Journal of the Korea Academia-Industrial cooperation Society
Vol. 15, No. 7 pp. 4290-4300, 2014

http://dx.doi.org/10.5762/KAIS.2014.15.7.4290
ISSN 1975-4701 / eISSN 2288-4688

호환 가능한 HL7 파서의 개발

박현상[1], 김화선[2], 조 훈[1*]
[1]경북대학교 의료정보학과, [2]대구한의대학교 IT의료산업학과

Development of Compatible Health Level 7 Parser

Hyun Sang Park[1], Hwa Sun Kim[2], Hune Cho[1*]
[1]**Dept. of Medical Informatics, Kyungpook National University**
[2]**Dept. of Medical Information Technology, Daegu Haany University**

요 약 기존의 HL7 인터페이스는 서로 다른 버전의 HL7 메시지를 처리하기 위해 버전마다 별도의 패키지 또는 변환 모듈로 개발해야 한다. 본 연구는 HL7 V2.5에 정의된 버전 호환성 요구사항을 기반으로 버전 간 호환 가능한 HL7 파서를 설계하고 개발하였다. 파서의 구조는 객체지향 프로그래밍에서의 상속 개념을 이용하여 하위 버전의 HL7 메시지 요소의 클래스 객체를 상위 버전의 클래스 객체가 상속하였다. 따라서 HL7 메시지의 버전에 상관없이 상위 클래스 객체만을 사용하여 모든 버전의 HL7 메시지를 처리할 수 있었다. 개발한 파서의 호환성 평가는 700건의 류마티스 입원 환자 데이터를 이용하였으며 성공적으로 테스트를 수행하였다. 향후에는 버전 간 상호 호환 가능한 HL7 파서의 구조에 대한 연구를 지속적으로 할 것이다.

Abstract The previous HL7 interface should be developed as a separate package or conversion module for each version to process HL7 messages from different versions. This study designed and developed an HL7 parser compatible among different versions based on the requirements of compatibility defined in HL7 V2.5. According to the structure of the parser, the inheritance concept in object-oriented programming was adopted so that the class object of the HL7 message from the lower version could be inherited to the class object of the upper version. Therefore, every version's HL7 messages could be processed using only the upper class' object regardless of the version. To evaluate the compatibility of the developed parser, 700 data sets about inpatients with rheumatoid arthritis were used. The 700 cases underwent the compatibility test successfully. In the near future, further research on the Inter-compatibility HL7 parser is planned.

Key Words : Compatibility, HL7, Object Oriented Programming, Parser

1. 서론

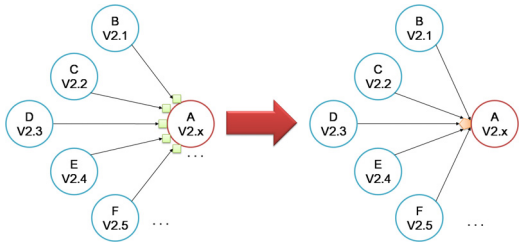
최근 컴퓨터 및 정보통신 기술의 급속한 발전으로 인해 타 분야와 마찬가지로 의료 분야에서도 비용 절감 및 의료 서비스의 질 향상을 위해 전자의무기록(Electronic Medical Record, EMR), 평생전자건강기록(Electronic Health Record, EHR) 등 다양한 의료정보시스템이 개발되고 있다. 그러나 이러한 의료정보시스템들은 각 의료기관 마다 개별적으로 개발되어 있으므로, 병원 간 정보교환 또는 병원 내 독자적인 시스템 간의 통합을 위해서는 정보 교환을 위한 표준이 필요하다. 이에 따라, 이 기종 의료정보시스템 간의 상호운용성(Interoperability)을 위해 HL7(Health Level Seven), DICOM, HL7의 CDA(Clinical Document Architecture), HL7의 CCD(Continuity of Care Document), ASTM의 CCR(Continuity of Care Record)의 다양한 의료정보 표

이 논문은 2013년 경북대학교 학술연구비 및 2013년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2012R1A1A2004829)
*Corresponding Author : Hune Cho(Kyungpook National Univ.)
Tel: +82-53-420-4896 email: hunecho@knu.ac.kr
Received March 11, 2014 Revised May 7, 2014 Accepted July 10, 2014

준이 개발되었다.

현재까지 HL7은 임상정보시스템과 병원정보시스템과 같은 전혀 다른 이 기종 정보 시스템 사이에 있어 의료정보 교환을 위한 최선의 선택으로 알려져 있다[1,2]. HL7을 활용할 경우 각 의료 기관은 자신의 정보 체계에 인터페이스 프로그램만을 개발하여 편리하게 의료기관 간의 의료정보 교환을 자유롭게 시도할 수 있고, 의료정보시스템의 유지 보수도 용이하게 된다[3]. 만약 HL7이 없다면 새로운 시스템이 추가될 때마다 기존 시스템과의 정보 교환을 위해 별도의 인터페이스가 필요하므로 엄청난 개발 비용이 소모될 것이다. HL7은 의료 업무에서 발생하는 대부분의 내용을 표준화하였기 때문에 의료정보 공유를 위한 손쉬운 방법이지만, 방대한 내용과 다양한 선택 사항을 지정해야 되는 문제로 HL7 메시지를 처리하는 업무는 어려운 일이며 오류가 발생할 가능성이 높다[4]. 이러한 문제점을 해결하는 방법으로는 HL7 인터페이스를 이용하는 방법이 있는데, 미국은 이미 대다수 병원 시스템에 공급되어 있고, 호주, 독일, 네덜란드, 일본, 영국, 오스트리아, 이스라엘에서도 사용되고 있다[5]. 대표적인 인터페이스는 Orion사의 Symphonia [6], Neotool 사의 NeoBrowse TCP for HL7 [7], Interfaceware 사의 Chameleon [8], Eversolove 사의 Medi7 [9], HAPI(HL7 Application Programming Interface) [10] 등이 있다. 국내에서는 경북대학교 병원과 전남대학교 병원 간의 퇴원요약정보를 교환 및 공유하는 것을 목표로 국내 최초의 HL7 인터페이스와 웹 기반의 메시지 브라우저를 개발한 사례가 있다[11, 12]. 삼성의료원은 기존 병원정보시스템과 방사선정보시스템을 확장한 의료영상저장전송시스템(Picture Archiving Communication System, PACS) 간의 연동을 위해 HL7을 이용한 사례가 있고[13], 서울대학교 병원과 전남대학교 병원에서 PACS를 도입하면서 기존의 의사처방전달시스템(Order Communication System, OCS)과의 환자 정보 공유를 위하여 HL7을 적용한 사례가 있다[14,15].

