http://dx.doi.org/10.5392/JKCA.2012.12.06.092

안드로이드 기반의 웹 서비스 프로토콜 커스터마이제이션 기법

Customization Technique of Web Service Protocol based on Android

김철진
인하공업전문대학 컴퓨터시스템과

Chul-Jin Kim(cjkim@inhatc.ac.kr)

요약

모바일 어플리케이션 규모는 급성장하고 있으며, 이에 따라 모바일과 웹 서비스와의 결합도가 높은 어플리케이션들이 증가하고 있다. 이러한 모바일 어플리케이션의 증가는 가변성을 고려한 개발이 이루어져야 함을 의미한다. 현재 웹 서비스와 연동하는 모바일 어플리케이션을 변경할 경우 어플리케이션 전체를 재설치 해야 한다. 그러나 이러한 재설치는 결합도가 큰 어플리케이션인 경우 부작용이 발생할 가능성이 높다. 따라서 본 논문에서는 안드로이드 플랫폼 기반에서 웹 서비스와 연동 시 웹 서비스 프로토콜을 변경하기 위한 커스터마이제이션 기법을 제안한다. 프로토콜 커스터마이제이션 기법은 선택 기법과 플러그인 기법으로 구분한다.

■ 중심어 : | 모바일 서비스 | 웹서비스 프로토콜 | 커스터마이제이션 | 가변성 |

Abstract

According to the scale of mobile applications has been expanded, the high coupled application combined mobile and web service are growing. The growth of mobile application's size means that predicting design for variability should be involved. If mobile application's change is occurred, application should be reinstalled totally. However this reinstallation can raise side-effects in case of high-coupling application. Therefore, this paper proposes a technique of customization for changing web service protocol as the mobile applications are connected with web service in android platform. Proposed protocol customization technique is consist of selection and plug-in technique.

■ keyword : | Mobile Service | Web Service Protocol | Customization | Variability |

I. 서 론

어플리케이션의 시장 규모가 폭발적으로 증가하는 모바일 어플리케이션들은 대부분 독자적으로 실행되는 스탠드 얼론(Stand Alone) 어플리케이션 이지만, 모바일 어플리케이션들과 웹 서비스와의 연동을 통해 어플리케이션 결합도가 높아지고 있다. 이러한 결합도의 증가는 어플리케이션의 변경에 따른 오류를 발생시킬 수 있는 부작용을 안고 있다. 따라서 어플리케이션을 웹 서비스와 연동 시 효과적으로 변경하기 위한 기법이 요구된다[1][2].

본 논문에서는 모바일 어플리케이션에서 웹 서비스와 연동 시 효과적으로 웹 서비스 프로토콜을 변경하기 위한 커스터마이제이션 기법을 제안한다. 본 논문에서

접수번호 : #120502-009
접수일자 : 2012년 05월 02일
심사완료일 : 2012년 05월 17일
교신저자 : 김철진, e-mail : cjkim@inhatc.ac.kr

제안하는 커스터마이제이션 기법으로는 프로토콜 선택 기법과 프로토콜 플러그인 기법으로 구분하여 제안한다.

본 논문의 구성으로 2절에서는 관련연구로 가변성과 안드로이드 프레임워크 구조에 대해 설명하며, 3절에서는 본 논문의 핵심인 프로토콜 커스터마이제이션 기법을 선택 기법과 플러그인 기법으로 구분하여 제안한다. 4절에서는 3절에서 제안한 기법을 웹 서비스 프로토콜 변경 실 사례에 적용하여 적합성을 검증한다.

II. 관 련 연 구

1. 가변성

가변성(Variability)은 공통성(Commonality)과 함께 제품군(Product Family)의 요소[3]로서 제품군 멤버들 사이의 차이[4]를 말한다. TV 제품군에서 가변성은 화면 사이즈가 되며, 공통성은 화면 분사 방식이나 방송파 수신 방식 등이다. 이와 같이 가변성은 동일 기능을 하는 제품군의 멤버들 간의 차이로서 컴포넌트에서 가변성은 소프트웨어 구성 요소와 일치하는 속성, 오퍼레이션, 그리고 메시지의 흐름이 된다.

