소프트웨어 임대 서비스를 위한 사용 요금 계산 기법

Charge Calculation Scheme for Software Rental Service

주 한 규*
Hankyu Joo

요 약

상용의 소프트웨어를 사용하기 위해서 대부분의 소프트웨어 사용자는 그 소프트웨어를 구입하여야한다. 하지만 해당 소프트웨어에 대한 사용 시간이 길지 않은 경우, 그 소프트웨어 비용은 과다한 지출로 인식된다. 소프트웨어 임대 서비스는 이러한 경우 효과적인 대안일 수 있다.

소프트웨어 임대를 지원하기 위해서는 요금 부과 기법이 필요하다. 두 종류의 요금 부과 기법이 생각될 수 있다. 하나는 특정 기간 소프트웨어 사용에 대하여 일정한 요금을 부과하는 것이며 다른 하나는 실제 사용한 시간에 따라 요금을 부과하는 것이다.

이 논문에서는 소프트웨어의 실제 사용 시간에 따라 요금을 부과하는 기법을 제안한다. 소프트웨어의 실제 사용 시간에 따라 요금을 부과하기 위하여 사용 시간을 정확하게 측정할 수 있는 기법을 고안하였다.

Abstract

To use commercial software, most software users purchase the software. Some software users, who do not use the software frequently, regard purchasing the software as undue expense. Software rental service can be an effective substitute.

To support the software rental service, charging scheme is necessary. Two categories of charging scheme can be considered. One is charging a fixed amount of fee for a fixed period of time and the other is charging a fee based on the actual usage time. In this paper, the software pay-per-use approach based on the amount of time that the software user has used is proposed. The proposed approach gives the capability to calculate the usage time.

☞ keyword : 소프트웨어 임대 서비스, 전자 상거래, 요금 계산, 부인 방지, Software Rental Service, Electronic Commerce, Charge Calculation, Non-Repudiation

1. 서 론

대부분의 소프트웨어 사용자는 상용의 소프트웨어를 사용하기 위해서는 그 소프트웨어를 구입하여 사용한다. 많은 소프트웨어는 상당히 비싸다. 특히 사용자가 구입한 소프트웨어에 대한 사용 시간이 많지 않은 경우, 사용자는 그 소프트웨어가 과도하게 비싸다고 인식하게 된다. 이러한 경우 필요한 소프트웨어를 사용하지 못하거나 경우에 따라 소프트웨어의 불법 복제 및 사용이 발생하게 된다.

이러한 문제점을 해결 방안으로 소프트웨어 임대를 생각할 수 있다. 소프트웨어 사용자는 소프트웨어를 구입하는 대신 소프트웨어 제공자로부터 소프트웨어를 임대하여 사용할 수 있을 것이다. 소프트웨어 제공자는 소프트웨어를 대여하여 줄 수 있는 자격을 가진 사업자로 소프트웨어 개발자로부터 소프트웨어를 대여하여줄 수 있는 권리를 획득한 사업자이다.

소프트웨어 임대 서비스 필요성은 꾸준히 제기되었으며ASP(Application Service Provider)[1, 2, 3]와 SaaS(Software as a Service)[4] 등의 임대 서비스 개념이 도입되어 사용되고 있다. 또한 Salesforce.com [5]과 BizmekaKanOffice[6] 등과 같은 상용의 소프트웨어 임대 서비스 기관 존재하여 소프트웨

* 종신회원 : 한림대학교 정보통신공학부 교수
hkjoo@hallym.ac.kr
[2007/11/22 투고 - 2007/12/20 심사 - 2008/01/21 심사완료]

어 임대 서비스를 제공하고 있다. ASP와 SaaS는 네트워크를 통하여 사용자의 요청된 작업을 서비스 제공자의 컴퓨터에서 수행한다. 사용자의 작업이 서비스 제공자의 컴퓨터에서 수행되므로 사용자가 자신의 자료를 서비스 제공자에게 전송하고 그곳에서 작업을 수행하고 저장하므로 이에 따른 정보보호의 문제점이 있다[3].

최근 고속 네트워크의 발전으로 소프트웨어 사용자가 소프트웨어 제공자로부터 소프트웨어를 다운로드 받는 것은 많은 비용을 필요로 하지 않는다. 소프트웨어 사용자는 소프트웨어를 사용자의 컴퓨터에 다운로드 받아서 자신의 컴퓨터에서 수행할 수 있다. 이 경우 자신의 자료를 제공자의 컴퓨터에 넘겨주지 않으므로 정보보호의 문제점을 경감할 수 있다. 소프트웨어 제공자는 사용자에게 요금을 부과하는 방법을 사용하여 소프트웨어 임대를 지원할 수 있다.

소프트웨어 임대 체계를 위하여서는 요금 부과 기법이 필요하다. 두 종류의 요금 부과 기법이 생각될 수 있다. 하나는 정해진 기간에 일정 요금을 부과하는 것이며 다른 하나는 실제 사용 시간에 따라 요금을 부과하는 것이다. 정해진 기간에 일정 요금을 부과하는 방식은 이미 상용으로 서비스되고 있다[5, 6]. 이 기법은 구현하기 간단하나 해당 소프트웨어를 조금만 사용하는 사용자의 경우도 동일한 요금을 지불해야 하는 문제가 있다. 실제 사용 시간에 따라 요금을 부과하는 기법은 복잡한 방법이나 사용 시간에 비례하여 요금을 부과하므로 모든 사용자가 요금 부과에 납득할 수 있을 것이다.

실제 사용 시간에 맞추어 요금을 부과하기 위해서는, 사용 시간을 정확하게 측정할 수 있는 방법이 필요하다. 이 방법에 의하여 측정된 시간은 소프트웨어 제공자와 사용자가 모두 납득할 수 있어야 한다. 의도적이건 실수로 인해서이건 일방에 의하여 방해 받지 않고 정확하게 계산하며 또한 그 근거를 보일 수 있어야 한다. 실제 요금 부과는 소프트웨어 제공자에 의하여 이루어지므로 사용 시간 계산의 책임 또한 제공자에게 있다. 따라서 제공자는 사용자의 사용 시간을 정확히 측정하며 또한 그 사용에 대하여 사용자가 부인할 수 없는 방법으로 사용 시간을 증명할 수 있어야 한다.

