(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2008-0100348
(43) 공개일자 2008년11월17일

(51) Int. Cl.
G06F 17/21 (2006.01) *G06F 17/00* (2006.01)
(21) 출원번호 10-2008-7020673
(22) 출원일자 2008년08월22일
심사청구일자 없음
번역문제출일자 2008년08월22일
(86) 국제출원번호 PCT/US2007/001546
국제출원일자 2007년01월19일
(87) 국제공개번호 WO 2007/097848
국제공개일자 2007년08월30일
(30) 우선권주장
11/361,263 2006년02월24일 미국(US)

(71) 출원인
마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
가우라브, 수라즈
미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이
마치라주, 수렌드라
미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이
(74) 대리인
양영준, 백만기

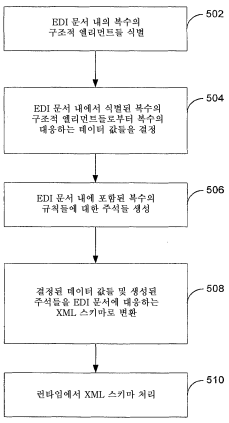
전체 청구항 수 : 총 20 항

(54) ＥＤＩ 문서 모델링 방법, 시스템, 및 컴퓨터 판독 가능 매체

(57) 요 약

XML(eXtensible Markup Language)을 이용하는 EDI(electronic data interchange) 문서 모델링이 개시된다. 복수의 구조적 엘리먼트들이 EDI 문서 내에서 식별된다. EDI 문서 내에서 식별된 복수의 구조적 엘리먼트들로부터 복수의 대응 데이터 값들이 식별된다. EDI 문서에 포함된 복수의 규칙들에 대한 복수의 주석들이 생성된다. 복수의 규칙들은 복수의 대응 데이터 값들과 연관된 연산들을 규정한다. 결정된 데이터 값들 및 생성된 주석들은 EDI 문서에 대응하는 XML 스키마로 변환된다. XML 스키마는 런타임시 처리된다.

대 표 도 - 도5



특허청구의 범위

청구항 1

런타임시 XML(eXtensible Markup Language)를 이용하여 EDI(electronic data interchange) 문서를 모델링하는 방법에 있어서,

상기 EDI 문서 내의 복수의 구조적 엘리먼트(structural element)들을 식별하는 단계;

상기 EDI 문서 내의 상기 식별된 복수의 구조적 엘리먼트들로부터 복수의 대응 데이터 값을 결정하는 단계;

상기 EDI 문서에 포함된 복수의 규칙들에 대한 주석들(annotations)을 생성하는 단계 - 상기 복수의 규칙들은 상기 복수의 대응 데이터 값들과 연관된 연산들(operations)을 규정함 - ;

상기 결정된 데이터 값들 및 상기 생성된 주석들을 상기 EDI 문서에 대응하는 XML 스키마로 변환하는 단계; 및

런타임시 상기 XML 스키마를 처리하는 단계

를 포함하는 EDI 문서 모델링 방법.

청구항 2

제1항에 있어서, 상기 식별하는 단계는 상기 복수의 구조적 엘리먼트들에 포함된 하나 이상의 루프들 및 하나 이상의 세그먼트들을 식별하는 단계를 포함하는 EDI 문서 모델링 방법.

청구항 3

제2항에 있어서, 상기 식별된 세그먼트들 중 하나 이상과 연관된 하나 이상의 서브세그먼트들을 식별하는 단계를 더 포함하는 EDI 문서 모델링 방법.

청구항 4

제1항에 있어서, 상기 주석들을 생성하는 단계는 상기 복수의 규칙들에 대한 주석들을 생성하는 단계를 포함하며, 상기 복수의 규칙들은 교차 필드 확인 규칙들(cross field validation rules), 트리거 필드 정보, 레퍼런스 지정자(reference designator), 및 스플릿 포인트 정보(split point information) 중 하나 이상을 포함하는 EDI 문서 모델링 방법.

청구항 5

제1항에 있어서, 상기 결정된 데이터 값들의 구조를 규정하기 위한 복수의 스키마 규칙들을 따르는 XML 스키마를 구성하는 단계를 더 포함하는 EDI 문서 모델링 방법.

청구항 6

제1항에 있어서, 상기 주석들을 생성하는 단계는 상기 복수의 규칙들에 대한 하나 이상의 속성들을 포함하는 상기 주석들을 생성하는 단계를 포함하는 EDI 문서 모델링 방법.

청구항 7

제1항에 있어서, 하나 이상의 컴퓨터 판독 가능 매체가 제1항의 방법을 수행하기 위한 컴퓨터 실행 가능 명령어들을 갖는 EDI 문서 모델링 방법.

청구항 8

런타임시 XML을 이용하여 EDI 문서를 모델링하는 시스템(100)에 있어서,

상기 EDI 문서 내의 복수의 구조적 엘리먼트들을 식별하는 단계;

상기 EDI 문서 내의 상기 식별된 복수의 구조적 엘리먼트들로부터 복수의 대응 데이터 값들을 결정하는 단계;

상기 EDI 문서에 포함된 복수의 규칙들에 대한 주석들을 생성하는 단계 - 상기 복수의 규칙들은 상기 복수의 대응 데이터 값들과 연관된 연산들을 규정함 - ;

상기 결정된 데이터 값들 및 상기 생성된 주석들을 상기 EDI 문서에 대응하는 XML 스키마로 수정하는 단계; 및

상기 XML 스키마를 처리하는 단계

를 위한 컴퓨터 실행 가능 명령어들을 실행하는 프로세서(106)를 포함하는 EDI 문서 모델링 시스템(100).

청구항 9

제8항에 있어서, 상기 식별된 복수의 구조적 엘리먼트들 내의 복수의 데이터 값들을 파싱(parsing)하기 위한 파서(112)를 더 포함하는 EDI 문서 모델링 시스템(100).

청구항 10

제8항에 있어서, 상기 프로세서(106)는 상기 복수의 구조적 엘리먼트들에 포함된 하나 이상의 루프들 및 하나 이상의 세그먼트들을 식별하도록 구성된 EDI 문서 모델링 시스템(100).

청구항 11

제10항에 있어서, 상기 프로세서(106)는 상기 식별된 세그먼트들 중 하나 이상과 연관된 하나 이상의 서브세그먼트들을 식별하도록 더 구성된 EDI 문서 모델링 시스템(100).

청구항 12

제8항에 있어서, 상기 프로세서(106)는 상기 복수의 규칙들에 대한 주석들을 생성하도록 구성되며, 상기 복수의 규칙들은 교차 필드 확인 규칙들, 트리거 필드 정보, 레퍼런스 지정자, 및 스플릿 포인트 정보 중 하나 이상을 규정하는 EDI 문서 모델링 시스템(100).

청구항 13

제8항에 있어서, 상기 프로세서(106)는 상기 결정된 데이터 값들의 구조를 규정하기 위한 복수의 스키마 규칙들을 따르는 XML 스키마를 규정하도록 더 구성되는 EDI 문서 모델링 시스템(100).

청구항 14

제8항에 있어서, 상기 프로세서(106)는 상기 복수의 규칙들에 대한 하나 이상의 속성들을 포함하는 주석들을 생성하도록 구성되는 EDI 문서 모델링 시스템(100).

청구항 15

런타임시 XML을 이용하여 EDI 문서를 모델링하는 컴퓨터 실행 가능 컴포넌트들을 갖는 하나 이상의 컴퓨터 판독 가능 매체(402)에 있어서, 상기 컴퓨터 실행 가능 컴포넌트들은,

상기 EDI 문서 내의 복수의 구조적 엘리먼트들을 식별하는 엘리먼트 컴포넌트(404);

상기 EDI 문서 내의 상기 식별된 복수의 구조적 엘리먼트들로부터 복수의 대응 데이터 값들을 결정하는 데이터 컴포넌트(406);

상기 EDI 문서에 포함된 복수의 규칙들에 대한 주석들을 생성하는 주석 컴포넌트(408) - 상기 복수의 규칙들은 상기 복수의 대응 데이터 값들과 연관된 연산들을 규정함 - ;

상기 결정된 데이터 값들 및 상기 생성된 주석들을 상기 EDI 문서에 대응하는 XML 스키마로 변환하는 변환 컴포넌트(410); 및

런타임시 상기 XML 스키마를 처리하기 위한 페이로드 컴포넌트(412)

를 포함하는 컴퓨터 판독 가능 매체(402).

