메디컬 Expert 시스템을 위한 Drools와 JavaFX 기반의 사용자 인터페이스 설계 및 구현

장원용, 최유나, 양성수, 최은미*
국민대학교 소프트웨어학부
*e-mail:emchoi@kookmin.ac.kr (교신저자)

User Interface Design and Development based on Drools and JavaFX for a Medical Expert System

Wonyong Jang, Yuna Choi, Seongsoo Yang, Eunmi Choi*
College of Computer Science, Kookmin University

요 약

본 논문은 Medical Expert 시스템을 효율적으로 운영하기 위해서 Drools 와 JavaFX기반으로 한 사용자 인터페이스를 설계 및 구현을 한다. 본 Medical Expert System을 구축하기 위하여 Drools의 내부의 구조인 BlackBoard 아키텍처 스타일을 이해하고 JavaFX을 이용하여 Medical Expert 시스템을 설계 및 구현 하였다. 이 시스템의 진행 프로세스는, 설문지 및 환자 진단으로 인해 환자에 대한 증상 정보들을 파악 후 미리 정해 둔 rule들에 적용시켜서 그 결과 값을 도출한다. 또한 본 시스템은 JavaFX에 scene builder 이용하여 인터페이스를 구성함으로써 기존의 Swing의 단점을 보완하고 장점을 부각시키도록 구현하였다.

1. 서론

본 논문에서는 Expert System을 사용하는 것이 여러 분야에서 점점 확산되어 가고 있는 현재 동향을 비추어 볼 때 Medical 분야에서 Expert System이 좀 더 효율적이고 확장가능성이 있는 시스템의 필요성이 요구되고 있다. 진단 지원 시스템인 Medical Expert System은 Java기반의 라이브러리인 Drools를 이용하여 함으로써 Medical 분야에서의 병원의 특징적인 rule들을 추가하고 변경사항이 있을 때는, 전체 프로그램을 수정할 필요 없이 drl 파일만을 수정하여 진단을 지원할 수 있도록 한다. 하지만 rule을 추가하기 위해 Expert system을 이루고 있는 Drools를 이해해야 하는 어려움이 생길 수 있다. 그렇기 때문에 다른 도메인의 어떤 전문가도 Expert system을 쉽게 사용하고, rule을 추가하기 위해 JavaFX기반의 사용자 인터페이스를 설계 및 제공할 필요성이 있다. 그래서 Drools와 JavaFX를 최대한 활용하여 Medical Expert System과 사용자 인터페이스를 설계 및 구현하였다.

2. 관련 연구

2.1 Drools

Drools는 BRMS(Business Rule Management System)[1]로써 Java Rule Engine API(JSR-94)를 지원[2]한다. Drools는 비즈니스 규칙 관리 및 복잡한 이벤트 처리가 빠르고, 신뢰할 수 있는 평가를 가능하게 하는 전방 연쇄(Forward-chaining) 및 후방 연쇄(Forward-chaining) 기반 규칙 엔진을 갖추고 있다[1]. Drools의 규칙은 Java 또는 DRL과 같은 언어로 정의된다. 관련된 연구인 [3]에서는 텍스트 기반 모델링 언어로 안드로이드 응용 프로그램을 생성하여 규칙 기반 모델 접근 방식을 소개를 하고 있다. 제안된 규칙 엔진은 모델, 뷰, 컨트롤러의 관점으로 구성된다. 모델은 추상 구문 모델(ASM)을 Java 객체(POJO)로 변환하고, 뷰에서는 추상 구문 모델(ASM)을 사용자 인터페이스 XML로 변환, 컨트롤러는 SAM을 액티비티 클래스로 변환한다. 안드로이드 응용 프로그램의 개발자는 Java 개발도구를 이용하여 POJO 클래스 및 Android 클래스의 코드를 생성하고, JDOM을 이용하여 XML 사용자 인터페이스를 생성한다. 또한 Drools 규칙 엔진을 기반으로 응용 프로그램 코드의 생성을 개선한다.

