New Technology Reports

Network-Assisted Mobile Terminal Support Technology

Daisuke Ochi, Kenichi Yamazaki and Satoshi Tanaka

The processing power of mobile terminals is greatly limited due to wireless links that disconnect under certain conditions of service interuption. This article introduces a mobile terminal support technology called "Twin Agents" that aims to solve this problem. In Twin Agents architecture, a proxy node representing the mobile terminal is placed on the network side to enable distributed processing and disconnected operations to be performed as needed.

1. Introduction

Mobile terminals are evolving rapidly in performance, capacity, and functions while broadband wireless access, reduced fees, and fixed-rate services continue to expand. Unimaginable applications and content in the past are now appearing one after another, and their enhancements and expansions are expected in the years to come. In short, mobile terminals are becoming capable of diverse applications, and under continuous-connection conditions, they should be able to not only receive services as a client but also to provide various types of services. We can foresee a mobile terminal used as an information-providing server in response to an external request and a group of mobile terminals connected in a Peer to Peer (P2P) mutually equal type, which should lead to new service formats.

Mobile terminals, however, are susceptible to suspended services since wireless links can disconnect under certain circumstances. In comparison to fixed nodes like personal computers, this greatly limits processing power and the types of services that can be provided. To overcome this instability of wireless links and to compensate for the relatively low processing power of mobile terminals, it is essential to improve terminal performance and wireless technology and upgrade facilities, but these measures in themselves would not completely make up for the limitations of mobile terminals. This article introduces Twin Agents [1] architecture and its programming model as a network-assisted mobile terminal support technology for offsetting

these limitations from the software side. In Twin Agents, a proxy node that acts for the mobile terminal is placed on the network side, and a program-execution environment is provided to enable cooperative operations between the proxy node and mobile terminal. Here, by introducing a mechanism for synchronous activation and execution of a proxy node program and a mobile terminal program, and a mechanism for adaptively changing proxy node and mobile terminal processing according to the state of the wireless link and that of other resources, a mobile terminal user can achieve program-load distributed processing and disconnected operations as needed.

The below presents a detailed description of Twin Agents architecture and an application example.

2. Requirements of Twin Agents Architecture

The characteristics of mobile terminals have the following limitations with regard to diversified service formats of the future.

- The very nature of mobile communications makes for unstable wireless links under certain conditions with disconnection occurring in the worst case scenario. It is also difficult to completely eliminate out-of-range areas.
- Because the processing power of mobile terminals is low compared to personal computers, the types of content and services that can be used are limited. Similarly, mobile terminals cannot be operated in server or P2P types without difficulty.

To overcome the above limitations, Twin Agents must satisfy the following requirements.

1) Retain Connection

To achieve stable communications, the application running on the mobile terminal and the correspondent node must be able to communicate even in the event of a wireless-link disconnection. Although communication is not actually possible while a wireless link is down, communication from an application standpoint can be maintained transparently by maintaining a session connection.

2) Perform Processing at Time of Disconnection

To minimize the effects of a disconnected wireless link and to continue the session, the mobile terminal and correspondent node must be able to receive alternate services through special processing at the time of a disconnection.

3) Achieve Service and Resource Independence

A mobile terminal must be able to receive and provide services independent of its processing power.

To satisfy the above requirements, Twin Agents architecture also makes use of non-mobile terminals to ensure retention of the communication session and continuity of services through disconnected operations, to substitute the processing power of mobile terminals.

3. Design of Twin Agents Architecture

To satisfy the requirements described in Chapter 2, Twin Agents places a proxy node on the network side dedicated to the mobile terminal in question and provides a mechanism for distributed processing between the proxy node and mobile terminal (**Figure 1**). In particular, the proxy node is placed between the highly stable core network and relatively unstable wireless network to provide a program execution environment and data storage function necessary for interacting with the mobile terminal. The mobile terminal, in turn, communicates with the correspondent node via the proxy node. The introduction of a proxy node in this way means that the proxy node can continue communications in place of the mobile terminal if the wireless link becomes disconnected. It also means that the proxy node can perform some of the mobile terminal's processing in a cooperative fashion to supplement the processing power of the mobile terminal.

