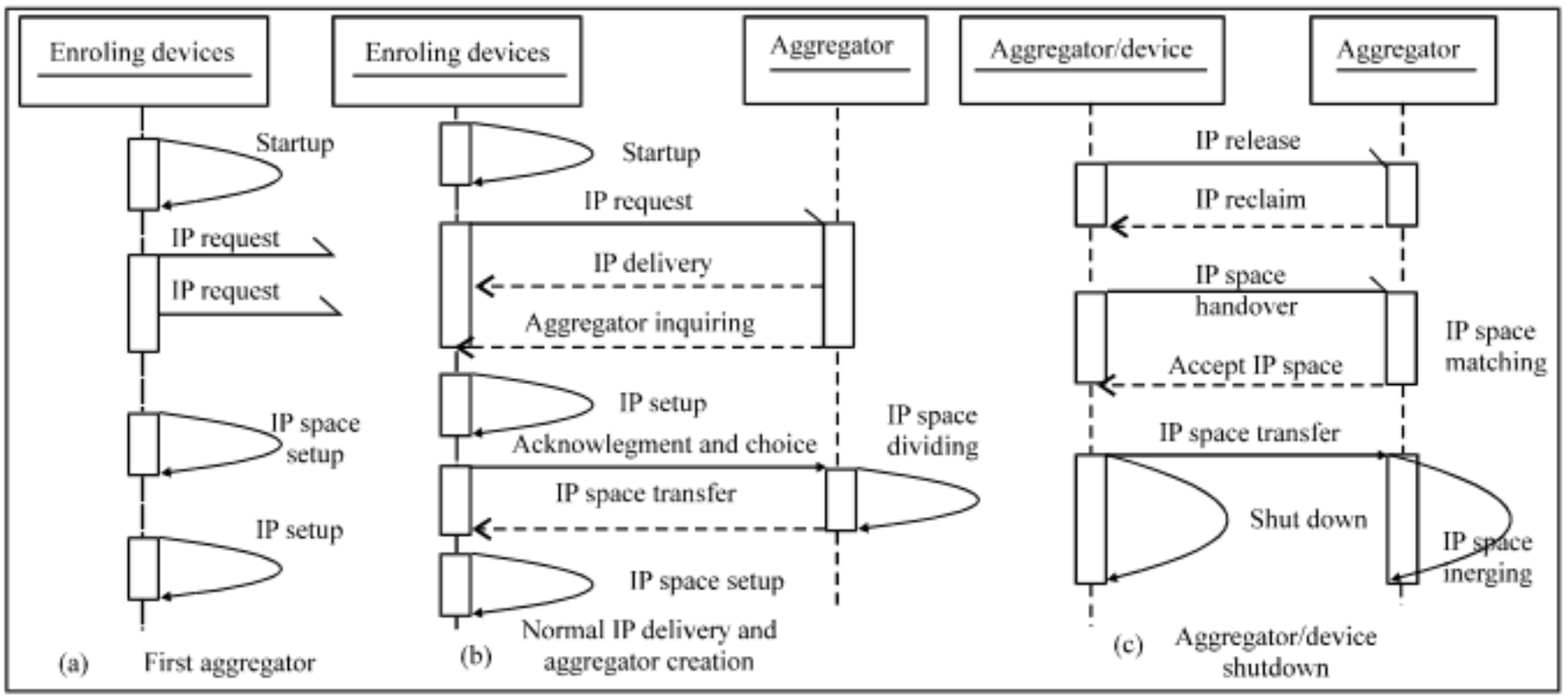

Fig. 2: IP configuration

When a device is about to exit from the network, it must returns its IP. For a common device, this is completed by broadcast. The aggregator that its IP space covers this IP reclaims the released IP. For an aggregator, its private IP is usually managed by itself and thus what it should do is to alter the state of its IP in the IP space, or else it has to do as a common device. Additionally, the IP space it holds and the state of each IP must be handed over to another aggregator. The successor is not arbitrary IP aggregator but is chosen according to the adjacency of their IP spaces. The corresponding process is shown in Fig. 2c.