본 논문에서는 사용자가 제공자로부터 필요한 소프트웨어를 다운로드 받아 자신의 컴퓨터에서 수행하는 소프트웨어 임대 서비스의 요금 계산 기법을 제안한다. 제안된 방법은 사용자의 사용 시간을 정확히 측정하며 또한 그 사용에 대하여 사용자가 부인할 수 없는 방법으로 사용 시간을 증명할 수 있다.

2. 관련 연구

소프트웨어 임대 서비스 필요성에 따라 ASP[1, 2, 3]와 SaaS[4] 등의 개념이 도입되어 사용되고 있다. 이미 Salesforce.com[5]과 BizmekaKanOffice[6] 등과 같이 상용의 소프트웨어임대 서비스 기관이 소프트웨어 임대 서비스를 제공하고 있다. ASP와 SaaS는 네트워크를 통하여 사용자의 요청된 작업을 서비스 제공자의 컴퓨터에서 수행한다. 사용자의 작업이 서비스 제공자의 컴퓨터에서 수행되므로 사용자가 자신의 자료를 서비스 제공자에게 전송하고 그곳에서 작업을 수행하고 저장하므로 이에 따른 정보보호의 문제점이 있다 [3].

소프트웨어 임대 서비스의 요금 부과 방법은 일정 기간에 일정 요금 부과가 일반적이다. 상용의 서비스인 BizmekaKanOffice[5]와 Salesforce.com[6]에서의 요금 부과는 일정 기간에 따라 정해진 요금을 부과하는 방식이다.

사용 시간에 따른 요금 부과 또한 제공될 수 있다. 기법은 따로 명시되고 있지 않으나 제공자에게 기록되는 로그 데이터를 사용하여 시작 시간과 종료 시간을 파악할 수 있을 것이다. 이 경우 문제점은 사용자가 이에 납득하지 못하는 경우 또는 의도적을 부인하는 경우 사용 시간을 증명할 수는 없다. 서비스 제공자의 로그 데이터를

사용자가 신뢰하지 않을 수도 있기 때문이다.

Ferreria 등[7]은 소프트웨어 사용에 따른 요금 부과 방식을 제안하였다. 그러나 [7]에서 제안된 기법은 소프트웨어의 수행 횟수에 따라 요금을 계산한다. 따라서 한 번 수행을 시작하고 오랜 시간 동안 종료하지 않고 사용하는 경우 짧은 시간만 사용하는 경우와 동일한 요금이 부과된다.

HP는 서버를 위한 PPU(Pay-Per Use) 소프트웨어(HP PPU)를 사용한다[8]. HP PPU는 서버 사용자가 서버를 구입하여 사용하거나매월 서버 임대료를 지불하는 대신 서버의 사용 분량에 따라 이용 요금을 지불할 수 있도록 한다. 이용 요금은 프로세서 사용율과 활성 프로세서의 수에 의하여 계산된다. HP PPU는 정확한 하드웨어 사용 분량을 계산하기 위하여 사용된다. 사용자가 자신의 사용 분량에 대한 이의를 제기하면 계산된 분량이 정확함을 증명할 수는 없다.

웹 기반 VOD(Video-on-Demand) 시스템의 시청 시간을 측정하는 기법은 제안되어 있다[9, 10]. 이들은 일방향 해쉬 함수의 반복으로 사용자의 시청 요청을 받아 사용자의 시청 시간을 정확히 측정한다.

3. 서비스 모델

사용 시간에 따른 요금 부과 기법을 사용하는 소프트웨어임대 방법은 3단계로 이루어져 있다. 사용자 등록, 소프트웨어 사용, 그리고 요금 부과 및 이의 해소가 그들이다.

새로운 고객이 소프트웨어 제공자가 제공하는 소프트웨어를 임대하여 사용하고자 하는 경우, 고객은 먼저 사용자 등록을 하여야 한다. 고객은 자신의 신상 정보, 요금 청구를 위한 주소, 공개키 인증서 등의 정보를 제공자에게 전송하여 사용자 등록을 한다. 제공자는 사용자를 위한 계정을 새로 생성한다.

등록된 고객이 소프트웨어를 사용하기 위해서는 제공자의 온라인 카탈로그를 참조하여 사용 가능한 소프트웨어와 그 임대 가격을 확인한다. 필요한 소프트웨어 사용을 결정하면 고객은 소프트웨어 제공자에게 사용 요청을 한다. 사용 요청을 받은 제공자는 소프트웨어를 제공한다 - 다운로드를 허용한다. 사용자는 소프트웨어를 필요한 만큼 사용하고 종료한다. 사용자는 종료 시 제공자에게 종료 메시지를 보낸다.

소프트웨어 제공자는 사용자에게 사용 요금을 부과한다. 계산된 요금은 사용자의 소프트웨어 사용 시간에 기반한다. 사용자의 부과된 요금에 동의하면 요금을 지불한다. 만약 부과된 요금에 이의가 있는 경우 사용자와 제공자간에 논란이 생기게 된다. 논쟁이 생기는 경우 요금 계산이 제공자에 의하여 수행되었으므로 제공자가 그 요금이 올바른 요금임을 증명하여야 한다.

4. 제안 기법

4.1 고려 사항

항상 정상적으로 소프트웨어가 사용되고 종료되는 것은 아니다. 임대된 소프트웨어를 사용하던 사용자가 사용 도중 의도치 못한 사건으로 인하여, 예를 들어 컴퓨터 전원 문제로 인하여, 사용이 중지될 수도 있다. 여러 이유로 종료 메시지가 제공자에게전달되지 않을 수도 있다. 예를 들면 네트워크 문제로 종료 메시지가 손실될 수도 있다. 또한 사용자가 의도적으로 또는 실수로 종료 메시지를 보내지 못하는 경우도 있다. 경우에 따라서는 사용자가 종료 메시지를 보내고 난 이후에도 계속해서 소프트웨어를 사용하고자 하는 사용자도 있을 수 있다.

정상적으로 소프트웨어가 사용되고 종료되는 경우뿐만 아니라 어떠한 경우에도 소프트웨어의 사용 시간이 정확하게 측정되고 그에 따라 사용 요금이 계산되어야 한다. 또한 계산된 사용 요금이 정당함을 제공자는 사용자에게 증명할 수 있어야 한다.