2.2 JavaFX

JavaFX[5] 은 리치 클라이언트 플랫폼으로서의 Java에서 upgrade 단계를 제공한다. 이것은 엔터프라이즈 비즈니스 애플리케이션을 위한 가볍고 하드웨어 가속 Java UI 플랫폼을 제공하도록 설계되어 있다. JavaFX를 사용하면 개발자는 응용 프로그램에서 Java 라이브러리를 재사용하는 강점을 가질 수 있다. 또한, 기본 시스템 기능에 액세스하거나 서버 기반의 미들웨어 응용 프로그램에 원활하게 연결할 수 있다.

JavaFX는 대규모 데이터 기반 비즈니스 애플리케이션을 처리 할 수 있는 강력한 Java 기반 UI 플랫폼을 제공한다. JavaFX 응용 프로그램은 Java기반 개발 된 표준 프로그래밍 방법과 디자인 패턴을 활용하고 있다. JavaFX는 고성능 하드웨어 가속 된 그래픽 및 미디어 엔진을 갖춘 다양한 UI 컨트롤, 그래픽, 미디어 API를 제공하고 민감한 비주얼 애플리케이션의 개발을 단순화할 수 있다.

3. Expert System 의 구조 연구

3.1 Expert System 의 Drools 구조 및 시스템 특성

본 연구에서는 Open-source project 인 Drools를 이용하여 메디컬 Expert System 에 대하여 설계하고 룰을 정의한다. 병원에서 환자에 대하여 처리할 수 있는 증상에 대하여 룰(Rule) 들을 정립하여 환자의 증상들을 가지고 그 조건에 맞는 결과를 도출하여 준다. Drools는 소프트웨어 시스템 아키텍처의 분석 시 전형적인 Blackboard 아키텍처 스타일을 갖추고 있다.

블랙보드 아키텍처 스타일은 Control, Knowledge Sources, Blackboard 로 구성되어 있다. Black Board의 역할은 중앙 데이터의 저장소이고 문제에 대한 현재 상태를 제시해 준다, Knowledge Sources 는 특정 도메인에 대한 해법을 제시해 주고 Blackboard에 그 해법을 적용시킨다. 마지막으로 Control은 Blackboard 상태를 모니터링하고 Knowledge Sources 스케줄을 관리하고 실행시킨다. 아래의 그림 1에서와 같이 경우에 따라 출장에 관한 룰, 또는 사원에 관한 룰이 될 수도 있다. Expert System에서는 환자 증상들에 관해서 진단 및 처방에 관한 룰이 drl 파일에 저장되어 있다. 그럼 이 부분이 Knowledge Source 가 되고, Rule Engine이 Control이 된다. 여기서 k session 은 Knowledge Base에서 나와서 Rule Engine 에 똑같은 공간을 만들어 놓는다. 그러면 환자라는 객체가 들어 왔을 때 Knowledge Base에 쌓이게 되는데 정립된 룰과 효율적으로 비교하며 결과 값을 도출하기 위해서 똑같은 공간인 k session에서 비교하면서 더 효율적으로 조건에 만족하는 결과 값을 도출하게 된다. 따라서 Expert System에서 Drools를 이용하여 구현함에 따라서 수많은 룰(Rule) 들 중에서 조건에 맞는 결과 값만을 효율적으로 도출할 수 있었다.

