Published in IET Communications
Received on 11th December 2008
Revised on 20th March 2009
doi: 10.1049/iet-com.2008.0710

ISSN 1751-8628

# Novel web agent framework to support seamless mobility for data networks

*Y.-J. Lee*[1] *D.-W. Lee*[2] *M. Atiquzzaman*[3]

[1]*Department of Technology Education, Korea National University of Education, San 7 Darak-Ri, Chungwon-Kun, Chungbuk 363-791, Korea*
[2]*Department of Computer Science, Woosong University, 17-2 Jayang-dong Dong-ku Daejeon, 300-718, Korea*
[3]*School of Computer Science, University of Oklahoma, 200 Felgar Street, Norman, OK 73019, USA*
*E-mail: lyj@knue.ac.kr*

**Abstract:** In this study, the authors address a mobile web agent framework based on stream control transmission protocol (SCTP) to overcome the deficiencies, which the typical mobile web agent framework based on transmission control protocol (TCP) suffers from, such as performance degradation, head-of-line blocking and unsupported mobility in mobile wireless environment. The proposed SCTP-based mobile web agent framework supports seamless transport layer mobility in a ubiquitous environment. It consists of an application engine and a transport engine, to use the hypertext transfer protocol and to deploy SCTP with dynamic address reconfiguration, respectively. The authors explore and describe the components necessary to implement the proposed mobile web agent framework in an ubiquitous environment. The performance of the proposed SCTP-based mobile web agent is compared with that of a typical TCP-based mobile web agent using ns-2 simulator. The simulation results show that the proposed web agent based on SCTP has a remarkably lower mean response time than a typical web agent based on TCP.

## 1 Introduction

Ubiquitous or pervasive computing environment consists of cheap, small and robust networked processing devices. Its network architecture is expected to have an IPv6 backbone network and an attached IPv4 Internet, that is, all-IP network. It provides users with customised, wired and wireless services while supporting host and user mobility. For example, while a user is roaming, portable web agents installed in personal digital assistants or net books can use predefined profile to download objects from web servers.

In the pervasive environment, a web agent uses hypertext transfer protocol (HTTP) for mobile users to retrieve objects from Internet. It is an application and connection-oriented protocol. Since most of IP traffic is transmitted using transmission control protocol (TCP), typical web agents use TCP as transport protocol. HTTP/1.0, however, does not support any means to request multiple objects. Therefore we must set up separate TCP connection for retrieving several objects from the server. In addition, it requires two extra round trip times (RTT) to establish a new TCP connection between the client and the server. Hence, HTTP/1.1 was introduced to enhance HTTP/1.0. It reduces the extra setup time with persistent connections and allows the client to send all requests simultaneously by using pipelining. However, even though we use any other enhanced HTTP versions, including HTTP/1.1, there is a gap between the requirements of HTTP and the functionality provided by TCP. That is, when multiple embedded objects are requested using HTTP, it is desirable that each object is reliably transferred by TCP. However, there is no requirement for ordered delivery of the objects in HTTP. Instead, it is more important to decrease the perceived latency of users. Most users are only concerned about the quick response time to retrieve web objects.

Typical mobile web agents based on TCP suffer from the following three deficiencies – performance degradation, head-of-line (HOL) blocking and unsupported mobility [1, 2]. IETF announced stream control transmission protocol (SCTP) [3] to overcome the defects of TCP.

SCTP relieves the performance degradation by incorporating the enhanced features of TCP congestion control schemes, such as fast retransmit algorithm and selective acknowledgement (SACK). It speeds up loss detection and increases bandwidth utilisation [4]. And it mandatorily uses the SACK. In addition to it, when the congestion window size (cwnd) is saturated, it can increase cwnd to improve the throughput [5].

SCTP's multi-streaming feature solves the HOL blocking problem of TCP. Since it transmits each web object in a separate stream, it eliminates the HOL effect between different objects. Even if one object is lost during the transfer, other objects can still be delivered to the web agent at the upper layer. And SCTP retransmits the lost object from the web server. It results in a better user response time and speeding up transfer of web objects.

Finally, SCTP's multi-homing feature can be utilised to resolve the unsupported mobility problem of TCP. It can allow a single SCTP endpoint to support multiple IP addresses. In its current form, the feature is only for redundancy. Recently, various extended SCTPs have been proposed [6–10]. The load-sharing SCTP [6] was proposed to aggregate the bandwidth of all the active transmission paths between the communicating endpoints. Stewart *et al.* [9] presented the extended SCTP multi-homing feature (called dynamic IP address reconfiguration) able to support transport layer mobility. This feature allows an SCTP endpoint to dynamically add and delete IP addresses during the lifetime of an SCTP association. Mobile SCTP [7] and transport layer seamless handover (SIGMA) [8] utilise the dynamic IP address reconfiguration feature and do not require any modification to the IP structure. Secure SCTP [10] tackles the traffic redirection attack problem raised from dynamic IP address reconfiguration.