4.2 접근 방법

본 논문에서 제안된 기법에서는 입회자라고 불리는 프로그램과 반복 해쉬 함수가 사용된다. 입회자 프로그램은 간단한 프로그램으로 사용자의 컴퓨터에서 수행된다. 독립된 프로그램일 필요는 없으며 사용자가 임대하여 사용하고자 하는 소프트웨어에 내장될 수 있다. 입회자는 사용자의 부정행위를 탐지하고 부정행위 시 소프트웨어 사용을 중지시키는 역할을 한다. 입회자는 사용자의 컴퓨터에 어떠한 악영향을 끼치지 않으며 또한 어떠한 정보도 사용자의 컴퓨터로부터 외부로 보내지 않아야 한다.

입회자 프로그램은 사용자가 임대 소프트웨어를 시작하고자 하면 실행되며 사용자가 그 소프트웨어의 사용을 종료하면 동시에 종료된다. 입회자가 하는 일은 사용자가 소프트웨어 제공자의 허락 없이 소프트웨어를 사용하는 경우 강제로 그 소프트웨어를 종료 시키는 일을 한다.

반복 해쉬[9, 10, 11, 12]는 사용자의 사용 시간을 계산하기 위하여 사용된다. MD5[13] 또는 SHA[14]와 같은 해쉬 함수, $H(x)$, 는 일방향함수의 특성을 가지고 있다. 즉, 주어진 메시지 n 으로부터 $h = H(n)$ 을 만족하는 h 는 쉽게 구할 수 있으나, 주어진 h 로부터 $h = H(n)$ 을 만족하는 n 을 구하는 것은 현실적으로 불가능하다. 반복 해쉬란 이러한 해쉬 함수를 반복적으로 수행함을 말한다.

사용자가 소프트웨어 임대를 요청할 때, 무작위로 큰 정수 n 을 선택하고 그 n 에 m 번 해쉬를 수행한다.

$H^i(n) = H(H^{i-1}(n))$, *where* $i = 1, 2, 3, ..., m$, *and* $H^0(n) = n$.

사용자는 $H^0(n)$, $H^1(n)$, ..., $H^{m-1}(n)$ 은 비밀스럽게 보관하고 $H^m(n)$ 은 전자 서명을 첨부하여 소프트웨어 제공자에게 전송한다. 소프트웨어 제공자는 사용자에게소프트웨어를 제공하여 사용할 수 있도록 한다. 이 때 입회자 프로그램도동시에 사용자의 컴퓨터에서 수행되어야 한다.

소프트웨어를 사용하는 동안 사용자는 주기적으로 소프트웨어 제공자에게 $H^i(n)$ 을 전송함으로써 계속 사용하고자 하는 의사를 전달한다 - 여기에서 i 는 $m-1$, $m-2$, ..., 로 매 번 1 씩 감소한다. $H^i(n)$ 를 수신한 소프트웨어 제공자는 이전에 받은 $H^{i+1}(n)$ 을 이용하여 $H^{i+1}(n) = H(H^i(n))$ 임을 확인하여 그 해쉬 값의 올바름을 판별한다. 제공자는 올바른 해쉬 값을 받았음이 확인되면 사용자에게 계속 사용 허락 메시지를 전송한다. 사용자는 그 허락 메시지를 입회자에게 전달하여 자신이 그 소프트웨어를 계속 사용할 수 있는 허락을 제공자로부터 획득하였음을 알린다. 주기적으로 입회자 프로그램은 사용자에게서 전달되는 계속 사용 허락 메시지(제공자로부터 생성 됨)를 확인하여 사용자가 계속 사용할 권한이 있음을 확인한다. 입회자 프로그램은 올바른 계속 사용 허락 메시지를 받지 못하면 사용자에게 사용 권한이 더 이상 없는 것으로 간주하여 그 소프트웨어의 사용을 중단시킨다.

계속 사용 허락 메시지 또한 반복 해쉬 함수를 이용한다. 소프트웨어 제공자는 소프트웨어 사용 요청을 받은 후 임의의 큰 정수 w 를 선택하고 m 보다 작지 않은 수 p 를 선택하여 이를 p 번 해쉬를 수행하여 $H^p(w)$를 사용자에게전송하여 입회자에게 전달되도록 하며 $H^0(n)$, $H^1(n)$, ..., $H^{p-1}(n)$ 은 비밀스럽게보관한다. 이 때 $H^p(w)$ 가 소프트웨어 제공자로부터 새로 생성된 것임을 입회자가 확인할 수 있어야 한다. 이를 위하여 새로운 정수 하나를 입회자가 생성하여 제공자에게 전송하며 제공자는 이 수와 $H^p(w)$ 에 함께 전자서명을첨부하여 전송함으로써 확인할 수 있도록 한다. 전자서명을 확인하기 위한 제공자의 공개키는 입회자가 가지고 있어야 한다. 매 번 사용 허락 메시지로 $H^j(n)$ 을 전송한다 - 여기에서 j 는 $p-1$, $p-2$, ..., 로 매 번 1 씩 감소한다.

사용자가 소프트웨어 사용을 마치면 소프트웨어 제공자는 사용 시간을 계산한다. 제공자가 마지막으로 받은 해쉬 값이 $H^{m-k}(n)$ 이면 그 사용자의 소프트웨어 사용 시간은 k 만큼이 된다. 사용 시간에 대한 사용자의 이의 제기가 있는 경우, 제공자는 처음 받은 해쉬 값($H^m(n)$)과 마지막 받은 해쉬 값($H^{m-k}(n)$), 그리고 $H^m(n) = H^k(H^{m-k}(n))$ 을 보임으로써 계산된 시간이 올바른 사용 시간임을 보인다.

4.3 표기법

프로토콜 기술을 위하여 다음의 표기법을 사용한다.