청구항 16

제15항에 있어서, 상기 엘리먼트 컴포넌트(404)는 상기 복수의 구조적 엘리먼트들에 포함된 하나 이상의 루프들 및 하나 이상의 세그먼트들을 식별하는 컴퓨터 판독 가능 매체(402).

청구항 17

제16항에 있어서, 상기 엘리먼트 컴포넌트(404)는 상기 식별된 세그먼트들 중 하나 이상과 연관된 하나 이상의 서브세그먼트들을 더 식별하는 컴퓨터 판독 가능 매체(402).

청구항 18

제15항에 있어서, 상기 주석 컴포넌트(408)는 상기 복수의 규칙들의 주석들을 생성하고, 상기 복수의 규칙들은 교차 필드 확인 규칙들, 트리거 필드 정보, 레퍼런스 지정자, 및 스플릿 포인트 정보 중 하나 이상을 규정하는 컴퓨터 판독 가능 매체(402).

청구항 19

제15항에 있어서, 상기 변환 컴포넌트(410)는 상기 결정된 데이터 값들의 구조를 규정하는 복수의 스키마 규칙들을 따르는 XML 스키마를 더 규정하는 컴퓨터 판독 가능 매체(402).

청구항 20

제15항에 있어서, 상기 주석 컴포넌트(408)는 상기 복수의 규칙들에 대한 하나 이상의 속성들을 포함하는 주석들을 생성하는 컴퓨터 판독 가능 매체(402).

명 세 서

배 경 기 술

<1> EDI(electronic data interchange)는 승인된 포맷팅 표준들 및 스키마들에 기초한 컴퓨터 간 기업 정보 교환을 위하여 기업이 사용하는 방법 중 하나이다. 예를 들어, 전 세계적으로 수백만의 회사들이 교역을 하기 위해 EDI를 이용하여 기업 트랜잭션(business transactions)과 연관된 데이터(예를 들면, 구매 오더, 선하/항공 화물 증권, 송장 등)를 송신한다.

<2> 전형적인 EDI 트랜잭션 모델에서, 대기업 실체(large business entity) 또는 EDI 통합 브로커는 많은 파트너들과 무역을 하며 많은 EDI 트랜잭션 데이터를 다양한 EDI 포맷들 및 스키마들로 처리하는 기술적 능력을 갖는다. "허브들(hubs)"로도 알려진 이 실체들은 "스포크들(spokes)"로도 알려진 하나 이상의 공급자들과 트랜잭션한다. 스포크들의 각각은 전형적으로는 하나의 허브만을 처리할 수 있는 비교적 소기업 실체이다.

<3> 스포크들이 EDI를 통해 허브와의 트랜잭션들을 개시하려고 시도하기 전에, 허브는 전형적으로는 스포크들이 EDI 스키마들에 따라 EDI 트랜잭션들을 적절히 포맷할 수 있도록 다양한 EDI 스키마들을 스포크들에 송신한다. 현재는, EDI 스키마들은 크기가 크며 각각의 EDI 스키마의 파일 크기는 보통은 1 MB 내지 3 MB의 범위에 있다. 또한, 허브 또는 큰 무역 파트너들은 스포크들의 하드웨어 성능 부족을 고려하지 않고, 많은 양의 스키마들을 스포크들에 관례적으로 송신한다. 이와 같이, 송신 중에 대역폭에서 수 기가바이트를 차지할 수 있는 그러한 수천의 스키마들은 허브로부터 스포크들로 송신된다.

<4> 스포크들의 송신 대역폭을 소모하는 많은 양의 스키마들을 송신하는 것 이외에, 관련된 시스템들 또는 기술들은 선형 또는 플랫 파일 기반으로 EDI 스키마들을 규정하여 EDI 스키마들을 나타낸다. 이러한 표현(representation)의 유형들 또는 모델들은, 단순한 구성을 가지면서, EDI 스키마들과 연관된 부가적인, 때로는 중요한 정보를 나타내는 완전한 능력을 갖지 못한다. 이와 같이, 스포크들과 허브는 마찬가지로 EDI 트랜잭션들의 편리성의 완전한 이익을 최대화시키지 못할 수 있다.

발명의 상세한 설명

<5> 발명의 개요

<6> 본 발명의 실시예들은 XML 구성들을 이용하여 구조적 EDI 문서들을 기술함으로써 EDI 스키마들을 나타내는 이전의 방법들 및 실시들을 강화한다. 런타임 동안 EDI 페이로드(payload)를 XML 페이로드로서 모델링함으로써, 본 발명의 실시예들은 스키마들 및 변환 맵들을 개발하는 시간을 감소시킨다. 또한, 대안의 실시예들은 EDI 데이터에 존재하는 복수의 확인 규칙들(validation rules)과 연관된 정보를 포함하는 XML EDI 스키마들 내에 주석들(annotations)을 생성한다.

<7> 이 개요는 상세한 설명부에서 후술되는 개념들의 선택을 간략화된 형태로 소개하기 위하여 제공된다. 이 개요는 본원의 청구물의 주요한 특징 또는 본질적인 특징을 식별하기 위한 것이 아니며, 본원의 청구물의 범위를 결정하는 데에 도움이되기 위한 것도 아니다.

<8> 기타 특징들은 이하에서 부분적으로 자명해지고 부분적으로 지적될 것이다.

실 시 예

<16> 먼저 도 1을 참조하면, 블럭도는 본 발명의 일 실시예에 따른 EDI 트랜잭션을 수행하기 위한 시스템(100)을 도시한다. 시스템(100)은 하나 이상의 스포크들(spokes, 104)에 링크되고 그들과 통신하는 허브(hub, 102)를 포함한다. 일 실시예에서, 허브(102)는 스포크들(104)을 서브하기 위한 컴퓨터 판독 가능 명령어들을 실행하기 위한 하나 이상의 프로세서들(예를 들면, 프로세서(106)) 또는 프로세싱 유닛들을 서브하는 서버 컴퓨터 또는 컴퓨팅 장치를 포함한다. 일 예에서, 스포크들(104)은 도 6에 도시된 바와 같은 컴퓨터(130)에 포함되거나 그것과 연결된 하나 이상의 컴포넌트들을 갖는 컴퓨팅 장치를 포함한다.

<17> 일 예에서, 허브(102)는 또한 EDI 스키마(110)와 같은 하나 이상의 EDI 스키마들을 저장하기 위한 메모리 영역(108)을 포함한다. 처음에, 허브(102) 및 스포크들(104)은 그들 간에 트랜잭션 데이터를 송신하기 위해 이용될 EDI 포맷들 또는 표준들에 대한 합의를 구축한다. 당사자들(parties)이 이용할 특정의 EDI 포맷들 또는 표준들을 결정하면, 허브(102)는 스포크들(104)에 송신될 적절한 EDI 스키마들을 선택한다. 다른 예에서, 허브(102)는 구매 오더들(purchase orders), 선하증권들(bills of lading), 송장들(invoices), 임금 지급장들(payrolls) 등과 같은, 스포크들(104)로의 모든 트랜잭션 유형들에 대한 모든 EDI 스키마들을 선택할 수 있다. 허브(102)와 스포크들(104) 간의 통신은 개인 또는 공중(public) 통신 네트워크, 유선 또는 무선 네트워크일 수 있지만, 스포크들(104)은 허브(102)로부터 송신된 많은 양의 EDI 스키마들을 처리하기 위한 하드웨어 자원들이 부족한 것이 전형적이다. 또한, 스포크들(104)에 대한 컴퓨팅 네트워크 통신의 유형 및 대역폭은, 데이터 크기가 수 기가바이트에 이를 수 있는 수천의 EDI 스키마들에 의해 부과된 요구를 처리하도록 갖추어져 있지 않다.