Cooperation between the proxy node and mobile terminal is achieved by middleware installed on both sides. The middleware on the proxy node side is called Network side Agent (NsA) and that on the mobile terminal side is called User side Agent (UsA). This NsA and UsA configure a single virtual terminal that is basically recognized by the correspondent node as the communicating partner without being conscious of the NsA and UsA.

The NsA and UsA can each run a program. A program executing under such an environment is called a "Taplet," which can be programmed either personally or by an Application Service Provider (ASP), for example. For any one application, there are two Taplets, one inside the NsA and the other inside the UsA, which are created as a pair by a programmer. These Taplets can be installed and executed as desired by the mobile terminal user.

These two Taplets operate in a cooperative manner (cooper-

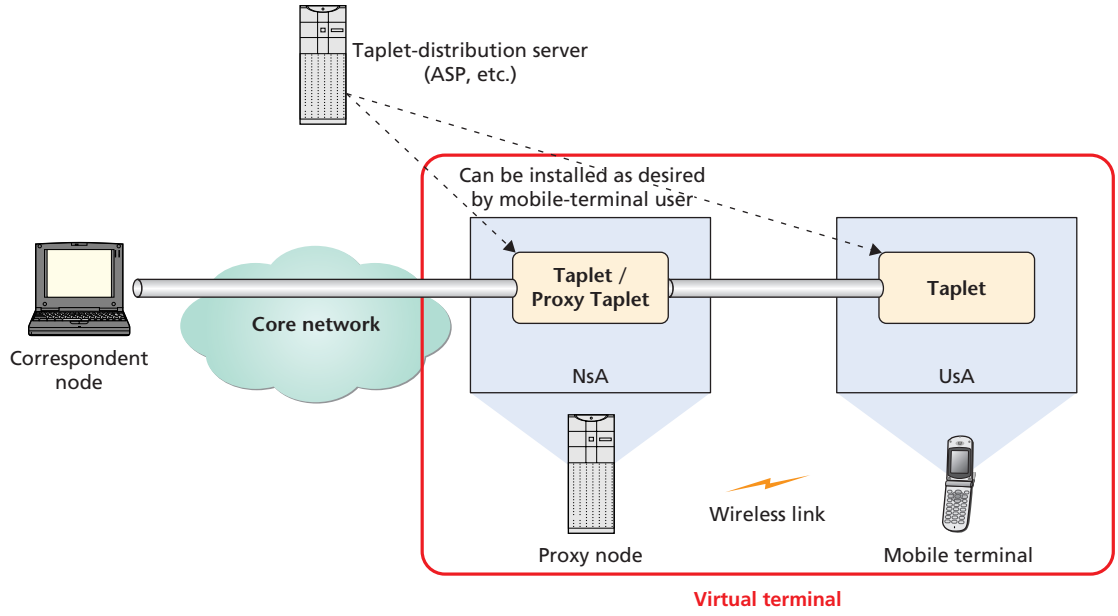

Figure 1 Twin Agents overview

ative mode) when a link is established between the NsA and UsA or in an independent manner (autonomous mode) when the link is disconnected. In other words, Taplet operation switches from cooperative mode to autonomous mode when a link becomes disconnected so that processing can continue in an alternative way for the mobile terminal user and the correspondent node. In this way, efforts to deal with the disconnection of an unstable wireless link can be isolated between the NsA and UsA, and the correspondent node can continue its session without having to worry about instability on the mobile terminal side. This scenario corresponds with requirements 1) and 2) in Chapter 2.

A Taplet can also be described to perform distributed processing using the storage facilities and the central processing unit (CPU) on the proxy node to supplement the processing power of the mobile terminal. This enables the correspondent node to communicate without concerning the processing power of the mobile terminal, which corresponds to requirement 3) in Chapter 2. Placing a server-type program in the NsA, moreover, enables the virtual terminal to operate like a server at the time of a link disconnection.

In addition to link state, Taplet operation can be switched in response to other mobile terminal resource states (such as remaining battery capacity).

Twin Agents is equipped with the features described above. Implementing these features in both the NsA and UsA eliminates the need to alter the correspondent node.

4. Technical Issues and Functions

The following technical issues must be addressed in achieving the Twin Agents scheme described in Chapter 3.

1) Means of synchronizing and executing the programs on the proxy node and mobile terminal.
2) Means of concealing disconnection of actual connection and disclosing it to the application if necessary without significantly changing the conventional socket interface.
3) Means of simplifying program description of proxy node NsA despite the fact that changing between cooperative mode and autonomous mode makes the program somewhat complicated.
4) Means of keeping NsA and UsA in a non-contradictory state in the event of a disconnection during data transfer.

