HL7 V2.4 메시지 인터페이스 엔진 설계

이상민[O*] 송진태* 김일곤* 조훈** 곽연식**
경북대학교 컴퓨터 과학과*
경북대학교 의과대학 의료정보학 교실**
lsrn9525@cs.knu.ac.kr[O*], redsusia@dreamwiz.com*, ikkim@knu.ac.kr*, hunecho@knu.ac.kr**, yskwak@knu.ac.kr**

The Design of the HL7 V2.4 Message Interface Engine

Sangmin Lee[O*] Jintae Song* Ilkon Kim* Hune Cho** Yunsik Kwak**
Dept. of Computer Science, Kyungpook National University, Korea*
Department of Medical Informatics, Kyungpook National University School of Medicine, Korea**

요 약

의료 정보의 표준 데이터 교환 환경을 구축하기 위해서 의료정보분야의 전송표준인 HL(Health Level)7[1]을 이용하는 방법이 있다. 방대한 HL7 V2.4 메시지를 의료 관련 어플리케이션이나 시스템에서 사용하고자 한다면, 메시지 처리만 전담하는 인터페이스 엔진이 필요하다. 이 논문에서는 인터페이스 엔진을 만들기 위해, HL7에서 정의한 메시지 구조를 바탕으로 HL7 V2.4 메시지를 모델링 하였고, 인터페이스 엔진의 세 가지 기능(Message Validation, Message Building, Message Transmission)을 제안한다.

1. 서 론

컴퓨터 및 정보통신 기술의 급속한 발전으로 인해 의료계에서도 병원전산화 작업이 활성화되고 있으며, 개별 의료기관의 정보 디지털화 작업도 진척되고 있다.[2] 그러나 이러한 정보시스템들이 각 의료기관마다 개별적으로 개발되어 있어 계열 병원간 정보교환 또는 시스템 통합 등이 쉽게 이루어지지 않는 환경으로 구성되어 있다. 환자 진료 수준의 질을 높이고, 의료기관의 업무의 효율화를 위해서는 각 의료기관간의 환자 정보를 공유할 수 있어야 한다.
의료 정보의 표준 데이터 교환 환경을 구축하기 위해서 의료정보분야의 전송표준인 HL7을 이용하는 방법이 있다. HL7은 의료 업무에서 발생하는 대부분의 내용을 표준화 하였기 때문에 의료 정보 공유를 위한 손쉬운 방법이지만, 그 방대한 내용과 다양한 선택 사항이 있기 때문에 HL7 메시지를 처리하는 것이 어려운 일이며 오류가 발생할 가능성이 높다. 여러 시스템과 어플리케이션 마다 이 작업을 반복하여 수행한다면 시간과 자원의 낭비이다. 그래서, HL7 메시지 처리에 관한 기본적이고, 핵심적인 기능만을 모아놓은 인터페이스 엔진을 제안한다. 인터페이스 엔진의 주요 기능에는 메시지 검증, 메시지 생성, 메시지 전송 이 있다.
본 논문에서는 HL7 메시지 구조를 살펴보고, 인터페이스 엔진의 틀인 메시지 모델링을 수행하며, 그 세부적인 기능으로 위의 주요 세 가지 기능을 하는 인터페이스 엔진을 제안한다. 현재 가장 최신의 표준인 HL7 버전 2.4 표준을 이용한다.

2. HL7 V2.4 메시지 구조

HL7 V2.4 메시지는 메시지, 세그먼트, 필드, 데이터타입으로 이루어진다. 하나의 메시지는 여러 개의 세그먼트로 이루어 졌고, 세그먼트는 다시 여러 개의 필드들로 만들어졌다. 필드와 데이터타입과는 일대일의 관계이며, 데이터타입은 자기 자신을 포함하는 구조이다. 아래의 내용은 HL7 V2.4 표준에서 메시지 모델링에 필요한 요소에 대한 설명을 요약한 것이다.