While SCTP can overcome the above-mentioned deficiencies of TCP, it does not support the location management function that is intrinsically provided by Mobile IP. Hence, location management in SCTP can be performed by a domain name server (DNS) in the application layer [11] or the mobile IP in the network layer [7]. However, both approaches cause problems. That is, the scheme to use DNS and mobile IP cause scalability problem and, complexity and inefficiency, respectively. In this paper, therefore, we do not consider the location management problem assuming that a web agent always initiates the connection setup to web server.

We propose a SCTP-based mobile web agent framework with the dynamic IP address reconfiguration. The proposed web agent overcomes the above deficiencies of a typical TCP-based web agent and shows better performance. We explore and describe the components necessary to implement the proposed mobile web agent in a pervasive environment. Mean response time between HTTP requests and replies is the most important performance measure in a web environment. The performance of the proposed mobile web agent is compared with that of a typical TCP-based mobile web agent using ns-2 simulator. The results show that the mean response time of SCTP-based web agent is lower than that of the TCP-based web agent.

The earlier works of this paper are presented in [12, 13]. They, however, only deal with the simulation results in wired environment through real implementation. That is, they do not address the validation of mobility which is difficult to experiment in real environment. This paper extends the previous works to present the results in both wired and wireless environment. Sasikala *et al.* [14] shows only the comparison between HTTP over TCP and HTTP over SCTP using ns-2 simulation, but does not present the mobile web agent architecture, which this paper focuses on.

The main contributions of this paper are: (i) proposing an architecture for an SCTP-based mobile web agent, (ii) describing functions of the web agent components and (iii) demonstrating performance enhancement of the proposed web agent over a TCP-based web agent.

The rest of the paper is organised as follows. The proposed SCTP-based framework for mobile web agent is described in Section 2. Section 3 compares mean response time of SCTP and TCP-based web agents. Conclusions are given in Section 4.

## 2 Framework for the SCTP-based mobile web agent

The architecture of the proposed mobile web agent based on SCTP is depicted in Fig. 1. It mainly consists of an application engine and a transport engine based on SCTP, performing the function of user interface and supporting the transport mobility, respectively. There exists SCTP API as the interface between the two engines. The application engine interacts with web servers exchanging the HTTP message in the application layer. The transport engine consists of a mobility manager and a data manager in the transport layer. The mobility manager consists of a movement handler and a handover handler. The data manager includes an event handler, a transmission control block (TCB) and a common database (CDB) to maintain the information related to the current association and mobility support.

### 2.1 Application engine

The application engine provides users with the HTTP services and interacts with the transport engine supporting SCTP. Using the multi-streaming feature of SCTP, it prevents HOL blocking occurred in TCP-based applications and reduces the response time of an application. A typical web page usually contains several multimedia objects, such as image, voice and text. If any



*IET Commun.*, 2009, Vol. 3, Iss. 12, pp. 1861–1869
doi: 10.1049/iet-com.2008.0710

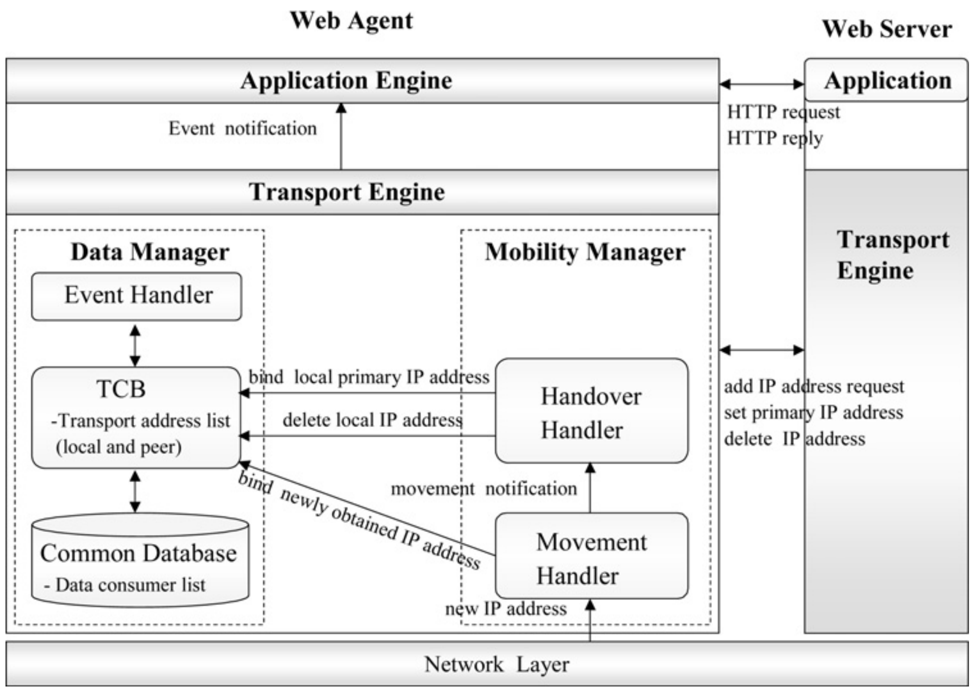


**Figure 1** *Architecture for the SCTP-based mobile web agent*

object is lost during receiving, the TCP-based receiver does not display the previously received object until completely receiving the lost object. Because of the HOL blocking, the response time of the TCP-based application increases especially in wireless networks with high packet loss. The SCTP-based application engine, however, can avoid the HOL blocking and display objects more promptly.