The following describes each of the elements making up Twin Agents focusing on the above technical issues. **Figure 2** shows an overview of Twin Agents architecture.

1) Application Manager

Each of the two application managers installs a Taplet in its corresponding agent (NsA or UsA) and manages Taplet version number. They also manage Taplet execution and the simultaneous activation and termination of those two Taplets. To prevent conflict in activation state and version number and to solve technical issue 1) above, these application managers exchange Taplet management information held by the NsA and UsA.

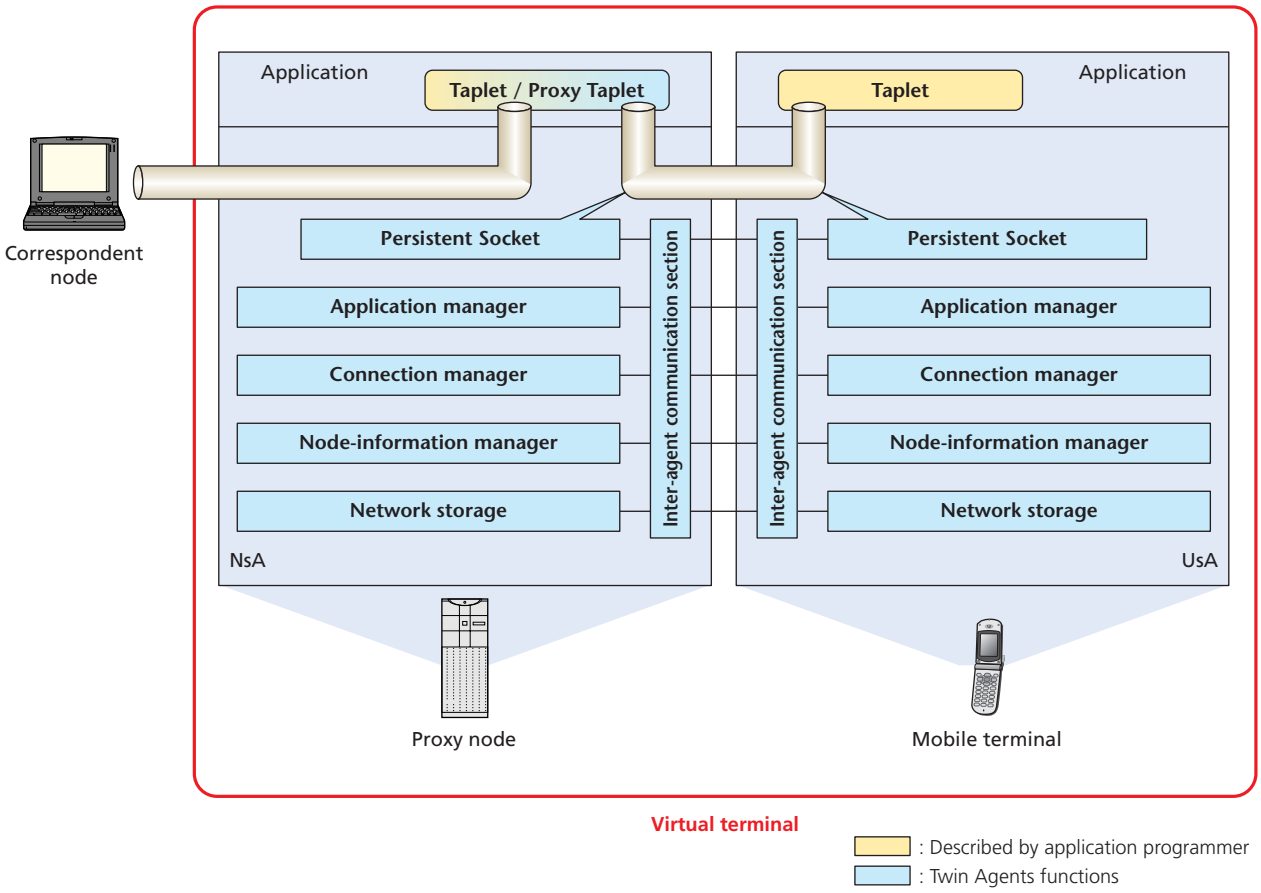

Figure 2 Twin Agents architecture

2) Connection Manager

The connection managers manage the creation and cancellation of the connection between NsA and UsA and the connection manager on the proxy node side manages that of the connection between NsA and the correspondent node. They also manage the Persistent Socket ID (PSID) described below.