<18> 이제 도 2를 참조하면, 블럭도는 본 발명의 일 실시예에 따른 XML을 이용하는 구조적 EDI 스키마를 도시한다. 전술된 바와 같이, 관련된 관행들은 간략화된, 선형, 또는 플랫(flat) 구성을 이용하여 EDI 스키마를 나타낼 것이다. 표 1은 좌측 컬럼에 본 발명의 양태들을 구현하는 구조적 EDI 스키마에서의 세 개의 EDI 트랜잭션들(transactions) 및 그에 대응하는 전통적인 EDI 표현에서의 세 개의 트랜잭션들을 예시한다.

표 1

구조 내의 EDI 거래들	플랫 EDI 거래들
BeginOfTransaction#1 POHeaderSegment POLine1 POSchedule1.1 POSchedule1.2 POLine1Totals POLine2 POSchedule2.1 POLine2Totals POTotals EndOfTransaction#1	BeginOfTransaction#1a POHeaderSegment POLine1 POSchedule1.1 POLine1Totals POTotals EndOfTransaction#1a BeginOfTransaction#1b POHeaderSegment POLine1 POSchedule1.2 POLine1Totals POTotals EndOfTransaction#1b BeginOfTransaction#1c POHeaderSegment POLine2 POSchedule2.1 POLine2Totals
	POTotals EndOfTransaction#1c

<19>

<20> 표 1: 구조 내의 세 개의 EDI 트랜잭션들(좌측 컬럼) 및 세 개의 EDI 문서들(우측 컬럼)

<21> 예시된 바와 같이, 플랫 EDI 스키마는 더 많은 문자들을 사용하므로 좌측 컬럼에 있는 구조적 EDI 스키마보다 더 많은 데이터 크기를 차지한다. 일 실시예에서, EDI 스키마들의 구조는 런타임동안의 더 빠른 프로세싱 및 공지의 XML 편집 툴들을 이용하는 사용자들에 의한 손쉬운 구성(configuring), 편집(editing), 및 수정

(modifying)을 가능하게 하는 XML을 이용하여 나타내어 진다.

<22> 일 예에서, EDI 스키마는 명칭 공간(namespace) 및 루트 노드 명칭(root node name)의 조합을 포함할 수 있는 DocType에 의해 식별된다. 일 예에서, DocType은 TargetNamespace '#' RootNodeName으로 정의된다. X12 표준이 이용되는 예에서, X12 스키마는 다음의 포맷을 이용한다:

<23> $$X12\ Schemas = X12\_\{Version\}\_\{TsId\}$$

<24> 이것이 가리키는 바는 다음과 같다.

<25> 1) 모든 X12 스키마들은 X12로 시작하는 루트 노드 명칭을 갖고;

<26> 2) "Version"은 문서의 버전 정보를 나타내며, 이것은 구성 또는 인스턴스 구동형인 정보의 동적인 부분이며(a dynamic piece of information which is configuration or instance driven);

<27> 3) "TsId"는 프로세스 중인 문서의 "트랜잭션(transaction) ID"를 의미하는 것이며 항상 입력 인스턴스로부터 판독된다.

<28> 다른 예로서, 다른 공지의 표준 EDIFACT는 EDIFACT 스키마들을 나타내기 위한 다음의 포맷을 포함한다;

<29> $$EDIFACT\ Schemas = Efact\_\{Version\}\_\{Tsid\}$$

<30> 이것이 가리키는 바는 다음과 같다.

<31> 1) 모든 EDIFACT 스키마들은 Efact로 시작하는 루트 노드 명칭을 갖고;

<32> 2) "Version"은 문서의 버전 정보를 나타내며, 이것은 구성 또는 인스턴스 구동형인 정보의 동적인 부분이며;

<33> 3) "TsID"는 프로세스 중인 문서의 "트랜잭션 ID"를 의미하며 항상 입력 인스턴스로부터 판독된다.

<34> 도 2에 도시된 바와 같이, EDI 스키마(202)의 구조적 개요는 하나 이상의 루트 노드들(204)을 포함할 수 있다. 각각의 루트 노드(204)는 자식(children)으로서 루프들(206) 및 세그먼트들(208)의 시퀀스를 포함한다. 루프들(206)은 하나 이상의 중첩된(nested) 서브 루프들(210) 또는 서브 세그먼트들(212)을 포함할 수 있다.

<35> 유사하게, 세그먼트(208)는 하나 이상의 컴포넌트 데이터 엘리먼트들(214) 및 데이터 엘리먼트들(216)을 포함할 수 있다. 일 실시예에서, EDI 스키마를 나타내는 XML 페이로드 스키마는 다음의 구조를 소유할 수 있다:

1. Schema -> RootNode
2. RootNode -> (Block)+
3. Block -> Segment | Loop
4. Loop -> (Block)+
5. Segment -> (DataElement)+, RuleSet
6. DataElement -> SimpleField | CompositeField
7. SimpleField -> data
8. CompositeField -> (data)+
9. RuleSet -> (Rule)+ | BlankRuleSet
10. Rule -> PairedRule | RequiredRule | ExclusionRule | ConditionalRule | ListRule.

<36>

<37> 일 실시예에서, XML을 이용하여 EDI 스키마들을 모델링하는 경우, 복수의 스키마 규칙들이 적용되어 XML 스키마를 구성한다. 예를 들어, 복수의 규칙들은 다음을 포함할 수 있다:

<38> 1. 하나 이상의 엘리먼트들(예를 들면, 루프, 세그먼트 등)은 속성들이 없는 각각의 레벨에 있다.

<39> 2. 모든 ComplexType은 'sequence' 컴포지터(compositor)를 이용할 것이다. 일 예에서, ComplexType은 계층 데이터 실체들을 모델링하기 위한 구성체(construct)이다. 각각의 ComplexType은 그 자식을 그룹화하기 위해 "sequence", "choice" 또는 "any" 컴포지터를 이용할 수 있으며, 각각의 컴포지터는 다음에 따라 이용될 수 있

다.

<40> a) Sequence - 자식은 XML 인스턴스 내에서 특정한 순서로 발생해야 한다.

<41> b) Choice - 자식 중 하나만 XML 인스턴스에 나타날 수 있다.

<42> c) Any - 자식은 XML 인스턴스에서 임의의 순서로 나타낼 수 있다.

<43> 일 예에서, EDI 스키마는 "sequence" 컴포지터만을 이용한다. 대안의 실시예에서, "choice" 및 "any" 컴포지터가 이용될 것이다. ComplexType은 오로지 자식으로서 포함된 엘리먼트들일 수 있다.

<44> 3. 문서의 루트 노드로부터 시작하여, 서브스트링 "루프(Loop)"를 포함하는 임의의 엘리먼트는 루프 엘리먼트로 간주될 것이다. 그렇지 않은 경우라면, 세그먼트(segment)로 간주될 것이다(이하의 부가 설명 참조).

<45> 4. 세그먼트가 식별되었다면, 세그먼트의 서브스트링들은 식별되지 않을 것이다.

<46> 5. 세그먼트들은 구조적 XML 스키마에서 자식 및 손자를 가질 수 있다. 예를 들어, 세그먼트 노드로부터 거리 1 및 2에 있는 자손들(descendants)만 구조의 일부로서 허용될 것이다.

<47> 6. EDI 스키마들 내의 규칙들을 나타내기 위하여 하나 이상의 주석들이 생성된다. 예를 들어, 다음 중 하나 이상을 나타내기 위하여 주석들이 이용될 수 있다.

<48> a) 교차 필드 확인 규칙들(cross field validation rules);

<49> b) HIPAA(Health Insurance Portability and Accountability Act)에 순응하는 것과 연관된 스키마들에 대하여 이용될 트리거 필드 정보;

<50> c) 레퍼런스 지정자(reference designator);

<51> d) 스플릿 포인트 정보 규칙들(split point information rules). 예를 들어, 스플릿 포인트 정보는 EDI 스키마들의 요건들을 충족시키는 EDI 스키마 또는 EDI 트랜잭션들이 복수의 하위문서들(sub-documents)로 스플릿되었을 경우의 정보를 나타낸다.