In addition, the proposed SCTP-based web agent has the following additional advantages comparing with a TCP-based web agent:

1. *Increasing the security of an application remarkably:* The TCP-based web agent is defenseless against blind SYN attacks because of its three-way-handshake connection setup. On the other hand, the proposed SCTP-based web agent can avoid the attack using the signed state cookie in its four-way-handshake connection setup. Even though it uses four packets for connection establishment, it combines the HTTP request into the third packet. Therefore it does not have extra overhead compared with the TCP-based web agent.

2. *Enhancing the fault tolerance of an application:* Since a TCP-based web agent uses only one connection path, if the TCP path is broken due to the physical layer problems, data cannot be transferred. However, the proposed agent can continue to communicate using the alternate path provided by the SCTP multi-homing feature.

3. *Supporting the seamless mobility for an application:* An existing TCP-based web agent suffers from the disconnection because it cannot modify the currently bound IP address in TCB into the new IP address. However, the proposed web agent keeps maintaining the seamless connection with web server because it can change the TCB using the dynamic address reconfiguration of SCTP.

The application programmers, who are already familiar with TCP and UDP, quickly adapt to SCTP. By using the SCTP API, they can easily transform a typical TCP application into the SCTP application. For example, the TCP application with socket(AF_INET, SOCK_STREAM, IPPROTO_TCP) as the socket system call can be transformed to the SCTP application with socket(AF_INET, SOCK_STREAM, IPPROTO_SCTP). For address binding they can use system calls, bind() and sctp_bindx(). Several predefined events such as the new address acquisition are delivered to the application engine through SCTP event notification. The application engine can obtain the status information such as contents and addresses of events by using sctp_opt_info(). To enhance flexibility for application programmers, the implementation of SCTP including the linux kernel SCTP (lksctp) [15] should contain SCTP API to specify some parameters peculiar to SCTP, such as the number of outgoing streams to set up during negotiation, stream IDs used etc.

## 2.2 Transport engine

In this section, we describe in detail the various components of the transport engine to support transport mobility. The transport engine consists of a mobility manager and a data manager in the transport layer.

*2.2.1 Data manager:* The data manager consists of an event handler, a TCB, and a CDB. It defines necessary parameters to implement the SCTP protocol. The CDB maintains information of SCTP instance: (i) Data consumer list for the currently connected association. The data consumer means the information to identify process, such as file descriptor, pipe pointer and table pointer. (ii) Secret key for the end-user's security. (iii) Address list for end points. (iv) Port number for the bound port of end point.

The TCB contains some important parameters for each association: (i) peer verification tag for the authentication value of the corresponding node, (ii) my verification tag for the authentication value of local node, (iii) state for the current status of SCTP, that is, connection complete, shutdown and stop, (iv) Peer transport address list for the transport addresses of the corresponding node, (v) local transport address list for the local IP addresses and (vi) primary path for the primary destination address of the corresponding node.

The event handler delivers events, such as change of IP address or modification of transport address list in the TCB, to the application engine.

*2.2.2 Mobility manager:* The mobility manager consists of a movement handler and a handover handler. It provides a web agent with the seamless transport mobility using the SCTP address configuration change (ASCONF) extension [9]. ASCONF extension describes the new IP address insertion (add_ip_address), the old IP address deletion (delete_ip_address) and primary IP address change (set_primary_address) chunks. Receiving the above chunks, data manager dynamically modifies the peer transport address list, local transport address list and primary address in the TCB.

Generally, a mobility manager in the wireless mobile network contains functions of movement detection, handover management and location management. The movement handler supports movement detection for a mobile node (MN) to identify and trace its own location change. The role of location management enables correspondent node (CN) to trace the current location of MN in order to initiate the connection establishment with MN. The handover handler supports handover management for both MN and CN to provide the roaming MN with the seamless handover.

In this paper, we assume that MN (web agent) always initiates connection setup to CN (web server) in the web environment. Therefore the location management is not needed for CN. Nevertheless, if the location management is needed, it can be added into the mobility manager in Fig. 1. The procedure of mobility manager is shown in Fig. 2. Timeline of the mobility manager is described in Fig. 3.

1. *Movement handler:* Initially, a mobile web agent probes old access router (AR) to hear router advertisement (RA) and