3) Persistent Socket

The Persistent Socket can switch from cooperative to autonomous mode to support processing at the time of a disconnection. **Figure 3** shows an overview of the Persistent Socket. This is a communication socket created only between NsA and UsA. It automatically maintains a virtual connection between the two Taplets during link disconnection and reestablishes an actual connection between NsA and UsA when that becomes possible.

Twin Agents makes use of an original PSID to identify the connection between NsA and UsA. This identifier is independent of lower-level transmission protocol (such as Transmission Control Protocol/Internet Protocol (TCP/IP)), which enables Taplet connection to be removed from any change to link status such as a disconnection of the NsA/UsA link, change or nullification of the mobile terminal network interface, and change of the mobile terminal address.

Twin Agents also describes a cooperative and autonomous mode using a programming model prescribed by Persistent Socket, and enables switching of Taplet operation according to link status. And for each mode, it can separately describe processing in that mode (ongoing operation) and processing for switching that mode to the other mode (changing operation). This provides a clear distinction between a common and automated application section and an application-specific section, and simplifies the description of Taplet operation at the time of a disconnection. A “Persistent Socket,” moreover, is simply an extension of a general-purpose communication program module called a “socket,” which means that it can be treated in much the same way as an ordinary socket.

A Persistent Socket therefore solves technical issue 2).

4) Proxy Taplet

A proxy generally refers to functions for storing, converting, and transferring data at a node placed along a communication