<52> 일 실시예에서, XML 스키마는 하나 이상의 2 또는 3 문자 태그들을 이용하여 구조화된다. 도 3은 본 발명의 양태들을 구현하는 XML 스키마를 예시하는 도면이다. 예를 들어, 패널(302)은 하나 이상의 2 또는 3 문자 태그들을 예시한다. 예를 들어, 태그들은 "ST", "BEG", "CUR"을 포함한다.

<53> 일 실시예에서, 세그먼트 엘리먼트의 명칭에 기초하여, 처음 세 문자들이 tagID로 간주될 수 있다. 대안의 실시예에서는, 3 문자 태그의 세번째 문자가 "-" 문자라면, 세그먼트 ID는 오직 처음 두 문자들일 것이다. 또한, 두 문자들 중 하나가 "-" 문자를 포함하면, 컴파일 에러(compilation error)가 발생할 것이다. 예를 들어,

<54> a) "BEG_BeginTransaction"은 세그먼트 ID가 BEG인 것을 의미하고;

<55> b) "N1_850Segment"는 세그먼트 ID가 N1인 것을 의미하고;

<56> c) "N_abc"는 무효(invalid) 세그먼트를 의미하고;

<57> d) "N1_Loop"는 이것이 세그먼트가 아니라 루프라는 것을 의미하며;

<58> e) "N_Loop"는 이것이 무효 세그먼트가 아니라 루프라는 것을 의미한다.

<59> 다른 실시예에서, 하나 이상의 특별한 규칙들이 세그먼트 태그들의 세트에 적용 가능하다. 예를 들어, "ISA", "IEA" 등과 같은 태그들은 특별한 중요도를 가지며 이들은 제어 세그먼트들로 호칭된다. 유사하게, "ST" 및 "SE"와 같은 태그들은 제어 및 데이터 세그먼트들이다. 이하의 리스트는 제어 세그먼트 태그들의 예시적인 세트를 제공한다.

ISA

GS

ST

<60> SE

GE

IEA

UNOA

UNB

UNG

UNH

UNT

UNE

<61> UNZ

<62> 다른 예에서, 본 발명의 실시예들은 주석들을 이용하여 하나 이상의 규칙들을 설명함으로써 EDI 스키마들의 기능을 강화한다. 예를 들어, 하나 이상의 교차 필드 확인 규칙들의 처리시, 규칙들은 세그먼트 레벨 노드들에서 주석들로서 표현된다. 모든 세그먼트는 RuleSet으로도 알려진 선택적인 규칙 리스트를 포함한다. 예를 들어, 패널(304)은 구조적 XML 스키마 내에서의 하나 이상의 주석 태그들을 예시한다. 예로서, 다음의 문단들에서 예시적인 규칙들의 세트가 설명된다.

<63> Paired/Multiple - (all, absent) - 모든 주제 명칭들(subject names)이 존재 또는 부재한다.

<64> 예를 들면:

```
<b:Rule subjects="X12ConditionDesignatorX_Paired ">
        <b:Subject name="@CUR11" />
        <b:Subject name="@CUR12" />
</b:Rule>
```

<65>

Required - (grouped, absent) – 주제들 중 적어도 하나가 요구된다 :

```
<b:Rule subjects="X12ConditionDesignatorX_Required ">
        <b:Subject name="@CUR11" />
<66>
        <b:Subject name="@CUR12" />
</b:Rule>
```

Exclusion – (any, absent) - 단 하나의 주제만이 제공될 수 있다 :

```
<b:Rule subjects="X12ConditionDesignatorX_Exclusion ">
        <b:Subject name="@CUR11" />
        <b:Subject name="@CUR12" />
</b:Rule>
```

Conditional - (all, value) - 한정자(qualifier)가 존재한다면, 모든 주제가 요구된다 :

```
<b:Rule subjects="X12ConditionDesignatorX_Conditional"
qualifier="@CUR05">
        <b:Subject name="@CUR11" />
        <b:Subject name="@CUR12" />
</b:Rule>
```

List - (any, value) - 한정자가 존재한다면, 적어도 하나의 주제가 요구된다 :

```
<b:Rule subjects="X12ConditionDesignatorX_ListConditional"
qualifier="@CUR05">
        <b:Subject name="@CUR11" />
        <b:Subject name="@CUR12" />
</b:Rule>
```

<67>

<68> EDI 스키마들의 이전의 표현 또는 모델과 달리, 본 발명의 실시예들은 XML을 이용하여 EDI 스키마들과 연관된 규칙들을 설명하기 위한 풍부한 정보 세트를 제공한다.

<69> 상기 예시된 바와 같이, 각각의 규칙은 "주제(subject)" 속성, 선택적인 한정자(qualifier) 속성, 및 주제 엘리먼트들의 리스트를 갖는다. 예를 들어, 규칙들의 각각은 (주제들, 한정자) 쌍들의 값에 기초하여 규정된다.

<70> 일 실시예에서, 규칙의 오퍼랜드들(operands)은 세그먼트에서 발견된 데이터 엘리먼트들이다. 이 실시예에서, 규칙 평가 동안, 오퍼랜드의 특정한 값은 필요치 않다. 오히려, 필요한 것은 데이터 엘리먼트가 값이 정해졌는지(valued) 여부에 대한 정보이다. 예로서 도 1의 시스템(100)을 이용하여, 파서/시리얼라이저(parser/serializer)(112)에 의한 파싱 또는 직렬화 동안, 세그먼트가 나타나면, 파서/시리얼라이저(112)는 블랭크 아닌(non-blank) 값을 가진 데이터 엘리먼트들을 추적할 수 있다. 세그먼트 프로세싱이 종료된 후, 파서/시리얼라이저(112)는 모든 규칙 오퍼랜드들을 평가할 수 있으므로 RuleSet를 실행할 수 있다. 규칙 오퍼랜드들의 프로세싱시 에러들이 발생한 경우, 하나 이상의 에러 코드들이 발행된다. 일 예로서, 특정의 규칙 위반들에 대응하는 적절한 에러 코드들의 예시적인 세트가 이하 설명된다:

<71> 에러 코드 2 - 조건부 요구 데이터 엘리먼트 상실(Conditional required data element missing) - 이것은 조건부 규칙이 위반될 때 보고될 것이다. 보고된 필드 번호는 부재(absent)하는 첫번째의 것(규칙에서 정의된 순서에서)이 될 것이다.

<72> 에러 코드 10 - 배제 조건 위반(Exclusion condition violated) - 이것은 배제 규칙이 위반될 때 보고될 것이다. 보고된 필드 번호는 적어도 2개 필드가 값이 정해지게 하는 첫번째의 것(규칙에서 정의된 순서에서)이 될 것이다.

<73> 에러 코드 14 - (BTS-EDI 특정의) - 교차 필드 확인 규칙들 위반 - 이 에러는 모든 다른 종류의 규칙들에 대하여 일어날 것이며 다음의 상황에서 일어날 것이다:

<74> a) 쌍을 이룬 규칙(paired rule) - 보고된 필드 번호는 부재하는 첫번째의 것이 될 것이고;

<75> b) 요구된 규칙(required rule) - 이것은 모든 주제들이 부재하는 경우에만 위반될 수 있다. 이 유형의 시나리오에서, 규칙에서 언급된 첫번째 필드는 위반을 야기시킨 것으로서 보고될 것이고;

<76> c) 리스트 규칙(list rule) - 이것은 한정자가 존재하고 모든 주제들이 부재일 때 위반될 수 있다. 이러한 경우, 규칙에서 언급된 첫번째 필드는 위반을 야기시킨 것으로서 보고될 것이다.