Conflict handling the common conflicts and errors in IP management are the overlap of IP space and the loss of IP. For the overlap of IP space, two stages are involved to deal with it. The first is the overlap detection. When delivering an IP to an applicant, an aggregator always encloses the range of its IP space in the packet. Meanwhile, it monitors IP responding packets sent by other aggregators and determines whether the intersection of their IP spaces is empty. If the intersection is nonempty, it implies an IP space overlap occurs and the second stage should be brought into effect to resolve this conflict. Here, three scenarios have to be handled: two IP spaces are identical, two IP spaces are partially overlapped and one IP space is completely covered by another, which are illustrated in Fig. 3.

Where subvector(A, B, C) means to obtain the sub-vector of vector A from B to C in original order, LowerBindary(A) is to obtain the minimal element of vector A while UpperBoundary(A) to the maximum one, size of (A) acquires the element amount of vector A, New_private_IP() requests a new private IP and Inform_the_oppsite_aggregator() notifies the opposite aggregator that its IP space should be modified. Some necessary measures are taken to ensure the continuity of IP space of an aggregator. The pseudo-algorithm is self-explanatory and no further explanation is given here.

The principal cause for the loss of IP is the abnormal shutdown of an aggregator or a common device. A specific IP reclaiming algorithm, Lazy_Collector, has been developed to deal with this exception. It is launched just in the case of no available IP to be acquired in AAHN and by the first aggregator that observes this situation. Its underlying idea is to declare the possession of entire IP space by the resolver and it is evident that some conflicts must be caused. Subsequently, these conflicts would be observed by other aggregators and the troubleshooting process mentioned above would be put into action. Finally, the lost IP addresses are regained.

Management of cooperating circle: For a specific virtual device, it is desirable to record its partners for the future collaboration. The Cooperating Circle serves this idea and is managed by the agent, Circle manager. There are two sorts of Cooperating Circles in a virtual device, one for partners providing assistances to it and the other for those requesting assistances from it, which are denoted by Cons_circle and Prov_circle, respectively. The first one is implemented as a collection of lists, such as Cons_circle[i], where, $i \in [1, n]$ is an integer. Each list corresponds to a local service that needs some external assistances and each item in it is a partner record. The second one is instantiated as a single list, Prov_circle and each item also denotes a partner.

For a task that requires external service cooperation, Circle manager is requested by Task scheduling manager to seek the cooperating partner in its Cooperating Circle

```
Processing for IP space overlapping ( )
{
    A_set= the IP space range of aggregator A;   B_set = the IP space range of aggregator B;
    A_IP = the private IP of aggregator A;  B_IP = the private IP of aggregator B;

    If  ( A_set = B_set)            // two IP spaces are identical
    {       L_set= subvector (A_set , LowerBoundary( A_set ) , UpperBoundary( A_set ) / 2 );
            H_set= subvector (A_set , UpperBoundary( A_set ) / 2 + 1 , UpperBoundary( A_set ) );
            Start: If  (A_IP < B_IP) { A_set= L_set;  B_set= H_set; }
                    Else if  (A_IP > B_IP)  { A_set= H_set;  B_set = L_set; }
                    Else { A_IP = New_private_IP();  B_IP = New_private_IP();  Go to Start; } }
    Else
    {     If  (not((A_set ⊂ B_set) or (B_set ⊂ A_set)))        // partially overlapped
          {     If  ((sizeof( A_set ) < sizeof( B_set )){ B_set = B_set - A_set; }
                 Else { A_set = A_set - B_set; } }
          Else                                  // one IP space is completely covered by another
          {     If  (A_set ⊂ B_set)
                {  SL_set = subvector ( B_set , LowerBoundary( B_set ) ,  LowerBoundary( A_set ));
                    SH_set = subvector (B_set , UpperBoundary( A_set ) , UpperBoundary( A_set ));
                    If ((sizeof( SL_set) > (sizeof( SH_set) ){ A_set= A_set  U  SH_set; }
                    Else { A_set = A_set U  SL_set;  }
                    B_set = B_set – A_set;  }
                Else
                {  SL_set  = subvector ( A_set , LowerBoundary( A_set ) , LowerBoundary( B_set ));
                    SH_set = subvector ( A_set , UpperBoundary( B_set ) , UpperBoundary( A_set ));
                    If ((sizeof( SL_set) > (sizeof( SH_set) ){  B_set= B_set U SH_set; }
                    Else { B_set = B_set U SL_set;  }
                    A_set = A_set - B_set;  }  }
    }
     Inform_the_opposite_aggregator();
}
```

Fig. 3: IP space overlap resolution

in accordance with the service and credit record of each partner. In case of no suitable partner found in its present Cooperating Circle, a virtual device has to find one on the network. In both cases, the Cooperating Circle should be modified properly after the service cooperation.