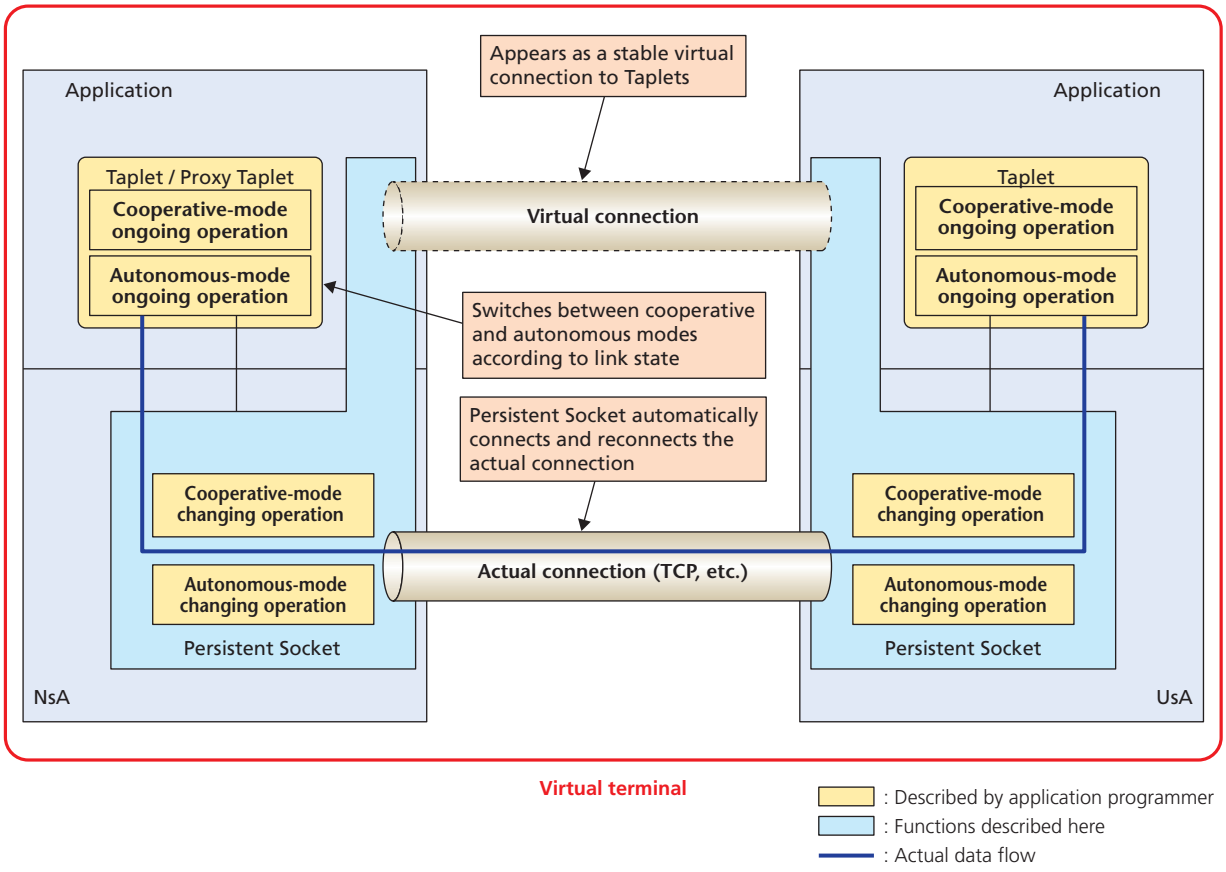

Figure 3 Overview of Persistent Socket

path. A Taplet in the NsA often has a proxy-like role with respect to the Taplet in the UsA. A Proxy Taplet provides common functions required by programs in the NsA for converting and transferring data, plus a description model for the section dependent on the application. These common functions are achieved by a program module equipped with common proxy functions as a Taplet extension.

Common functions include one for data exchange with the UsA and the correspondent node and one for assessing input data and processing it accordingly. Using and extending the Proxy Taplet in this way simplifies the use of such functions thereby solving technical issue 3).

Figure 4 shows an overview of the Proxy Taplet. Here, the UsA Taplet can maintain a communication session with the correspondent node by starting up the NsA Proxy Taplet or its program-module extension via a program module called a "Proxy Connector." A programmer can extend data transfer processing in the Proxy Taplet and data send/receive processing in the proxy connector as desired. This makes it possible to perform some type of encoding at the Proxy Taplet and corresponding decoding at the proxy connector.

Twin Agents also allows for the preparation of multiple Proxy Taplets in accordance with the level of functional implementation desired. For example, a "Unit Proxy Taplet" extends the Proxy Taplet to provide a general-purpose Proxy Taplet equipped with a function for guaranteeing fixed units of communication. A unit Proxy Taplet can therefore be used to solve technical issue 4). A programmer is free to select and extend Proxy Taplets for the purpose at hand.

5) Node-Information Manager

A node-information manager manages information on static and dynamic resources possessed by the proxy node or mobile terminal. Static-resource information includes CPU processing power, storage capacity, available network media, input/output capabilities (display resolution, input/output devices, etc.), and type of power supply. Dynamic-resource information includes CPU load, remaining storage capacity, state of communications (network load, radio intensity, communication mode, etc.), remaining battery capacity, mobile terminal location, and mobile terminal state (such as manner mode). The above information can be used to modify Taplet operation according to current conditions. For example, Taplet operation could be made to