<77> 또한, EDI 스키마들을 모델링할 때, 본 발명의 실시예들은 EDI 스키마들에 각종 데이터 유형들이 포함되는 것을 적절히 고려한다. 예로서, X12 포맷 내의 하나 이상의 데이터 유형들이 이하 설명된다:

<78> A.X12 데이터 유형들:

<79> X12는 다수의 데이터 유형들을 규정한다. 그러한 유형들은 파서/시리얼라이저에 의해 인식될 것이다. 이들은 그러한 유형들을 확인하고 임의의 필요한 변환을 수행하는 능력을 가질 것이다. 그러나, 문서 스키마는 그것에 연관된 임의의 종류의 단순한 유형을 가질 수 있다. 만일 그것이 EDI 데이터 유형이 아니라면, 스키마 확인 플래그(schema validation flag)가 턴온되어 있는지 아닌지에 따라, XmlValidatingReader와 같은 컴퓨터 실행 가능 명령어들, 확인된 코드들, 라우팅들(r일 것이다.

<80> 1. "N" - 이것은 선택적인 길이 제한을 갖는 정수 데이터 유형이다. 이 유형의 간단한 엘리먼트는 로컬 명칭 "N"을 갖는 유형에 대한 제한인 간단한 유형을 가져야 한다(A simple element of this type, should have a simple type that is a restriction on a type with local name "N".). 그리고 "N"은 본질적으로는 xs:string이다.

```
<xs:element minOccurs="0" name="ITD07">
<xs:simpleType>
<xs:restriction base="N"/>
<xs:minLength value="1" />
<xs:maxLength value="5" />
</xs:simpleType>
```

<81> </xs:element>

<82> 2. "Nn" - 이것은 n>0에 의해 특정된, 함축된 소숫점(implied decimal point)를 갖는 정수 유형이다. 따라서, 유형이 N2라면, 파서는 1234를 12.34로 변환할 것이다. 유사하게, 12.34는 시리얼라이저에 의해 1234로 변환될 것이다. 소숫점을 만나지 않거나 길이 제한이 따르지 않는다면, 적절한 EDI 에러 코드가 발생될 것이다. 이 유형의 간단한 엘리먼트는, 예를 들면, 로컬 명칭 "N2"을 갖는 유형에 대한 제한인 간단한 유형을 가져야 한다. 그리고 "N2"는 본질적으로 xs:string이다.

```
<xs:element minOccurs="0" name="ITD08">
<xs:simpleType>
                    <xs:restriction base="N2"/>
                        <xs:minLength value="4" />
                        <xs:maxLength value="8" />
                </xs:simpleType>
        </xs:element>
```

<83>

<84> 3. "ID" - 이것은 선택적인 길이 제한을 갖는 열거 데이터 유형(enumeration data type)이다. 이 유형의 간단한 엘리먼트는 로컬 명칭 "ID"를 갖는 유형에 대한 제한인 간단한 유형을 가져야 한다. 그리고 "ID"는 본질적으로 xs:string이다. 열거의 리스트가 비어있다면, 어떠한 데이터 값도 허용될 것이다.

```
<xs:element minOccurs="0" name="ITD09">
<xs:simpleType>
                    <xs:restriction base="ID"/>
                        <xs:enumeration value="00" />
                        <xs:enumeration value="10" />
                </xs:simpleType>
        </xs:element>
```

<85>

<86> 4. "AN" - 이것은 길이 제한을 갖는 영숫자(alphanumeric) 데이터 유형이다. 이 유형의 간단한 엘리먼트는 로컬 명칭 "AN"을 갖는 유형에 대한 제한인 간단한 유형을 가져야 한다. 그리고 "AN"은 본질적으로 xs:string이다.

```
<xs:element minOccurs="0" name="ITD09">
<xs:simpleType>
                    <xs:restriction base="AN"/>
                        <xs:minLength value="1" />
                        <xs:maxLength value="5" />
                </xs:simpleType>
        </xs:element>
```

<87>

<88> 5. "R" - 이것은 실수(real number)이다. 이 유형의 간단한 엘리먼트는 로컬 명칭 "R"을 갖는 간단한 유형을 가져야 한다. 그리고 "R"은 포맷 [sign]integral-digits[.[fractional-digits]]을 갖는데 이 경우 부호는 - 만 될 수 있고, +는 허용되지 않을 것이다.

<xs:element minOccurs="0" name="ITD09">

<xs:simpleType>

<xs:restriction base="R"/>

<xs:minLength value="1" />

<xs:maxLength value="5" />

</xs:simpleType>

</xs:element>

<89>

<90> 6. "Date" - 이것은 날짜 데이터 유형이다. 이 유형의 간단한 엘리먼트는 로컬 명칭 "date"를 갖는 간단한 유형을 가져야 한다. 이 포맷은 CCYYMMDD 이다.

<91> 7. "Time" - 이 유형의 간단한 엘리먼트는 로컬 명칭 "time"을 갖는 간단한 유형을 가져야 한다. 이 포맷은 HHMM[[SS][d[d]]] 이다.

<92> B. EDIFACT 데이터 유형:

<93> 유사하게, EDIFACT 포맷의 스키마들을 설명하는 데에 있어서, EDIFACT 스키마는 다음 중 하나와 같은 다수의 데이터 유형들을 규정한다.

<94> 1. "a" - 이것은 길이 제한을 갖는 알파벳 데이터 유형이다. 이 유형의 간단한 엘리먼트는 로컬 명칭 "a"를 갖는 유형에 대한 제한인 간단한 유형을 가져야 한다. 그리고 "a"는 본질적으로 xs:string이다. <xs:element minOccurs="0" name="ITD07">

<xs:simpleType>

<xs:restriction base="a"/>

<xs:minLength value="1" />

<xs:maxLength value="5" />

</xs:simpleType>

</xs:element>

<95>

<96> 2. "n" - 이것은 길이 제한을 갖는 수치 데이터 유형이다. 이 유형의 간단한 엘리먼트는, 예를 들면, 로컬 명칭 "n"을 갖는 유형에 대한 제한인 간단한 유형을 가져야 한다. 그리고 "n"은 본질적으로 xs:string이다.

```
<xs:element minOccurs="0" name="ITD08">
<xs:simpleType>
                    <xs:restriction base="n"/>
                        <xs:minLength value="4" />
                        <xs:maxLength value="8" />
                </xs:simpleType>
        </xs:element>
```

<97>

<98> 3. "ID" - 이것은 선택적 길이 제한들을 갖는 열거 데이터 유형이다. 이 유형의 간단한 엘리먼트는 로컬 명칭 "ID"를 갖는 유형에 대한 제한인 간단한 유형을 가져야 한다. 그리고 "ID"는 본질적으로 xs:string이다. 열거의 리스트가 비어 있다면, 어떠한 데이터 값도 허용될 것이다. 949 코드 페이지에 들어오는 EDIFACT 데이터의 경우, 스키마 내의 Enum 값들은 그들의 유니코드 등가물(Unicode equivalent)일 것이다.

<99>

```
<xs:element minOccurs="0" name="ITD09">
<xs:simpleType>
                    <xs:restriction base="ID"/>
                        <xs:enumeration value="00" />
                        <xs:enumeration value="10" />
                </xs:simpleType>
        </xs:element>
```

<100>

<101> 4. "AN" - 이것은 길이 제한을 갖는 영숫자 데이터 유형이다. 이 유형의 간단한 엘리먼트는 로컬 명칭 "AN"을 갖는 유형에 대한 제한인 간단한 유형을 가져야 한다. 그리고 "AN"은 본질적으로 xs:string이다.

```
<xs:element minOccurs="0" name="ITD09">
<xs:simpleType>
                    <xs:restriction base="AN"/>
                        <xs:minLength value="1" />
                        <xs:maxLength value="5" />
                </xs:simpleType>
        </xs:element>
```

<102>

<103> 이전에 설명된 바와 같이, 본 발명의 양태들을 구현하는 XML 스키마에 의해 주석이 달린 규칙들 중 하나는 레퍼런스 지정자(reference designators)이다. 레퍼런스 지정자들은 필드 레벨에서 존재하며 그들은 에러 디버깅 동안 사용될 수 있는 '메타데이터-사전(metadata-dictionary)'에서 숫자를 조회한다. 엔진은 AK502 레벨에서 파싱하는 동안 이 정보를 보고할 것이다. 예를 들어, 스키마 내에 존재한다면 지정자 값을 보고할 것이다. 다음의 주석이 필드 레벨에서 사용된다. AK502는 선택적 필드이며, 주석이 존재하지 않는다면, 값이 정해지지 않을 것이다. 다음의 예시적인 XML 문들(statements)은 레퍼런스 지정자의 일 예를 예시한다:

<104> `<xs:annotation>`

<xs:appinfo>

<b:fieldInfo edi_reference_designator="725"

xmlns:b="http://schemas.company.com/software/2003" />

</xs:appinfo>

<105> </xs:annotation>

<106> 유사하게, 규칙들 중 하나인 트리거 필드 정보가 본 발명의 일 실시예에 따라 주석이 달린다. 이 주석은 트리거 필드로서 기능하는 세그먼트 내의 필드를 식별하기 위하여 루프 또는 세그먼트 레벨에서 존재할 수 있다. 그리고 또한 그 값은 트리거 전이(trigger transition)를 야기시켜야 한다. 이 정보는 더욱 사용자 친화적인 Xml을 생성하기 위하여 HIPAA 스키마들에서 이용된다. 주석은 아래에 도시된다: 이 예에서, TS270A1_2100A_Loop 는 트리거 루프이다. 트리거 세그먼트는 XML에서 NM1_InformationSourceName_TS270A1_2100A 로서 기입될 수 있고 트리거 필드는 NM101_EntityIdentifierCode 이다. 트리거 값은 2B; 36; GP; P5; 또는 PR 중 어느 하나일 수 있을 것이다. 다음은 예시적인 XML 문 또는 코드를 예시한다:

- <xs:element name="TS270A1_2100A_Loop">

- <xs:annotation>

<xs:appinfo>

<b:recordInfo

trigger_field="NM1_InformationSourceName_TS270A1_2100A/NM101__EntityIdentifie

rCode" trigger_value="2B 36 GP P5 PR/>

</xs:appinfo>

<107> </xs:annotation>

<108> 도 4는 본 발명의 양태들이 저장될 수 있는 예시적인 컴퓨터 실행 가능 매체(502)를 도시하는 블럭도이다. 예를 들어, 컴퓨터 판독 가능 매체(402)는, 본 발명의 일 실시예에 따라 런타임(runtime)시 XML을 이용하여 EDI 문서를 모델링하는 동작들을 설명하는 흐름도인 도 5에 도시된 연산들을 수행하기 위한 하나 이상의 컴퓨터 실행 가능 컴포넌트들을 포함한다.

<109> 예를 들어, 엘리먼트 컴포넌트(404)는 단계(502)에서 EDI 문서 내에서 복수의 구조적 엘리먼트들을 식별한다. 단계(504)에서, 데이터 컴포넌트(406)는 EDI 문서에서 식별된 복수의 구조적 엘리먼트들로부터 복수의 대응 데이터 값들을 결정한다. 단계(506)에서, 주석 컴포넌트(408)는 EDI 문서에 포함된 복수의 규칙들에 대한 주석들을 생성한다. 복수의 규칙들은 복수의 대응 데이터 값들과 연관된 연산들을 규정한다. 변환 컴포넌트(410)는 결정된 데이터 값들 및 생성된 주석들을 단계(508)에서 EDI 문서에 대응하는 XML 스키마로 수정 또는 변환한다. 단계(510)에서, 페이로드 컴포넌트(512)는 런타임시 XML 스키마를 프로세스한다.

<110> 도 6은 컴퓨터(130) 형태의 범용 컴퓨팅 장치의 일례를 도시한다. 본 발명의 일 실시예에서, 컴퓨터(130)와 같은 컴퓨터는 여기에 도시되고 설명된 다른 도면들에서의 사용에 적합하다. 컴퓨터(130)는 하나 이상의 프로세서들 또는 처리 장치들(132) 및 시스템 메모리(134)를 갖는다. 예시된 실시예에서, 시스템 버스(136)는 시스템 메모리(134)를 포함하는 각종 시스템 컴포넌트들을 프로세서들(132)에 연결한다. 버스(136)는 메모리 버스 또는 메모리 컨트롤러, 주변 버스, 가속 그래픽 포트, 및 각종 버스 아키텍처들 중 임의의 것을 이용하는 프로세서 또는 로컬 버스를 포함하는 몇몇 유형의 버스 구조들 중 임의의 것의 하나 이상을 나타낸다. 예로서, 이러한 아키텍처는 ISA(industry standard architecture) 버스, MCA(micro channel architecture) 버스, EISA(Enhanced ISA) 버스, VESA(video electronics standard association) 로컬 버스, 그리고 메자닌 버스

(mezzanine bus)로도 알려진 PCI(peripheral component interconnect) 버스 등을 포함하지만 이에 제한되는 것은 아니다.

<111> 컴퓨터(130)는 통상적으로 적어도 어떤 형태의 컴퓨터 판독가능 매체를 포함한다. 컴퓨터(130)에 의해 액세스 가능한 매체는 그 어떤 것이든지 컴퓨터 판독가능 매체가 될 수 있고, 이러한 컴퓨터 판독가능 매체는 휘발성 및 비휘발성 매체, 이동식 및 비이동식 매체를 포함한다. 예로서, 컴퓨터 판독가능 매체는 컴퓨터 저장 매체 및 통신 매체를 포함하지만 이에 제한되는 것은 아니다. 컴퓨터 저장 매체는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보를 저장하는 임의의 방법 또는 기술로 구현되는 휘발성 및 비휘발성, 이동식 및 비이동식 매체를 포함한다. 예를 들어, 컴퓨터 저장 매체는 RAM, ROM, EEPROM, 플래시 메모리 또는 기타 메모리 기술, CD-ROM, DVD(digital versatile disk) 또는 기타 광 디스크 저장 장치, 자기 카세트, 자기 테이프, 자기 디스크 저장 장치 또는 기타 자기 저장 장치, 또는 컴퓨터(130)에 의해 액세스되고 원하는 정보를 저장할 수 있는 임의의 기타 매체를 포함하지만 이에 제한되는 것은 아니다. 통신 매체는 통상적으로 반송파(carrier wave) 또는 기타 전송 메커니즘(transport mechanism)과 같은 피변조 데이터 신호(modulated data signal)에 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터 등을 구현하고 모든 정보 전달 매체를 포함한다. 당업자는, 신호 내에 정보를 인코딩하도록 그 신호의 특성들 중 하나 이상을 설정 또는 변경시킨 신호를 의미하는 피변조 데이터 신호에 익숙하다. 예로서, 통신 매체는 유선 네트워크 또는 직접 배선 접속(direct-wired connection)과 같은 유선 매체, 그리고 음향, RF, 적외선, 기타 무선 매체와 같은 무선 매체를 포함한다. 상술된 매체들의 모든 조합이 또한 컴퓨터 판독가능 매체의 영역 안에 포함되는 것으로 한다.

<112> 시스템 메모리(134)는 이동식 및/또는 비이동식, 휘발성 및/또는 비휘발성 메모리 형태의 컴퓨터 저장 매체를 포함한다. 예시된 실시예에서, 시스템 메모리(134)는 ROM(138) 및 RAM(140)을 포함한다. 시동 중과 같은 때에, 컴퓨터(130) 내의 구성요소들 사이의 정보 전송을 돕는 기본 루틴을 포함하는 기본 입/출력 시스템(BIOS)(142)은 통상적으로 ROM(131)에 저장되어 있다. RAM(140)은 통상적으로 처리 장치(132)가 즉시 액세스할 수 있고 및/또는 현재 동작시키고 있는 데이터 및/또는 프로그램 모듈을 포함한다. 예로서, 도 6은 운영 체제(144), 애플리케이션 프로그램(146), 기타 프로그램 모듈(148) 및 프로그램 데이터(150)를 도시하고 있지만 이에 제한되는 것은 아니다.