그러나 기존에 개발된 HL7 인터페이스는 특정한 한 개의 버전으로 개발되거나 버전마다 별도의 패키지를 가지기 때문에, 서로 다른 버전의 HL7 메시지 처리 결과를 이용하기 위해서 해당 버전의 패키지 또는 버전 간 변환 모듈을 따로 개발하여야 한다. 예를 들어, Fig. 1(a)와 같이 n개의 시스템으로부터 전송된 메시지를 처리하기 위해서는 n개의 패키지 또는 변환 모듈이 필요하다. 이러한 이유로 HL7에서는 특정 버전의 HL7 인터페이스의 재사용성과 상호운용성을 보장하기 위해 V2.3부터 버전 호환성에 대한 요구사항을 정의해오고 있다. 그럼에도 불구하고 기존의 HL7 인터페이스는 이러한 요구사항을 충족하지 않은 채 개발되므로, 새로운 버전이 발표될 때마다 추가적인 비용과 개발 시간이 소요되었다. 즉, HL7 표준의 목적과 일치하지 않기 때문에 이 기종 의료정보시스템 간의 상호운용성을 보장할 수 없다. 그러므로 완벽한 상호 운용성을 위해 Fig. 1(b)와 같이 버전 간 호환 가능한 HL7 파서의 개발이 필요하다.



[Fig. 1] Necessity of compatibility HL7 parser
(a) Non-Compatibility Parser
(b) Compatibility Parser

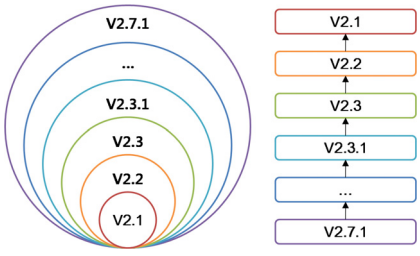
본 논문은 HL7 V2.5에 정의된 버전 호환성 요구사항[16]을 기반으로 버전 간 호환 가능한 HL7 파서의 구조를 설계하고 ADT^A01 메시지 처리를 위한 파서를 개발하였다. 또한 버전 간 HL7 메시지 분석, 변환 및 유효성 검사를 수행하여 개발한 파서를 평가하였다.

2. 재료 및 방법

2.1 HL7 V2 표준 구조

HL7 V2는 1992년 12월 31일에 발표된 V2.1부터 2012년 7월 9일에 발표된 V2.7.1까지 약 10년 동안 10가지 버전이 발표되었으며 현재 새로운 버전이 계속 개발되고 있다. HL7 V2는 특정 버전과의 호환성을 위해 V2.3부터 버전 호환성에 대한 요구사항을 정의하였고, 표준 구조를 Fig. 2(a)와 같이 상위 버전이 하위 버전을 포함되도록 설계하였다. 즉, 상위 버전은 하위 버전에 영향이 안 미치는 범위 내에서 새로운 HL7 메시지 요소를 추가하거나 기존의 요소를 변경 또는 삭제할 수 있다. 이러한 구조는

객체지향 프로그래밍 개념 중 상속과 동일하다. 상속은 자식 클래스가 부모 클래스의 속성(지역 변수, 메소드)을 물려받는 개념으로, 자식 클래스는 부모 클래스의 속성을 물려받고 새로운 속성을 정의하거나 물려받은 부모의 속성을 변경할 수 있다[17]. 만약 자식 클래스들에 중복하여 정의될 공통부분이 부모 클래스에 한번만 정의된다면, 자식 클래스들에서 재사용이 가능하므로 중복 코드의 작성에 드는 시간과 비용을 절감할 수 있다[18]. Fig. 2(b)는 상속 개념을 이용하여 HL7 V2.x 간의 관계를 클래스 다이어그램으로 나타낸 것이다. 즉, 상위 버전의 HL7 메시지 요소 클래스가 하위 버전의 HL7 메시지 요소 클래스를 상속하도록 함으로써, 상위 버전에서는 새로운 요소만 추가하거나 기존 요소만 변경하면 되기 때문에 프로그램 개발에 필요한 시간과 비용을 절감할 수 있다. 결과적으로 상속 구조는 HL7 V2.x 구조와 버전 호환성에 대한 요구사항을 충족하는 가장 적합한 구조라고 할 수 있다.



(a) (b)

[Fig. 2] HL7 V2.x diagram
(a) Venn diagram (b) Class diagram

2.2 국내외 연구 현황

HL7 V2는 현재 미국의 의료 기관 95%가 사용하고 있으며 35개 이상의 HL7 회원국에서도 HL7 V2를 개발하고 있다[19]. 이에 반해 HL7 V3는 시장 규모가 가장 작으며 발달 속도가 더 이상 증가하지 않는 단계에 머물러 있다[20]. 이러한 현상에 대한 원인 중 하나가 HL7 V3는 구조적 의미가 복잡하여 개발이 어렵고 많은 시간이 소요되기 때문이다. 이와 대조적으로 HL7 V2의 시장 규모는 점점 증가하고 있으며 현재 다양한 공개용 라이브러리, 툴킷, 인터페이스 엔진이 개발되었다. 따라서 본 논문에서는 HL7 V2 파서의 최근 연구 현황을 조사하였다.

경북대학교 의료정보원천기술연구소는 사용자 친화적인 GUI를 통해 HL7 인터페이스를 좀 더 쉽게 이용할 수 있는 HL7 Toolkit을 개발하였다. 개발한 툴킷은 기존의 프로그램 전문가가 직접 소스코드를 작성하여야만 이용 가능했던 HL7 인터페이스를 프로그램 비전문가도 GUI를 통한 단순 조작만으로 HL7 메시지와 데이터베이스 사이의 정보 교환이 가능한 방법을 제공한다[21]. 그러나 HL7 Toolkit의 목적은 특정 버전의 HL7 메시지 생성, 처리, 전송이므로 자체적으로 버전 호환 및 변환 기능을 제공해주지 않는다.

유케어소프트는 다양한 도구를 이용하여 HL7 메시지를 처리할 수 있는 HITookit을 개발하였다. HIToolkit은 HL7 메시지 구조를 Tree 또는 List 구조로 출력하여 사용자가 직관적으로 파악할 수 있는 인터페이스, HL7 메시지 생성, 검증, 송 수신 기능과 메시지 구조를 추가, 수정, 삭제 기능을 제공한다[22]. 그러나 HIToolkit은 특정 버전(V2.3.1, V2.4, V2.5)만을 지원해준다.

Merge Healthcare는 저렴하고 기존의 의료정보시스템에 영향이 적은 Merge HL7 Toolkit을 개발하였다. Merge HL7 Toolkit은 간단하게 HL7 메시지를 파싱 및 생성할 수 있으며 MLLP을 사용하여 HL7 메시지를 송수신할 수 있다[23]. 그러나 Merge HL7 Toolkit은 특정 버전(V2.3.1, V2.4, V2.5.1, V2.6)만 지원해준다.

SoftTeam은 다양한 의료정보시스템 또는 모듈을 저렴하고 효율적으로 통합하기 위해 HL7 Tool kit을 개발하였다. HL7 Tool kit은 다양한 종류의 HL7 메시지를 처리할 수 있는 파서와 HL7, XML, ASC와 같은 다양한 형태로 입출력할 수 있는 인코더 및 디코더, TCP/IP로 HL7 메시지를 전송 및 수신할 수 있는 통신 기능을 제공한다[24]. 그러나 HL7 Tool kit은 특정 버전의 HL7 메시지만 지원해준다.