- *S*: 제공자(서버)
- *C*: 사용자 (클라이언트)
- *O*: 입회자
- Id_C: *C* 의 계정 이름
- $H(n)$: 메시지 n 의 일방향 해쉬 함수
- $H^i(n)$: 메시지 n 의 반복 해쉬. $H^i(n) = H(H^{i-1}(n))$ *and* $H^0(n) = n$
- *Ps*: *S* 의 공개키
- *Ss*: *S* 의 개인키
- *Pc*: *C* 의 공개키
- *Sc*: *C* 의 개인키
- $S_k(X)$: 메시지 *X* 대하여 키 k (*Ss* or *Sc*)로 수행한 전자서명
- N_A: *A* (*C*, *S*, 또는 *O*)에 의하여 생성된 nonce (랜덤 정수)

4.4 프로토콜

(1) 사용 요청

C 는 *S* 에게 사용자 등록을 한 것으로 가정한다. *C* 는 온라인 카탈로그를 이용하여 사용 가능한 소프트웨어와 그 소프트웨어의 시간당 임대 가격을 확인한 후, 해당 소프트웨어를 사용하기로 결정하면 제공자에게 사용 요청 메시지를 보낸다. *C*는 요금 부과 시간 단위 *L* 을 결정한다. *C* 는 또한 현재 세션에서의 선택된 소프트웨어를 사용할 최대 시간(최대 시간 단위의 수) *m* 을 선택한다. *C* 는 랜덤 정수 *R1* 을 선택하고 이 정수를 *m* 번 반복 해쉬를 수행한다. 즉, $H^1(R1)$, $H^2(R1)$, …, $H^m(R1)$ 을 계산한다. *C* 는 *R1*, $H^1(R1)$, $H^2(R1)$, …; $H^{m-1}(R1)$ 을 비밀스럽게 보관한다. 그리고 다음의 프로토콜이 수행된다.

1. $C \rightarrow S$: Id_C
2. $S \rightarrow C$: N_S
3. $C \rightarrow S$: Id_C, *Title*, *Pr*, *L*, *m*, $H^m(R1)$, N_C, $S_{Sc}(Id_C, Title, Pr, L, m, H^m(R1), N_S)$
4. $S \rightarrow C$: *Ob*, *SW*, $S_{Ss}(Ob, SW, N_C)$
5. $O \rightarrow C$: N_O
6. $C \rightarrow S$: N_O
7. $S \rightarrow C$: *p*, $H^p(R2)$, $S_{Ss}(p, H^p(R2), N_O)$
8. $C \rightarrow O$: *p*, $H^p(R2)$, $S_{Ss}(p, H^p(R2), N_O)$

C 는 자신의 계정 이름 (Id_C)을 *S* 에게 전송하며 프로토콜을 시작한다. *메시지 1*을 *C* 로부터 수신하면 *S* 는 새로 생성한 정수(N_S)를 생성하여 *C* 에게 전송한다. *S* 는 *C* 를 위한 세션을 시작한다. 이 세션은 *C* 가 소프트웨어 사용을 완료하면 종료된다.

C 는 Id_C 와 사용하고자 하는 소프트웨어 이름 (*Title*), 단위 시간 당 소프트웨어 사용 가격(*Pr*), 시간 단위(*L*), 그리고 최대 시간 단위 수(*m*), 해쉬 값($H^m(R1)$), 그리고 새로 생성한 정수(N_C)를 *S* 에게 전송한다. *C* 는 또한 Id_C, *Title*, *Pr*, *L*, *m*, $H^m(R1)$, 그리고 N_S 에 전자서명하여 그 서명을 *S* 에게 전송한다.

S 는 *C* 로부터 *메시지 3*을 이용하여 소프트웨어 사용 요청을 받은 후, 전자서명을 확인하여 올바른 요청인지 확인한다. 만약 올바른 요청으로 확인되면, *S* 는 입회자 프로그램(*Ob*)과 요청된 소프트웨어(*SW*)를 *C* 에게 전송한다.

C 는 *메시지 4*를 받으면 전자서명을 확인하여 올바른 메시지임을 확인하고 *Ob* 프로그램을 시작시킨다 (수행되는 *Ob* 프로그램은 *O* 라는 객체가 된다.). *O* 는 수행되는 동안 사용자 컴퓨터의 메모리에 상주한다.

O 는 새로운 정수(N_O)를 생성하여 이를 *C* 에게 전송한다. *C* 는 N_O 를 *S* 에게 전달한다.

S 는 N_O 를 수신한 후 *m* 보다 큰 랜덤 정수 *p* 를 선택한다. *S* 는 또한 랜덤 정수 *R2* 를 선택하여 이를 *p* 번 반복 해쉬를 수행한다. 즉, $H^1(R2)$, $H^2(R2)$, …, $H^p(R2)$를 계산한다. *S*는 *R2*, $H^1(R2)$, $H^2(R2)$, …, $H^{p-1}(R2)$는 비밀스럽게 보관하고, *p*, $H^p(R2)$, 그리고 {p, $H^p(R2)$, N_O}에 대한 전자서명을 *C* 에게 전송한다. *C* 는 이 메시지를 그대로 *O* 에게 전달한다.

O 는 *p*, $H^p(R2)$, $S_{Ss}(p, H^p(R2), N_O)$를 가지고 있는 메시지를 수신한 후 전자 서명을 확인하여 이 메시지가 *S* 로부터 생성된 메시지임을 확인한다. *O* 는 자신이 생성하여 도전 메시지로 보낸 N_O 와 *S* 가 생성한 해쉬 값($H^p(R2)$)에 생성된 전자서명을 확인하여 수신한 해쉬 값($H^p(R2)$)이 *S*에 의하여 현재의 세션을 위하여 새로 생성되었음을 확인한다.

올바른 메시지임을 확인하면 *O*는 요청된 소프트웨어의 수행을 시작한다.

(2) 소프트웨어 사용

1. $C \rightarrow S$: $H^i(R1)$
2. $S \rightarrow C$: $H^i(R2)$
3. $C \rightarrow O$: $H^i(R2)$

임대된 소프트웨어를 사용하는 동안 주기적으로 *C* 는 *S* 에게 사용 중인 소프트웨어를 계속 사용하고 싶다는 의사를 알린다. 소프트웨어 사용 요청 프로토콜 수행 시 *C* 가 생성해 놓았던 해쉬 값이 계속 사용 요청을 위하여 사용된다. 매 시간 단위(*L*)마다 *C* 는 $H^i(R1)$를 *S* 에게 전송한다. 여기에서 *i* 는 *m-1*, *m-2*, …의 값을 순차적으로 가지게 된다.