<113> 컴퓨터(130)는 또한 기타 이동식/비이동식, 휘발성/비휘발성 컴퓨터 저장매체를 포함한다. 단지 예로서, 도 6은 비이동식·비휘발성 자기 매체에 기록을 하거나 그로부터 판독을 하는 하드 디스크 드라이브(154)를 도시한다. 도 6은 또한 이동식·비휘발성 자기 디스크(158)에 기록을 하거나 그로부터 판독을 하는 자기 디스크 드라이브(156), CD-ROM 또는 기타 광 매체 등의 이동식·비휘발성 광 디스크(156)에 기록을 하거나 그로부터 판독을 하는 광 디스크 드라이브(160)를 포함한다. 예시적인 운영 환경에서 사용될 수 있는 기타 이동식/비이동식, 휘발성/비휘발성 컴퓨터 저장 매체로는 자기 테이프 카세트, 플래시 메모리 카드, DVD, 디지털 비디오 테이프, 고상(solid state) RAM, 고상 ROM 등이 있지만 이에 제한되는 것은 아니다. 하드 디스크 드라이브(154), 및 자기 디스크 드라이브(156) 및 광 디스크 드라이브(160)는 통상적으로 인터페이스(166)와 같은 비휘발성 메모리 인터페이스를 통해 시스템 버스(136)에 접속된다.

<114> 위에서 설명되고 도 6에 도시된 드라이브들 또는 기타 대용량 저장 장치들 및 이들과 관련된 컴퓨터 저장 매체는, 컴퓨터(130)를 위해, 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 및 기타 데이터를 저장한다. 도 6에서, 예를 들어, 하드 디스크 드라이브(154)는 운영 체제(170), 애플리케이션 프로그램(172), 기타 프로그램 모듈(174), 및 프로그램 데이터(176)를 저장하는 것으로 도시되어 있다. 여기서 주의할 점은 이들 컴포넌트가 운영 체제(144), 애플리케이션 프로그램(146), 기타 프로그램 모듈(148), 및 프로그램 데이터(150)와 동일하거나 그와 다를 수 있다는 것이다. 운영 체제(170), 애플리케이션 프로그램(172), 기타 프로그램 모듈(174) 및 프로그램 데이터(176)에 다른 번호가 부여되어 있다는 것은 적어도 이들이 다른 사본(copy)이라는 것을 나타내기 위한 것이다.

<115> 사용자는 키보드(180), 포인팅 장치(182)(예를 들면, 마우스, 트랙볼, 펜, 또는 터치패드) 등의 입력 장치 또는 사용자 인터페이스 선택 장치를 통해 명령 및 정보를 컴퓨터(130)에 입력할 수 있다. 다른 입력 장치(도시 생략)로는 마이크, 조이스틱, 게임 패드, 위성 안테나, 스캐너 등을 포함할 수 있다. 이들 및 기타 입력 장치는 시스템 버스(136)에 결합된 사용자 입력 인터페이스(184)를 통해 처리 장치(132)에 접속되지만, 병렬 포트, 게임 포트 또는 USB(universal serial bus) 등의 다른 인터페이스 및 버스 구조에 의해 접속될 수도 있다. 모니터(188) 또는 다른 유형의 디스플레이 장치도 비디오 인터페이스(190) 등의 인터페이스를 통해 시스템 버스(136)에 접속될 수 있다. 모니터(188) 외에, 컴퓨터는 종종 프린터 및 스피커 등의 기타 주변 출력 장치를 포

함할 수 있고, 이들은 출력 주변장치 인터페이스(195)를 통해 접속될 수 있다.

<116> 컴퓨터(130)는 원격 컴퓨터(194)와 같은 하나 이상의 원격 컴퓨터로의 논리적 접속을 사용하여 네트워크화된 환경에서 동작할 수 있다. 원격 컴퓨터(194)는 퍼스널 컴퓨터, 서버, 라우터, 네트워크 PC, 피어 장치 또는 기타 통상의 네트워크 노드일 수 있고, 통상적으로 컴퓨터(130)와 관련하여 상술된 구성요소들의 대부분 또는 그 전부를 포함한다. 도 6에 도시된 논리적 접속으로는 LAN(196) 및 WAN(198)이 있지만, 기타 네트워크를 포함할 수도 있다. LAN(136) 및/또는 WAN(138)은 유선 네트워크, 무선 네트워크, 그들의 조합 등일 수 있다. 이러한 네트워킹 환경은 사무실, 전사적 컴퓨터 네트워크(enterprise-wide computer network), 인트라넷, 및 글로벌 컴퓨터 네트워크(예를 들면, 인터넷)에서 일반적인 것이다.

<117> LAN 네트워킹 환경에서 사용될 때, 컴퓨터(130)는 네트워크 인터페이스 또는 어댑터(186)를 통해 LAN(196)에 접속된다. WAN 네트워킹 환경에서 사용될 때, 컴퓨터(130)는 통상적으로 인터넷과 같은 WAN(198)을 통해 통신을 설정하기 위한 모뎀(178) 또는 기타 수단을 포함한다. 내장형 또는 외장형일 수 있는 모뎀(178)은 사용자 입력 인터페이스(184) 또는 기타 적절한 메커니즘을 통해 시스템 버스(136)에 접속된다. 네트워크화된 환경에서, 컴퓨터(130) 또는 그의 일부와 관련하여 기술된 프로그램 모듈은 원격 메모리 저장 장치(도시하지 않음)에 저장될 수 있다. 예로서, 도 6은 원격 애플리케이션 프로그램(192)이 메모리 장치에 있는 것으로 도시하고 있지만 이에 제한되는 것은 아니다. 도시된 네트워크 접속은 예시적인 것이며 이 컴퓨터들 사이에 통신 링크를 설정하는 기타 수단이 사용될 수 있다.

<118> 일반적으로, 컴퓨터(130)의 데이터 프로세서들은 컴퓨터의 각종 컴퓨터 판독 가능 저장 매체에 상이한 시간에 저장된 명령어들에 의해 프로그래밍된다. 프로그램들 및 운영 체제들은 통상적으로, 예를 들면, 플로피 디스크들 또는 CD-ROM들로 배포된다. 그것으로부터, 프로그램들 및 운영 체제들은 컴퓨터의 보조 메모리에 설치 또는 로드된다. 실행시, 이들은 적어도 부분적으로는 컴퓨터의 주메모리에 로드된다. 여기에 기술된 본 발명의 양태들은 컴퓨터 판독 가능 매체가 마이크로프로세서 또는 기타 데이터 프로세서와 연계하여 이하에 설명된 단계들을 실시하기 위한 명령어들 또는 프로그램들을 포함하는 경우 그러한 매체의 상기 및 기타 다양한 유형들을 포함한다. 또한, 본 발명의 양태들은 여기에 설명된 방법들 및 기술들에 따라 프로그래밍되는 경우 컴퓨터 자체를 포함한다.

<119> 예시의 목적으로, 운영 체제와 같은 프로그램들 및 기타 실행가능 프로그램 컴포넌트들은 개별 블럭들로 도시된다. 그러나, 그러한 프로그램들 및 컴포넌트들은 컴퓨터의 상이한 저장 컴포넌트들에 다양한 시간에(at various times) 존재하고, 컴퓨터의 데이터 프로세서(들)에 의해 실행되는 점이 인식된다.

<120> 컴퓨터(130)를 포함하는, 예시적인 컴퓨팅 시스템 환경과 관련하여 설명되었지만, 본 발명의 실시예들은 다수의 기타 범용 또는 전용 컴퓨팅 시스템 환경들 또는 구성들에서 사용할 수 있다. 컴퓨팅 시스템 환경은 본 발명의 임의의 양태의 용도 또는 기능성의 범위에 관해 어떤 제한을 암시하고자 하는 것이 아니다. 또한, 컴퓨팅 시스템 환경이 예시적인 운영 환경에 도시된 컴포넌트들 중 임의의 하나 또는 그 컴포넌트들의 임의의 조합과 관련하여 어떤 의존성 또는 요구사항을 갖는 것으로 해석되어서는 안된다. 본 발명의 양태들에서 사용하는 데 적합할 수 있는 잘 알려진 컴퓨팅 시스템, 환경, 및/또는 구성의 예로는 퍼스널 컴퓨터, 서버 컴퓨터, 핸드-헬드 또는 랩톱 장치, 멀티프로세서 시스템, 마이크로프로세서 기반 시스템, 셋톱 박스, 프로그램가능한 가전제품, 이동 전화, 네트워크 PC, 미니컴퓨터, 메인프레임 컴퓨터, 상기 시스템들이나 장치들 중 임의의 것을 포함하는 분산 컴퓨팅 환경, 기타 등등이 있지만 이에 제한되는 것은 아니다.

