공유객체를 기반으로 한 그룹웨어 세션관리 툴킷

이경옥[0] 양재헌
한국과학기술원 전자전산학과 전산학전공
{kolee, jhyang}@dsg.kaist.ac.kr

A Session Management Toolkit for Synchronous Groupware Using Shared Objects

Kyung-Ok Lee[0] Jae-Heon Yang
Division of Computer Science, Dept. of EECS, KAIST

요 약

그룹웨어는 시·공간적으로 떨어져 있는 사용자들이 공동작업을 수행할 수 있도록 만들어진 어플리케이션이다. 따라서 싱글 유저 어플리케이션과 비교하여 사용자간 데이터 공유 지원, 통신 지원, 사용자 관리 등 추가적으로 구현해야 될 사항이 많다. 이 논문에서는 이러한 그룹웨어 어플리케이션 개발에 편의를 제공하고자 그룹웨어 어플리케이션이 공통적으로 개발하여야 하는 루틴을 라이브러리로 시스템 차원에서 지원해 주는 툴킷, SessionKit을 개발하여 소개한다. SessionKit은 자바로 구현된 순수 객체 모델 기반의 툴킷으로 일반 객체와 공유 객체 사이에 사용 방법 상의 차이를 없앰으로써 메시지 전달 방식에 의한 데이터 공유에 비해 개발자에게 한 단계 높은 abstraction을 제공한다. 또한 일반적으로 그룹웨어 어플리케이션이 어플리케이션 단위로 데이터를 공유하는데 반해 SessionKit 시스템은 개별 객체를 그 공유 단위로 함으로써 서로 다른 어플리케이션 간에도 정보 공유가 가능하도록 한다.

1. 서론

네트워크의 발전으로 이제는 여러 사람의 공동 작업이 필요한 일도 굳이 모두 한 장소에 모여서 할 필요가 없어졌다. 네트워크로 연결된 지역이면 어디서든지 같이 일할 수 있게 된 것이다. 시·공간적으로 떨어져 있는 사람들 사이에 공동 작업이 가능하도록 하는 분산 어플리케이션을 그룹웨어라고 한다. [1] 그룹웨어 어플리케이션은 여러 사람의 공동작업을 지원해야 하므로 일반적으로 싱글 유저 어플리케이션에 비해 개발 시 고려해야 할 점들이 많다.[2][5] 여러 사람이 서로 의사를 교환하고 작업을 공유할 수 있도록 사용자간 통신과 정보 공유가 가능해야 하고, 여러 사용자 어플리케이션이 하나의 그룹을 형성하여 공동 작업을 수행하므로 프로세스 그룹 관리도 해야 한다. 그리고 상호 협조할 수 있도록 다른 사용자를 인식할 수 있는 정보를 제공해야 한다. 이처럼 고려해야 될 점들이 많은 이유로 그룹웨어 어플리케이션은 개발하기가 까다롭다.

이 논문에서는 그룹웨어 어플리케이션 개발자의 부담을 줄이고자 만들어진 세션 관리 라이브러리인 SessionKit이라는 툴킷에 대해 소개한다. SessionKit은 위에서 언급된 프로세스간 통신, 데이터 공유, 프로세스 그룹 관리, 다른 사용자 인식 정보 제공 등 그룹웨어에 공통적으로 필요한 루틴들을 라이브러리로 만들어 개발자에게는 API만을 제공하는 미들웨어 형태의 시스템이다. SessionKit은 사용자에게 보다 나은 abstraction(추상화)을 제공하기 위해 자바 언어를 사용하여 오브젝트 기반으로 개발되었다. 개발자들이 공유 데이터 객체를 생성하려면 SessionKit 라이브러리에서 제공하는 SharedData class를 상속하면 되고, 세션이라는 프로세스 그룹에 참여하기 위해서는 ReplSession class를 상속하면 된다. SessionKit에서 제공하는 이 두 베이스 객체가 사용자간 데이터 공유나 세션관리를 담당한다.

SessionKit이 제공하는 특징적인 장점은 먼저 공유 객체와 일반 객체가 그 사용법 상 차이가 없어 쉽게 일반 어플리케이션을 그룹웨어 어플리케이션으로 바꿀 수 있다는 것이다. 또 다른 장점은 실제 객체가 서버에 있고 클라이언트들은 그것의 proxy를 가지는 중앙집중적 구조인 RMI나 CORBA에 비해 클라이언트들이 각각 복사본을 가지는 replicated 구조를 취하고 있어 response time과 latency가 작다는 것이다. 마지막으로 일반적인 그룹웨어 세션은 어플리케이션 단위로 데이터 공유가 이루어지지만 SessionKit은 객체 단위로 공유가 가능하기 때문에 서로 다른 어플리케이션도 쉽게 정보를 공유할 수 있다. 즉, SessionKit은 동일 어플리케이션간 서로 공동 작업을 수행할 수 있도록 지원할 뿐만 아니라, 서로 다른 어플리케이션간에도 공동 작업이 가능하도록 해 준다.