행위(Behavior) 가변성은 동일 컴포넌트 멤버들 간에 동일 기능을 하는 오퍼레이션이 서로 다른 알고리즘으로 기능을 제공하는 경우에 행위 가변성이라고 한다. 예를 들면, 은행 도메인의 계좌 컴포넌트에서 이자 계산 기능은 은행 마다 존재하지만 서로 다른 알고리즘으로 기능을 제공하기 때문에 은행 마다 이자 계산의 기능은 각각 다르다. 따라서 이자 계산은 행위 가변성에 해당한다.

워크플로우(Workflow)[5][6] 가변성은 동일 컴포넌트 멤버들 간에 동일 기능이 서로 다른 메시지 흐름을 제공하는 경우에 워크플로우 가변성이라고 한다. 예를 들면, 회원 등록하는 회원 컴포넌트에서 회원 입력, 회원 인증, 데이터 베이스 저장의 프로세스를 요구할 수도 있지만 특정 도메인에서는 데이터 베이스 저장이 아니라 레거시(Legacy) 시스템으로 연동되는 프로세스로 요구할 수 있기 때문에 동일한 회원 등록에 대해 다른 워크플로우를 요구하는 경우에 워크플로우 가변성이라고 한다.

속성(Attribute) 가변성은 동일 컴포넌트 멤버들 간에 동일 역할의 속성이 서로 다른 타입이나 값을 가지는 경우 속성 가변성이라고 한다[7][8]. 예를 들면 동일 컴포넌트에서 날짜를 나타내는 속성이 도메인의 요구사항에 따라 문자 타입이나 날짜 타입으로 요구할 수 있는데, 이러한 경우를 속성 가변성이라고 할 수 있다.

2. 안드로이드 어플리케이션 프레임워크

안드로이드의 어플리케이션 프레임워크는 [그림 1]의 컴포넌트 재사용 프레임워크처럼 안드로이드 플랫폼 상에서 모바일 컴포넌트(어플리케이션 또는 액티비티)를 유연하게 조합하여 다양한 서비스를 제공하기 위한 프레임워크이다. [그림 1]에서와 같이 가변부(Variation Point)에 대해 설정하기 위해 인텐트(또는 인텐트 필터) 라는 메타정보를 이용한다.

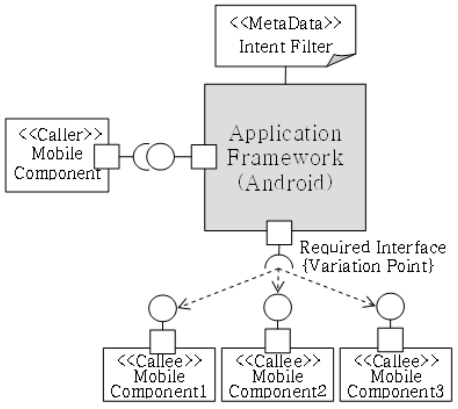


그림 1. 안드로이드 프레임워크 컴포넌트 다이어그램

안드로이드 플랫폼 상에서 재사용 프레임워크 기능을 제공하기 위해 어플리케이션 프레임워크 내의 액티비티 관리자(Activity Manager)와 패키지 관리자(Package Manager)가 핵심적인 역할을 한다.

[그림 2]와 같이 액티비티 관리자는 컴포넌트로부터 가변적으로 호출되는 데이터 정보를 인텐트를 통해 전달하며, 패키지 관리자는 인텐트 필터에 정의된 가변적인 호출 컴포넌트 정보를 필터링하여 해당 컴포넌트(어

플리케이션)를 호출한다[3]. 인텐트 필터나 인텐트를 변경하면 액티비티 관리자와 패키지 관리자가 변경된 메타정보에 따라 가변적으로 컴포넌트를 변경한다.

