IHE MHD 메타데이터와 IHE XDS.b 메타데이터의 자동 변환

전형석, 이장수, 도형호, 이성기
경북대학교 IT대학 컴퓨터학부
e-mail:pebble876@gmail.com

Automatic Conversion between MHD Metadata and XDS.b Metadata

Hyeongseok Jeon, Jangsoo Lee, Dongwook Park, Hyoungho Do, Sungkee Lee
School of Computer Science and Engineering,
Kyungpook National University

요 약

IHE(Integrating the Healthcare Enterprise)의 XDS.b(Cross-Enterprise Document Sharing.b) 프로파일은 의료기관 간 환자의 전자진료기록을 공유하기 위해 개발되었으나 모바일 환경을 고려하지 않았다. 이를 보완하고자 IHE에서는 모바일기기에서도 전자진료기록을 공유할 수 있는 MHD(Mobile access to Health Documents) 프로파일을 개발했다. XDS.b 메타데이터는 ebXML(electronic business XML)을 사용하고 MHD 메타데이터는 JSON(JavaScript Object Notation)을 사용하므로 MHD와 XDS.b를 연동하기 위해서는 MHD 메타데이터를 XDS.b 메타데이터로 변환이 필요하다. 본 논문에서는 MHD 메타데이터를 XDS.b 메타데이터로 변환하는 규칙을 정의했다.

1. 서론

IHE(Integrating the Healthcare Enterprise)는 다양한 의료정보시스템의 상호운용성 표준 프로파일들을 정의·출판하는 기관이다. IHE에서는 의료기관 간 환자의 전자진료기록을 등록 및 공유하기 위해 통합 프로파일인 XDS.b(Cross- Enterprise Document Sharing.b)를 개발하였다[1]. 그러나 XDS.b는 모바일 환경을 고려하지 않았기 때문에 이를 보완하기 위해 IHE는 MHD(Mobile access to Health Documents) 프로파일을 개발하여 모바일 기기에서도 환자의 전자진료 기록을 쉽게 관리할 수 있도록 했다[2]. MHD는 XDS.b Infrastructure와 연결해서 사용할 수 있고 XDS.b의 메타데이터는 ebXML (electronic business XML) 형식을, MHD의 메타데이터는 JSON(JavaScript Object Notation) 형식을 따른다. 따라서 XDS.b와 MHD가 결합하여 사용될 경우, 상호 간에 의료정보의 공유 및 등록을 위해서는 ebXML과 JSON 상호 간에 데이터를 변환해 줄 수 있는 인터페이스가 필요하다. 본 논문에서는 MHD의 메타데이터를 XDS.b의 메타데이터로 자동 변환해주는 인터페이스 모듈을 구현하는 방법을 제안한다. 본 논문의 구성은 다음과 같다.

2장에서 관련 연구를 소개하고, XDS.b 메타데이터 형식인 ebXML과 MHD 메타데이터 형식인 JSON에 대해 기술한다. 3장에서는 MHD 메타데이터를 XDS.b 메타데이터로 변환하기 위한 인터페이스 모듈을 구현하는 방법에 대해서 기술하고, 4장에서 논문의 결론을 맺는다.

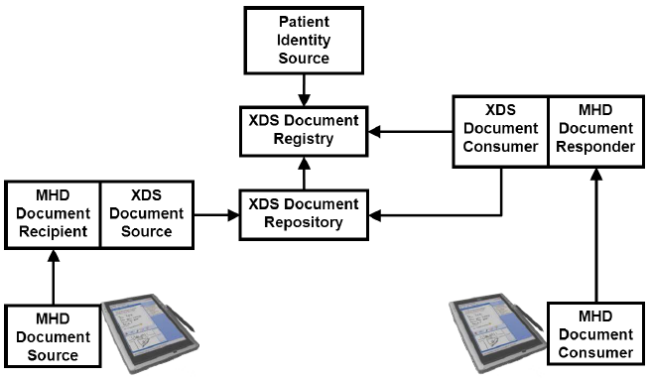
2. 관련 연구

2.1 IHE XDS.b와 IHE MHD

XDS.b 프로파일은 IHE에서 전자진료기록을 저장하고 교류하기 위해 정의된 표준 구현 가이드라인이며 메타데이터 포맷으로 ebXML을 사용한다. Document Source, Document Consumer, Document Repository, Document Registry, Patient Identity Source 로 구성된 5개의 액터와 5개의 트랜잭션으로 정의되어 있다. 액터는 정보를 공유하는 주체로써 시스템에서 사람이나 소프트웨어에 해당하며 각각의 전자진료기록 관리에 필요한 트랜잭션을 수행한다.