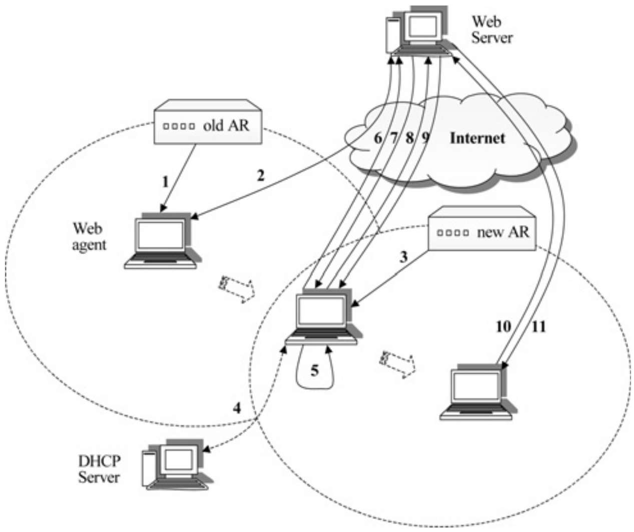


**Figure 2** *Procedure of the mobility manager*

finds the network prefix included in RA (1, Figs. 2 and 3). The web agent takes its own IP address by using stateless auto-configuration of IPv6 based on the network prefix (when using IPv6) or asking to dynamic host configuration protocol (DHCPv4/v6) server (when using IPv4/IPv6). The web agent and a web server exchange IP address to set up the association. At this time, each end point identifies the primary IP address on which data are sent (2, Figs. 2 and 3). The information of the association is written in each TCB of a web agent and a web server, followed by exchanging data.

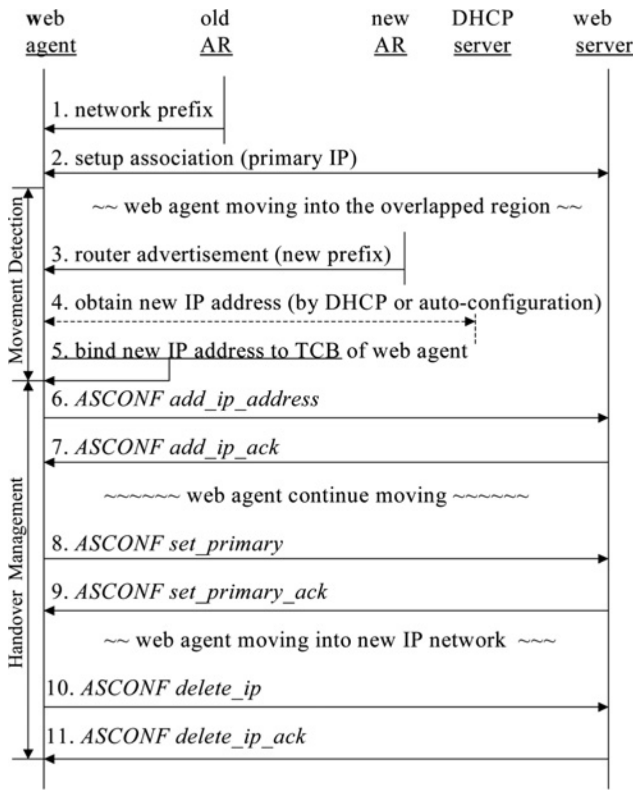


**Figure 3** *Timeline of the mobility manager*




doi: 10.1049/iet-com.2008.0710

While communicating with a web server, the web agent moves from the coverage of old AR to the overlapped region which is managed by both old AR and new AR. The web agent acquires new router prefix from new AR (3, Figs. 2 and 3), and recognises its movement into new network by comparing its current network prefix (1, Figs. 2 and 3) with new network prefix (3, Figs. 2 and 3). If using IPv6, the web agent can itself configure new IP address using stateless auto-configuration based on the network prefix. Otherwise, it can obtain a new IP address from the DHCPv4/v6 server (4, Figs. 2 and 3), which increases the required signalling time. At any rate, the newly acquired IP address is bound on the local transport address list in the TCB of a web agent (5, Figs. 2 and 3). The event handler delivers these events to the application engine as SCTP notification.

2. *Handover handler:* After binding the new IP address on TCB, the web agent notifies the web server of that it is going to use the new IP address by sending ASCONF add_ip_address (6, Figs. 2 and 3). The web server adjusts its own TCB by adding the received new IP address of the web agent and responds to the web agent by sending an ASCONF add_ip_ack (7, Figs. 2 and 3). At this moment, the web agent changes to multi-homed and is thus reachable in two different networks. It can obtain data from both old and new IP addresses. Therefore if there is a physical problem with the path of the primary address, the new IP address can become an alternate address.

As a web agent departs from the overlapped region and comes in the coverage of new AR, it encounters more packet loss on the primary path. If the amount of obtained packets from new IP address becomes greater than from the primary IP address, the web agent forwards the ASCONF set_primary chunk to the web server. When receiving the chunk, the web server uses the new IP address as primary address for data communications (8, Figs. 2 and 3). The web server responds to web agent (9, Figs. 2 and 3) by sending the ASCONF set_primary_ ack chunk.

As the web agent continuously moves into the core coverage of new AR, the previous primary IP address becomes useless. The web agent forwards the ASCONF delete_IP chuck to the web server, which deletes the previous primary IP address (10, Figs. 2 and 3). The reason to delete the useless IP address is as follows: it is assumed that the newly set primary path is broken in the coverage of new AR. If we did not remove the previous primary IP address from the binding list, it might become an alternate path. Therefore the data from the web server may be redirected to the alternate path. However, the previous primary IP address cannot obtain any data from the coverage of new AR. Thus, there is unnecessary traffic in the network. Handover is completed when the web server replies to the web agent by sending ASCONF delete_ip_ack (11, Figs. 2 and 3).