Partners in a Cooperating Circle are prioritized based on their performances in the past service cooperation, which are called Credit and are quantified as decimals between-1 and 1. Upon the completion of service cooperation, the performance of the service supplier is evaluated by an agent, Partner assessor, of service applicant. Many factors are involved in this processing, such as the echo time and the fault rate of data transmission. The new Credit for a supplier is calculated on the current performance and the past Credit, by which the new position of the supplier in the Cooperating Circle is adjusted. Therefore, the Cooperating Circle can be viewed as the clue to the optimal choice of partner to some extent.

Auto-announcement: After a device started up and obtained its IP, an important task is to announce its existence, by which it could be learnt by other devices on the network. The device description is broadcasted on the network by Messenger. For other devices, they can either check or neglect this description. If one device checks the description and finds some services of interest, it might keep the device as a potential partner for the future cooperation. Nonetheless, neglecting this description does not mean a loss of a partner, as described in the following sections, this device could be found again in some other ways.

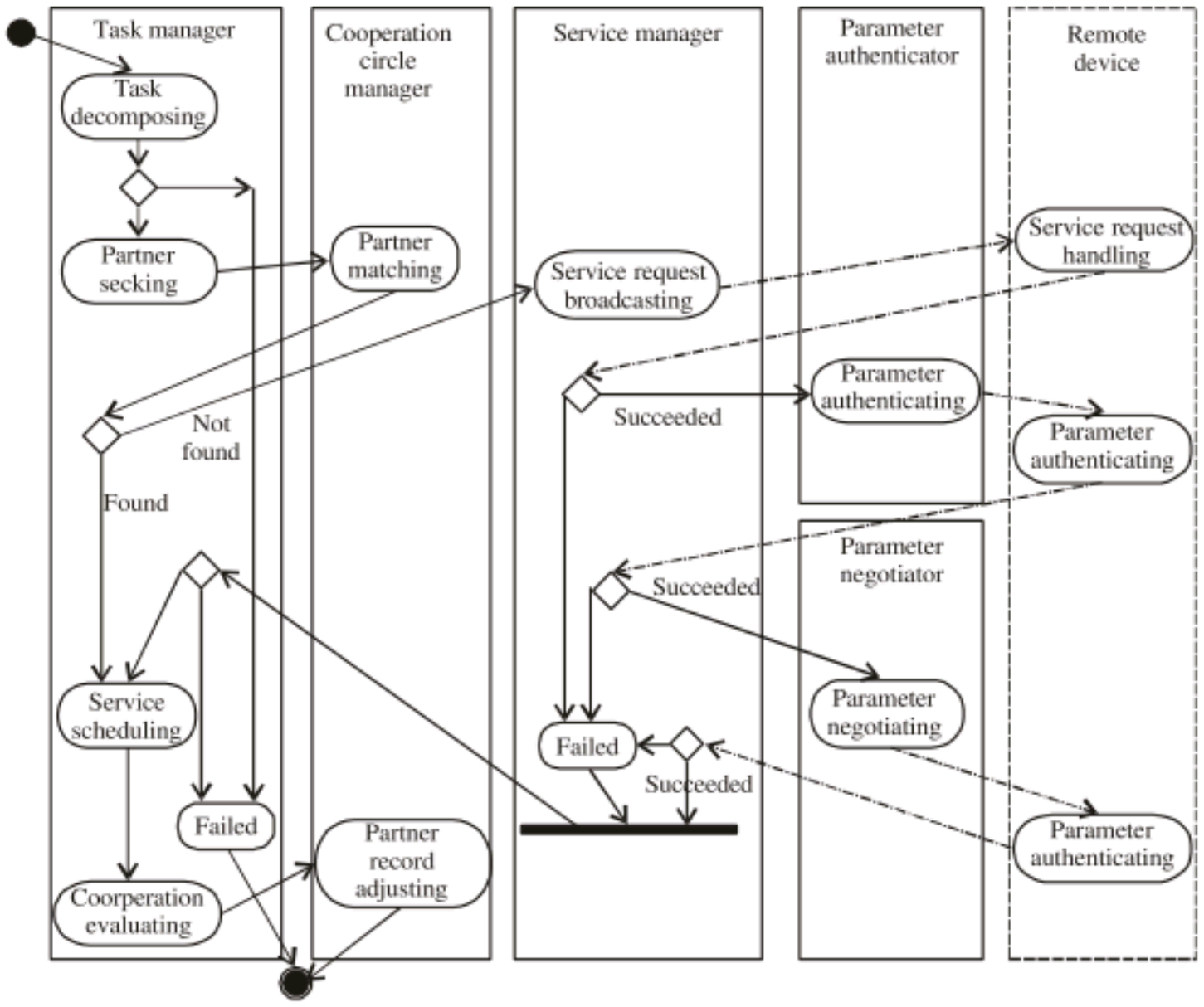

Fig. 4: Service cooperation between virtual devices

Auto-service discovery: When given some tasks, especially for those that require external assistances, a virtual device usually follows the procedures in Fig. 4 to complete them. Many agents are involved, not the ones mentioned here but those would be described in the following sections. There are two approaches to finding a partner: discovering it in the Cooperating Circle or seeking it on the network. To address this issue, Collaboration Degree is introduced into AAHN. Based on the comparison between contextual parameters of two services, Collaboration Degree is used to estimate how similar two services are and further determine the possibility of their cooperation.