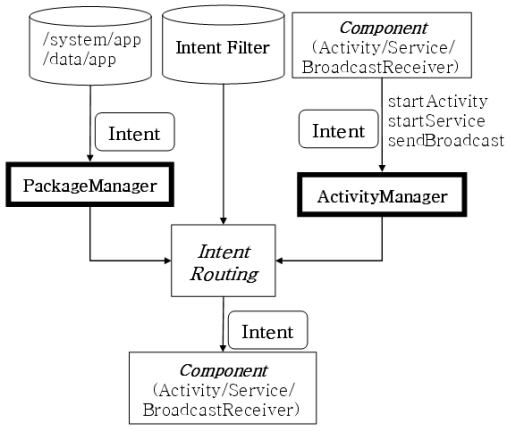


그림 2. 안드로이드 프레임워크 구조

3. 모바일 지능형 추천 에이전트[9]

연구 [9]에서는 모바일 환경에서 지능형 멀티 에이전트를 통해 사용자에게 도움 되는 정보를 능동적으로 제공할 수 있는 시스템을 제안한다. 프로파일 모듈, 규칙 생성 모듈, 필터링 모듈, 서비스 모듈로 구성된다. 추천 에이전트를 이용하여 사용자의 정보를 기반으로 지능적인 사용자의 요구 사항에 대해 파악 가능하도록 구성한다. 연구 [9]는 추천 에이전트에 의한 상황인지 기반 서비스를 제공하는 것으로, 커스터마이제이션에 대한 방안은 제한적으로 연구 되었다.

4. 콘텐츠 추천 시스템 설계 및 구현[10]

연구 [10]은 인터넷에 존재하는 방대한 양의 콘텐츠에서 사용자에게 적합한 콘텐츠를 제공하는 추천 시스템에 대한 설계 및 구현을 제안한다. 사용자의 복합 상황정보를 이용하여 콘텐츠를 검색하고 추천 가중치를 적용하여 콘텐츠를 추천한다. 연구 [10]이 콘텐츠를 상황정보에 의해 추천하는 것이며, 본 논문의 가변성 정보를 통해 적합한 웹 서비스 프로토콜을 추천하기 위한 서비스 변경에 대한 연구는 미흡하다.

III. 웹서비스 프로토콜 커스터마이제이션 기법

안드로이드 플랫폼의 어플리케이션 프레임워크에 기반하여 웹 서비스 프로토콜을 커스터마이제이션 하기 위한 기법으로 선택 기법과 플러그인 기법을 제안한다.

1. 웹서비스 프로토콜 선택기법

선택기법은 모바일 어플리케이션에서 여러 웹 서비스 프로토콜 컴포넌트 중에 하나의 컴포넌트를 선택하여 요구하는 프로토콜을 제공하는 기법이다.

선택 기법은 어플리케이션 내의 프로토콜을 다양하게 제공할 수 있도록 서비스 실행 중에 프로토콜을 변경하기 위한 기법이다. [그림 3]과 같이 “Activity” 서비스에서 선택적으로 “Protocol_001”이나 “Protocol_002” 프로토콜 서비스를 선택할 수 있다. 이러한 서비스의 선택은 서비스 사용자에 의해 선택되며 모바일 서비스 실행 중에 웹 서비스 프로토콜 변경이 가능하다.

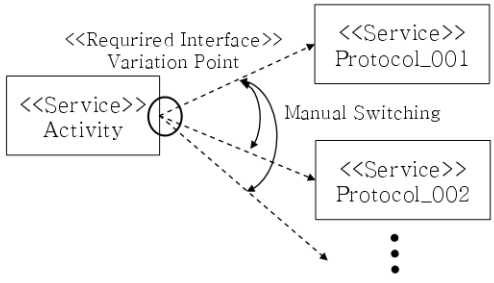


그림 3. 선택기법 구조

안드로이드 플랫폼 기반의 선택 기법을 위한 클래스 다이어그램은 [그림 4]와 같다. “Application Framework”은 안드로이드 플랫폼 상에서 어플리케이션의 라이프사이클을 관리하거나 컴포넌트를 변경하기 위한 메커니즘을 제공한다. 동적인 서비스 구성을 위해 호출 컴포넌트는 인텐트와 인텐트 필터를 포함해야 하며 어플리케이션 프레임워크는 해당 인텐트 필터를 통해 서비스를 선택할 수 있도록 필터링을 해 준다.