# 3 Performance evaluation

In this section, we compare the performance of TCP-based web agent framework with SCTP-based web agent framework in terms of mean response time.

## 3.1 Mean response time for TCP-based and SCTP-based framework

In the TCP-based framework, response time is defined as the time taken between the initial request from the client to the server, and the completion of the response from the server to the client. After a three-way handshake, the client receives the HTML file with $M$ embedded objects. The client, supporting persistent connections and pipelines, then sends its requests, that is the client sends $M$ requests simultaneously using pipelining. The server sends its responses to those requests in the same order that the requests were received.

In the SCTP-based framework, the initialisation of a SCTP association is completed after the exchange of four messages. The passive side of the association does not allocate resources for the association until the third of these messages has arrived and been validated. Last two messages of the four-way handshake can already carry user data. With this piggybacking, SCTP has the same connection-establishment delay as TCP, namely one RTT. Since SCTP has multi-streaming feature, it avoids the HOL blocking. Furthermore, SCTP does not limit maximum number of objects for pipelining. To summarise, main difference between TCP-based framework and SCTP-based framework is that SCTP can receive multiple objects in parallel by using its own multi-streaming feature.

In Sections 3.2 and 3.3, we describe the performance comparison of TCP-based framework and SCTP-based framework in wired and wireless environment. In both environments, we have used ns-2 discrete event simulator version 2.29 [16] for the performance comparison of the frameworks. A script language OTcl is used to glue the network components (nodes, links, agents, applications etc) provided by ns-2 simulator, configure the parameters (bandwidth, delay, routing protocol etc) and launch activities (data transfer, topology change etc). We have used TCP and SCTP as the transport protocol for the HTTP traffic in ns-2 simulation.

## 3.2 Wired environment

The simulation topology for wired environment is shown in Fig. 4. In the topology node 0 represents web client and node 1 represents web server. Fig. 5 depicts the web client and web server in ns-2 simulation. In simulation model, the client initiates sessions with the web server. The results were obtained from the trace file generated in the simulation.

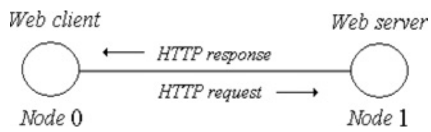


**Figure 4** *Simulation topology for wired environment*

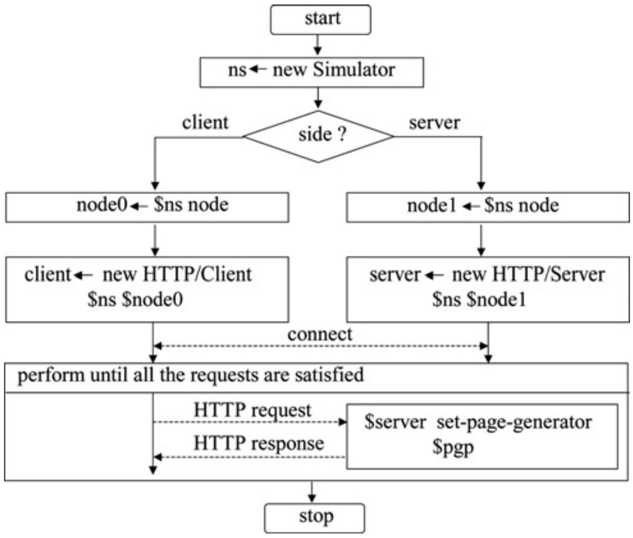


**Figure 5** *Web client and web server in ns-2 simulation*

The test parameters such as object size, number of objects, bandwidth and RTT are given in Table 1.

The default parameters we have used in our simulation are object size of 13.5 kB and maximum segment size of 536 B. The plots of packet loss rate against mean response time for TCP-based framework as well as SCTP-based framework is presented in Fig. 6. The packet loss rate was varied from 0.4 to 5% and the mean response time was observed. It can be seen from Fig. 6 that for both frameworks, the mean response time increases as the packet loss rate increases. The increase in packet loss rate tends to HOL blocking problem when TCP is used for web transaction and this in turn increases the response time. This can be eliminated by SCTP's multi-streaming feature, which allows multiple streams in an association.

The effect of bandwidth on mean response time for both frameworks is presented in Fig. 7. The figure shows the mean response time is inversely proportional to the

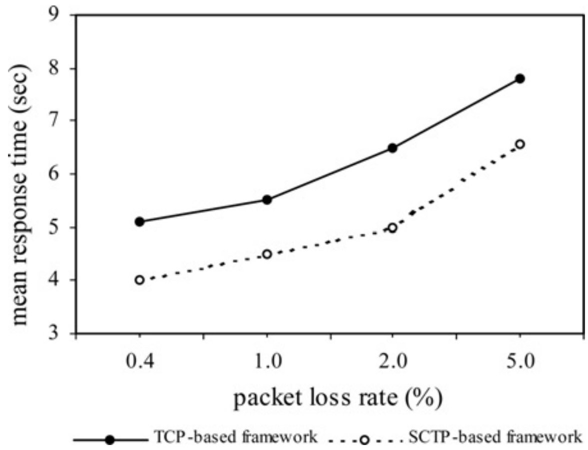


**Figure 6** *The effect of packet loss rate on mean response time*

bandwidth. As the bandwidth increases, it reduces the transfer time, as well as the mean response time. This also means to reduce the transaction time for web objects.