① 메시지

메시지는 데이터 전송의 가장 작은 단위이며, 세그먼트들의 모임으로 구성된다. 각 메시지들은 메시지의 목적에 맞는 메시지 구조를 가지고 있다. MSH(Message Header) 세그먼트를 시작으로 여러 세그먼트들이 모여서 메시지가 구성된다.

② 세그먼트

세그먼트는 필드들의 논리적 그룹이다. 메시지의 세그먼트들은 필수 세그먼트이거나 선택 세그먼트이다. 필수세그먼트는 메시지 내에서 반드시 필요한 세그먼트를 의미한다. 선택 세그먼트는 메시지 내에서 한번 만 사용되거나 반복을 허용할 수 있다. 각 세그먼트는 그 쓰임에 따라 3문자로 고유한 이름을 가지고 있다. Z 로 시작하는 세그먼트는 지역적으로 정의된 메시지에 사용된다.

세그먼트를 문자열로 표현할 때는 첫번째 필드에 세그먼트 ID를 붙이고, 각 필드의 값이 연속해서 정의되고, 마지막에 세그먼트 구분자 (segment seperators : <CR>)를 붙인다.

③ 필드

필드는 콤포넌트들의 집합이며, 콤포넌트는 다시 서브콤포넌트로 구성된다. 메시지의 실제 내용은 세그먼트의 필드에서 표현 된다 . 필드는 널 값을 허용한다. 필드는 세그먼트 내부의 순서 번호를 가지고 있다. 필드의 최대 길이 속성은 한 필드에 표현할 수 있는 최대 문자 개수를 정의한다. 개념적으로 최대 길이는 중요하지 않고, 각 도메인에 따라 조절할 수 있다. 최대길이는 콤포넌트와 서브콤포넌트를 포함한 길이이고, 최대 길이는 한번 표현될 때의 길이이으로 반복 구분자에 의한 반복 필드는 다르게 계산해야 한다. 콤포지트는 자기 내부의 가장 긴 콤포넌트의 최대 길이보다 짧은 값을 가질 수 없다. 필드는 세그먼트와 같이 반복이 가능하다. 하지만, 조건에 따라 반복 불가, 무한 반복, 유한 반복으로 구분된다.

④ 데이터타입

데이터타입은 데이터 필드의 내용을 제한한다. 데이터 타입은 다중 콤포넌트 혹은 서브콤포넌트를 보유하고 있다. 일반적으로 HL7에서는 데이터타입에 길이를 제한하지 않고, 세그먼트, 필드에서 그 길이를 제한하는 속성을 가진다. 데이터타입은 두 가지 형태로 구분된다. 데이터타입은 여러 개의 데이터타입으로 이루어진 콤포지트와 그렇지 않은 프리미티브로 구분된다.

3. 메시지 모델링

HL7 V2.4 메시지를 컴퓨터에서 표현하려면 먼저 모델링이 필요하다. 모델링은 만드는 사람마다 다르게 작성될 수 있으며, 다양한 방법이 존재한다. 본 논문에서는 두 가지 방법으로 진행되었다. 첫 번째는 HL7에서 제공하는 데이터베이스를 이용하는 방법이고, 두 번째는 세부적인 클래스와 부분적으로 데이터베이스를 이용하는 방법이다.
HL7 에서는 메시지 구조를 정의한 데이터베이스를 제공하고 있다. 여기에는 메시지, 세그먼트, 필드, 콤포넌트, 테이블 등 메시지 모델링에 관련한 모든 내용을 제공하고 있다. 이 데이터베이스에 질의를 통해 동적으로 메시지를 모델링이 가능하다. 그래서, 첫 번째 방법은 프로그래밍 하기 편리한 점이 많다. 몇 개의 클래스만으로도 전체 메시지에 대한 모델링 방법을 제시할 수 있고, 코딩에 관련한 시간을 줄일 수 있다. 하지만 이 방법은 다양한 예외성이 존재하는 HL7 표준을 만족하지 못한다. 예를 들어, OBX 세그먼트의 5번째 필드는 'varies' 이다. 이 형식은 OBX 세그먼트의 2번째 필드의 값에 따라 데이터 타입을 결정하는 필드이다. 즉, 이 형식은 메시지를 형태를 만들 때 결정 되는 것이 아니고, OBX 세그먼트 2번째 필드의 값이 결정될 때 결정 되어 진다. 그리고, 메시지 형이 결정 될 때, 기본적으로 입력되어야 하는 필수 필드들이 있다. MSH 세그먼트의 필드 Separator, Encoding Characters, Message Type 필드들은 메시지의 형태를 결정하는 가장 중요한 내용이다. 이 내용이 변화하면 메시지 전체의 구조가 변화하게 되므로, 메시지가 처음 생성될 때 결정되어야 한다. 그리고, 메시지 처리 속도가 현저히 떨어진다.