MHD 프로파일은 모바일 기기에서 전자진료기록을 저장하고 공유하기 위해 IHE에서 출판한 표준개발 가이드라인으로서, 모바일 기기에서도 전자진료기록을 교류하기 쉽도록 XDS.b와 유사한 환경을 조성하기 위한 HTTP 인터페이스를 정의하고 있다. 메타데이터 포맷으로 JSON을 사용한다. Document Source, Document Recipient, Document Consumer, Document Responder로 구성된 4개의 액터와 4개의 트랜잭션으로 구성되어 있다. Document Recipient,

Document Responder는 독립적으로 전자진료기록을 관리하거나 혹은 그림 1과 같이 XDS.b와 결합하여 사용된다.



<그림 1> XDS.b와 결합된 MHD

2.2 ebXML과 JSON

ebXML은 비즈니스 거래를 위해 개발된 표준 기술이다. 인터넷 환경에서 데이터 통신의 상호운용성을 보장하여 규모나 위치와 관계없이 안전한 비즈니스 거래가 이루어지도록 설계되었다. XML 기반의 마크업 언어로 다양한 형태의 정보를 트리구조로 저장한다[3].

JSON은 작은 크기의 데이터를 교환하기 위해 만들어진 표준 기술이다. JSON의 포맷은 자바스크립트를 따르지만, 자바스크립트 외에도 다양한 언어에서 사용된다. JSON은 문자열과 숫자, boolean, null 값을 표현할 수 있고 객체 또는 배열 구조로 이루어진다. JSON에서 객체는 중괄호의 쌍으로 둘러싸인 순서가 존재하지 않는 값들의 집합이다. 배열은 대괄호의 쌍으로 둘러싸인 순서가 존재하는 값들의 집합이다[4].

3. MHD 메타데이터를 XDS.b 메타데이터로 변환하기 위한 인터페이스 모듈 구현

JSON에서 XML로 자동 변환하는 다양한 라이브러리가 존재한다. 그러나 MHD 메타데이터를 XDS.b 메타데이터로 변환할 경우 attribute에 따라 문법변환방법이 다르다. 따라서 attribute 별로 적합한 변환 규칙을 찾고, 이를 토대로 메타데이터 포맷 변환이 이루어져야 한다.

3.1 변환 규칙

전자진료기록의 메타데이터는 Document Entry, Submission Set, Folder로 나뉜다. 본 논문에서는 메타데이터를 구성하는데 필수적으로 포함되어야 하는 Document Entry과 Submission Set의 attribute들만을 변환 대상으로 한다.

메타데이터를 구성하는 attribute들의 목록은 MHD와 XDS.b가 같으며 변환 규칙별로 attribute를 분류할 수 있다. 표 1은 Document Entry와 Submission Set에 속하는 attribute를 변환 규칙별로 분류한 것이다. 변환 규칙의 이름은 임의로 정한 것이다.

<표 1> attribute 별 변환 규칙

Attribute 이름	변환 규칙
classCode confidentialityCode formatCode practiceSettingCodes healthcareFacilityTypeCode typeCode contentTypeCode	Coded Value
languageCode uniqueId creationTime submissionTime sourcePatientID	Slot
patientId sourceId	ExternalIdentifier

Coded Value 변환 규칙은 표 2과 같다.

<표 2> Coded Value 변환 규칙

JSON 표현

"attribute 이름" : {
"code":"String1",
"codeName":"String2"
"codingScheme":"String3",
}

ebXML 표현
<rim:Classification classificationScheme="UUID" nodeRepresentation="String1"> <rim:Name> <rim:LocalizedString value="String2"/> </rim:Name> <rim:Slot name ="codingScheme"> <rim:ValueList> <rim:Value> String3 </rim:Value> </rim:ValueList> </rim:Slot> </rim:Classification>

XDS.b 메타데이터에서 UUID(Universally unique identifier)는 attribute 별로 지정된 고유한 값이다.