Fig. 8 shows the mean response time increases as the RTT increases. SCTP-based framework has better performance because of its multi-streaming feature. The RTT is varied from 55 to 1000 ms and the mean response time is observed. As TCP has no multi-streaming feature, the web objects require extra RTT. This increases the mean response time of TCP-based framework than SCTP-based framework. Long RTTs are becoming prevalent with the introduction of satellite links into the Internet. A long

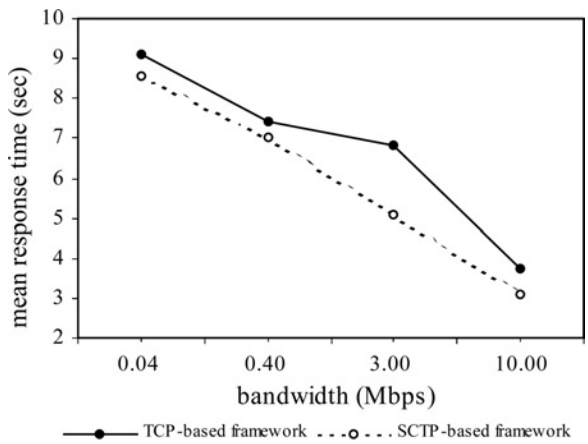


**Figure 7** *The effect of bandwidth on mean response time*

**Table 1** Test parameters in wired environment

| Performance measure | Simulation parameters | Values |
|---|---|---|
| mean response time (s) | packet loss rate (%) | 0.4, 1, 2, 5 |
| | bandwidth (Mbps) | 0.04, 0.4, 3, 10 |
| | round trip time (ms) | 55, 80, 256, 1000 |
| | number of objetcs | 3, 5, 10, 20 |



doi: 10.1049/iet-com.2008.0710

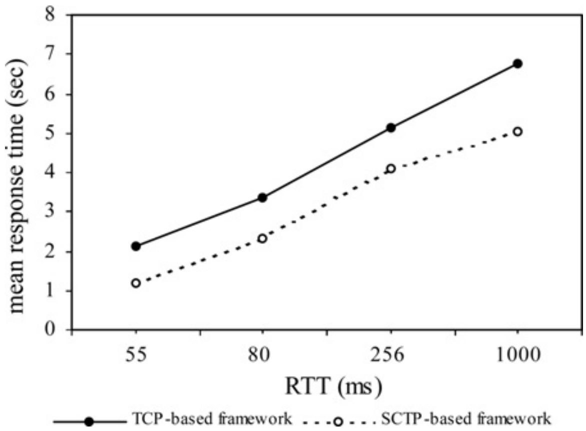


**Figure 8** *Mean response time for varying RTT*

RTT reduces the rate at which the window increases, which is a function of the number of ACKs received and does not account for the RTT. This poses many problems to TCP. First, it increases the duration of slow-start, which is a transitory phase designed to quickly but smoothly fills the pipe. Given the low bandwidth utilisation during slow-start, this deteriorates the performance of TCP transfers, particularly short ones (e.g. Web transfers). Second, it causes unfairness in the allocation of the bottleneck bandwidth. Small RTT connections increase their rates more quickly and grab most of the available bandwidth.

Fig. 9 shows the increase in mean response time as the number of objects increases. As the number of object increases, if any of the objects fails, the number of queued up objects increases until the object is retrieved. This causes the increase in transfer time. SCTP-based framework has better performance compared with TCP-based framework because of its multi-streaming feature.

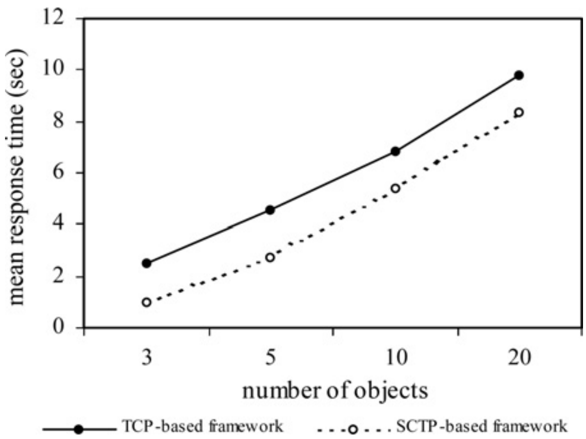


**Figure 9** *Mean response time for varying the number of objects*

Table 2 shows the average value of simulation results in wired environment. It is shown that mean response times of SCTP-based framework are less than those of TCP-based framework. The reason is why the multi-streaming feature of SCTP-based framework reduces mean response time.

## *3.3 Wireless environment*

The wireless simulation model and topology is shown in Fig. 10. The CN is located in the wired network and an MN is moving between two wireless stations. The default parameters are object size of 13.5 kB and maximum segment size of 536 B. In Fig. 10, node 0 represents the CN(web server) and nodes 1 and 2 represent the MN(web client). The arrowhead shows the MN movement from base station1 to base station 2.