2장에서는 사용자 관점에서 SessionKit 시스템의 프로그래밍 모델에 대해 살펴보겠고, 3장에서는 실제 SessionKit 을 사용하여 구현한 세가지 그룹웨어 어플리케이션을 예로 들어 설명하겠다. 4장에서는 SessionKit의 세부 설계와 구현에 대해 설명하겠고, 5장에서 관련 연구에 대해 살펴보고 6장에서 결론을 맺겠다.

2. SessionKit 프로그래밍 모델

본 장에서는 SessionKit 시스템의 프로그래밍 모델을 사용자 관점에서 살펴보겠다. 2.1절에서는 전체 SessionKit의 구조에 대해 설명하고, 2.2절에서는 프로세스 그룹(세션) 객체의 생성과 그 역할에 대해 2.3절에서는 데이터 공유 객체의 생성과 그 역할에 대해 설명하겠다.

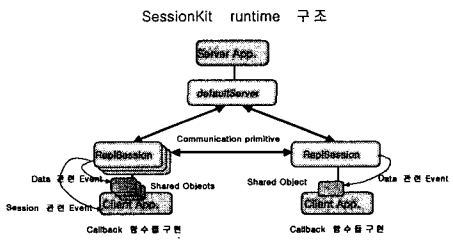
2.1 SessionKit 구조

SessionKit은 세 개의 자바 패키지로 구성되어 있다. SessionKit.client 패키지는 사용자간 데이터 공유나 프로세스 그룹 관리, 프로세스간 통신 등을 담당하는 클래스들의 묶음이고, SessionKit.events 패키지는 세션 관련, 데이터 관련 이벤트와 그 이벤트 감시자(listener) 클래스들의 묶음이며, SessionKit.server 패키지는 관리자용 서버 어플리케이션이 필요할 경우 이용되는 클래스들의 묶음이다.(보통의 경우 SessionKit에서 제공하는 defaultSessionServer를 서버로 이용)

2.2 세션 관리

그룹웨어 어플리케이션은 기존에 한 프로세스의 시작과 종료만을 다루던 싱글 유저 어플리케이션과 달리 하나의 공동작업을 수행하는 네트워크로 연결된 프로세스 그룹이라는 세션 개념을 가지게 된다. 따라서 세션의 시작과 종료, 세션 참가자 관리, 통신을 위한 연결관리, 상호인식 정보 제공 등 실제 어플리케이션 작업 외에 부수적으로 개발자가 해주어야 할 일들이 많다. 이렇게 모든 그룹웨어에 공통적으로 필요한 루틴들을 라이브러리 차원에서 제공해주면 개발자 부담이 훨씬 줄어들 것

이다.[2][3][4] SessionKit 에서는 ReplSession객체가 참가자 당 클라이언트 측에 생성, 이러한 루틴들을 지원한다. ReplSession 객체간 통신은 latency를 최소화하기위해 일대일로 이루어지는데, 일대일 통신 채널은 객체 생성시 만들어진다. (객체 생성시 먼저 서버와의 통신채널을 만들고, 서버로부터 세션에 있는 참가자 정보를 얻어와 그 각각에 대해 일대일 통신채널을 연다.-서버와의 연결이후 참가하는 참가자 정보는 이벤트를 통해 동기적으로 정보를 얻는다.) 다음 그림은 SessionKit을 이용하여 개발한 그룹웨어의 runtime 구조이다.



2.3 객체 단위 데이터 공유

각 클라이언트들이 복사본을 가지는 공유 데이터를 지원할 경우 고려해야 될 점은 한 클라이언트가 값을 바꾸었을 경우 그 변화 값을 다른 클라이언트들에게 전달하는 것이다. 이것은 SessionKit의 SharedData class가 담당하는 일로 이 베이스 클래스는 read(SharedDataUpdateEvent)와 update()함수를 가진다. update()함수는 현재 공유 데이터 값을 ReplSession을 통해 다른 사용자들에게 전달하는 일을 하고, read(-)는 다른 사용자가 업데이트 한 값을 이벤트로 전달 받아 자신이 가진 값을 업데이트 하는 일을 한다. 여기서 개발자가 신경써야 할 것은 공유 데이터 값이 변했을 때 호출되는 콜백 함수의 구현이다. SessionKit은 공유 객체 단위로 정보를 공유하므로,공유 객체마다 적절한 콜백 함수를 구현해 주어야 한다.