S 는 *C* 로부터 $H^i(R1)$를 수신한 후, $H(H^i(R1)) = H^{i+1}(R1)$ 임을 확인한다. 만약 $H(H^i(R1)) = H^{i+1}(R1)$ 이면, *S* 는 *C* 가 해당 소프트웨어를 *L* 시간만큼 더 사용하고자 하는 의사가 있음을 안다. *S* 는 *C* 로부터 올바른 계속 사용 요청을 받았음을 확인한 후, 계속 사용 허락 메시지를 전송한다. 계속 사용 허락 메시지를 위해서는 소프트웨어 사용 요청 프로토콜 수행 시 *S* 가 생성해 놓았던 해쉬 값($H^j(R2)$)이 사용된다. *S* 는 *C* 로부터 받은 계속 사용 요청 메시지 확인 후, 즉시 ($H^j(R2)$)를 전송한다. 여기에서 *j* 는 *p-1*, *p-2*, …의 값을 가진다.

C 는 수신한 $H^j(R2)$를 *O* 에게 전달한다. *O* 는 $H(H^j(R2)) = H^{j+1}(R2)$ 임을 확인한다. 만약 $H(H^j(R2)) = H^{j+1}(R2)$ 이면, *O* 는 *C* 가 시간 단위 *L* 만큼 현재 사용 중인 소프트웨어를 계속 사용할 권리가 있음을 알 수 있다.

만약 *S* 가 *C* 로부터 올바른 사용 요청을 받지 못하면, *S* 는 *C* 가 해당 소프트웨어를 계속 사용할 의사가 없는 것으로 간주하여 현재 세션을 종료하고 사용 시간을 계산한다. 이 경우 *S* 는 계속 사용 허락 메시지를 전송하지 않으며 *O* 는 계속 사용 허락 메시지를 수신할 수 없다. *O* 는 올바른 계속 사용 허락 메시지($H(H^j(R2)) = H^{j+1}(R2)$)를 수신하지 않으면 현재 수행 중인 소프트웨어를 적절한 종료 예고 메시지와 함께 종료한다. 또한 자신의 수행(*O*)을 종료한다.

(3) 종료

1. $C \rightarrow S$: *SIG_Term*, Id_C, *Title*, N_C, N_S, S_{Sc}(*SIG_Term*, Id_C, *Title*, N_C, N_S)
2. $C \rightarrow O$: *SIG_Term*

종료 메시지는 생략 가능하다. *C* 는 종료 메시지 없이 현재 사용 중인 소프트웨어를 종료할 수

있다. C 가 명시적으로 종료 메시지를 보내고 싶은 경우, 위에 기술된 종료 메시지를 S 와 O 에게 전송할 수 있다. 이 경우 S 는 현재 세션을 종료하고 사용 시간을 계산한다. O 또한 이 경우 현재 수행 중인 소프트웨어를 종료 종료하고 자신의 수행(O)을 종료한다.

(4) 요금 계산

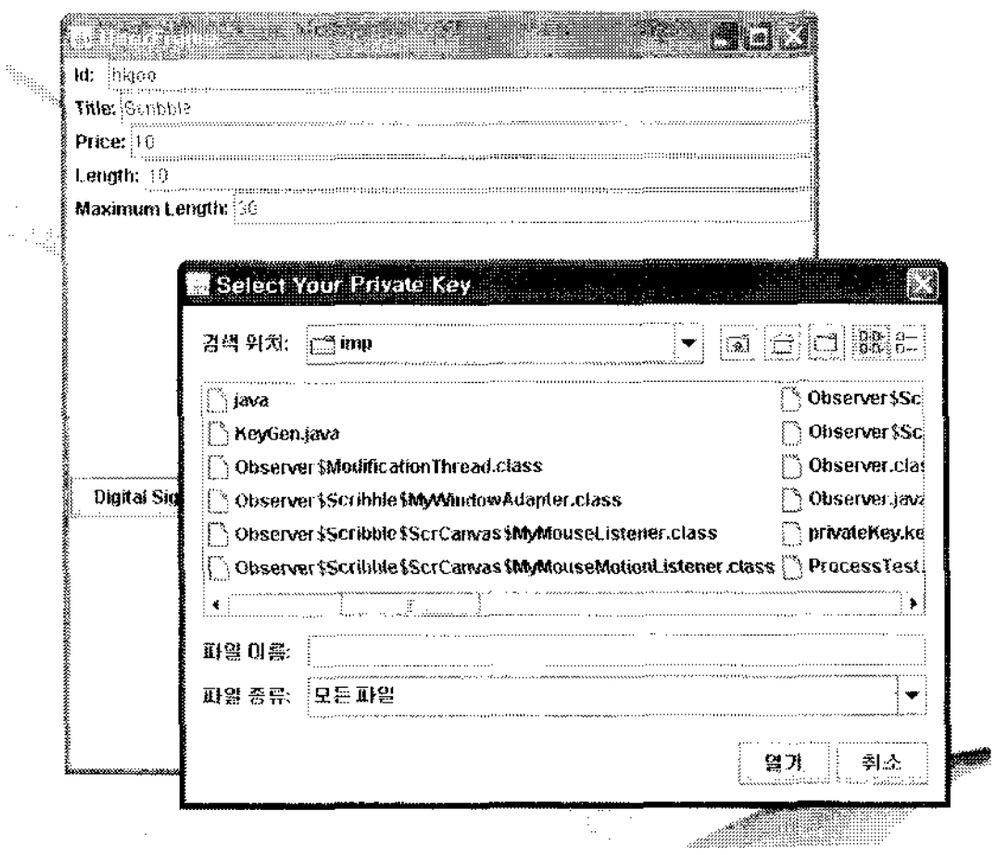
C 가 소프트웨어 수행을 마치면 S 는 소프트웨어 사용에 대한 요금을 계산한다. C 가 의도적으로 소프트웨어를 종료하였을 수도 있고 의도하지 않고 소프트웨어가 종료되었을 수도 있다. 어떠한 경우이건 S 는 세션을 종료하고 요금을 계산한다. 요금 계산은 C 가 소프트웨어를 사용한 시간 단위의 수와 시간 단위 당 요금에 의하여 결정된다.