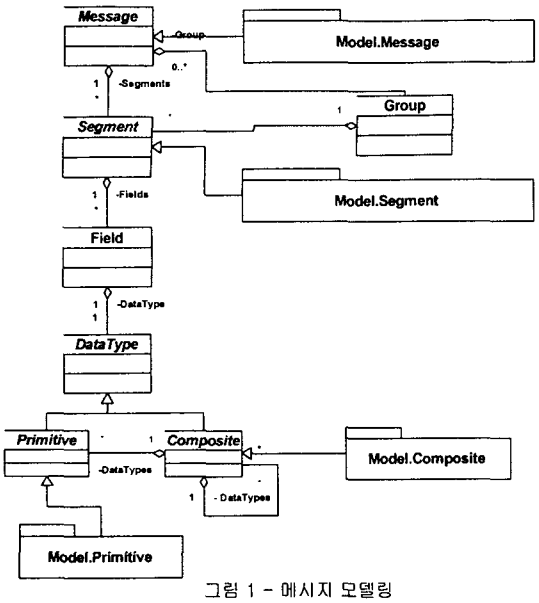

그림 1 - 메시지 모델링

이러한 점을 보완하고자 진행 된 것이 두 번째 방법으로, 그림 1에서 보듯이 메시지, 세그먼트, 필드, 데이터타입, 콤포지트, 프리미티브 기본 클래스에서는 각 역할에 맞는 공통적이고, 기본적인 기능을 구현하고, 메시지, 세그먼트, 콤포지트, 프리미티브에서 상속 받은 클래스에서는 각 HL7 표준에서 정의된 기본 속성을 가지면서, 다양한 요구사항을 각 역할에 맡게 개별적으로 구현 하는 방법이다. 이 방법은 속도를 크게 향상 시켰고, HL7의 다양한 예외성을 충족시킬 수 있다. 하지만, 방대한 량의 정보를 표현하려면 많은 코딩 시간이 필요하고, 조그만 변화를 수용하는데도 시간이 많이 걸리는 단점이 있다. 본 논문에서는 코딩 시간이 많이 걸리더라도, 속도와 다양한 요구사항을 충족시켜줄 수 있는 두 번째 방법으로 모델링 되었다.
메시지내의 세그먼트를 처리할 때는 필수 옵션과 반복 옵션을 처리해야 하며, 다른 예외적 것으로 여러 세그먼트가 하나의 단위로 묶어서, 세그먼트 역할을 하는 경우가 있다. 이 경우를 그룹이라 명하겠다. 그룹은 메시지 내에서는 세그먼트 역할과 동일하며, 그룹 내부에서는 메시지와 같이 여러 세그먼트를 관리한다. 그룹도 필수 옵션과 반복 옵션에 영향을 받고, 내부의 세그먼트들도 필수 옵션과 반복 옵션에 의해 동작한다.
세그먼트는 한 개 이상의 필드들로 이루어졌다. 세그먼트에서 바로 데이터 타입과 관련한 클래스를 호출하지 않고, 필드를 중간 단계로 이용한다. 이때, 데이터 타입의 이름과 필드에서 정의된 속성들을 정의하게 된다. 필드에서는 데이터 타입의 이름을 이용하여 동적으로 각 데이터 타입과 연결한다. 이 방식을 이용하는 이유는 필드 반복을 지원해야 하며 필드와 데이터 타입과는 속성이 다르기 때문이다. 필드와 데이터타입은 일대일의 관계이다. 하나의 필드에는 반드시 하나의 데이터타입과 연결되어있다. 필드는Data 필드의 다양한 정보를 관리하는 역할을 하고, 데이터타입은 HL7표준에서 정의한 데이터 타입을 통합하는 역할을 한다.
데이터타입에서 콤포지트와 프리미티브가 파생되고, 콤포지트는 내부적으로는 자신을 포함해서 프리미티브를 가지는 데이터타입에 관련한 배열이 존재한다. 콤포지트는 자신을 포함하기 때문에 무한 반복을 할 가능성이 있지만, HL7 표준에 의해 3단계 이상 반복 되지는 않는다. 모든 데이터의 입력, 삭제, 수정으로 발생되는 이벤트의 시작점은 프리미티브이고, 콤포지트를 거치거나 아니면 직접 필드 단계에 까지 이벤트 발생을 알려준다.