Suppose that the same task has ever been handled by the virtual device, the appropriate partners are probably recorded in its Cooperating Circle. In this case, Circle manager is required to pick it out based on Credit. Next, a service request is sent to this partner. In case that the previous partner is power off or it is the first time to confront this task, Task scheduling manager has to commit Service searcher to discovering a new partner on the network. Say that a TV is given the task to play DVD video and it is the first time to handle this task. It is aware that it needs a DVD player as soon as it gets this task. After receiving a negative response from its Circle manager, TV forms the description of required service, which is as follows by and large:

```
<service-name>video-playing, DVD-playing</service-name>
<data-type>MPEG4</data-type>
<action-name>send, play</action>
<parameter-list>
    <parameter>…</parameter>
</parameter-list>
```

Upon forming the description, Service searcher would broadcast it on the network. If no response returns in some time, it means that no partner could be found. The virtual device has to give up this task. Supposing that the DVD player on the network receives this request and is free, Request handler of DVD player calculates the Collaboration Degree and learns that it could satisfy the request. Then the processing steps into the next stage.

Auto-authentication and auto-negotiation: As the DVD driver learns that it could serve the TV, they authenticate each other. It is done by their Partner authenticators. The common approach of trust transfer and integration is adopted. The entire authenticating procedure is shown in Fig. 5. Here, PDVD and PTV are private keys of DVD driver and TV, respectively. The operator ? means the concatenation of strings. $E_{PDVD}()$ and $E_{PTV}()$ are the