C 가 소프트웨어 사용을 종료하였을 때, S 가 마지막으로 수신한 계속 사용 요청 메시지의 해쉬 값이 $H^{m-k}(R1)$ 이면 S 는 C 가 해당 소프트웨어를 k 단위 시간 이상 사용 하였음을 알 수 있다. 만약 C 가 사용 시간(요금)에 대한 이의를 제기하는 경우, S 는 $H^{k}(H^{m-k}(R1)) = H^{m}(R1)$ 임을 보임으로써 C 가 최소한 k 단위 시간만큼 해당 소프트웨어를 사용 하였음을 증명할 수 있다.

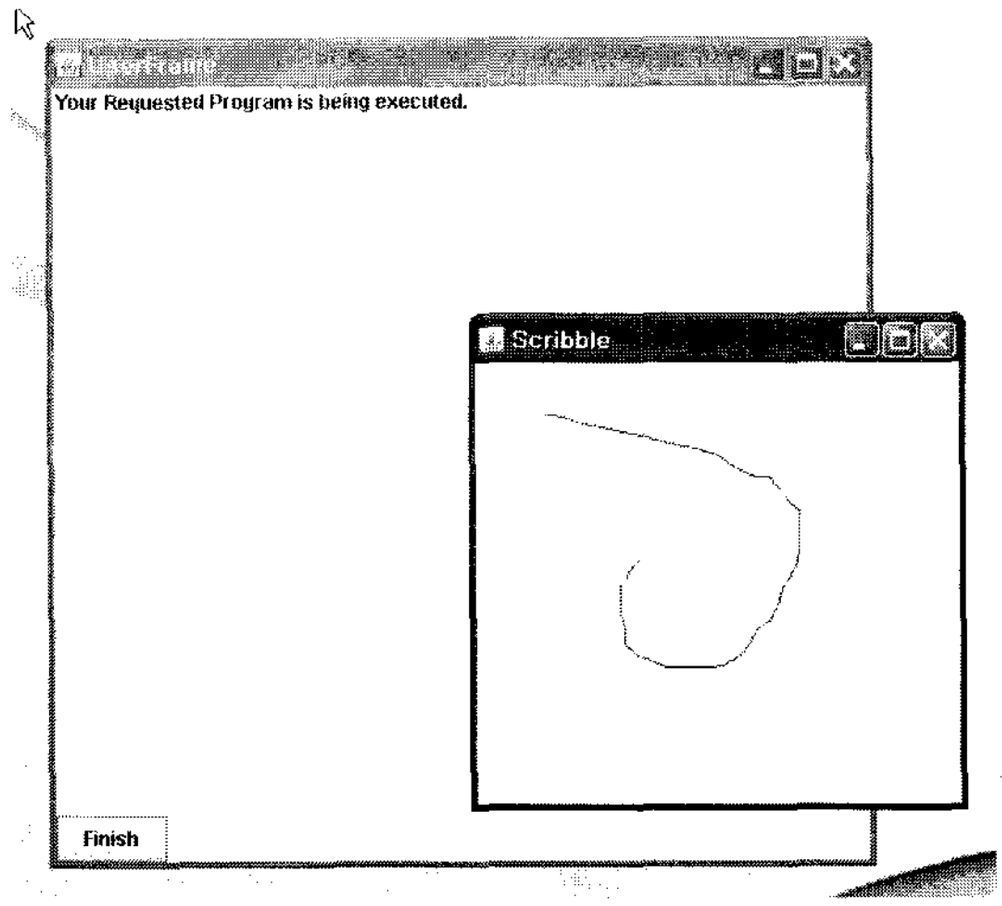
5. 구현 시 고려 사항

본 논문에서 기술된 기법에 대한 소프트웨어 프로토타입이 제작되었다. 프로토타입 제작은 Java 프로그래밍 언어를 사용하였으며 암호화 관련 부분은 암호화 패키지를 제공하는 JCA(Java Cryptography Architecture)[15]를 이용하여 수행되었다.

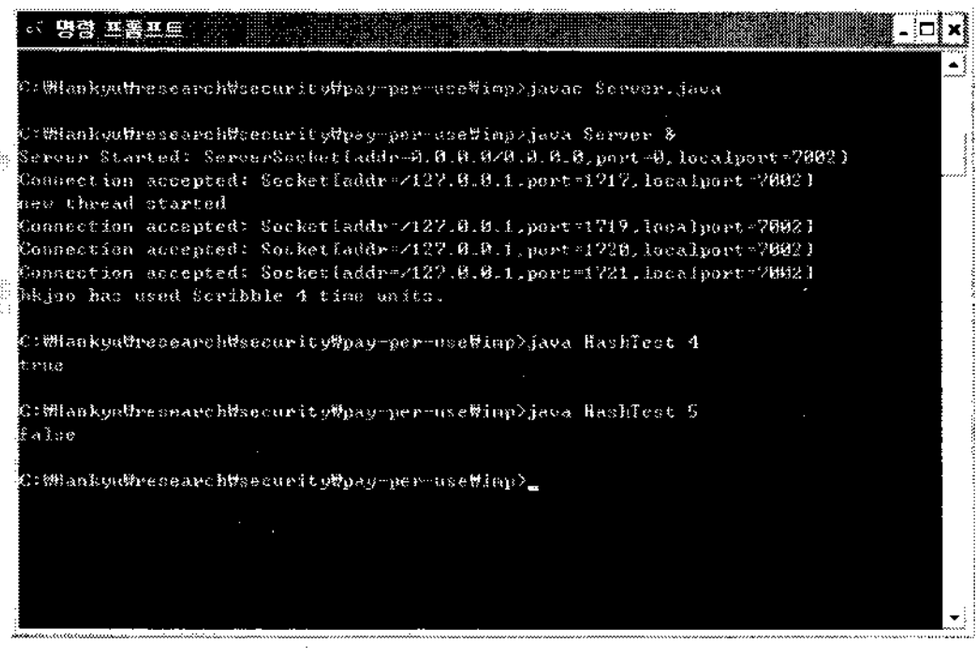
(그림 1)은 구현된 프로토타입의 사용 신청을 보여준다. 사용자가 자신의 계정 이름, 사용할 소프트웨어, 가격, 시간 단위, 그리고 최대 사용 시간을 입력한 후 자신의 개인키를 이용하여 전자서명을 수행한다. 제공자로부터 사용허락이 되면 (그림 2)와 같이 사용 중임을 알리는 화면과 사용자가 원하는 응용 프로그램이 수행된다. 사용자는 자신의 응용 프로그램을 종료하거나 종료(Finish) 버튼을 누름으로써 사용을 종료할 수 있다. 사용이 종료되면 제공자는 (그림 3)에서 보듯이 사용자의 사용 시간을 계산한다. 또한 사용자의 사용 요청 시의 해쉬 값과 마지막 해쉬 값을 저장함으로써 사용 시간 계산이 올바름을 증명할 수 있다.