4. 인터페이스 엔진

메시지 모델링를 통해, HL7 V2.4 메시지를 객체지향방법으로 접근을 해서, 클래스 형태로 그 틀을 제작하였다. 인터페이스 엔진은 이 틀안에 HL7 메시지 처리를 위한 가장 기본적 기능인 메시지 검증, 메시지 생성, 메시지 전송을 구현 하여야 한다.

① 메시지 검증

HL7 V2.4 메시지를 검증에는 여러 단계가 있다. 이는 MSH의 검증, 세그먼트의 순서와 존재에 대한 검증, 필드의 개수, 길이, 반복, 존재에 대한검증, 데이터 타입에 맞는 값인지에 대한 검증, ID/IS 데이터 타입의 경우에는 테이블 값을 참조 하는지에 대한 검증으로 이루어진다.

- MSH 검증
모든 HL7 메시지의 첫 세그먼트는 MSH이어야 한다. MSH는 HL7 메시지를 파싱하기 위한 델리미터 정보와 메시지 구조에 대한 정보 그리고 전송에 관련된 정보를 가지고 있다. MSH의 검증에는 다음과 같은 단계가 있다.
· 메시지의 첫 세그먼트가 MSH인가?
· 메시지 구조에 대한 정보를 담고 있는가?
· 송수신을 위한 Application 과 Facility 정보를 담고 있는가?
- 세그먼트 검증
MSH의 검증이 완료되면 HL7 메시지의 구조에 대한 정보를 알 수 있다. 이를 이용하여 HL7 데이터베이스에 질의를 하면 메시지를 구성하는 세그먼트들을 얻을 수 있다. 이 세그먼트들은 순서를 가지고 있으며, 옵션들을 가지고 있다. 세그먼트가 가지는 옵션은 2가지인데, 이를 조합하면 4가지의 경우가 생성된다. 이는 다음과 같다.
· Required , Non-Repeat
· Required, Repeat
· Optional , Non-Repeat
· Optional, Repeat
메시지를 구성하고 있는 각 세그먼트는 위와 같은 특성 중의 하나를 가진다. 메시지를 구성하고 있는 세그먼트가 위의 규칙을 따르는지를 검사하기 위해서는 정규 표현식을 이용하는 것이 효과적이다. 정규 표현식은 패턴 매칭을 검사하기 위한 최상의 방법이다.
예를 들어, ADT_A02 메시지에 대해서 살펴 보도록 한다. 다음은 ADT_A02 메시지에 대한 상태 다이어그램이다.