Table 3 shows test parameters – the moving speed of MN and the overlapped region size that affect on mean response time and mean packet loss. In order to view the impact of moving speed, the speed of MN is varied from 5 to 20 m/s. The overlapped region size is varied from 1 to 15 m and the effect on mean response time and mean packet loss is observed.

Figs. 11 and 12 show the effect of moving speed on mean packet loss and mean response time, respectively. From the

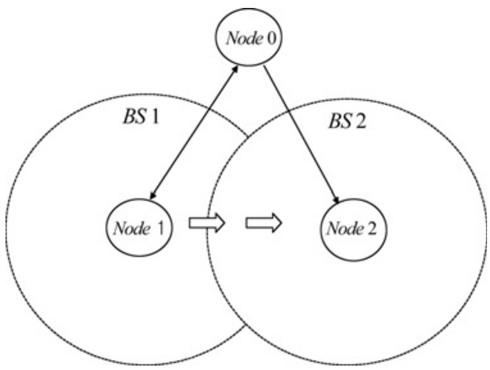


**Figure 10** *Simulation topology for wireless environment*

**Table 2** Average value of simulation results in wired environment

| Performance measure | Simulation parameters | TCP-based framework | SCTP-based framework |
|---|---|---|---|
| mean response time (s) | packet loss rate | 6.23 | 5.02 |
| | bandwidth | 6.76 | 5.94 |
| | round trip time | 4.34 | 3.16 |
| | number of objects | 5.91 | 4.38 |

 

**Table 3** Test parameters in wireless environment

| Performance measures | Simulation parameters | Values |
|---|---|---|
| mean response time (s) | moving speed (m/s) | 5, 10, 12.5, 15, 20 |
| mean packet loss (packets) | region size (m) | 1, 5, 10, 12.5, 15 |
| | moving speed (m/s) | 5, 10, 12.5, 15, 20 |
| | region size (m) | 1, 5, 10, 12.5, 15 |

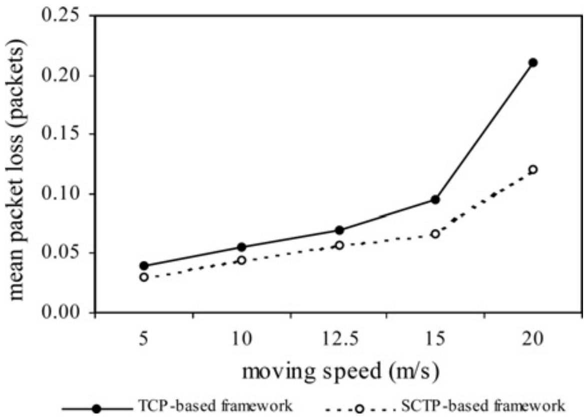


**Figure 11** *Effect of moving speed on mean packet loss*

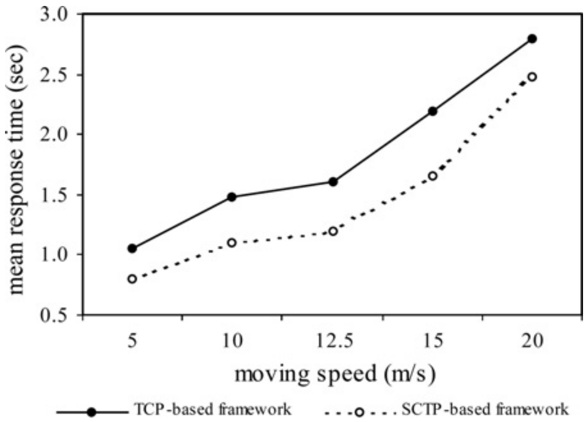


**Figure 12** *Effect of moving speed on mean response time*

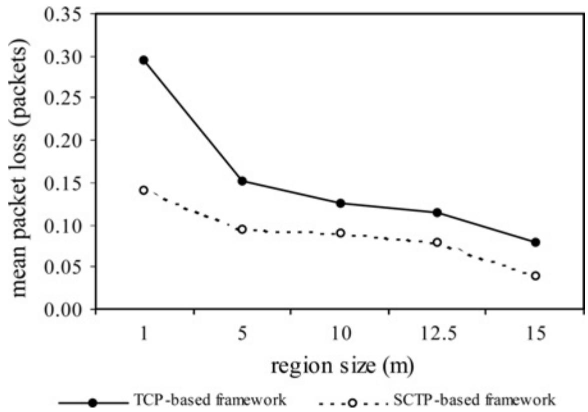


**Figure 13** *Effect of region size on mean packet loss*

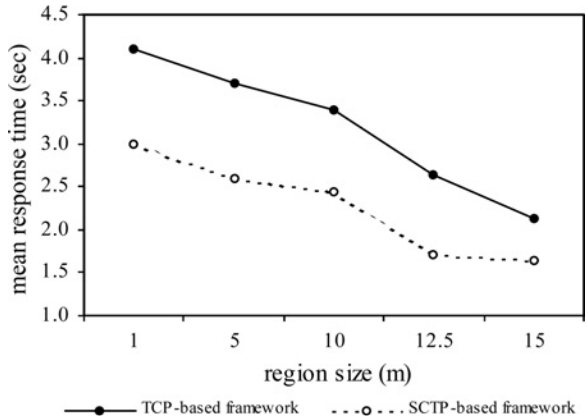


**Figure 14** *Effect of region size on mean response time*

figures, it can be seen that SCTP has better performance than TCP. The mean packet loss increases as the moving speed increases and hence increases the mean response time.