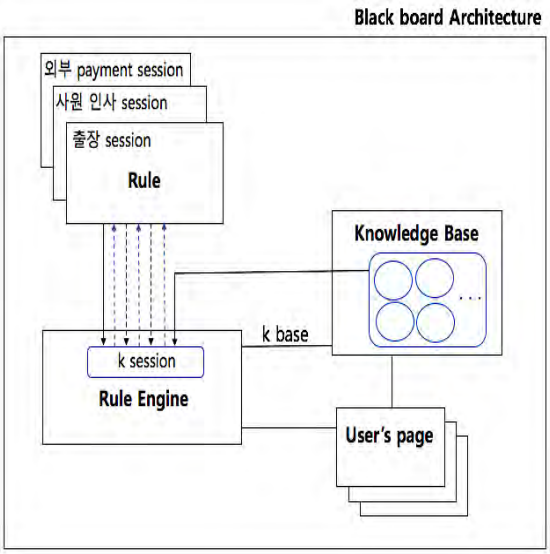

그림 1. Drools 내부 동작 원리

3.2 JavaFX의 GUI 인터페이스 구성

JavaFX는 기존 Java의 표준 라이브러리인 Swing의 단점을 보완하기 위해 개발 되었고 훨씬 메모리 면에서 가볍고 확장성이 뛰어나다는 장점을 살려서 구현하게 되었다. 특히 JavaFX는 Swing에 비해 오류가 적고 단 몇 줄만으로 여러 이벤트 리스터를 구현 할 수 있고 또한 깔끔하게 Model과 View, Controller 사이를 분리 해주게 되어서 시스템이 커졌을 때 확장 가능성을 제공해 준다. (MVC 모델), 또한 새로운 애플리케이션을 만들게 될 때 이제는 Swing 또는 SWT는 고려 대상에서 제외 된다. 이제는 JavaFX와 HTML5만이 고민 대상이 될 것이다. HTML5는 다양한 플랫폼(태블릿, 데스크톱, 휴대폰)뿐만 아니라 다양한 OS(리눅스, 윈도우, 맥)를 지원한다. 또한 도움을 얻을 수 있는 다양한 오픈소스와 프레임워크, 대형 개발 커뮤니티가 있다. 하지만 다양한 플랫폼에서 사용할 수 있는 용이성을 갖는 대신 브라우저 내에서만 작동한다는 단점이 있다 JavaFX로 개발을 하려면 유저가 무언가를 설치해야 한다는 단점이 생기지만 만약 문제가 되지 않기만 한다면 JavaFX를 사용하여 브라우저를 벗어나 Java의 성능을 한층 더 활용 할 수 있기 때문에 여기서는 JavaFX를 택했다. 여기서 FXML은 인터페이스를 정의하기 위한 XML Markup Language이다[4]. 이 프로젝트에서는 Scene builder를 사용하여 편리하게 인터페이스를 구성하였다. 아래 그림 2을 보면 MainApp 클래스의 소스코드 중 일부분 이다. RootLayout.fxml 이라는 layout의 인터페이스를 scene builder로 만든 후 FXMLLoader 에 연결하는 코드이다.

```
/**
 * 상위 레이아웃을 초기화한다.
 */
public void initRootLayout() {
    try {
        // fxml 파일에서 상위 레이아웃을 가져온다.
        FXMLLoader loader = new FXMLLoader();
        loader.setLocation(MainApp.class.getResource("view/RootLayout.fxml"));
        rootLayout = (BorderPane) loader.load();

        rootLayout.setMinSize(1000, 830);
        // 상위 레이아웃을 포함하는 scene을 보여준다.
        Scene scene = new Scene(rootLayout);
        primaryStage.setScene(scene);
        primaryStage.show();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

그림 2. MainApp 클래스의 일부 소스코드

3.3 JavaFX property 사용

JavaFX에서는 property를 모델 클래스의 모든 필드에서 사용하는 것은 흔한 일이다. property는 예를 들면 어떤 변수 값이 바뀌면 자동으로 알려주어서 뷰 와 데이터가 계속 일치하게 유지시켜 준다. 또한 환자 리스트를 관리하려면 환자의 정보에 변경이 일어나면 JavaFX 뷰 클래스에 바로 알려야 한다. 뷰와 데이터는 항상 일치해야 하기 때문에 컬렉션 클래스를 도입한다. 예를 들면 Observablelist<Patient> person = FXCollection.observableArrayList 로 쓸 수 있다.
아래의 그림 3은 Patient 클래스의 변수를 property로 정의한 소스 코드 일부이다.

```
private final StringProperty number;
private final StringProperty name;
private final StringProperty sex;
private final IntegerProperty age;
private final StringProperty height;
private final IntegerProperty weight;
private final ObjectProperty<LocalDate> birthday;
```

그림 3. PatientModel 클래스의 소스코드 일부

3.4 JavaFX 의 Controller

FXML[4]로 인터페이스를 정의했다면 화면에 보여주기 위해 컨트롤러가 필요하다. 컨트롤러는 뷰 와 모델 사이에서 데이터 관리를 중재하는 역할을 한다. 또한 아래 그림 4은 MainApp 클래스에서 컨트롤러에 연결해서 다른 환자 리스트에 대한 정보를 연결 할 수 있게 해주었다.