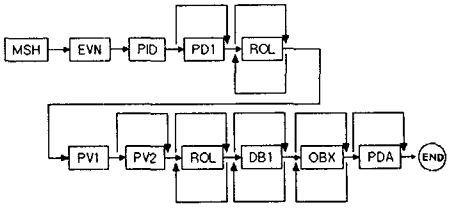

그림 2 - 상태 다이어그램 (ADT_A02)

이를 정규 표현식으로 표현하면 다음과 같다.
(MSH)(EVN)(PID)(PD1)?(ROL)*(PV1)(PV2)?(ROL)*(DB1)*(OBX)*(PDA)?$
위의 정규 표현식과 메시지에 존재하는 모든 세그먼트의 ID를 맺칭하면 메시지의 세그먼트가 규약에 맞게 존재하는지를 쉽게 알 수 있다.
- 필드 검증
각 세그먼트는 필드들의 집합이다. 필드는 하나의 데이터 타입을 가지는데, 필드에서는 이 데이터 타입에 다음의 속성을 부여한다.
· 길이
· Required / Optional
· Repeat / Non-Repeat
동일한 데이터 타입이더라도 필드의 속성에 따라 최대 길이가 다르게 적용된다. MSH에서 얻은 필드 델리미터를 이용하여 세그먼트를 분할하여 얻은 필드의 값이 널 인지를 보고 Required/Optional을 검증하게 된다. 그리고 필드 값에서 반복 델리미터의 존재를 보고 Repeat / Non-Repeat를 검증하게 된다.
- 데이터타입 검증
데이터타입에는 크게 프리미티브 타입과 콤포지트 타입으로 나누어진다. 프리미티브 타입은 가장 세분화된 형태이며 콤포지트 타입은 콤포지트 타입이나 프리미티브 타입을 자식으로 가지는 데이터 타입이다. 그러므로 콤포지트를 검증하기 위해서는 하위에 있는 콤포지트 타입과 프리미티브 타입의 검증이 우선시 되어야 하며, 결국 프리미티브 타입의 검증이 데이터타입 검증의 핵심이 된다.
프리미티브 데이터타입에는 문자열의 특성을 지닌 데이터타입과 숫자로 표시되는 특성을 지닌 데이터타입이 있다. 이 두 가지 특성을 모두 만족시키는 검증을 위해서는 앞서 세그먼트의 검증에서 사용한 바 있는 정규 표현식을 사용하는 것이 효과적이다.
SI 타입은 양의 정수만을 취할 수 있는 데이터 타입이다. 이를 검증하기 위한 정규 표현식은 다음과 같다.
(+)?[1-9]*[0-9]$
이 정규 표현식을 이용하면 간단하게 모든 양의 정수에 대한 검증을 할 수 있다. 그 외의 다른 숫자로 표시되는 데이터 타입도 이와 같은 방법을 이용하여 검증을 한다. 그리고 ST,TX 와 같은 문자형 데이터 타입은 특별하게 검사할 필요가 없다.
- 테이블 Value 검증
프리미티브 데이터 타입 중에서 ID 타입과 IS 타입은 HL7이나 사용자가 이미 정한 값들 중에서 하나의 값을 가지는 데이터 타입이다. 이 값들은 테이블에 저장되어 있다. ID 타입과 IS 타입의 차이점은 ID 타입은 HL7이 정한 값들 중에서 하나의 값을 가져야 하지만, IS는 사용자가 정한 값들 중에 하나를 선택해야 한다는 강제성이 없다. ID 타입과 IS 타입의 검증을 위한 테이블 번호는 필드의 속성에 정의되어 있거나 콤포지트 타입에 속한 ID/IS의 경우에는 콤포지트 타입에서 정의하고 있다. 테이블 번호를 이

용하여 HL7 데이터 베이스의 테이블 값을 검색하여 사용할 수 있는 값을 가져 올 수 있다.