HAPI는 2001년부터 지속적으로 업데이트하고 있는 Java 기반의 공개용 HL7 V2 파서 라이브러리이다. HAPI는 V2.1부터 V2.6까지 다양한 종류의 HL7 메시지를 처리할 수 있으며 HL7 메시지 생성, 검증, 송수신 기능을 제공한다. HAPI는 완성도가 높은 공개용 라이브러리이지만, HL7 버전 간의 상하 관계를 배제하고 버전마다 별도의 패키지로 구분하여 버전 호환을 지원해준다. 따라서 라이브러리의 확장성과 재사용성이 없으므로 새로운 버전이 출시될 때마다 해당 버전의 패키지를 처음부터 다시 개발해야 되는 문제점이 있다.

Interfaceware은 사용자가 배우기 쉽고 다루기 편리한 HL7 메시징 툴킷인 Chameleon을 개발하였다. Chameleon

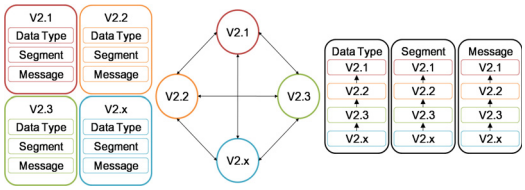
은 HL7 메시지 정의, 생성, 전송 관리, 파싱, 매핑 기능을 제공하며 HL7 메시지뿐만 아니라 XML, X.12, HIPAA, IHE 등의 다양한 메시지 규격을 지원한다. Chameleon은 XML/HL7 변환 모듈을 이용하여 자체적으로 버전 호환 기능을 제공한다. 그러나 이러한 변환 모듈의 문제점은 HL7 버전의 개수만큼 변환 모듈이 필요하며 서로 다른 버전의 HL7 메시지를 처리하기 위해서는 최소 한 번의 변환 과정을 거쳐야한다.

2.3 기존 연구와의 차별성

HL7 V2는 90년대 초반부터 등장하여 현재 다양한 공개용 라이브러리, 툴킷, 인터페이스 엔진이 개발되었지만, 이들 중 버전 호환 및 변환을 지원해주는 제품은 드물다.

일부 제품(HAPI, Chameleon)에서는 버전 호환성에 대한 문제를 인식하여 자체적으로 버전 호환을 지원해주고 있다. Fig. 3(a)은 HAPI를 포함한 일부 제품의 구조로써, HL7 버전의 상하 관계를 배제하고 버전마다 별도의 패키지로 개발하여 버전 호환을 지원해준다. 그러나 이러한 구조의 문제점은 새로운 버전이 출시될 때마다 해당 버전의 모든 HL7 메시지 요소를 처음부터 다시 개발해야 된다. 예를 들어, HL7 V2.7과 버전 호환을 위해 별도의 패키지를 개발해야 되는 경우 87개의 자료형과 175개의 세그먼트, 193개의 메시지 구조, 총 455개의 HL7 메시지 요소 클래스를 처음부터 다시 개발해야 되는 불상사가 발생한다. Fig. 3(b)은 Chameleon을 포함한 일부 제품의 구조로써, 버전 간 변환 모듈을 개발하여 버전 호환을 지원해준다. 그러나 이러한 구조의 문제점은 버전의 개수만큼 변환 모듈이 필요하며 새로운 버전이 출시될 때마다 기존 버전과의 호환을 위해서는 추가로 변환 모듈을 개발해야 된다. 그리고 서로 다른 버전의 HL7 메시지를 처리하기 위해서는 최소 한 번의 변환 과정을 거쳐야 된다. 결과적으로 이러한 구조들은 HL7 V2.x 구조와 버전 호환성에 대한 요구사항을 충족하지 않은 구조라고 할 수 있다.

본 연구는 HL7 V2.x 구조와 버전 호환성에 대한 요구사항을 충족하는 객체지향 프로그래밍의 상속 개념을 HL7 파서의 구조에 적용하여 버전 호환 문제 및 기존 연구의 문제를 해결하였다. 즉, Fig. 3(c)처럼 버전이 아닌 HL7 메시지 요소마다 별도의 패키지로 구분하고 상위 버전의 HL7 메시지 요소가 하위 버전의 HL7 메시지 요소를 상속하도록 HL7 파서의 구조를 설계하고 개발하였다. 따라서 상속 개념의 장점인 재사용성과 확장성으로 인해 새로운 버전이 출시되더라도 개발 시간 및 비용을 절감할 수 있다. 그리고 상속 구조는 버전 호환뿐만 아니라 생성된 객체를 하위 버전으로 업캐스팅하여 재사용할 수 있다는 장점을 가진다.

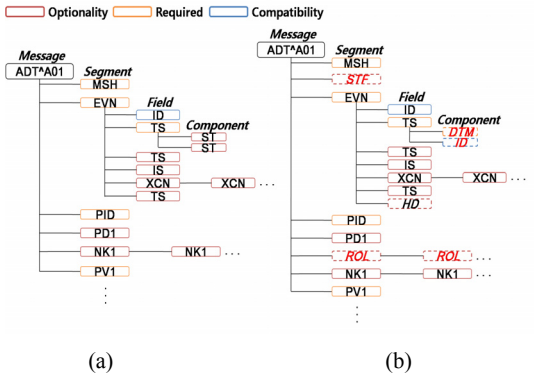


(a) (b) (c)

[Fig. 3] Version compatible Methods
(a) Separated Structure
(b) Conversion Module Structure
(c) Inheritance Structure

2.4 버전 호환성 요구사항

HL7 V2.5에 정의된 버전 호환성의 두 가지 핵심 개념은 다음과 같다. (1) 상위 버전의 메시지를 수신하는 하위 버전의 시스템은 오류 없이 메시지를 수신할 수 있어야 한다. (2) 상위 버전의 시스템은 하위 버전의 메시지를 이해할 수 있어야 한다. 이 개념에서는 다음과 같은 요구사항들을 적용한다. (a) 새로운 메시지, 세그먼트 그룹, 세그먼트, 자료형, 표가 도입될 수 있으며 새로운 세그먼트, 필드, 컴포넌트는 끝 부분에 추가될 수 있다. 그러나 세그먼트는 필요에 따라 메시지의 어느 곳에든지 추가될 수 있다. (b) 새로운 자료형은 이전 자료형과 동일한 구조와 해석을 가지고 있다면 변경될 수 있다. (c) 선택적(Optional) 요소는 조건적(conditional) 또는 필수(required)로 변경되며, (d) 비 반복 요소는 반복으로 변경될 수 있다. 그리고 (e) 필드, 자료형, 컴포넌트의 길이는 길어질 수 있다. 즉, HL7에서 상위 버전의 구조는 하위 버전의 구조를 포함하며, 상위 버전의 메시지를 하위 버전의 메시지로 정보 손실 없이 변환이 가능하다. 예를 들어, Fig. 4는 버전 호환성 요구사항을 충족한 HL7 V2.3, V2.5 ADT^A01 메시지 구조를 나타낸다. MSH, EVN 필수 세그먼트 사이에 STF 선택적 세그먼트가 추가되었고, PD1, NK1 선택적 세그먼트 사이에 반복 가능한 ROL 선택적 세그먼트가 추가되었다. 그리고 EVN 필수 세그먼트의 끝 부분에는 HD 복합 자료형으로 정의된 선택적 필드가 추가되었다.