3. 예제 프로그램

본 장에서는 실제 SessionKit 라이브러리를 사용하여 구현한 세가지 그룹웨어 어플리케이션을 예로 들어 설명하겠다. 3.1절과 3.2 절에서는 각각 네트워크로 연결된 오목게임 그룹웨어와 화이트 보드 그룹웨어의 실제 구현 코드를 보고 SessionKit 이 어느 정도 개발자 편의를 제공하는지 살펴본다. 3.3절에서는 위의 예로 쓰인 서로 다른 두 그룹웨어 간 정보 공유에 대해 설명한다.

3.1 오목 게임

먼저 세션관리를 담당하는 ReplSession 객체를 생성한다.

```
public class omokClient extends ReplSession {
  public omokClient() {
    super("오목게임"); //세션이름을 주고 세션담당객체 생성
    omok omokuser = new omok(this); //오목판 인터페이스
    OmokUserFrame frame = new OmokUserFrame(omokuser);
//omok, OmokUserFrame class는 오목판 User Interface class
    addUserChangeListener(frame);
  } //참여자 정보 listener로 등록,동기적으로 정보를 받는다.
}
```

세션관련 부분을 완성하면 오목판 Board를 공유 객체로 만든다.

```
public class Board extends SharedData {
  public int[][] m_Board=new int[19][19]; //19*19 오목판
  public Board(ReplSession session, omok user) {
    super(session); //세션객체를 참조하는 공유객체 생성
      - //공유 데이터의 update정보는 ReplSession객체를
        //통해 전달되므로, ReplSession객체를 참조한다.
  }
  public void SharedDataValueChanged(DataUpdateEvent e) {
      read(e); //바뀐 데이터 값을 읽는다.
      omokuser.repaint(); //오목판을 새로 그린다.
  } //공유 객체 당 콜백 함수를 구현한다.
}
```

오목게임의 메인 함수이다. ReplSession 객체를 확장한 omokClient객체를 하나 생성한다.

```
  public static void main(String[] args) {
    new omokClient();
  }
```

위의 예제에서 알 수 있듯이 SessionKit을 사용할 경우 세션관리와 참가자간 데이터 공유를 지원하는데 드는 개발자 비용이 작다.

3.2 화이트 보드

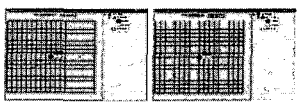
아래 예제는 점 찍기, 선 그리기, 텍스트 입력하기, 원 그리기 이 네 개 사용자 명령을 수행하는 그래픽 에디터이다. 각 사용자들은 에디터의 데이터가 아닌 취한 명령을 서로 공유하면서 공동작업을 한다. 따라서 아래 화이트 보드 그룹웨어는 사용자 명령을 공유 변수로 한다. 세션관리를 위해서는 오목과 동일하게 ReplSession객체를 이용한다.

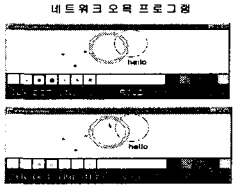
다음은 공유 변수 Command를 선언한 것이다.

```
public class Command extends SharedData {
  public String command;
  private WhiteBoardUser whiteboarduser; //사용자 인터페이스 객체
  public Command(ReplSession session, WhiteBoardUser user) {
    super(session); //ReplSession객체를 참조,오목게임과 같은 이유
    this.whiteboarduser = user;
  }
  public void SharedDataValueChanged(DataUpdateE vent e) {
    read(e); //update된 명령어 데이타를 읽는다.
    //명령에 맞춰 다시 그린다.
    whiteboarduser.controls.repaint();
    whiteboarduser.drawingArea.repaint();
  }
}
```

WhiteBoardUser 인데페이스 객체로부터 사용자 입력을 받아, 그 입력값을 공유객체 command로 바꾼 다음 command.update()를 통해 다른 사용자에게 알린다. 그러면 같은 세션 내 command 객체를 가진 모든 다른 사용자의 SharedValueChanged 함수가 호출되고 각 사용자들은 read()를 통해 update된 값을 가지게 된다.