(그림 1) 사용 신청



(그림 2) 소프트웨어 사용



(그림 3) 제공자

프로토타입 제작을 통하여 확인된 소프트웨어 구현 시 고려할 사항은 다음과 같다.

입회자 프로그램은 프로그램 자체에 제공자의 공개키를 가지고 있어야 한다. 입회자가 제공자의 공개키를 프로그램 자체에 가지고 있지 않으면 입회자 프로그램이 사용자의 컴퓨터에서 수행되는 특성을 이용하여 사용자가 제공자를 가장하여 공개키를 생성하여 입회자에게 제공하며 대응되는 개인키를 이용하여 전자서명 함으로써 언제든지 계속 사용 허락 메시지를 생성할 수 있는 가능성이 있게 된다.

입회자 프로그램과 임대된 프로그램은 개념상 상이한 두 개의 객체로 보이나 구현 시 하나의 프로그램으로 작성하는 것이 효과적이다. 최대한 강력히 결합되도록 하여 입회자 수행 없이 임대 소프트웨어만 실행 시키는 것이 불가능하도록 하여야 한다.

사용자와 제공자는 네트워크로 연결되어 있어야 한다. 그러나 임대된 소프트웨어가 수행되는 동안 항상 사용자와 제공자가 네트워크로 연결되어 있지 않는 경우에도 수행될 수 있도록 지원하는 것이 효율적이다. 사용 요청 프로토콜을수행하는 동안과 계속 사용 요청 메시지와 계속 사용 허락 메시지가 전송 되는 순간만 연결 상태를 유지하면 될 것이다. 이를 위하여 소프트웨어 계속 사용 요청 메시지를 수행하는 프로토콜 수행하는 순간 새로운 연결을 생성하고 계속 사용 허락 메시지가 도착하는 순간 연결을 종료하는 것이 바람직하다. 이 경우 세션을 유지시켜 줄 필요가 있다. 사용자는 계속 사용 요청 메시지 ($H^i(R1)$)와 함께 자신의 계정 번호(Id_C)와 식별자(예, N_s)를 함께 전송하며 세션이 유지되는 동안 제공자는 이를 이용하여 특정 사용자의 특정 세션임을 확인한다.

사용자가 제공자와 일정 시간 동안 네트워크 연결 상태를 유지하기 어려우면 매 시간 단위마다 사용 요청을 하고 사용 허락을 받는 대신 시간 예약을 할 수도 있을 것이다. 이는 사용자가 연속으로 여러 번 사용 요청 메시지를 전송하고 사용 허락 메시지를 수신하여 저장한 후, 매 시간 단위마다 저장된 사용 허락 메시지를 입회자에게 전송함으로써 가능하다. 이 경우 예약된 시간 이전에 종료하는 경우, 예약된 시간을 반환하는 것이 불가능한 문제점을 가지게 된다.

6. 토론

본 논문은 사용자가 필요한 소프트웨어를 소프트웨어 임대 사업자로부터 다운로드 받아 자신의 컴퓨터에서 수행하는 소프트웨어 임대 서비스의 요금 계산 기법을 제안하고 있다. 사용하고자 하는 소프트웨어가 사용자의 컴퓨터에서 수행되며 사용 요청 이외의 메시지가 사용자 컴퓨터로부터 전송될 필요가 없으므로 사용자는 ASP나 SaaS 등에서 느끼는 정보보호의 문제점 없이 소프트웨어를 사용할 수 있다.

본 논문에서 기술된 기법을 사용하면 사용자의 소프트웨어 사용 시간을 정확하게 계산할 수 있다. 제공자가 어떠한 경우에도 사용자의 사용 시간을 과다하게 계산할 수 없다. 만약 소프트웨어 제공자가 $H^{m-k}(n)$ 을 가지고 있고 초기에 사용자로부터 받은 해쉬 값이 $H^m(n)$ 이라면 사용자는 k 시간 단위 만큼 해당 소프트웨어를 사용한 것이

된다. 해쉬 함수는 일방향 함수이므로 $H^a(n)$ 로부터 $H^b(n)$ (여기에서 $b < a$) 을 생성하는 것은 불가능하다. 즉 소프트웨어 사용자 이외에 누구도 현재 자신이 가지고 있는 해쉬 값 $H^i(n)$ 로부터 $H^{i-k}(n)$ 를 생성하는 것은 불가능하다. 따라서 제공자가 가지고 있는 $H^{m-k}(n)$는 소프트웨어사용자가 제공자에게 전송한 것이며 이는 k 시간 단위만큼 소프트웨어를 사용한 후 계속 사용 의사를 밝힌 것이므로 최소한 k 시간 단위 만큼 소프트웨어를 사용한 것이다.

소프트웨어 사용자 또한 제공자에게 알리지 않고 소프트웨어를 사용할 수 없다. 매 주기마다 입회자 프로그램은 사용자가 소프트웨어를 계속 사용할 권한이 있는지 (사용자에게서 허락 메시지를 받았는지) 확인하여 제공자의 허락 메시지가 없으면 소프트웨어를 종료한다. 허락 메시지 또한 반복적인 일방향 해쉬 함수를 사용하므로 제공자만이 생성할 수 있다. 즉 제공자 외의 누구도 제공자임을 가장할 수 없다. 특히 제공자로부터 전송된 초기 해쉬 값이 입회자가 생성한 임시 정수와 함께 전자서명 되도록 하여 사용자가 이전에 사용했던 해쉬 값을 재사용하는 것이 불가능 하도록 하였다.