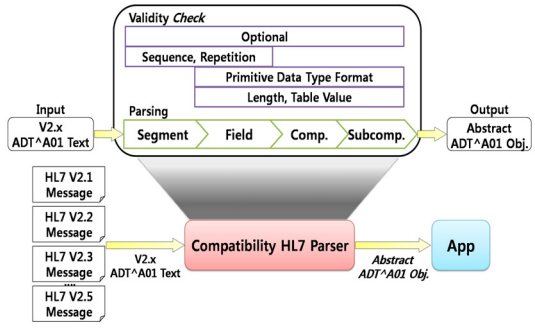


(a) (b)

[Fig. 4] ADT^A01 Message Architecture
(a) HL7 V2.3 ADT^A01
(b) HL7 V2.5 ADT^A01

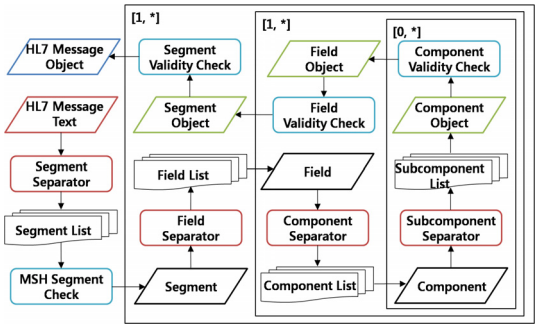
2.5 파서의 설계

Fig. 5과 같이 버전 간 호환 가능한 HL7 파서의 구조를 설계하였다. 즉, 하위 버전의 HL7 메시지 요소의 클래스 객체를 상위 버전의 클래스 객체가 상속하도록 함으로써, 파서를 이용한 의료정보시스템에서는 HL7 메시지의 버전에 상관없이 상위 클래스 객체만을 사용하여 모든 버전의 HL7 메시지를 처리할 수 있도록 설계하였다.



[Fig. 5] Architecture of HL7 Parser

또한, HL7 파서는 입력 받은 메시지를 파싱(parsing)과 동시에 여러 단계의 메시지 유효성 검사(validity check)를 수행한다. 이는 MSH 세그먼트 검사, 세그먼트 순서 및 선택성 검사, 필드의 최대 길이, 반복, 선택성에 대한 검사, 기본 자료형의 형식에 대한 검사, ID 및 IS 기본 자료형인 경우 특정 테이블의 값을 사용하는 지에 대한 검사를 포함한다. Fig. 6는 HL7 V2.x 메시지 파싱 및 유효성 검사 과정을 나타낸다. 파싱은 메시지, 세그먼트, 필드, 컴포넌트, 서브 컴포넌트로 하향식(top-down)으로 수행하고, 유효성 검사는 메시지 구성 요소의 클래스 객체 생성 후 상향식(bottom-up)으로 수행하도록 설계하였다.

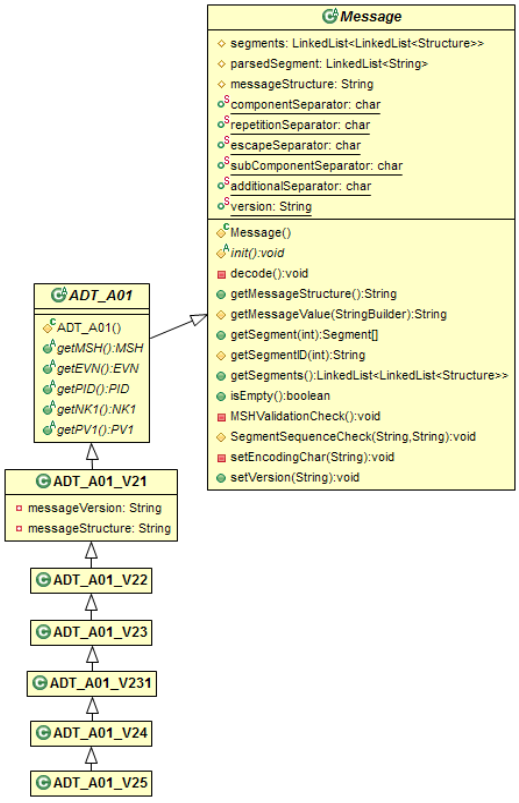


[Fig. 6] Parsing and validity check process of HL7 V2.x

2.6 구성요소 설계

2.6.1 메시지 구조

HL7 파서는 버전에 비 종속적으로 동일한 메소드(method)를 사용하지만, Fig. 7와 같이 버전마다 서로 다른 결과를 반환하기 위해 하위 버전의 ADT_A01 메시지



[Fig. 7] Class diagram of ADT^A01 Message

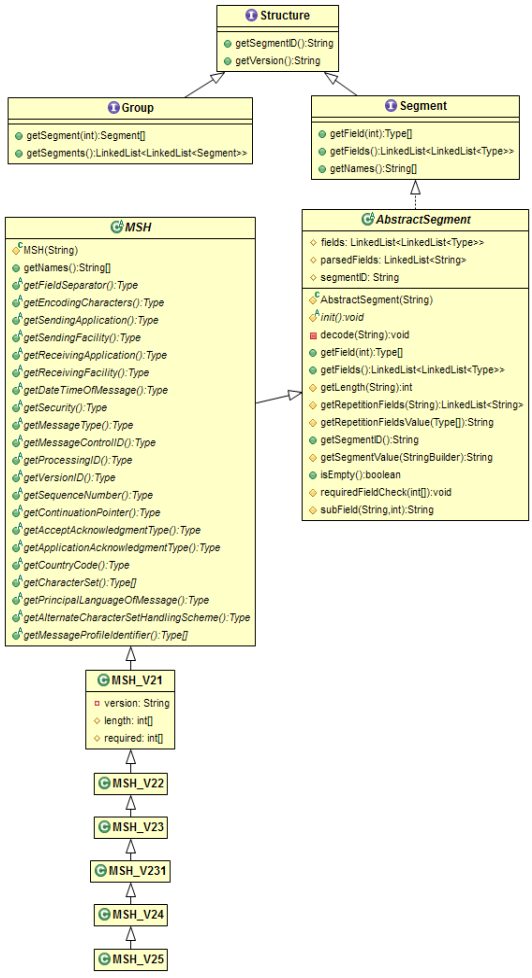
클래스를 상위 버전의 ADT_A01 메시지 클래스가 상속하는 구조로 설계하였다. 예를 들어, 파서가 V2.5 ADT^A01 메시지를 파싱하면 ADT_A01_V25 클래스의 객체가 생성되고, 이를 의료정보시스템에 전송할 때는 업캐스팅(upcasting)하여 ADT_A01 추상 클래스의 객체로 전송한다.