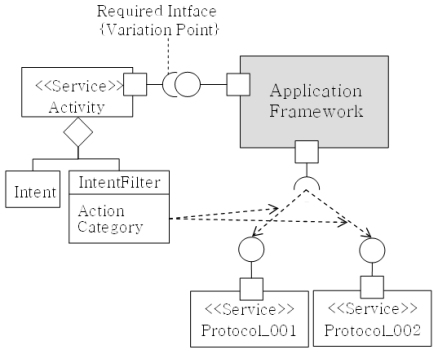


그림 4. 선택기법 클래스다이어그램

선택 기법을 위해 선택 영역을 가변성 영역으로 정의해야 하며, 안드로이드 플랫폼에서 인텐트 필터를 활용하여 정의할 수 있다.

표 1. 선택 기법의 가변성을 위한 인텐트 필터

Filter Type	Filter Value	Etc
Action	android.intent.action.MAIN	
Category	android.intent.category.DEFAULT	Filter for Implict Intent
	com.app.VARIABILITY_NAME	Variability Name

[표 1]에서와 같이 “Action”과 “Category” 필터를 정의할 수 있으며, 선택적으로 호출하기 위한 어플리케이션들은 “Category” 필터에 프로토콜 가변성 식별자(“com.app.VARIABILITY_NAME”)를 정의해야 한다. 또한 호출되는 프로토콜들도 해당 가변성 식별자를 정의해야 한다. [그림 5]은 [표 1]에 대한 가변성 정보를 안드로이드 플랫폼 상에서 설정한 설정 코드이다.

```
...
<activity android:name="Protocol_001" >
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="com.app._VARIABILITY_NAME" />
</intent-filter>
</activity>
<activity android:name="Protocol_002" >
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="com.app._VARIABILITY_NAME" />
  </intent-filter>
</activity>
</application>
...
```

그림 5. 선택 기법의 가변성을 위한 설정

[그림 6]와 같이 인텐트 필터를 적용하여 가변성 부분을 동적으로 선택할 수 있도록 구현할 수 있다. 사용자는 서비스 실행 중에 요구하는 서비스를 선택하여 웹 서비스 프로토콜을 변경할 수 있다.

2. 웹서비스 프로토콜 플러그인(Plug-In) 기법

플러그인 기법은 기존 어플리케이션에 새로운 프로토콜 서비스를 플러그인 하여 새로운 프로토콜을 제공하는 방법이다. 선택 기법이 어플리케이션 내에 존재하는 웹 서비스 프로토콜들을 변경하는 것에 반해 플러그인 기법은 어플리케이션 내에 존재하지 않는 새로운 프로토콜 서비스를 플러그인 하여 새로운 프로토콜을 제공한다.

그림 6. 선택 기법 사례

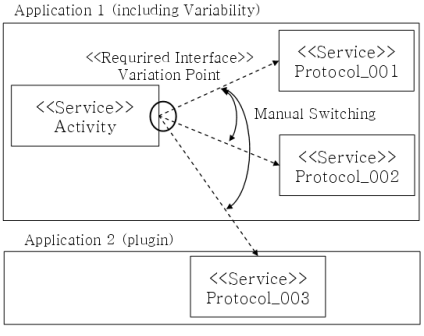


그림 7. 플러그인 기법 구조

[그림 7]에서와 같이 기존 어플리케이션 “Application 1”은 새로운 프로토콜 서비스인 “Protocol_003”을 플러그인 하여 새로운 웹 서비스를 제공할 수 있다. 플러그인 되는 프로토콜 서비스는 어플리케이션 “Application 2” 내에 포함되어 디바이스에 설치된다. “Application 1”과 “Application 2”는 독립된 어플리케이션이므로 독립적으로 설치될 수 있다. “Activity”은 가변부를 통해 “Protocol_001”, “Protocol_002”, “Protocol_003” 프로토콜로 변경이 가능하다.