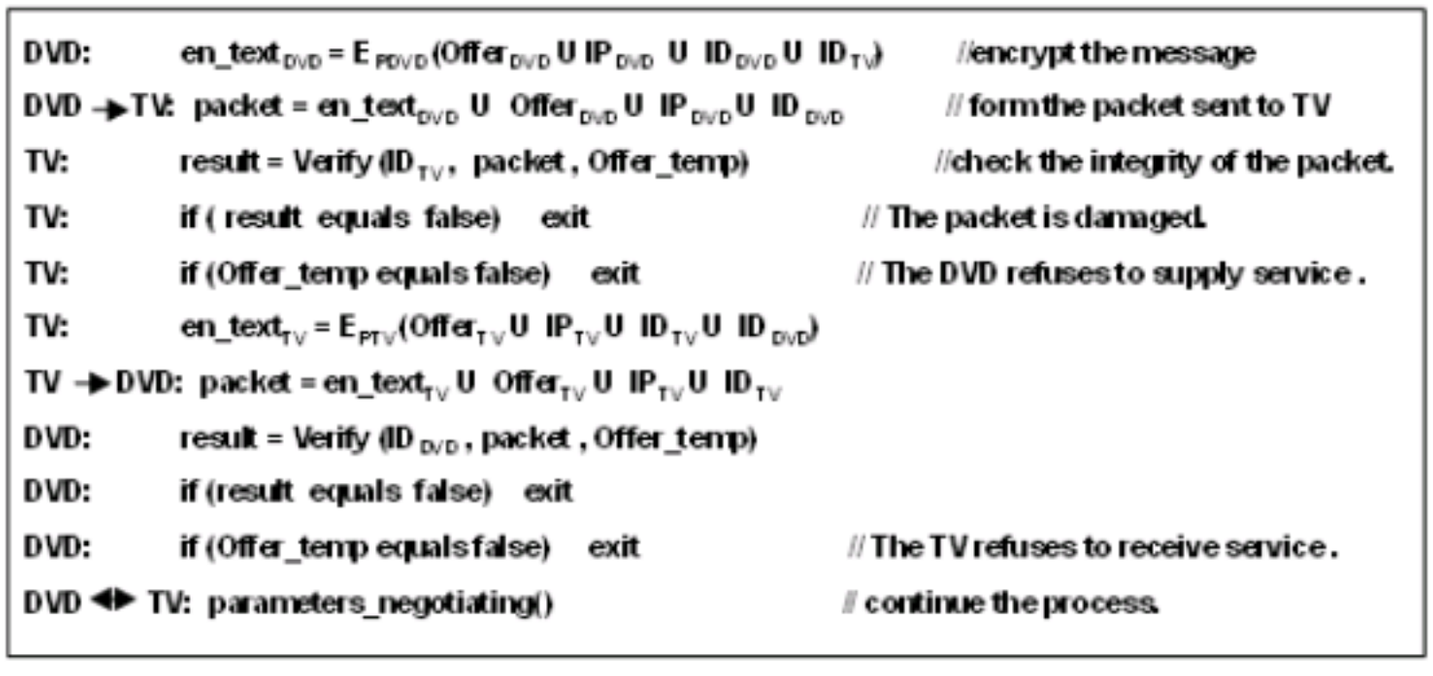

Fig. 5: Authentication

encrypting procedures used by DVD driver and TV, respectively. Verify() is to verify the validity and integrity of encrypted messages. If the mutual authentications passed successfully, parameters negotiation is the next job to be completed.

For each parameter involved in the service, the accepted ranges for two participants would be exchanged by Parameter negotiator. Each participant chooses and exchanges the possible value until reaching the final assignment or failing.

For the example illustrated here, supposing that the agreement of the parameter assignments has been reached after the negotiation, preparations are finished and the service cooperation could be launched. Except that the functions of physical device are controlled by the virtual device, it is analogous to service cooperation on other network, thus the description is omitted here.

Auto-strategy adjustment: In AAHN, decision-making implies two functions for a virtual device: one is the ability to form cooperation with other partners automatically and the other is to adjust its cooperating strategy in line with its states. The first function has been described in sections above. For the second one, two ways are adopted now. The first way tends to supply more services for other virtual devices, hence it is suitable for the situation when the load of virtual device is light or a request is sent by an affinitive partner. In contrast, the second way render the virtual device attaching more attention to its own task and neglect more external service requests. Correspondingly, it is more necessary when the load of virtual device is heavy, the resource is nearly running out or the service applicant is insignificant. In AAHN, strategy models can be loaded or dropped dynamically so that some flexibility of decision making is obtained.

When a device starts up, Decision model controller is launched. It inquires about the system status periodically from System assessor. Based on this status, a proper decision model is launched by the Decision model controller to deal with the coming service requests. Upon the arrival of a service request, Request handler consults Decision model controller for the acceptance or reject of this request.

Device wrapper: For the functions of a device, they are wrapped as a family of service agents in the virtual device, by which the physical device is controlled and invoked. Additionally, a supervising agent, Service hook, is implemented in each virtual device. It is in charge of installing/uninstalling and updating service agents; hence the function of a device could be upgraded or augmented to some extent. Another important feature is the migration of wrapping agent. In some scenarios, it is the best way to dispatch a proxy to the destination to run on behalf of the initial device. The technique of mobile agent has been proved to be an effective way to carry out this object. In the demo system of AAHN, the migration of function has greatly reduced the interaction between devices and enhanced the security and integrity of the system.