따라서 의료정보시스템은 수신된 메시지의 버전과는 상관없이 추상 클래스의 메소드를 통해 메시지의 데이터를 사용할 수 있다. 이러한 구조의 또 다른 장점은 생성된 객체를 이전 버전으로 업캐스팅하여 재사용할 수 있다는 것이다. 즉, V2.5 클래스의 객체를 이전 버전 클래스로 업캐스팅하면 메소드 내부에서 해당 버전에 맞는 메시지 구조로 자동으로 변환하게 된다. Message 클래스는 모든 메시지 클래스의 최상위 클래스이므로 메시지 파싱, 세그먼트 객체 반환, MSH 세그먼트 유효성 검사, 세그먼트 순서 및 선택성 검사, 버전 설정과 같은 공통적으로 사용되는 메소드를 정의하였다. 특히, 버전 설정 메소드는 HL7 메시지의 모든 요소를 특정 버전으로 자동 변환해주는 기능을 제공한다. ADT_A01 추상 클래스는 ADT_A01 메시지의 모든 세그먼트를 반환하는 추상 메소드를 정의하며, 이 추상 클래스를 상속하는 버전 클래스에서는 자신에게 필요한 메소드만 재정의(overriding)하여 사용한다.

2.6.2 세그먼트 구조

세그먼트 또한 버전에 비 종속적으로 동일한 메소드를 호출해야 하므로, 하위 버전의 세그먼트 클래스를 상위 버전의 세그먼트 클래스가 상속하는 구조로 설계하였다. 그리고 메시지는 세그먼트뿐만 아니라 여러 세그먼트들의 그룹으로써 선택적 또는 반복적으로 구성되기 때문에, 이를 효과적으로 관리하기 위해 Structure 인터페이스를 Segment, Group 인터페이스가 상속하는 구조로 설계하였다. 따라서 Message 클래스에서는 Structure형 연결 리스트(LinkedList)를 정의함으로써, 세그먼트와 세그먼트 그룹과 상관없이 이들을 관리할 수 있다.

Fig. 8은 MSH 세그먼트의 클래스 다이어그램을 나타낸다. AbstractSegment 클래스에서는 모든 세그먼트에서 공통적으로 사용되는 필드 파싱 및 필드 최대 길이, 반복, 선택성에 대한 유효성 검사 메소드를 정의하였다. 즉, 특정 필드가 표준에서 정의한 최대 길이를 벗어나는지, 표준에서 정의한 최대 반복 횟수를 초과하는지, 세그먼트에서 반드시 있어야 하는 필드가 없는지를 확인함으로써 필드의 유효성을 보장한다. MSH 추상 클래스는 MSH 세그먼트의 모든 필드 이름과 테이블 번호를 저장하는 변수와 모든 필드를 반환하는 추상 메소드를 정의한다. 이 추상 클래스를 상속하는 버전 클래스에서는 자신에게 필요한 메소드만 재정의하여 사용한다.

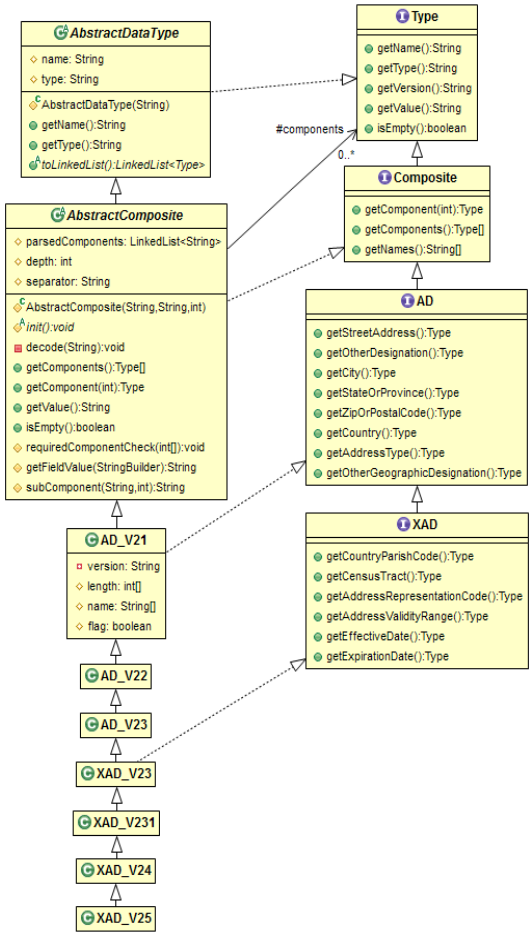


[Fig. 8] Class diagram of ADT^A01 segment

2.6.3 자료형 구조

본 HL7 파서에서는 버전 별 자료형 변환 유형을 기반으로 자료형 클래스를 설계하였다. 자료형 변환 유형은 크게 3 가지 유형, (1) 확장 구조, (2) 유사 구조, (3) 차이 구조로 구성된다. 확장 구조는 새로운 자료형이 이전 자료형과 동일한 구조를 가졌기 때문에, 이전 버전으로 변환할 시 기존 데이터의 손실이 발생하지 않는다. 대표적

인 확장 구조는 AD 복합 자료형에서 XAD 복합 자료형이 있다. 이에 비해, 유사 구조는 자료형 간에 일부만 동일한 구조를 가졌기 때문에, 이전 버전으로 변환할 시 기존 데이터의 손실이 발생할 수도 있다. Fig. 9은 AD 복합 자료형에서 XAD 복합 자료형으로 확장하는 구조를 클래스 다이어그램으로 나타냈다. AD 복합 자료형은 V2.3부터 XAD 복합 자료형으로 확장되었기 때문에, 각 자료형마다 한 개의 상위 클래스가 더 필요하다. 그러나 Java에서는 문법상 다중 상속을 허용하지 않으므로 인터페이스를 사용하였다. 따라서 AD_V21 클래스는 AbstractComposite 추상 클래스와 AD 인터페이스를 각각 상속 및 구현하고, XAD_V23 클래스는 AD_V23 클래스와 XAD 인터페이스를 각각 상속 및 구현하며 XAD 인터페이스가 AD 인터페이스를 상속하여 확장 구조를 설계하였다.



[Fig. 9] Class diagram of AD and XAD data type

3. 연구 결과

파서는 HL7 V2.5 기반으로 V2.4, V2.3.1, V2.3, V2.2, V2.1의 ADT^A01 메시지를 버전에 상관없이 처리할 수 있고, 이전 버전의 메시지로 자동 변환도 할 수 있다. 구현 언어로는 Java jdk 1.7.0을 사용하였다. HL7 V2.x 테이블의 값을 참조하기 위해 Microsoft Office Access 2007을 이용하였다.