```
// 메인 애플리케이션이 컨트롤러를 이용할 수 있게 한다.
PatientController controller = loader.getController();
controller.setMainApp(this);
```

그림 4. MainApp 와 Controller 연결

4. Medical Expert System 구현 결과

본 논문의 제안된 Medical Expert System에서 그림 5는 JavaFX로 구현된 초기 인터페이스이다. 환자들이 이름별로 구분되어 있고 환자들 이름을 각각 클릭하게 되면 이벤트를 발생을 해서 옆에 Person details 에 환자의 신상정보를 보여주게 된다. 그리고 설문 결과와 그 환자의 증상이 입력이 된다. 여기서 의사가 직접 환자를 추가하거나 신상정보를 변경 할 수 있고, 또한 증상들을 개별적으로 추가 할 수 있도록 구현하였다. 그림 6은 New 버튼을 누르게 되면 나타나는 화면이다. 그림과 같이 환자의 여러 신상정보를 입력 후 그 환자의 증상들을 직접 추가해 주는데 오른쪽에 TreeView 로 각 부위 별로 카테고리를 만들어서 분류 해 두었다. 그래서 직접 증상을 추가 해 줄 수 있다. 또는 검색기능을 이용하여 증상을 추가 할 수도 있게 구성하였다. 또한 Edit 버튼을 누르게 되면 똑같은 화면이 나오게 되지만 기존에 가지고 있었던 증상들을 그대로 보여주면서 수정할 수 있게 구현 하였다. 또한, Delete 버튼은 환자의 정보를 삭제 시킬 수 있도록 하였다.

증상들이 입력이 완료가 되면 FireAll 버튼을 클릭하게 되면 Drools 가 작동을 하게 된다. 여기서 drl 파일에 미리 설정 해둔 룰(rule)들을 검토 한 후 조건에 충족하는 결과 값을 도출해 낸다. 아래 그림5를 보게 되면 이러한 증상들에 해당하는 진단 결과는 감기이고 미리 정해둔 처방의 결과가 나오게 된다.

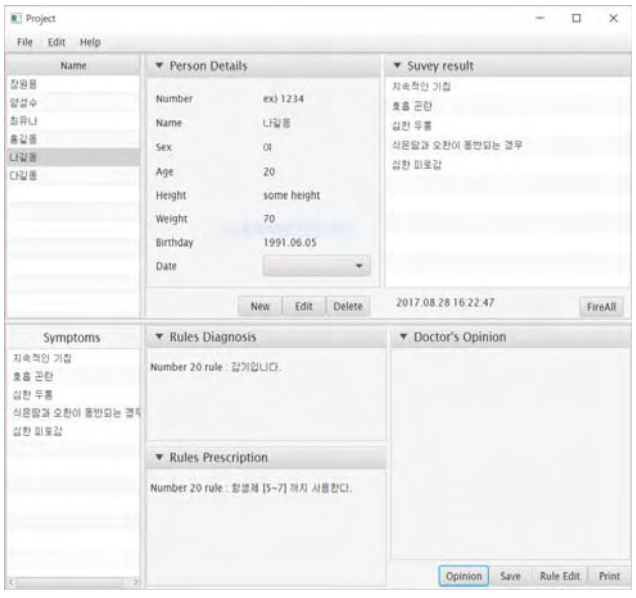

그림 5. JavaFX로 구현된 시스템 인터페이스

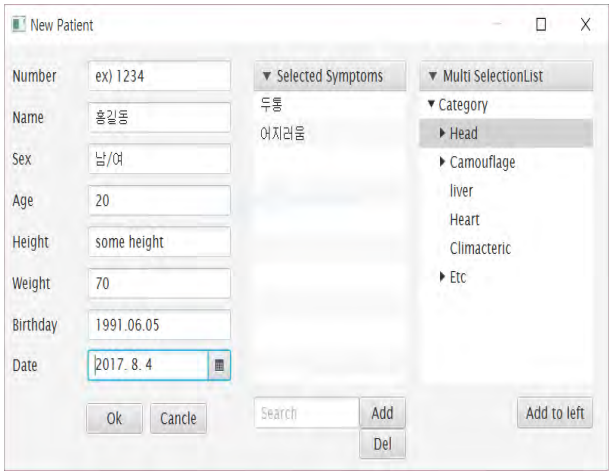

그림 6. New 버튼 클릭 시

또한, 여기서 의사는 추가로 처방에 대해서 보다 환자의 상황에 맞추어서 처방을 수정을 해 줄 수가 있다. 그림 7과 같이 Opinion 버튼을 누르게 되면 또 다른 화면이 뜨게 되면서 의사 소견을 추가해 줄 수가 있다. 또한 처방으로 항생제를 5~7까지 약을 처방 하도록 하여, 바로 옆 텍스트 필드를 나열해 두었다. 