Figs. 13 and 14 show the effect of overlapped region size on mean packet loss and mean response time, respectively. The mean packet loss decreases as the overlapped region size increases and hence decreases the mean response time.

**Table 4** Average value of simulation results in wireless environment

| Performance measures | Simulation parameters | TCP-based framework | SCTP-based framework |
|---|---|---|---|
| mean response time (s) | Moving speed | 2.02 | 1.61 |
| mean packet loss (packets) | region size | 2.97 | 2.10 |
| | moving speed | 0.09 | 0.06 |
| | region size | 0.12 | 0.08 |



doi: 10.1049/iet-com.2008.0710

Figures show that SCTP-based framework has better performance than TCP-based framework.

Table 4 shows the average value of simulation results in wireless environment. Mean response time and mean packet loss of SCTP-based framework are less than those of TCP-based framework. The reason is why the handover latency of SCTP-based framework with lower packet loss is less than that of TCP-based framework.

## 4 Conclusions

In this paper, we have proposed a SCTP-based mobile web agent framework to support the transport layer mobility. The proposed transport layer-based framework overcomes many of limitations of typical network layer-based frameworks. To demonstrate the efficiency of our mobile web agent framework, we have carried out experiments using ns-2 simulator in wired and wireless environment. Results show that our SCTP-based web agent framework reduces the mean response time when compared with a TCP-based web agent framework. Future work includes performance evaluation of the implementation-based framework of SCTP mobile web agent in a real wireless Internet environment.

## 5 Acknowledgment

This work was supported by the 2009 research grant from Korea National University of Education.

## 6 References

[1] BAI H., FU S., ATIQUZZAMAN M.: 'Transport layer design in mobile wireless network. Design and analysis of wireless networks' (Nova Science Publishers, 2004)

[2] JAMALIPOUR A.: 'The wireless mobile internet' (John Wiely & Sons Ltd, USA, 2003)

[3] FU S., ATIQUZZAMAN M.: 'SCTP: State of the art in research, products, and challenges', *IEEE Commun. Mag.*, 2004, **42**, pp. 64–76

[4] CARO A., IYENGAR J., AMER P., LADHA S., HEINZ G., SHAH K.: 'SCTP: a proposed standard for robust internet data transport', *IEEE Comput.*, 2003, **36**, pp. 20–27

[5] ALAMGIR M., ATIQUZZAMAN M., IVANCIC W.: 'Effect of congestion control on the performance of TCP and SCTP over satellite network' (NASA Earth Science Technology, 2002)

[6] AL A., SAADAWI T., LEE M.: 'Improving throughput and reliability in mobile wireless networks via transport layer bandwidth aggregation', *Comput. Netw.*, 2004, **46**, pp. 635–649

[7] KOH S., CHANG M., LEE M.: 'mSCTP for soft handover in transport layer', *IEEE Commun. Lett.*, 2004, **8**, pp. 189–191

[8] SIVAGURUNATHAN S., JONES J., ATIQUZZAMAN M., FU S., LEE Y.: 'Experimental comparison of handoff performance of SIGMA and mobile IP'. IEEE Workshop High Performance Switching and Routing, HongKong, China, 2005

[9] STEWART R., XIE Q., TUEXEN M., MARUYAMA S., KOZUKA M.: 'Stream control transmission protocol (SCTP) dynamic address reconfiguration'. IETF Internet draft, draft-ietf-tsvwg-addip-sctp-06.txt, 2003

[10] UNURKHAAN E., PATHGEB E., JUNGMAIER A.: 'Secure SCTP – a versatile secure transport protocol', *Telecommun. Syst.*, 2004, **27**, pp. 273–296

[11] FU S., MA L., ATIQUZZAMAN M., LEE Y.: 'Architecture and performance of SIGMA: a seamless mobility architecture for data networks'. IEEE Int. Conf. Communications, Seoul, Korea, 2005

[12] LEE Y., HAM H., ATIQUZZAMAN M.: 'SCTP-based framework for mobile web agent'. Proc. 20th Int. Symp. Computer and Information Sciences (ISCIS), Istanbul, Turkey, 2005

[13] LEE Y., ATIQUZZAMAN M.: 'Web agent supporting transport layer mobility'. Proc. 2nd Int. Workshop on Mobility Aware Technologies and Applications (MATA), Canada, 2005

[14] SASIKALA P., SEO T., LEE Y.: 'Performance comparison of TCP and SCTP in wired and wireless internet environment', *J. Korean Inst. Ind. Educators*, 2008, **33**, pp. 287–299

[15] Linux Kernel Stream Control Transmission Protocol (lksctp): http://lksctp.sourceforge.net/

[16] ns-2 Network Simulator: http://www.isi.edu/nsnam/ns