3.1 버전 간 자료형 변환

파서를 이용하는 사용자는 버전에 따라 변환되는 자료형의 유형에 대해 알 필요가 없으며, 편의성을 위해 메소드 내부에서 변환 과정을 처리하였다. Fig. 10은 MSH_V23 클래스에서 메시지 버전을 반환하는 메소드의 소스 코드를 나타낸다. 이 메소드는 Message 추상 클래스의 version 클래스 변수와 자신의 version 변수를 비교하여 각 버전에 맞는 자료형이 반환되도록 구현하였다. 버전이 일치하지 않을 경우, super 키워드를 통해 상위 클래스의 메소드를 호출하고, 일치하면 현재 버전에 맞는 자료형으로 변환하여 반환한다.

```
public Type getVersionID() throws Exception {
    if (!Message.version.equals(this.version)) {
        return super.getVersionID();
    }
    if (getField(11) != null && getField(11)[0].getType()
!= "ID") {
        VID vid = (VID) getField(11)[0];
        return              new              ID(name[11],
subField(vid.getComponent(0).toString(), length[11]),
        tableNums[11], version);
    } else {
        return (getField(11) != null) ? getField(11)[0] :
null;
    }
}
```

[Fig. 10] Code of getVersionID method

버전이 일치하더라도 버전 변환의 유무에 따라 두 가지의 경우를 추가로 처리해야한다. 버전 변환을 하는 경우에는 해당 버전에 맞는 자료형 변환 과정을 거치고, 버전 변환을 하지 않는 경우, 파싱된 데이터를 사용하기 위해 메소드를 호출할 때에는 별도의 자료형 변환 과정을 거치지 않는다. 예를 들어, MSH 세그먼트의 VersionID

필드는 V2.2, V2.3에서는 ID 기본 자료형으로 정의되었지만, V2.3.1부터는 VID 복합 자료형으로 정의되었다. 따라서 Fig. 10와 같이 현재 저장된 필드의 자료형이 ID가 아닌지를 확인하는 두 번째 조건문을 추가하였다. 필드의 자료형이 ID가 아니라면 VID 복합 자료형의 첫 번째 컴포넌트를 이용하여 ID 기본 자료형 클래스의 객체를 생성한 후 이를 반환하고, 일치하면 현재 저장된 필드를 반환한다.

3.2 메시지 파싱 및 변환

개발한 파서는 HL7 V2 표준에 대한 지식이 없는 개발자라도 간단한 객체 생성 및 메소드 호출만으로 메시지를 파싱하고, 파싱한 데이터를 사용할 수 있다. 개발한 파서는 ADT^A01 메시지를 파싱하고, 파싱된 데이터 일부를 출력하는 소스 코드를 나타냈다. 소스 코드 실행 결과, MSH 세그먼트와 인코딩 문자, 전송 응용 프로그램, 메시지 구조 및 버전이 출력되었다. 또한 개발한 파서는 HL7 메시지의 모든 요소를 특정 버전으로 자동 변환해주는 메소드를 제공한다. Fig. 11은 Message 추상 클래스의 setVersion 메소드를 사용하여 V2.5 ADT^A01 메시지를 V2.1 ADT^A01 메시지로 변환하는 소스 코드를 나타낸다.

```
public static void main(String[] args) throws Exception {
    ADT_A01 adt = new ADT_A01_V25();
    System.out.println(adt);
    adt.setVersion("2.1");
    System.out.println(adt);
}
```

[Fig. 11] Transformation from V2.5 ADT^01 message to V2.1 ADT^01 message

3.3 호환성 실험 및 유효성 검사

개발한 파서는 버전 간 호환성 실험과 메시지 유효성 검사를 수행하였다. 테스트 환경은 HL7 메시지를 전송하는 클라이언트와 전송된 메시지를 파싱하는 서버로 구성되며, 이들은 소켓 통신으로 연결하였다.

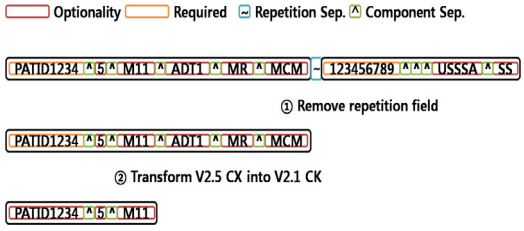
3.3.1 호환성 실험

개발한 파서의 호환성 기능을 평가하기 위해 다양한 버전(V2.5, V2.4, V2.3.1, V2.3, V2.2, V2.1)별 ADT^A01 메시지를 이용하여 호환성 실험을 수행하였다. 실험 방법은 (1) 2006-2009년 K대학교병원 류마티스 질환으로 입원한 환자 데이터 700건을 이용하여 V2.5 ADT^A01 메시지를 생성한다. (2) 개발한 파서를 이용하여 생성한 메시지를 파싱한 후 이전 버전으로 변환하고 변환된 메시지를 저장한다. (3) 변환된 메시지를 파싱하고 이전 버전으로 변환한다. (4) 실험 (2), (3)을 반복 수행한다. (5) 각 단계에서 메시지 파싱 및 이전 버전으로 변환 시 오류가 발생하는지 확인한다. 테스트 결과, 개발한 파서가 예외 없이 메시지가 파싱 및 변환하였음을 확인하였다 [Table 1]. 파싱된 메시지 객체를 추상 클래스로 업캐스팅한 후 메소드를 호출하여 버전에 상관없이 파싱된 메시지의 데이터를 사용할 수 있었다. 그리고 이전 버전으로 메시지를 변환할 시, 각 표준에서 명시한 필드의 개수, 최대 길이, 자료형으로 자동 변환되었다.

[Table 1] Results of compatible test

Version	2.1	2.2	2.3	2.3.1	2.4
2.5	700/700	700/700	700/700	700/700	700/700
2.4	700/700	700/700	700/700	700/700	
2.3.1	700/700	700/700	700/700		
2.3	700/700	700/700			
2.2	700/700				

그러나 이전 버전으로 메시지를 변환하는 경우, 일부 필드의 최대 길이나 반복 횟수가 줄어들기 때문에 특정 필드에서는 데이터 손실이 발생하였다. 왜냐하면 일반적으로 데이터 표현 범위가 넓은 자료형에서 좁은 자료형으로, 즉, 상위 버전에서 하위 버전으로 유형을 변환하는 경우에는 부득이하게도 추가된 데이터는 손실되기 때문이다. Fig. 12는 이전 버전으로 변환 시, 필드에서 발생하는 데이터 손실 과정을 나타낸다. V2.5에서는 PID 세그먼트의 3번째 필드인 환자 식별자 목록(Patient Identifier List)은 CX 복합 자료형으로 무한 반복이 가능하지만, V2.1에서는 CK 복합 자료형으로 단일 반복만 가능하다. 따라서 이전 버전으로 변환할 시, 추가된 환자 식별자 목록과 기존 환자 식별자 목록의 일부가 손실된다.