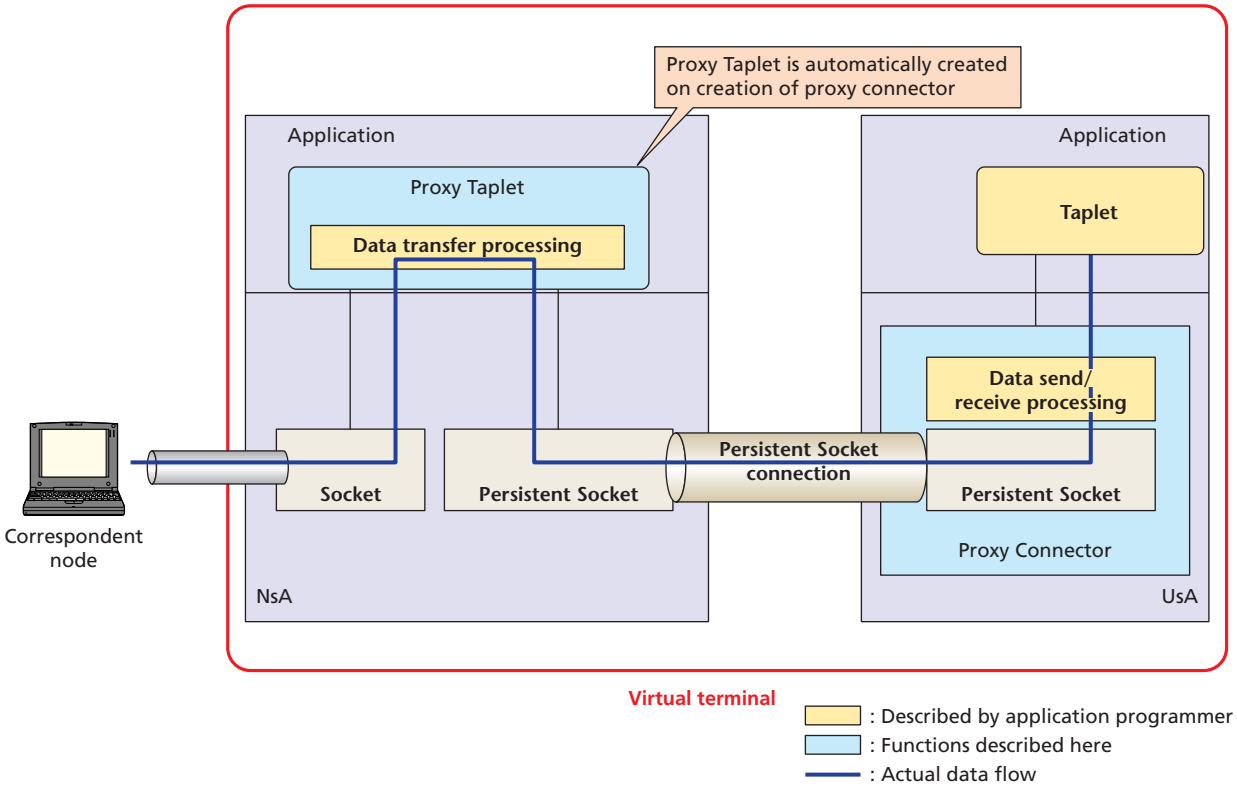

Figure 4 Overview of Proxy Taplet

vary according to the mobile terminal's remaining battery capacity.

6) Network Storage

Because the NsA and UsA form a single virtual terminal, there is a need to save common information. Twin Agents provides network storage that can be accessed by both NsA and UsA programs by a common access technique to maintain data consistency.

Twin Agents is configured by the above group of functions enabling the instability of wireless links to be overcome and providing a mechanism in the form of Taplets for offsetting the relatively low processing power of mobile terminals. While the above description assumes only one NsA in the proxy node, multiple NsAs may also be used.

5. Twin Agents Application Example and Evaluation

Since Twin Agents operates independently of any particular protocol, it is general enough to be used for a wide range of applications that involve disconnected operations and distributed processing. The tool group provided by Twin Agents, moreover, can accommodate low-level to high-level Application Program Interfaces (APIs) making it relatively easy to create complex processing. The following describes typical kinds of applications that can be achieved by Twin Agents and evaluates a video conference application as one specific example.

5.1 Types of Applications

As a platform that can supplement the relatively low processing power of mobile terminals, Twin Agents can do more than simply provide special processing at the time of a disconnection. Applications using Twin Agents can be classified into several application types as described below. Many applications targeted by Twin Agents are actually composites of these application types.

1) Disconnected Operation Type

Performs some sort of processing at a proxy node (such as a relay node) either during or in preparation for a disconnection of a mobile terminal link. This type can be further divided, such as into an application type that provides a proxy response (e.g., out-of-range message, redirection to Web) for the mobile terminal to the correspondent node; one that transfers communications to another node (e.g., call transfer service, mail transfer service); and one that saves communications (e.g., answering service, service for saving history of received calls).

2) Communication Recording Type

Stores the contents of a call between the mobile terminal and

correspondent node or saves node information (of the mobile terminal, for example) at a proxy node. Applications that have subsequent need for such stored history can be considered, such as an application that plays back a stored call at a later date.

3) Communication Conversion Type

Converts the communication contents passed between the mobile terminal and correspondent node into another format at a proxy node. This might consist of encryption, communication-path multiplexing, and media conversion, for example.

4) Substitute Processing Type

Performs a task that needs to be done by the mobile terminal at a proxy node to distribute mobile terminal load or reduce costs. Some examples are proxy authentication, proxy information collection, and proxy information dispatching. Here, if the mobile terminal is placed in charge of only the user interface (UI) section, it can be treated as a controller of the proxy node.