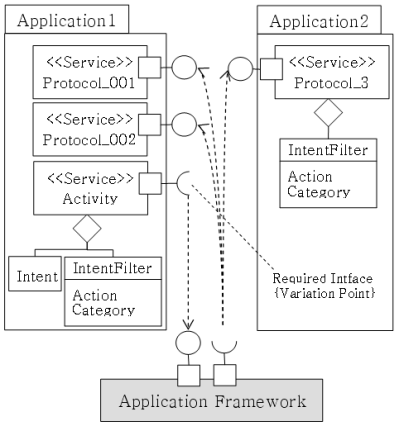


그림 8. 플러그인 기법 구조

안드로이드 플랫폼 기반의 플러그인 기법을 위한 클래스 다이어그램은 [그림 8]과 같다. “Application 1”의 “Activity” 서비스는 안드로이드의 “Application Framework”으로 인텐트 필터를 전달하여 가변부에 해당하는 프로토콜 서비스를 선택할 수 있다. 이때 “Application 1” 내부뿐 만 아니라, 플러그인 된 “Application 2”의 “Protocol_003” 프로토콜 서비스도 선택 가능하다.

표 2. 플러그인 기법의 가변성을 위한 인텐트 필터

Filter Type	Filter Value	Etc
Action	android.intent.action.MAIN	
Category	android.intent.category.DEFAULT	Filter for Implict Intent
	android.intent.category.ALTERNATIVE	Filter for PlugIn
	com.app.VARIABILITY_NAME	Variability Name

플러그인 기법의 가변성 정의는 [표 2]와 같이 인텐트 필터를 통해 정의할 수 있다. 플러그인 기법의 가변성 인텐트 필터는 선택 기법의 가변성 인텐트 필터에 추가적으로 “android.intent.category.ALTERN ATIVE”를 추가해야 한다. “android.intent.category. ALTERNATIVE”는 외부 어플리케이션과의 연동을 위한 인텐트 값 이다. [그림 9]에서와 같이 “category” 타입의 인텐트 값으로 “android.intent.category.ALTERNATIVE”를 설정하면 “Application 2”의 “Protocol_003” 프로토콜 서비스는 외부 어플리케이션에서 호출될 수 있다.

① Application 1's Intent Filter

```
...
<activity android:name="Protocol_002" >
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.DEFAULT" />
<category android:name="com.app._VARIABILITY_NAME" />
</intent-filter>
</activity>
...
```

② Application 2's Intent Filter

```
...
<activity android:name="Protocol_003" >
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.DEFAULT" />
<category android:name="android.intent.category.ALTERNATIVE" />
<category android:name="com.app._VARIABILITY_NAME" />
</intent-filter>
</activity>
</application>
...
```

그림 9. 플러그인 기법의 가변성을 위한 설정

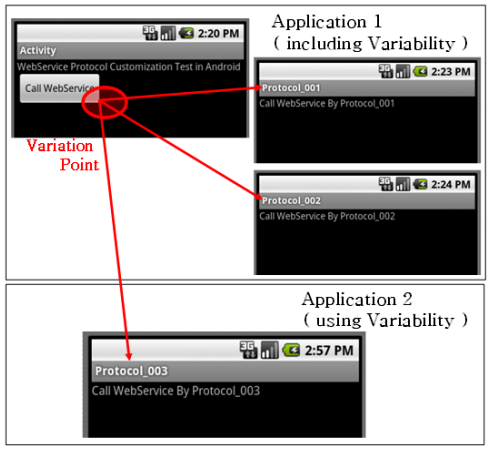


그림 10. 플러그인 기법 사례

[그림 10]은 [그림 9]의 플러그인 설정 정보를 통해 구현된 플러그인 기법의 사례이다. “Activity”에서 내부

프로토콜 서비스인 “Protocol_001”이나 “Protocol_002”를 호출할 수 있으며, 플러그인 된 외부 프로토콜 서비스인 “Protocol_003”을 호출할 수 있다.