다음 그림은 두 개 그룹웨어의 실행 화면이다.


네트워크 오목 프로그램



화이트 보드 프로그램

3.3 오목 게임과 화이트 보드간 정보 공유

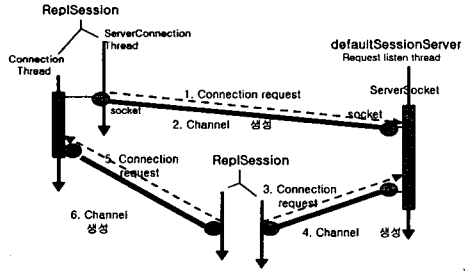
일반적인 그룹웨어가 어플리케이션 단위로 메시지 전달 방식에 의해 정보를 공유하는데 반해 SessionKit은 SharedData클래스를 베이스로 하는 객체 단위로 정보를 공유한다. 이것은 Java의 Reflection을 이용하여 런타임에 클래스 단위로 객체를 필터링 함으로써 가능한 것으로 서로 다른 어플리케이션 간에 정보를 공유할 수 있게 할 뿐만 아니라 서로 다른 어플리케이션이 같은 세션에 연결되어 있어도 서로 간섭 받지 않고 공동 작업을 수행할 수 있게 한다.

4. SessionKit 세부 설계 및 구현

본 장에서는 지금까지 설명한 SessionKit 라이브러리의 세부 설계와 그 구현에 대해 설명한다. 4.1 절에서는 세션의 생성과 소멸, 참가자 관리 등 세션관리에 대해 설명하고, 4.2절에서는 replicate된 공유 객체 데이터 값의 전파과정에 대해 설명한다. 마지막으로 4.3절에서는 세션 정보나 사용자 정보, 데이터 정보 등을 가진 이벤트의 전달과정에 대해 설명한다.

4.1 세션 관리

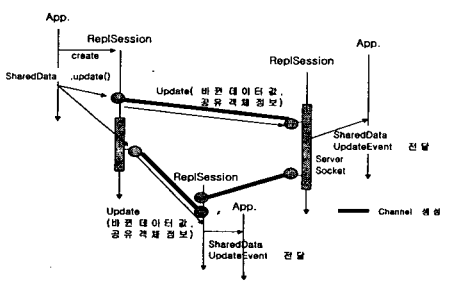
특별한 서버 관리자가 필요 없는 경우 defaultSessionServer를 수행시킨다. 이 서버는 클라이언트들로부터 세션 생성, 파괴, 세션 참가, 탈퇴 등의 요구를 받아 동적으로 세션 리스트, 참가자 리스트 등을 관리하고 해당 이벤트를 발생시키는 역할을 한다. 세션 서버를 수행시킨 상태에서 각 클라이언트들이 ReplSession객체를 생성하면 ReplSession객체 생성자는 서버와의 통신을 담당하는 ServerConnectionThread와 세션에 존재하는 다른 참가자들과 일대일 통신을 담당하는 ConnectionThread를 만든다. ConnectionThread를 통해 전체 클라이언트들이 통신채널을 연결함으로써 실제 하나의 프로세스 그룹(세션)이 형성되고, ServerConnectionThread를 통해 필요한 세션생성, 파괴, 사용자 참가, 탈퇴 등 세션관련 정보를 서버로부터 전달 받는다.



4.2 공유 데이터 값의 전파

SessionKit 공유 객체 모델은 response time과 latency를 최소화 하기위해 여러 클라이언트들이 공유 객체의 복사본을 가지고 각각 작업을 수행하도록 되어 있다. 따라서 공동작업을 원활히 수행하려면 공유 객체 복사본들의 값의 변화를 다른 클라이언트들에게 빨리 전달하는 것이 중요하다. 공유 데이터의 update와 그 값의 전파는 SessionKit 제공 클래스인 SharedData 클래스의 update() 함수가 담당한다. 업데이트 전파과정은 먼저 Reflection을 이용하여 공유 객체 정보를 얻고, 이 정보와 업데이트 데이터 값으로 SharedDataUpdateEvent로 만든다. 그런 후, 클라이언트가 참조하고 있던 ReplSession객체를 통해 현재 세션 참가중인 다른 참가자들에게 이 이벤트를 보낸다.(이벤트를 받은 클라이언트들은 이벤트로부터 공유 객체 정보를 얻어 해당 공유 객체 데이터를 바르게 update 한다.)

이는 공유 객체 모델인 RMI나 CORBA가 비동기적으로 업데이트 값을 확인할 수 있는데 반해 동기적으로 업데이트 값을 얻을 수 있는 장점을 제공한다.