VERIFICATION

Based on the development toolkit, ADK (Tryllian Agent Development Kit) (2008), a verifying system has been developed and examined in two scenarios: IP management and service cooperation.

Scenario 1: In this scenario, two virtual devices, saying VA and VB, are launched on two computers to verify the IP management. The creation of first aggregator on the network is shown in Fig. 6a. The result for the normal IP

```
Host: VA
Request an IP......
No aggregator found in the network.
Intend to play as an aggregator.
Obtained system IP space:192.168.0.*.
My IP is 192.168.0.1.
My IP space: 192.168.0.0 ~ 192.168.0.255.
Total/Occupied: 256/1.
192.168.0.1   occupied
Waiting......
```

(a)

```
Host: VB
Request an IP......
Received an response.
My IP is set as 192.168.0.128.
Intend to play as an aggregator.
Send my offer to 192.168.0.1....
Received my IP space:
192.168.0.128 ~ 192.168.0.255
Total/Occupied: 128/1.
192.168.0.128   occupied
Waiting......
```

(b)

```
Host: VA
Received an IP request.
Deliver IP 192.168.0.128.
Send the IP and inquiry......
Received the offer from 192.168.0.128.
Divide the IP space into two:
192.168.0.0 ~ 192.168.0.127
192.168.0.128 ~ 192.168.0.255
Hand over 192.168.0.128 ~ 192.168.0.255
to the host: 192.168.0.128.
My IP space: 192.168.0.0 ~ 192.168.0.127.
Total/Occupied: 128/1.
192.168.0.1   occupied
Waiting......
```

(d)

```
Host: VB
Received an handover request.
Accept and respond the request..
Received the IP space from
the host: 192.168.0.1.
My IP space is modified to:
192.168.0.0 ~ 192.168.0.255
Total/Occupied: 256/1.
192.168.0.128   occupied
Waiting......
```

(c)

Fig. 6: IP management, (a) creation of the first aggregator, (b) normal IP processing-VB, (c) normal IP processing-VA and (d) IP space handover

```
Habitat ID: 60000003-0000c35064453c8a
Habitat name: lg

----------------------------------------

Restored 0 checkpointed agents and created 10 agents from habitat XML file.
[IV(Task scheduling manager)]: Accept and break down the task,Play DVD.
[IV(Task scheduling manager)]: Ask Circle_Service_Searcher to find a partner.
[IV(Circle manager)]: The DVD player has not been found in the circle.
[IV(Circle_Service_Searcher)]: Request the Service searcher to discover it.
[IV(Partner authenticator)]: Find a potential partner: wyllg://tryllian/agents/6
000002a-00355f55550524fe
[IV(Partner authenticator)]: Authenticate the opposite peer: wyllg://tryllian/ag
ents/6000002a-00355f55550524fe
[IV(Service searcher)]: The [tryllian/agents/6000002a-00355f55550524fe] has pass
ed the authentiocation.
[IV(Partner authenticator)]: IV has been authenticated by wyllg://tryllian/agent
s/6000002a-00355f55550524fe. Inform the Task scheduling manager.
[IV(Parameter negotiator)]: Negotiating.............
[VA(Service searcher)]: The negotiation succeeded.
[IV(Task scheduling manager)]: The DVD player has been found .
[IV(Task scheduling manager)]: Launch the service cooperation......
[VA(Task scheduling manager)]: Ask the Circle_Manager to record the DVD driver.
[IV(Circle manager)]: The partner's record has been recorded.
```

Fig. 7: Service cooperation (TV)

delivery between a common device and an aggregator is shown in Fig. 6b and c. In Fig. 6d, the handover of IP space is illustrated. From these results, we can draw the conclusion that the approach adopted in AAHN is competent for both dynamic IP management and dynamic IP server.