All of the above application types can be achieved using Twin Agents. As mentioned above, there are also applications that can combine these elemental application types. Video conference and online real-time games are two key examples. These are considered to be easily achieved using Twin Agents architecture.

5.2 Implementation of Video Conference Application

We created a video conference application for three parties to demonstrate the advantages of Twin Agents. **Figure 5** shows the network configuration of this application. The three parties are connected in a P2P scheme with one of them implemented on Twin Agents (on a virtual terminal). Here, correspondent nodes A and B correspond to conventional video conference applications that do not consider disconnected operations.

In this video conference application using Twin Agents, both the NsA and UsA are equipped with a Taplet. Twin Agents includes a Proxy Taplet that can store received data (Buffering Proxy Taplet), while the NsA Taplet here extends that Taplet to implement a proxy response function for times of disconnection.

This application can achieve the following operations that were difficult for conventional video conference applications.

1) Operation during link disconnection (autonomous-mode ongoing operation)
   - NsA maintains a connection with correspondent nodes A and B.
   - NsA stores all multimedia information (video, audio, text, etc.) sent by correspondent nodes A and B.
   - NsA sends proxy video to correspondent nodes A and B in place of the mobile terminal user's picture.
   - NsA performs a proxy response to inquiries (schedule, etc.) from correspondent nodes A and B in place of the mobile terminal user.
   - UsA stores multimedia information input by the mobile terminal user.
2) Operation at link restoration (cooperative-mode changing operation)
   - UsA and NsA notify each other of multimedia information that each has saved and immediately plays back that information to resynchronize.

The virtual terminal user and users of correspondent nodes A and B can therefore continue conferencing without losing any part of their three-way video conference.

In addition to the above disconnection assistance, Twin Agents can assist the mobile terminal even during the normal

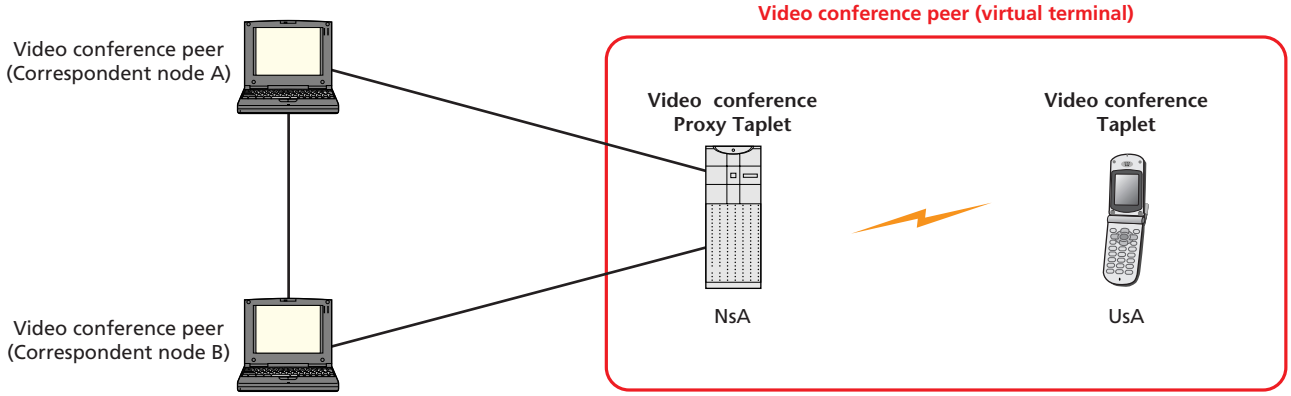

Figure 5 Network configuration of video conference application

operation. For example, video and audio sent by correspondent nodes A and B can be combined and compressed before transmitting to the mobile terminal, or that it can be converted to a data format supported by the mobile terminal. Transmitted video can even be recorded in real time. Since any media conversion here is performed at the NsA, correspondent nodes can transmit in general-purpose formats without having to worry about the functional limitations of the mobile terminal.

It can be seen from the above that Twin Agents can achieve at least the disconnected-operations, information-storage, and information-processing types of applications.

In this implementation of a video conference application, we found that the amount of coding required for the communication and proxy sections could be reduced by about 20% through by using a Persistent Socket and Proxy Taplet compared to processing described without them. We also found that coding could be reduced by about 30% for an IRC (Internet Relay Chat protocol) [2] client software that we created as another application for evaluation purposes.