4.3 이벤트 전달

세션 서버는 새로운 세션이 만들어지거나 파괴될 때, 새로운 사용자가 들어오거나 나갈 때, 공유 데이터 값이 변할 때 적절한 이벤트를 각 클라이언트의 ReplSession에게 보낸다. 이벤트를 넘겨받은 ReplSession은 해당 이벤트 listener로 등록한 객체들의 알맞은 이벤트 콜백 함수를 호출해 준다.

다음은 SessionKit에서 정의한 이벤트들이다.

DataUpdateEvent	공유 객체 데이타 값의 변화를 알린다.
LateComerUpdateEvent	새로운 사용자에게 현재 세션 데이터를 넘기도록 한다.
NewUserArrivedEvent	새로운 사용자의 세션 참가를 알린다.
UserLeavedEvent	사용자가 세션에서 나갔음을 알린다.
SessionCreateEvent	새로운 세션이 만들어졌음을 알린다.
SessionDestroyEvent	세션이 파괴되었음을 알린다.

5. 관련연구

일반적으로 가장 잘 알려진 그룹웨어 툴킷은 Groupkit이다[2]. 이것은 실제 세션관리 부분을 완전 그룹웨어 어플리케이션과 분리시켜 전혀 개발자가 신경 쓰지 않아도 되도록 하였다. 그러나 이로 인해 개발자가 특별한 그룹웨어 세션 관리를 원할 경우 유연하게 대처할 수 없다는 단점을 가지게 된다. 즉, Groupkit에서 제공하는 세션 관리 형태가 아닌 서로 다른 어플리케이션을 같은 세션에 연결시키고자 하는 경우나 explicit 세션 형성[6]을 원하지 않는 경우 시스템 차원에서 전혀 개발자에게 편의를 제공하지 못한다. 그리고 어플리케이션간 데이터 공유도 메시지 전달 방식에 의해 이루어지므로 객체보다는 abstraction 레벨이 떨어지고, 서로 다른 어플리케이션간에는 데이터 공유가 불가능하다.

Sun에서 Java 객체 데이터 공유를 위해 만들어진 툴킷인 JSDT(Java Shared Data Toolkit)는 SessionKit처럼 java로 구현되어 있고, 세션 정보나 공유 데이터 정보가 이벤트로 전달되는 등 비슷한 점이 많다[7]. 그러나 JSDT도 일반 그룹웨어처럼 어플리케이션 단위로 데이터가 공유 되고, 업데이트 된 데이터 값도 메시지 전달 방식에 의해 다른 클라이언트들에게 전달되는 등 객체 단위 공유에 비해 abstraction 레벨이 떨어진다. 그러나 JSDT는 데이터 update 정보를 보낼 때, priority와 ordering등 QoS를 정할 수 있다는 장점이 있다.

6. 결론

SessionKit 시스템은 그룹웨어 어플리케이션 개발에 공통적으로 필요한 세션관련 루틴을 라이브러리로 제공함으로써 개발자에게 편의를 제공해 주고자 개발되었다. 순수 객체 모델로서 공유 객체도 일반 객체처럼 사용하게 하여 메시지 전달 방식에 의한 데이터 공유에 비해 개발자에게 한층 더 나은 abstraction을 제공해 주며, 데이터 공유 단위도 어플리케이션에서 객체 단위로 세분화하여 지원, 서로 다른 어플리케이션도 필요한 데이터만을 쉽게 공유할 수 있도록 하였다.

7. 참고 문헌

[1] C.A.Ellis,S.J.Gibbs, and G.L.Rein, GROUPWARE-Some Issues and Experiences, CACM Vol.34, No.1, Jan, 1991

[2] Mark Roseman and Saul Greenberg, Groupkit-A Groupware Toolkit for Building Real Time Conferencing Applications, CSCW' 96

[3] Mark Roseman and Saul Greenberg. Registration for Real-Time Groupware. Research Report' 94

[4] Toe Urnes and Roy nejabi,Tools for Implementing Groupware: Surveys and Evaluation. Technical Report '1994

[5] John F.P, Mark Day and Jakov Kucan, Notification Servers for Synchronous Groupware, In Proceedings of the ACM CSCW ' 96

[6] Edwards,W.K.1994.Session management for collaborative applications.In Proceedings of the ACM CSCW' 94

[7] Sun, JSDT(Java Shared Data Toolkit), http://java.sun.com/products/java-media/jsdt/index.html