Scenario 2: In this scenario, it demonstrates preparations for service cooperation between virtual devices. The virtual device VA is used to simulate a TV while VB to a DVD player. Here we assume that it is the first time that they cooperate with each other, that is, DVD driver is not in the cooperating circle of TV. Here the interactions between them are show in Fig. 7 and 8. This scenario is simplified properly because it is only used for the verification of the feasibility of approaches taken in the research. As shown in these Fig. 7 and 8, preparations for the service cooperation have been completed by

```
Habitat ID: 60000003-0000c3500355f555
Habitat name: wyllg

----------------------------------------

Restored 0 checkpointed agents and created 6 agents from habitat XML file.
[DVD(Request handler)]: I have received the remote service request.
[DVD(Request handler)]: Request the authenticator to verify the applicant.
[DVD(Partner authenticator)]: I have passed the authentication conducted by lg:/
/tryllian/agents/6000002a-064453c8a3ecc57d
[DVD(Partner authenticator)]: Authenticate: lg://tryllian/agents/6000002a-064453
c8a3ecc57d
[DVD(Partner authenticator)]: Authentication finished successfully. Notify Reque
st handler.
[DVD(Parameter negotiator)]: Negotiating.........
[DVD(Parameter negotiator)]: The negotiation has succeeded
[DVD(Request handler)]: Launch the service cooperation......
```

Fig. 8: Service cooperation (DVD)

two virtual devices automatically and the final cooperation has been implemented successfully.

CONCLUSION AND FUTURE DIRECTION

Autonomy is an effective approach to addressing many problems encountered in the research of intelligent home network. In AAHN, a lot of effort has been invested in it. Based on the Development Kit, ADK, a verifying system is developed in our research. Without assistances provided by other auxiliary devices or end users, virtual devices can build the network from scratch, find the service supplier they need, complete the relevant work for service cooperation and implement it finally. Additionally, some useful tests for special requirements for AAHN have been carried out, for example, the dynamic creation of IP server. The verifying result reveals the practicability of AAHN. Considering that household appliances become more and more powerful, our target is to develop an embedded system that can be integrated into a device. Correspondingly, there are two directions in next stage: one is the feasibility and implementation of embedded system and the other is the intelligent human-device interaction between devices and end users.

REFERENCES

Brumitt, B., J. Krumm, B. Meyers and S. Shafer, 2000. Ubiquitous computing and the role of geometry. Personal Commun. IEEE, 7: 41-43.

Charles, G.V., A. Bessam, G. Sylvain and M. Mounir, 2008. Toward autonomic pervasive computing. Proceedings of the 10th International Conference on Information Integration and Web-based Applications and Services, Nov. 24-26, Linz, Austria, ACM, pp: 673-676.

Chung-Ming, H., L. Ming-Sian and W. Hon-Long, 2008. Ubiquitous audio access in the UPNP home network. Proceedings of International Conference on Information Networking, Jan. 23-25, Busan, pp: 1-5.

Gerald, T., M.C. David, E.W. William, D. Rajarshi and S. Alla *et al*., 2004. A multi-agent systems approach to autonomic computing. Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems, Jul. 19-23, IEEE Computer Society, New York, pp: 464-471.

Goth, G., 2008. Redefining the server as home networks emerge. IEEE Internet Comput., 12: 7-9.

Hindriks, K., F.S. De Boer, W.V.D. Hoek and J.J. Ch. Meyer, 1999. Agent programming in 3APL. Auton. Agent Multi-Ag., 2: 357-401.

Jeong-Joon, Y. and L. Dong-Ik, 2003. Scalable home network interaction model based on mobile agents. Proceedings of the 1st IEEE International Conference on Pervasive Computing and Communications, Mar. 26-26, Fort Worth, TX, pp: 543-546.

Jung-Tae, K., P. Sangwook, L. Jong-Hoon, P. Eui-Hyun and P. Kwang-Roh, 2008. Provision of the multimedia service framework in the ubiquitous home network. IEEE Trans. Consumer Electr., 54: 501-506.

Merabti, M., P. Fergus, O. Abuelmaatti, H. Yu and C. Judice, 2008. Managing distributed networked appliances in home networks. Proc. IEEE, 96: 166-185.

Michael, Y.K.C. and C. Robin, 2005. A hybrid transfer of control model for adjustable autonomy multiagent systems. Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems, Jul. 25-29, The Netherlands, ACM, pp: 1149-1150.

Shoham, Y., 1993. Agent-oriented programming. Artif. Intel., 60: 51-92.

Soldatos, J., I. Pandis, K. Stamatis, L. Polymenakos and J.L. Crowley, 2007. Agent based middleware infrastructure for autonomous context-aware ubiquitous computing services. Comput. Commun., 30: 577-591.