The above applications demonstrate that a Persistent Socket and Proxy Taplet can be applied to various applications and that their use can simplify programming.

6. Related Works

Rover [3], Odyssey [4], and Mervlet [5] are tools for constructing mobile-transparency and mobile-recognition applications. Though their objective is similar to that of Twin Agents, they are end-to-end-type systems that require server-side support. They are also systems that move server-control process code to the client for processing and synchronizing purposes, which means that they basically target disconnected operations for server control only—they cannot provide disconnected operations for client control. Furthermore, the mobile terminal in these systems is assumed to operate as a client. In contrast, Twin Agents enables disconnected operations for client control enabling the mobile terminal to operate also as a server.

In addition, the World Wide Web (WWW) information-dispatching system [6] proposes a mechanism for stable provision of information even if the mobile terminal should be disconnect from the network. This information, however, is limited to Web-based information preventing flexible disconnected operations from being performed. Twin Agents is not limited to the Web—it can provide diversified services in a flexible and stable manner via a virtual terminal.

7. Conclusion

This article described Twin Agents architecture as a means of dealing with the instability of wireless links and the low processing power of mobile terminals in the provision and reception of services with mobile terminals. Twin Agents solves these problems by flexible disconnected operations in response to current link conditions and by distributed processing between a proxy node and mobile terminal. This architecture simplifies program description for distributed processing between the proxy node and mobile terminal by using a Persistent Socket and Proxy Taplet. The former conceals the effects of a link disconnection and allows description of disconnected operations, while the latter conceals the operation of basic proxy functions. Twin Agents architecture is applicable to not only mobile terminals but also to fixed nodes such as desktop personal computers that use dial-up connections.

In addition to providing assistance at the time of a link disconnection, Twin Agents can be used for other service types such as substitute processing, communication conversion, and communication recording. It enables a proxy node on the network to assist a mobile terminal and provide it with additional power thereby easing the resource limitations of the mobile terminal and making use of a mobile terminal more convenient for the user.

With the coming of a ubiquitous computing [7] environment, we plan to broaden our research to include information appliances and sensor networks in addition to virtual terminals and mobile terminals. We are to expand Twin Agents architecture to provide an adaptive application environment that can assist sensors and other devices that are even more powerless than mobile terminals.

References

[1] D. Ochi and K. Yamazaki: "Twin Agents: Network-assisted Disconnected Operation and Distributed Processing for Mobile Communication," Proc. of IEEE Symposium on Applications and the Internet 2004 (SAINT2004), Jan. 2004.

[2] J. Oikarinen and D. Reed: "Internet Relay Chat Protocol," RFC 1459, May 1993.

[3] A. D. Joseph, J. A. Tauber and M. F. Kaashoek: "Mobile computing with the Rover toolkit," IEEE Transactions on Computers, Special Issue on Mobile Computing, Vol. 46, No. 3, pp. 337–352, Mar. 1997.

[4] B. D. Noble, M. Satyanarayanan, D. Narayanan, J. E. Tilton, J. Flinn and K. R. Walker: "Agile Application-Aware Adaptation for Mobility," Proc. the 16th ACM Symposium on Operating System Principles (SOSP-16),

pp. 276–287, Oct. 1997.

[5] N. Islam, D. Zhou, S. Shahid, A. Ismael and S. Kizhakkiniyil: "AOE: A Mobile Operating Environment for Web-based Applications," Proc.of IEEE Symposium on Applications and the Internet 2004 (SAINT2004), Jan. 2004.

[6] S. Tagashira, et al.: "Adapting a Mobile Information Announcement System for Network Connectivity," Proc. 1999 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '99), Vol. 2, pp. 970–976, Jun. 1999.

[7] Mark Weiser: "The computer for the 21st century," Scientific American, pp. 94–104, Sep. 1991.

ABBREVIATIONS

API: Application Program Interface
ASP: Application Service Provider
CPU: Central Processing Unit
IRC: Internet Relay Chat protocol
NsA: Network side Agent
P2P: Peer to Peer
PSID: Persistent Socket ID
TCP/IP: Transmission Control Protocol/Internet Protocol
UI: User Interface
UsA: User side Agent
WWW: World Wide Web