[Fig. 12] Process of Data loss

3.3.2 유효성 검사

개발한 파서의 유효성 검사 기능을 평가하기 위해 HL7 V2 표준 예제 메시지에 의도적으로 오류 내용을 포함시켜 오류 메시지를 생성하고, 이를 이용하여 유효성 검사 실험을 수행하였다. 오류 유형은 세그먼트 순서 및 선택성, 기본 자료형 형식, 테이블 참조 값, 필수 필드 오류로 구성하였다. 실험 방법은 (1) 버전 별로 ADT^A01 오류 메시지를 생성한다. (2) 개발한 파서를 이용하여 오류 메시지를 파싱한다. (3) 파싱 시 발생하는 예외와 오류 내용이 일치하는지 확인한다.

```
MSH|^~\&|ADT1|MCM|LABADT|MCM|198808181126|SECURITY|
ADT^A01|MSG00001|P|2.3|||||||<cr>
EVN|A01|198808181123||||<cr>
PID|1||PATID1234^5^M11^ADT1^123456789^^^USSSA^SS||JONES^
WILLIAM^A^III||19610615|X||C|1200 N ELM
STREET^^GREENSBORO^NC^27401-1020|GL|(919)379-1212|(919)2
71-3434||S||PATID12345001^2^M10|123456789|987654^NC|||||||||<cr>
NK1|1|JONES^BARBARA^K|WI^WIFE||||NK^NEXT OF
KIN|||||||||||||||||||||||||||<cr>
PV1||O|||||0148^ADDISON,JAMES|0148^ADDISON,JAMES|0148^AD
DISON,JAMES|AMB|||||||0148^ADDISON,JAMES|S|1400|A|||||||||||||
||||GE||||||||||||<cr>
```

[Fig. 13] HL7 V2.3 ADT^01 message Including intentional error

[Table 2] Detection results of error message

Validation check list	Error message version					
	2.1	2.2	2.3	2.3.1	2.4	2.5
Segment seq., Opt.	8/8	8/8	8/8	8/8	8/8	8/8
Data type Format	3/3	3/3	3/3	3/3	3/3	3/3
Reference Table value	5/5	6/6	6/6	7/7	9/9	9/9
Required Field	11/11	11/11	11/11	11/11	12/12	12/12
Total	27/27	28/28	28/28	29/29	32/32	32/32

실험 결과, 개발한 파서가 오류 메시지에 포함된 유효성 검사 항목의 오류를 모두 검출하였음을 확인하였다 [Table 2]. 오류 메시지를 파싱할 시, 오류의 종류와 발생 위치가 예외 메시지로 출력되어 오류 정보를 확인할 수 있었다.

본 실험에서는 필드의 최대 길이를 유효성 검사 항목에 포함하지 않았다. 왜냐하면 HL7 V2 표준에서는 메시지를 교환하는 시스템 간의 동의에 따라 필드의 최대 길이가 조정될 수 있기 때문에 개념적으로 중요하지 않다. 그래서 표준에서는 수신자는 최대 필드 길이까지 받을 수 있어야 하며, 송신자는 최대 필드 길이까지 전송할 수 있어야 된다고 정의되어있다. 따라서 본 연구에서는 필드의 데이터가 최대 길이를 넘길 시, 예외를 발생시키지 않고 최대 길이까지 데이터를 잘라내도록 파서를 개발하였다.

4. 향후 응용

의료 분야의 정보화가 활성화됨에 따라 서로 다른 병원정보시스템 간에 의료정보를 교환해야 할 필요성이 증대하고 있다. 이러한 이유로 인해 HL7 표준이 등장하였으며, 현재 선진국에서는 이 기종 의료정보시스템 통합을 위한 중요한 수단으로 제공되고 있다. 그러나 HL7 표준은 다양한 버전이 존재하며 계속해서 새로운 버전을 개발하는 중이기 때문에 이전 버전으로 개발된 의료정보시스템과의 버전 호환 문제는 지속적으로 발생할 것이다. 그럼에도 불구하고 현재 개발된 HL7 인터페이스는 이러한 문제를 방치하거나 버전마다 별도의 패키지 또는 변환 모듈을 개발하는 것과 같은 일시적인 해결책을 제시하기 때문에 새로운 버전이 발표될 때마다 추가적인 비용과 개발 시간 소요는 불가피하다. 따라서 HL7 V2 표준에 정의된 버전 호환성 요구사항을 충족하는 HL7 파서의 개발과 같은 근본적인 해결책이 절실히 필요한 시점이다.

본 논문은 HL7 V2.5 표준에 정의된 버전 호환성 요구사항에 기반으로 버전 간 호환 가능한 HL7 파서의 구조를 설계하였다. 설계한 구조를 기반으로 ADT^A01 메시지 처리를 위한 파서를 Java로 개발하였다. 설계한 구조는 객체지향 프로그래밍에서의 상속 개념을 이용하여 하위 버전의 HL7 메시지 요소의 클래스 객체를 상위 버전의 클래스 객체가 상속하도록 함으로써, 파서를 이용한 응용 프로그램에서는 HL7 메시지의 버전에 상관없이 상위 클래스 객체만을 사용하여 모든 버전의 HL7 메시지를 처리할 수 있도록 설계하였다. 이러한 구조의 또 다른 장점은 생성된 객체를 이전 버전으로 업캐스팅하여 재사용할 수 있다는 것이다. 그리고 개발한 파서의 효용성을 평가하기 위해 HL7 메시지를 전송하는 클라이언트와 전송된 메시지를 파싱하는 서버로 구성된 실험 환경에서 두 가지 실험을 수행하였다. 첫 번째 실험은 이전 버전과의 호환성 기능을 평가하기 위해 류마티스 입원 환자 데

이터 기반의 ADT^A01 메시지를 이용하여 메시지 분석 및 변환을 수행하였고, 두 번째 실험은 메시지 유효성 검사 기능을 평가하기 위해 오류 내용이 포함된 HL7 V2 표준 ADT^A01 예제 메시지를 이용하여 메시지 분석을 수행하였다. 실험 결과, 개발한 파서는 버전에 상관없이 파싱된 메시지의 데이터를 사용할 수 있었으며 이전 버전으로 메시지가 자동 변환하였다. 그리고 메시지 파싱 시 유효성 검사가 정확하게 수행하여 오류를 정확하게 검출하였다.

본 논문에서 설계한 파서의 구조는 하위 버전에서 상위 버전으로, 즉, 이후 버전과의 호환(Forward Compatibility)은 불가능하다. 왜냐하면 Java의 문법 상 다운캐스팅(downcasting)의 목적은 상속 관계인 두 클래스에서 업캐스팅한 클래스를 원래 형태로 돌려놓기 위한 것이기 때문이다. 따라서 하위 버전에서 상위 버전으로 변환하기 위해서는 먼저 상위 버전에서 하위 버전으로 변환해야 된다는 것이다. 또한 개발한 파서는 수많은 HL7 메시지 중에 가장 많이 활용되는 입원/퇴원/전원 중에서 ADT^A01 메시지를 구현하였기 때문에 실무 전역에서 사용하기에는 아직 이르다. 그러나 본 연구는 이전 버전과의 호환성 문제에 대한 근본적인 해결책을 제시하는 데 그 의의가 있다. 따라서 개발한 파서를 지속적으로 수정 보완하여 향후에는 버전 간 상호 호환 가능한 (Inter-compatibility) HL7 파서의 구조에 대한 연구를 지속적으로 할 계획이다.