소프트웨어 사용자는 한 번 다운로드한 응용 프로그램을 이 후 새로 다운로드 할 필요없이 반복적으로 사용할 수 있을 것이다. 만약 다운로드한 프로그램이 변경되지 않았고 입회자가 가지고 있는 소프트웨어 제공자의 공개키가 변경되지 않았으면 반복적으로 해당 소프트웨어를 사용할 수 있다. 그러나 이 경우에도 요금은 동일하게 부과된다. 입회자를 통하여 해당 응용 프로그램은 수행되며 입회자가 소프트웨어 제공자로부터 사용 허락 메시지(해쉬값과 새로운 전자서명)를 받은 후라야 해당 소프트웨어가 수행되기 때문이다.

이 기법은 최대 1 시간 단위(L)만큼의 부정확함이 있을 수 있다. 소프트웨어 사용자가 $H^{m-k}(n)$ 을 전송한 후 즉시 종료하였으면 k 만큼의 시간 단위를 사용한 것이며, 계속 사용 허락 메시지를 받아 그 시간 단위의 남아있는 시간을 소진하였으면 $k+1$ 만큼의 시간 단위를 사용한 것이 되나 제공자는 판별할 수 있는 방법이 없다.

이 기법은 다른 소프트웨어와 마찬가지로 소프트웨어 크래킹에 안전할 수는 없다. 즉 재공학 등으로 입회자와 응용 프로그램 사이의 연결 부분을 제거한 후 입회자 없이 응용 프로그램만 사용하거나 입회자가 가지고 있는 소프트웨어제공자의 공개키 부분을 사용자가 생성한 공개키로 치환하여 제공자에게 알리지 않고 소프트웨어를 사용할 수도 있을 것이다. 제공자가 크래킹에 안전한 소프트웨어는 계속 연구되어야한다.

7. 결론

소프트웨어 임대 서비스는 완전히 새로운 개념은 아니다. 이미 ASP나 SaaS 등의 개념이 등장하였으며 이를 이용한 상용의 소프트웨어임대 서비스가 존재한다.

본 논문에서는 소프트웨어를 다운로드를 통하여 임대한 사용자가 사용한 시간만큼 요금을 지불할 수 있는 기법이 제안되었다. 본 논문에서 제안한 방식은 소프트웨어가 사용자의 컴퓨터에서 수행될 수 있으므로 ASP나 SaaS 등에서와 같이 제공자 컴퓨터에서 수행될 때 수반되는 정보보호의 문제점을 최소화 할 수 있다.

소프트웨어 임대 서비스를 위하여 사용한 시간을 어떠한 환경에서든 정확하게 계산할 수 있는 기술이 필요하다 본 논문에서는 이를 이를 위하여 반복 해쉬를 사용하였다. 본 논문에서 제안된 기법은 소프트웨어 제공자에 의하여 사용 시간이 정확하게 계산되며 사용자가 부인할 수 없도록 시간 계산의 근거를 제시한다.

본 논문에서는 다운로드한 소프트웨어의 크래킹을 방지할 수 있는 기법은 제안되지 않았으며 추후 연구가 필요하다. 또한 소프트웨어 제공자의 컴퓨터에서 수행되는 ASP와 SaaS 등의 기법에 제안된 연구 기법을 접목하는 연구 또한 필요하다.

참 고 문 헌

[1] M. A. Smith and R. L. Kumar, "A theory of application service provider (ASP) use from a client perspective", Information & Management, pp. 977-1002, 2004.

[2] H. Lee, J. Kim, and J. Kim, "Determinants of success for application service provider: An empirical test in small businesses", International Journal of Human-Computer Studies, pp. 796-815, 2007.

[3] S. K. Sharma and F. N. D. Gupta, "Application service provider: issues and challenges", Logistics Information Management, pp. 160-169, 2002.

[4] W. Sun, K. Zhang, S.-K. Chen, X. Zhang, and H Liang, "Software as a service: an integration perspective" ICSOC 2007, LNCS 4947, pp. 558-569, 2007.

[5] salesforce.com, http://www.salesforce.com, 2008.

[6] BizmekaKanOffice, http://kanoffice.bizmeka.com, 2008.

[7] L. C. Ferreria, R. Dahab, M. P. Aragao, and J. A. P. Magalhaes, "Two approaches for pay-per-use software construction," Proc. of the 2nd IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, pp. 184-191, 2000.

[8] Hewlett Packard, HP Pay per use (PPU) User's Guide for versions 7.x, 5th edition, Manufacturing Part Number 5971-4840, 2005.

[9] J. Zhou and K. Lam, "A secure pay-per view scheme for web-based video service," Proc. of the 2nd International Workshop on Practice and Theory in Public Key Cryptography, LNCS 1560, pp. 315-326, 1999.

[10] H. Joo, "Private and fair pay-per-view scheme for web-based video-on-demand systems," IEEE Trans. Consumer Electronics, vol. 49, no. 2, pp. 403-407, 2003.

[11] L. Lamport, "Password authentication with insecure communication," CACM, vol. 24, no. 11, pp. 770-772, 1982.

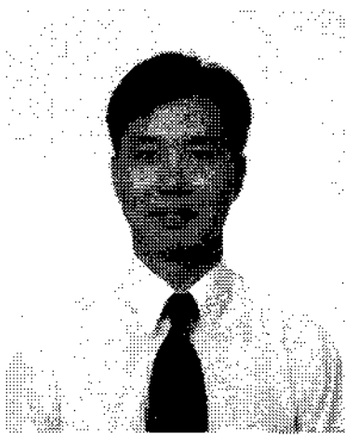
[12] T. Pedersen, "Electronic payments for small amounts," Proc. of the Cambridge Workshop on Security Protocols, LNCS 1189, pp. 59-68, 1996.

[13] R. Rivest, "The MD5 message digest algorithm," RFC 1321, IETF, 1992.

[14] NIST, "Secure hash standard," FIPS 180-2, 2002.

[15] Java Cryptography Architecture (JCA) Reference Guide for Java Platform Standard Edition 6, http://java.sun.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html, 2007

◐ 저 자 소 개 ◑



주 한 규(Hankyu Joo)

1988년 한림대학교 전자계산학과 졸업(학사)
1994년 아리조나주립대학교 대학원 컴퓨터공학과 졸업(석사)
1998년 아리조나주립대학교 대학원 컴퓨터공학과 졸업(박사)
2000~현재 한림대학교 정보통신공학부 교수
관심분야 : 정보보호, 소프트웨어공학.
E-mail : hkjoo@hallym.ac.kr