이는 의사가 약을 처방할 때 약의 정량을 편하게 보기위해 구현해 놓았고, 텍스트 필드에는 각각의 이벤트를 걸어두어서 텍스트 필드에 값을 입력을 하면 그 값이 바로바로 SUM 에 모두 더해진 값이 보여 지도록 하여서 확인 할 수 있다. 만약 처방 할 약의 종류가 여러 가지일 때는 여러 줄의 처방이 나오고 그에 따른 텍스트필드에 정량 값을 입력하면 된다. 또한 약 처방에 따른 처방 기간을 텍스트 필드에 입력 할 수 있도록 구현하였다.

그림 7. Opinion 버튼을 클릭 시 화면

마지막으로 모든 처방이 완료 되고 의사가 확인 단계에서 OK 버튼을 클릭하게 되면 그림 8와 같이 약의 정량과 처방 기간, 날짜, 추가적 의사 소견 등을 종합하여 출력을 해주게 된다.

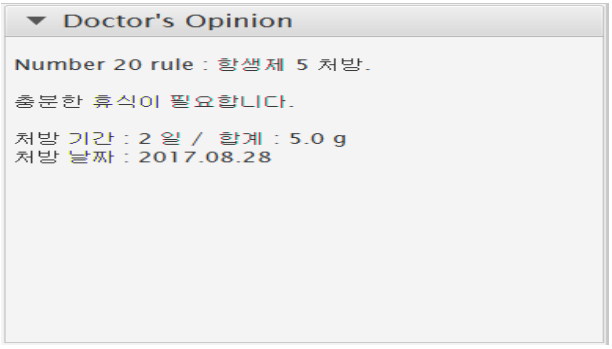

그림 8. 최종 의사 처방 화면

5. 결론

본 논문은 Medical System을 효율적으로 설계 및 구현하기 위하여 Drools의 내부 동작 과정을 정확히 이해함으로써 Drools 기반으로 한 병원의 증상 진단 지원 Expert System을 설계 및 구현 하였다. 이 시스템의 인터페이스를 구성할 때 JavaFX를 이용하였기 때문에 scene builder를 이용하여 기존의 자바 설계에서 쓰던 Swing 의 단점인 메모리 및 여러 버그 등의 단점을 보완하고 풍부한 기능과 확장성 및 연동 가능성 등의 장점들을 부각 시켜서 시스템의 성능과 효율을 높일 수 있었다. 본 연구를 통하여 메디컬 Expert System의 사용 시 프로세스에 따른 사용자 인터페이스를 정립 및 구현을 하였다.

참고문헌

[1] http://www.jboss.org/jboss.

[2] 박종문, 안형배, 이명준, “JCAF과 DROOLS를 이용한 상황 인식 어플리케이션 생성 도구” (A Toolkit for Generating Context-Aware Applications with JCAF and Drools), 한국정보과학회 학술발표논문집, Vol. 39, No. 6, pp 254-256, 2012.

[3] Jan Ruzicka, “Integrating DROOLS and R software for intelligent map system”, Geoinformatics FCE CTU, Vol 7, pp 85-92, 2012.

[4] https://www.tutorialspoint.com/javafx/javafx_event_handling.htm

[5] http://www.oracle.com