References

[1] P. H. Cheng, C. H. Yang, H. S. Chen, S. J. Chen, J. S. Lai, "Application of HL7 in a Collaborative Healthcare Information System", *Proceedings of 26th EMBC Conference*, pp. 3354-3357, 2004.

[2] Y. Alsafadi, O. R. L. Sheng, R. Nartinez, "Comparison of Communication Protocols in Medical Information Exchange Standards", *CBMS 1994, Proceedings of the 7th IEEE Symposium*, pp. 258-263, 1994.
DOI: http://dx.doi.org/10.1109/CBMS.1994.316022

[3] J. S. Kim, "Development of Composer-Based HL7 V2.4 Messaging Toolkit", *The Graduate School of Kyunpook National University Master's Thesis*, 2005.

[4] S. M. Lee, J. T. Song, I. K. Kim, H. Cho, Y. S. Kwak, "The Design of the HL7 V2.4 Message Interface Engine", *Proceedings KISC 2003 Autumn Annual Conference*, Vol. 30, No. 2, pp. 781-783, 2003.

[5] Health Level 7, Health Level Seven International [Internet]. Available From: http://www.hl7.org. (accessed Dec., 20, 2013)

[6] Orion Health, Orion System Symponia [Internet]. Orion Health group of companies, Available From: http://www.orionhealth.com. (accessed Dec., 20, 2013)

[7] Corepoint Health, NeoBrowse TCP for HL7 [Internet]. Corepoint Health Inc., Available From: http://neotool.com. (accessed Dec., 20, 2013)

[8] iNTERFACEWARE, Chameleon HL7 Messaging Toolkit [Internet]. Interfaceware, Available From: http://www.interfaceware.com/chameleon.html. (accessed Dec., 20, 2013)

[9] Eversolve, Medi7 [Internet]. Eversolve, Available From: http://eversolve.com/products. (accessed Dec., 20, 2013)

[10] HAPI, HL7 Application Programming Interface [Internet]. University Health Network, Available From: http://hl7api.sourceforge.net/. (accessed Dec., 20, 2013)

[11] Y. S. Kwak. *Development of prototype for transferring discharge summary information between hospitals using HL7*. p.3-68, Ministry of Health and Welfare, 2003.

[12] K. S. Um, Y. S. Kwak, H. Cho, I. K. Kim, "Development of an HL7 Interface Engine, Based on Tree Structure and Streaming Algorithm, for Large-size Messages which Include Image Data", *Computer Methods and Programs in Biomedicine*, Vol. 80, No. 2, pp. 126-140, 2005.
DOI: http://dx.doi.org/10.1016/j.cmpb.2005.07.004

[13] Y. S. Go, "Unified Message Development Framework for HL7", *The Graduate School of Korea University Master's Thesis*, 2005.

[14] J. W. Choi, D. H. Lee, *Introduction of Seoul National University Hospital PACS-HIS Interlokcing Technology*. p.19-54, Korea Health Industry Development Institute, 2004.

[15] J. W. Choi, H. W. Kim, H. I. Cho, "Implementation of HL7 Interface Engine for Unifying Medical Records", *Healthcare Informatics Research*, Vol. 4, No. 1, pp. 9-14, 1998.

[16] Health Level 7., *HL7 Version 2.5 Standard Chapter 2 Control*. p.21-26, HL7, 2003.

[17] Walid Al-Ahmad, "A Framework for Conceptual Modeling in OOP", *Journal of the Franklin Institute*, Vol. 343, No. 4/5, pp. 532-544, 2006.
DOI: http://dx.doi.org/10.1016/j.jfranklin.2006.02.035

[18] J. G. Lee, C. H. Cho, D. G. Lee, W. Choi, Y. G. Song, Y. S. Kim, "Object-Oriented Programming Concept and

Paradigm", *Electronics and telecommunications trends*, Vol. 8, No. 1, pp. 45-68, 1993.
[19] Health Level 7, HL7 Version 2 Product Suite [Internet]. Health Level Seven, Available From: http://www.hl7.org. (accessed Dec., 20, 2013)
[20] Corepoint Health, The HL7 Evolution [Internet]. Corepoint Health Inc., Available From: http://www.corepointhealth.com/sites/default/files/whitepapers/hl7-v2-v3-evolution.pdf. (accessed Dec., 20, 2013)
[21] H. S. Kim, H. Cho, I. K. Lee, "The Development of a Graphical User Interface Engine for the Convenient Use of the HL7 Version 2.x Interface Engine", *Healthcare Informatics Research*, Vol. 17, No. 4, pp. 214-223, 2011. DOI: http://dx.doi.org/10.4258/hir.2011.17.4.214
[22] UCARESOFT, HL7 Message Toolkit Product [Internet]. UCARESOFT Inc., Available From: http://hl7korea.knu.ac.kr/ucaresoft/product/HlToolkit.htm.(accessed Dec., 20, 2013)
[23] Merge Healthcare, Merge HL7 Toolkit [Internet]. Merge Healthcare, Available From: http://www.merge.com/Solutions/Toolkits/Merge-HL7-Toolkit.aspx. (accessed Dec., 20, 2013)
[24] SoftTeam, HL7 Tool kit [Internet]. SoftTeam Solutions Private Limited, Available From: http://www.softteam.com/hit.html. (accessed Dec., 20, 2013)

박 현 상(Hyun Sang Park) **[정회원]**

- 2013년 8월 : 대구한의대학교 IT의료산업학과 (이학사)
- 2013년 8월 ~ 현재 : 경북대학교 의료정보학과 (석사과정)

<관심분야>
의료정보표준, 모바일 헬스케어, HL7

김 화 선(Hwa Sun Kim) **[정회원]**

- 2003년 2월 : 인제대학교 컴퓨터공학과 (공학석사)
- 2007년 2월 : 경북대학교 의료정보학과 (의료정보학박사)
- 2009년 6월 ~ 2011년 2월 : 경북대학교 의과대학 연구교수
- 2011년 3월 ~ 현재 : 대구한의대학교 IT의료산업학과 교수

<관심분야>
모바일 헬스케어, 병원정보시스템, 표준용어체계

조 훈(Hune Cho) **[정회원]**

- 1986년 2월 : 남캐롤라이나 주립대학 전산학과 (전산학석사)
- 1992년 2월 : 유타주립대학 의료정보학과 (의료정보학박사)
- 1994년 8월 ~ 1999년 2월 : 아주대학교 의과대학 조교수
- 1999년 3월 ~ 현재 : 경북대학교 의료정보학과 교수

<관심분야>
유비쿼터스 헬스케어, 모바일 헬스케어, HL7