② 메시지 생성

HL7 V2.4 메시지는 스트링 형태로 되어있지만, 델리미터의 위치에 따라 데이터의 의미가 완전히 달라진다. 메시지를 만들 때, 공통된 모듈이 없으면 많은 오류가 발생하고 각 메시지마다 다른 형태로 제작되게 된다. 그래서, 인터페이스 엔진에 메시지를 만드는 모듈을 정의 함으로써, 메시지 처리를 스트링 단위로 하는 것이 아니라, 메시지 모델링을 통한 클래스 단위로, 각 필드의 정보를 바탕으로 메시지를 만들게 된다. 그러므로 오류를 줄이고, 편리성과 효율성을 제공한다.

- 전체 메시지 생성

HL7 V2.4 메시지는 트리 구조 형태로 되어있다. 가장 상위에 메시지가 있고, 그 밑으로 세그먼트, 필드, 데이터타입 들로 이루어 져 있다. 그리고, 이러한 계층의 구분은 델리미터에 의해 이루어진다. 전체 메시지를 만들 때는 가장 하위의 데이터타입부터 시작해서, 한 단계씩 올라갈 때 마다 델리미터를 넣어서 그 하위 노드의 요소를 결합하여 상위 노드의 단계로 올라가게 된다. 데이터타입의 콤포지트 내부에는 다시 콤포지트를 포함할 수 있다. 이론적으로는 무한 반복의 가능성으로 오류가 발생할 수 있지만 표준에서는 3단계이상 반복되지 않는다. 하지만, 서브콤포넌트의 델리미터가 다르므로 데이터타입은 그 단계에 따라 레벨로 구분을 한다.

- MSH 세그먼트 생성

MSH 세그먼트는 메시지 내에서 가장 중요한 세그먼트이다. 메시지 헤더의 역할로, 메시지의 대한 기본 정보를 보유하고 있다. 메시지의 이름, 메시지 전송과 관련한 송신자와 수신자, Event type, delimeter 정의등, 전체 메시지 형태를 이루는 필요한 정보를 보유하고 있다. 만약 MSH 세그먼트의 내용이 바뀌면 전체 메시지 구조가 바뀌게 되고, 해석하는데 오류가 발생할 가능성이 높다. MSH 세그먼트에서 Message Type의 경우에는 조금이라도 바뀌면 메시지 구조에 큰 영향을 끼치므로 처음 메시지가 생성될 때 외에는 값에 변화를 줄 수 없게 하고, 나머지 필드도 값을 초기화한 후에 값의 변화가 생길 때는 사용자에게 주의를 줄 필요가 있다.

- 반복 처리

HL7 V2.4 메시지에는 두 가지 경우의 반복이 존재한다. 하나는 세그먼트 단위이고, 다른 하나는 필드 단위이다. 그룹도 또한 반복이 가능한데, 이것은 세그먼트와 같은 영역으로 다룬다. 반복 처리는 반복 가능한 세그먼트와 필드에 대한 객체를 그 위치 다음에 추가하면 되지만, 메시지를 관리하는데 불편한 점이 있을 수 있다. 각 세그먼트와 필드에 대한 접근을 위치로 접근을 할 때, 고정된 위치가 아니라 유동적인 위치이기 때문에, 접근하는데 어려움이 따를 가능성이 높다. 그래서, 기본적인 메시지 구조에서, 반복 가능한 세그먼트와 필드가 추가 될 때에는 그 위치에서 트리 구조로 생각한다면, 선택된 노드에서 형제 노드에 추가하는 것이 아니고 자식 노드의 형태로 추가를 한다. 그럼으로써, 기본 메시지 구조의 형태를 그대로 유지하면서, 반복 처리가 가능하게 한다.