IV. 사례 연구

본 논문에서 제안한 모바일 기반의 웹 서비스 프로토콜 커스터마이제이션 기법을 인터넷 뱅킹 사례에 적용하여 적합성을 검증한다. 인터넷 뱅킹을 제공하는 대부분의 웹 서비스가 RESTful(Representational State Transfer) 프로토콜[11]로 전환되고 있으나 기존의 웹 서비스들 중에 SOAP(Simple Object Access Protocol) 프로토콜[12] 형태로 유지하고 있는 서비스들이 있기 때문에 가변적으로 웹 서비스 프로토콜을 변경할 수 있도록 서비스를 제공한다.

인터넷 뱅킹 웹 서비스 프로토콜을 가변적으로 수행하기 위한 구조는 [그림 11]과 같다. 잔액조회나 입금 서비스를 호출하기 위해 기존 “Application 1”에서는 SOAP 프로토콜을 이용하여 제공한다. 그러나, RESTful 프로토콜을 기반으로 하는 서비스로 변경하기 위해서 “Application 2”를 플러그인하여 RESTful 프로토콜 서비스로 변경할 수 있다.

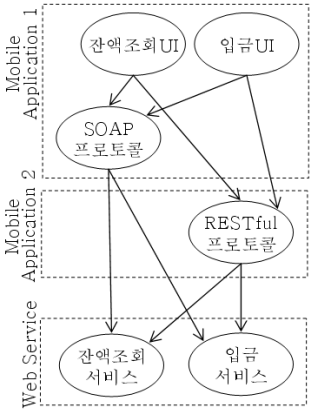


그림 11. 인터넷 뱅킹의 가변적 웹 서비스 시나리오

구축 환경은 SOAP 프로토콜 서비스를 위해 안드로이드용 SOAP 어뎁터로 ksoap[13]과 SOAP 서버로 아파치 Axis2[14]를 사용하였다. RESTful 프로토콜 서비스를 위해서는 웹 서버로 아파치 Tomcat과 JSP를 기반으로 구축되었다.

가변적 웹 서비스 프로토콜을 적용한 인터넷 뱅킹 클래스 다이어그램은 [그림 12]와 같다. 웹 서비스 호출을 위해 어플리케이션 내부 및 외부에서 호출 할 수 있도록 인텐트 필터를 포함하며, 안드로이드 플랫폼의 “Application Framework”을 통해 내부 및 외부에 존재하는 웹 서비스 프로토콜을 호출 할 수 있다.