Slot 변환 규칙의 대상이 되는 attribute들은 JSON에서 이름과 값의 쌍으로만 이루어진 간단한 구조를 가진다. Slot 변환 규칙은 표 3과 같다.

<표 3> Slot 변환 규칙

JSON 표현
"attribute 이름" : "Value"
ebXML 표현
<rim:Slot name="attribute 이름"> <rim:ValueList> <rim:Value> "Value" </rim:Value> </rim:ValueList> </rim:Slot>

표 4는 ExternalIdentifier의 변환 규칙을 보여준다.

<표 4> ExternalIdentifier 변환 규칙

JSON 표현
"attribute 이름" : "Value"
ebXML 표현
<rim:ExternalIdentifier identificationScheme="UUID" ... value = "Value" ... </rim:ExternalIdentifier>

3.2 구현

각 attribute의 이름과 JSON 트리구조에서의 위치를 이용해 MHD 메타데이터를 attribute 별로 파싱 한다. 본 논문에서는 파싱을 위해 JAVA의 json-simple 라이브러리 1.1.1 버전을 이용했다.

파싱한 MHD 메타데이터 attribute가 따르는 변환 규칙을 적용해 XDS.b 메타데이터 attribute를 생성한다.

ebXML 생성을 위해 JAVA의 DOM 라이브러리를 사용했다. 각 변환 공식을 함수로 구현하고, 함수의 매개변수로는 attribute가 MHD 메타데이터 상에서 속한 객체 또는 배열의 이름, attribute를 사용한다. attribute가 MHD 메타데이터 상에서 속한 객체 또는 배열의 이름을 매개변수로 전달해서 같은 이름의 attribute가 documentEntry나 submissionSet에서 각각 존재하더라도 각 attribute의 ebXML 트리 구조상에서의 위치를 알고 생성할 수 있다.

표 5는 구현한 인터페이스를 이용해 Coded Value 변환 규칙을 따르는 classCode를 MHD 메타데이터 형식에서 XDS.b 메타데이터 형식으로 변환한 결과이다.

4. 결론

모바일 기기에서 전자의료정보를 교환하기 위해 개발된 MHD 프로파일은 XDS.b와 연결되어 사용될 수 있지만, 상호 간 메타데이터를 변환하는 변환 규칙은 정의되어 있지 않다. 본 논문에서는 MHD 메타데이터를 XDS.b 메타데이터로 변환하는 변환 규칙을 정의하고 자동 변환 인터페이스를 구현했다. 구현된 인터페이스를 사용해 MHD 메타데이터를 XDS.b 메타데이터로 자동 변환하여 모바일 기기에서 XDS.b Infrastructure로의 접근성을 향상할 수 있었다.

5. 감사의 글

본 연구는 경북대학교 특성화사업의 지원으로 수행되었음.

6. 참고 문헌

[1] IHE. "Cross-Transaction Specifications and Content Specifications", 『IT Infrastructure Technical Frameworks』, Volume 3, 2012

[2] IHE. "Mobile access to Health Documents(MHD)", 『IT Infrastructure Technical Frameworks Supplement』, 2012

[3] ebXML Technical Architecture Project Team, ebXML Technical Architecture Specification v1.0.4, 2001

[4] Crokford, Duglas. "The application/json Media Type for JavaScript Object Notation (JSON)", 2006

<표 5 - classCode의 변환 결과>

MHD 메타데이터	XDS.b 메타데이터

"classCode": {
"code" : "2345-3",
"codingScheme" : "OINK",
"codeName" : "Author garbage"
}
	<rim:Classification
	classificationScheme="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"
	nodeRepresentation="2345-3">
	<rim:Name> <rim:LocalizedString value="Author garbage"/> </rim:Name>
	<rim:Slot name="codingScheme">
	<rim:ValueList> <rim:Value>"OINK"</rim:Value> </rim:ValueList>
	</rim:Slot>
	</rim:Classification