- Varies 필드 처리

MFE, OBX, QPD 세그먼트에는 'varies' 란 데이터 형이 있다. OBX를 예를 들면, OBX의 2번째 필드의 값에 따라 'varies' 필드의 데이터 타입을 결정한다. 즉, 이 형식은 메시지를 형태를 만들 때 결정 되는 것이 아니고, OBX 세그먼트 2번째 필드의 값이 결정될 때 결정 되어 진다. 이 'varies' 필드에 대해서는 데이터타입 결정을 뒤로 미루고, 데이터타입을 결정하는 필드에다 값이 변할 때, 이벤트를 발생시켜 알려 주게 한다. 이 이벤트를 받았을 때 필드의 데이터타입을 결정한다.

- 데이터 입력 처리

Data를 입력할 때는 단순히 값을 넣는 것이 아니라, 데이터에 대한 검증이 필요하다. 문자형이면, 검사가 어렵지만, 날짜와 시간을 나타내는 형, DT, TM, TS, 은 정수형을 기본으로 한 데이터타입이므로 검증이 가능하다. 그리고, 테이블과 연결된 데이터타입도 테이블에서 정의한 값인지 검사 해 봄으로써, 메시지의 오류를 줄일 수 있다.

HL7 V2.4 메시지는 트리 구조 형태이기 때문에, 한번 상위의 값을 얻기 위해서는 하위의 무수한 데이터타입에 대해 접근해야 한다. 이때 많은 시간이 소비될 수 있으므로, 접근 단계를 줄일 필요가 있다. 그래서, 데이터가 입력될 때, 이 값을 필드 단계에서 보유한다. 그럼으로써, 하위의 데이터타입에 까지 데이터를 접근하는 시간을 줄인다. 데이터타입에 값이 들어오면 이벤트를 발생시켜, 현재 데이터타입이 소속된 필드 전체의 값을 다시 재 정의하게 한다.

③ 메시지 전송

HL7은 의료 환경 내에서 전자 문서 교환을 위한 어플리케이션 레벨의 프로토콜이고, 클라이언트-서버 시스템 모델을 바탕으로 한다. 어플리케이션 레이어의 HL7메시지는 admission, discharge, and transfer; order entry and result reporting (generic and specific); queries; finance and patient account; and master file updates를 다룬다.[5]

HL7 이름에서 알 수 있듯이, OSI 레이어의 7 어플리케이션 프로토콜이다. 하지만, HL7은 단순히 어플리케이션 레이어에만 머물지 않고, presentation, session, transport 서비스 까지 염두에 두고 있다. Ad hoc 환경(RS-232), transport Level의 환경(TCP/IP, DECNET, SNA), presentation level의 환경(IBM's SNA LU6.2, SUN Microsystems's NFS), inter-process 통신(e.g., Pipes in a UNIX System) 서비스에 의한 두 어플리케이션이 결합된 환경을 지원하는 것을 허용한다. 하지만, 이런 다양한 환경에서도 error free transmission, character conversion, 메시지 길이에 대해 지원을 해야 한다.

- Error Free Transmission

Error Free Transmission은 전송자가 메시지를 보낼 때, 원하는 메시지의 전체 바이트가 전송되었다는 것을 보장하는 것이다. Low-level에서오류 검사를 하지만 acknowledgment 메시지를 받을 때 까지는 메시지를 완전히 전송되었다고 보장하지 못한다. 클라이언트가 서버로 메시지를 보내면 서버는 우선 메시지의 기본적인 구조와 크기에 대해 체크를 해서 전송 중에 오류를 검사해서 알려 주게 된다. 서버를 통해 최종 목적지에 도달하면 메시지 검증의 모든 기능을 이용해 정확하게 메시지를 검사하고, 오류가 있으면 오류 메시지를 다시 보내게 된다.