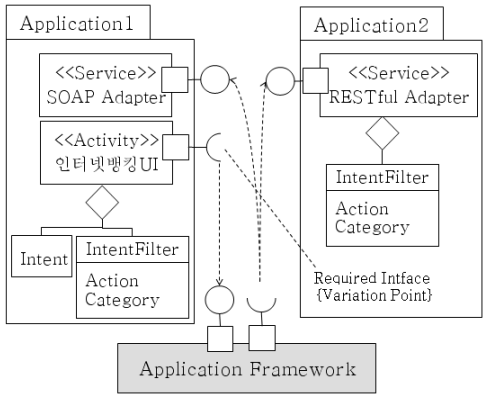


그림 12. 인터넷 뱅킹 모바일 웹 서비스 클래스 다이어그램

[그림 13]은 SOAP 서비스 프로토콜을 가변적으로 서비스하기 위한 인텐트 필터 정보이다. 인텐트 필터 값 “banking.deposite.variability.PROTOCOL”은 입금 서비스 프로토콜을 가변적으로 처리하기 위한 가변성 식별자이며, “android.intent.category.ALTERN ATIVE”는 플러그인 된 외부 서비스 프로토콜과의 가변적 처리를 위한 식별자이다. RESTful 서비스를 위한 설정정보 또한 SOAP 서비스 설정 파일과 동일하며 RESTful 서비스가 플러그인 될 경우 선택기법을 통해 두 프로토콜 중에 한 개의 프로토콜을 선택하여 서비스를 제공받을 수 있다.

```
<activity android:name="Account_deposite_SOAP"
                android:label="Account_deposite_SOAP">
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.ALTERNATIVE" />
    <category android:name="banking.deposit.variability.PROTOCOL" />
  </intent-filter>
</activity>
<activity android:name="Account_getBalance_SOAP"
                android:label="Account_getBalance_SOAP">
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.ALTERNATIVE" />
    <category android:name="banking.getBalance.variability.PROTOCOL" />
  </intent-filter>
</activity>
```

그림 13. SOAP 서비스 설정 파일

```
private static final String DEPOSIT_SOAP_ACTION = "urn:deposit";
private static final String DEPOSIT_METHOD_NAME = "deposit";

private static final String NAMESPACE = "http://ws.apache.org/axis2";
private static final String URL
        = "http://192.168.171.75:8080/webservice/services/Account";
...
private String deposit(String id, String money) {
 SoapObject Request
    = new SoapObject(NAMESPACE, DEPOSIT_METHOD_NAME);
 Request.addProperty("id", id);
 Request.addProperty("money", money);

 SoapSerializationEnvelope soapEnvelope
    =  new SoapSerializationEnvelope(SoapEnvelope.VER11);

 soapEnvelope.setOutputSoapObject(Request);

 AndroidHttpTransport aht = new AndroidHttpTransport(URL);
 SoapPrimitive resultString = null;
 try {
  aht.call(DEPOSIT_SOAP_ACTION, soapEnvelope);
  resultString = (SoapPrimitive)soapEnvelope.getResponse();
 } catch (Exception e) {
  e.printStackTrace();  }
  return resultString.toString();
}}
```

그림 14. SOAP 어뎁터(ksoap기반)(입금)

```
private String deposit(String id, String money) {
 String checkMessage = null;
 HttpGet mGetMethod =
 new HttpGet("http://192.168.171.75:8080
            /webservice/Account?id="+id+"&money="+money);
 try {
  HttpResponse mHttpRes = httpClient.execute(mGetMethod);
  Header[] headers = mHttpRes.getHeaders("balance");
  checkMessage = headers[0].getValue().toString();
 } catch (Exception e)
  {..}
 return checkMessage;
}}
```

그림 15. RESTful 어뎁터(입금)

[그림 14]는 ksoap을 기반으로 안드로이드에서 입 SOAP 프로토콜을 사용하여 입금 서비스를 호출하기 위한 어뎁터 코드이며, [그림 15]는 RESTful 프로토콜을 사용하여 입금 서비스를 호출하기 위한 어뎁터 코드이다. [그림 14]와 [그림 15]는 서로 다른 웹 서비스 프로토콜이지만, [그림 16]과 같은 동일한 입금 웹 서비스를 호출할 수 있다.

```
public String getBalance(String id) throws ClassNotFoundException {
  String balance = null;
  Class.forName("com.mysql.jdbc.Driver");
  try {
    String url = "jdbc:mysql://localhost/websoftdb";
    Connection conn = DriverManager.getConnection(url, "root", "1234");
    Statement stmt = conn.createStatement();
    String strSQL = "select balance from account where id = '" + id
                                                        + "';";
    ResultSet rs = stmt.executeQuery(strSQL);

    rs.next();
    balance = rs.getString("balance");

    stmt.close();
    conn.close();
  } catch (SQLException ex) {   }
 return balance;
 }
}
```

그림 16. 입금 웹 서비스 코드

Banking_RESTful_WebService_PlugIn.apk
패키지 설치 후

그림 17. 플러그인 기법을 통한 웹 서비스 프로토콜 커스터마이제이션

[그림 17]은 인터넷 뱅킹 웹 서비스 프로토콜을 가변적으로 적용하여 구현한 결과이다. SOAP 프로토콜이나 RESTful 프로토콜을 가변적으로 선택하더라도 웹 서비스에는 영향을 받지 않으며 선택적으로 웹 서비스 프로토콜을 가변적으로 적용할 수 있다.