- Character Conversion

HL7 에서는 Encoding rule로 ASCII를 기본으로 하고 있다. 필드 구분자는 ASCII의 표시가능 한 어떠한 문자를 사용해도 되지만, 세그먼트 구분자는 반드시 ASCII의 Carriage Return 문자를 사용해야 된다. 영어권외에도 다른 지역의 언어를 지원하기 위해 HL7에서는 다양한 Encoding rule을 지원한다. EC(The European Community)를 위해, ISO 8859나 기타 지역을 위해 Multi-character Code인 UNICODE를 또한 지원하고 있다. MSH 세그먼트의 18번째 필드에 character set을 정의하여 메시지의 encoding 방식을 설정한다. HL7 메시지를 받으면 이 필드에서 설정된 character set으로 character conversion을 하여 메시지를 처리한다. 유니코드 방식으로 전송한다면, 언어에 크게 영향을 받지 않기 때문에 유니코드 방식을 주요 encoding rule로 사용한다

- 메시지 길이

메시지 길이는 메시지의 종류와 도메인 따라 크게 다를 수 있다. 표준에서도 메시지의 길이에 대한 한계를 두지 않고 있다. 하지만, 메시지 전송을 할 때는 정해진 길이 만큼의 바이트만 한 번씩 이루어지게 된다. 그래서, 메시지의 시작과 끝에 대한 정의가 필요하다. HL7 V2.x에서는 메시지의 시작을 항상 MSH 세그먼트로 하기 때문에 시작점을 찾는 문제는 해결되었고, 메시지의 끝은 <EOF>필드를 붙여서 메시지의 끝점을 찾게 된다. 메시지가 발생하는 량이 많을 때는 큰 메시지가 계속 나누어져서 전송될 가능성이 높다. 그래서 메시지의 순서가 바뀌거나 다른 메시지와 결합될 수 있다. 수신측에서는 동기화 시키거나, 주고 받는 메시지의 길이를 최대한 적게 나누는 바이트 수를 실제 시스템에 적용하여 가장 효율적인 지점을 경험적으로 찾아야 한다.

5.Conclusion

HL7 V2.4 메시지 인터페이스 엔진은 API를 제공하기 위한 라이브러리 형태로 제작되었다. HL7 V2.4 메시지를 이용하는 의료관련 어플리케이션이나 시스템에 이 라이브러리를 포함하여 개발하면, HL7 V2.4 메시지를 사용할 수 있게 된다. 하지만, 인터페이스 엔진은 기본 기능만 제공하기 때문에, 개발자는 방대한 HL7 V2.4 표준에 대한 이해가 필요하다. 인터페이스 엔진을 사용하는 사용자의 요구사항과 개발 속도를 높이기 위해 인터페이스 엔진 기능을 확장한 인터페이스 툴킷 개발이 필요하다. 현재 후속 연구로 HL7 V2.4 메시지 인터페이스 툴킷을 개발하고 있다.

참고 문헌

[1] HL7 URL : http://www.hl7.org/

[2] Mandle KD, Kohane IS. Healthconnect: Clinical grade patient-physician communication. In Proceedings, AMIA Annual Symposium 1999

[3] Rhapsody Document, Orion, 2002

[4] Eah-Wen Huang, Sheng-Hsiung Hsiao, Der-Ming Liou, "Design and implementation of a web-based HL7 message generation and validation system", International Journal of Medical Informatics (2003) 70, 49-58

[5] Andrew Hutchison, Matthias Kaiserswerth, Michael Moser, and Andreas, "Electronic Data Interchange for Health Care", IEEE Communication Magazine, July 1996

[6] Alfredo Tirado-Ramos, Jingkun Hu, K. P. Lee, "Information Object Definition-based Unified Modeling Language Representation of DICOM Structured Reporting", JAMIA Volume 9 Number 1 Jan/Feb 2002

[7] Junghoo Cho, Sridhar Rajagopalan, "A Fast Regular Expression Indexing Engine", Proceeding of the 18th International Conference on Data Engineering, 2002

[8] Muhammad F. Mudawwar, "Multicode : A Truly Multilingual Approach to Text Encoding", IEEE, April 1997