위의 사례를 통해 안드로이드 모바일 플랫폼에서 웹 서비스 프로토콜을 가변적 적용할 수 있음을 검증하였으며, 제안된 웹 서비스 프로토콜 선택 기법과 플러그인 기법을 통해 모바일 웹 서비스 프로토콜의 유연한 가변성 처리가 가능함을 알 수 있다.

V. 결론 및 향후 연구

모바일 어플리케이션은 웹 서비스와 연동을 통해 서

비스를 제공하므로 결합력이 높아지고 있다. 또한 웹 서비스 형태도 다양화되고 있다. 이에 따라 모바일 웹 서비스 변경에 대한 부하를 줄이기 위해 본 논문에서는 웹 서비스 프로토콜을 커스터마이제이션하기 위한 선택 기법 및 플러그인 기법을 제안한다. 사례연구에서는 SOAP 프로토콜과 RESTful 프로토콜을 가변적으로 변경하는 실 사례를 통해 적합성을 검증하였습니다. 향후 본 모바일 커스터마이제이션 기법을 활용하여 자동화 업그레이드를 위한 스마트 업그레이드에 대한 연구를 진행할 예정이다.

참 고 문 헌

[1] B. Konig-Ries and F. Jena, "Challenges in Mobile Application Development," it-Information Technology, Vol.52, No.2, pp.69-71, 2009.

[2] Android Developers, http://developer.android.com/index.html (accessed January 31, 2011).

[3] G. Heineman and W. Councill, *Component - Based Software Engineering*, Addison-Wesley, 2001.

[4] C. Atkinson, J. Bayer, C. Bunse, E. Kamsties, O. Laitenberger, R.Laqua, D. Muthig, B. Paech, J. Wust, and J. Zettel, *Component-based Product Line Engineering with UML*, Pearson Education Ltd, 2002.

[5] C. J. Kim and S. D. Kim, "A Component Workflow Customization Technique," Korea Information Science Society, Vol.27, No.5, 2000.

[6] C. J. Kim, S. H. Lee, and E. S. Cho, "A Framework for Improving Reusability of Home Network System," Vol.1, No.2, ITIRC, 2008(9).

[7] C. J. Kim, E. S. Cho, and C. Y. Song, "A Design Technique of Configurable Framework for Home Network Systems", 한국산학기술학회 논문지, 제12권, 제4호, pp.1844-866, 2011.

[8] 조은숙, 김철진, 이숙희, "임베디드 소프트웨어를 위한 프레임워크의 재사용성 메트릭에 관한 연구", 한국산학기술학회 논문지, 제12권, 제11호, pp.5252-5259, 2011.

[9] 김만선, 주복규 "모바일 환경을 위한 지능형 추천 에이전트에 관한 연구", 한국콘텐츠학회논문지, Vol.6, No.4, pp.55-62, 2006.

[10] 이락규, 피준일, 박준호, 복경수, 유재수, "모바일 환경에서 콘텐츠 추천 시스템 설계 및 구현", 한국콘텐츠학회논문지, Vol.11, No.12, pp.40-51, 2011.

[11] www.w3.org/TR/soap/

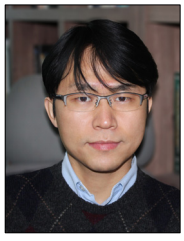
[12] https://www.ibm.com/developerworks/webservices/library/ws-restful/

[13] http://code.google.com/p/ksoap2-android/

[14] http://projects.apache.org/projects/axis2.html

저 자 소 개

김 철 진(Chul-Jin Kim) 정회원



- 2004년 : 숭실대학교 컴퓨터정보공학부(공학박사)
- 2004년 ~ 2009년 : 삼성전자 책임연구원
- 2009년 ~ 현재 : 인하공전 교수

<관심분야> : 클라우드 컴퓨팅, 웹 서비스, 모바일 서비스, 제품공학, 커스터마이제이션