<121> 본 발명의 실시예들은 일반적으로 하나 이상의 컴퓨터들 또는 기타 장치들에 의해 실행되는 프로그램 모듈과 같은 컴퓨터 실행가능 명령어와 관련하여 기술될 수 있다. 일반적으로, 프로그램 모듈은 특정 태스크를 수행하거나 특정 추상 데이터 유형을 구현하는 루틴, 프로그램, 개체, 컴포넌트, 데이터 구조 등을 포함하지만 이에 제한되는 것은 아니다. 본 발명의 양태들은 또한 통신 네트워크를 통해 연결되어 있는 원격 처리 장치들에 의해 태스크가 수행되는 분산 컴퓨팅 환경에서 실시될 수도 있다. 분산 컴퓨팅 환경에서, 프로그램 모듈은 메모리 저장 장치를 비롯한 로컬 및 원격 컴퓨터 저장 매체 둘다에 위치할 수 있다.

<122> 소프트웨어 아키텍처와 관련하여 인터페이스는 소프트웨어 모듈, 컴포넌트, 코드부(code portion), 또는 컴퓨터 실행 가능 명령어들의 기타 시퀀스를 포함한다. 인터페이스는, 예를 들면, 자신을 대신하여 컴퓨팅 태스크들을 수행하기 위한 제2 모듈에 액세스하는 제1 모듈을 포함한다. 일례에서, 제1 및 제2 모듈은 운영 체제에 의해 제공된 것과 같은 API들(application programming interfaces), COM(component object model) 인터페이스들(예를 들면, 피어-투-피어 애플리케이션 통신), 및 XML 메타데이터 인터체인지 포맷 인터페이스들(예를 들면, 웹 서비스들 간의 통신용)을 포함한다.

<123> 인터페이스는 J2EE(Java 2 Platform Enterprise Edition), COM, 또는 DCOM(distributed COM) 예들에서와 같은 밀결합 동기 구현(tightly coupled, synchronous implementation)일 수 있다. 대안적으로 또는 부가적으로, 인터페이스는 웹 서비스에서와 같은 소결합 비동기 구현(loosely coupled, asynchronous implementation)일 수 있다(예를 들면, 간단한 개체 액세스 프로토콜 사용). 일반적으로, 인터페이스는 밀결합, 소결합, 동기, 및 비동기의 특성들 중 임의의 조합을 포함한다. 또한, 인터페이스는 표준 프로토콜, 고유 프로토콜(proprietary protocol), 또는 표준 및 고유 프로토콜들의 임의의 조합에 따를 수 있다.

<124> 여기에 기술된 인터페이스들은 모두 단일의 인터페이스의 일부이거나 별개의 인터페이스들로서 구현되거나 또는 그들의 임의의 조합일 수 있다. 인터페이스들은 국부적으로 또는 원격으로 실행하여 기능성을 제공할 수 있다. 또한, 인터페이스들은 여기에 예시되고 설명된 것보다 추가적이거나 더 적은 기능성을 포함할 수 있다.

<125> 동작시, 컴퓨터(130)는 본 발명의 양태들을 구현하기 위하여, 도 5와 같은 도면에 도시된 바와 같은 컴퓨터 실행 가능 명령어들을 실행한다.

<126> 여기에 예시되고 설명된 본 발명의 실시예들에서의 동작들의 실행 또는 수행의 순서는, 달리 지정이 없다면, 본질적인 것은 아니다. 즉, 동작들은, 달리 지정이 없다면, 어떠한 순서로도 수행될 수 있고, 본 발명의 실시예들은 여기에 개시된 것들보다 부가적이거나 더 적은 동작들을 포함할 수 있다. 예를 들면, 본 발명의 양태들의 범주 내에 다른 동작이 있기 전, 다른 동작과 동시, 또는 다른 동작이 있은 후에, 특정 동작을 실행 또는 수행하는 것이 고려된다.

<127> 본 발명의 실시예들은 컴퓨터 실행 가능 명령어들로 구현될 수 있다. 컴퓨터 실행 가능 명령어들은 하나 이상의 컴퓨터 실행 가능 컴포넌트들 또는 모듈들로 조직될 수 있다. 본 발명의 양태들은 그러한 컴포넌트들 또는 모듈들의 임의의 수 및 조직으로 구현될 수 있다. 예를 들어, 본 발명의 양태들은 여기에서 설명되고 도면에 도시된 특정의 컴퓨터 실행 가능 명령어들 또는 특정의 컴포넌트들 또는 모듈들에 한정되는 것은 아니다. 본 발명의 다른 실시예들은 여기에 예시되고 설명된 것보다 더 많거나 적은 기능성을 갖는 다른 컴퓨터 실행 가능 명령어들 또는 컴포넌트들을 포함할 수 있다.

<128> 본 발명의 양태들 또는 그 실시예들의 구성요소들을 소개할 때, 관사들("a", "an", "the", "said")은 그 구성요소들이 하나 이상 존재하는 것을 의미하려는 것이다. 용어들 "포함하는(comprising, including)" 및 "갖는(having)"은 포괄적인 것이며 상기에 열거된 구성요소들 이외의 부가적인 구성요소들이 존재할 수 있음을 의미하려는 것이다.

<129> 본 발명의 양태들의 범주 내에서 상기 구성들, 제품들, 및 방법들에 각종 변화들이 이루어질 수 있으므로, 상기 상세한 설명에 포함되고 첨부도면에 도시된 모든 내용은 제한적인 의미가 아닌 예시적인 의미로 해석되어야 할 것이다.

도면의 간단한 설명

<9> 도 1은 본 발명의 일 실시예에 따른, EDI 트랜잭션들을 수행하기 위한 시스템을 도시하는 블럭도.

<10> 도 2는 본 발명의 일 실시예에 따른, XML을 이용하는 구조적 EDI 스키마를 도시하는 블럭도.

<11> 도 3은 본 발명의 양태들을 구현하는 XML 스키마를 도시하는 도면.

<12> 도 4는 본 발명의 양태들이 저장될 수 있는 예시적인 컴퓨터 판독 가능 매체를 도시하는 블럭도.

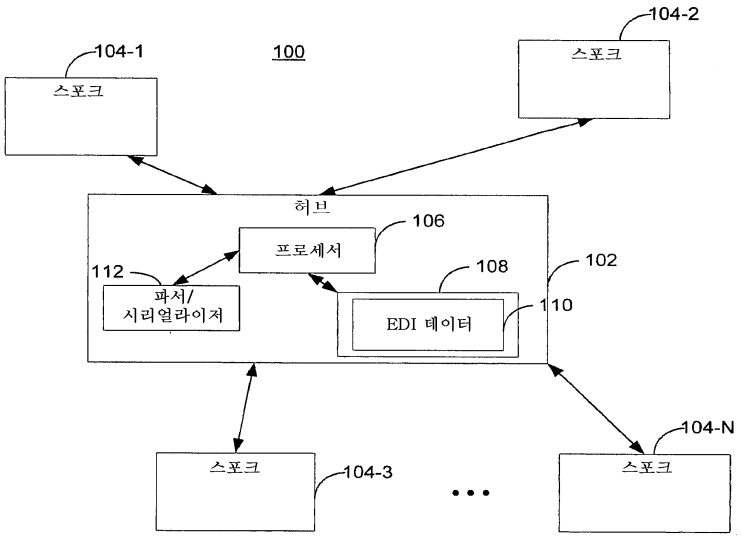
<13> 도 5는 런타임시 XML을 이용하여 EDI 문서를 모델링하는 동작들을 도시하는 예시적인 흐름도.

<14> 도 6은 본 발명이 실시될 수 있는 적절한 컴퓨팅 시스템 환경의 일 예를 도시하는 블럭도.

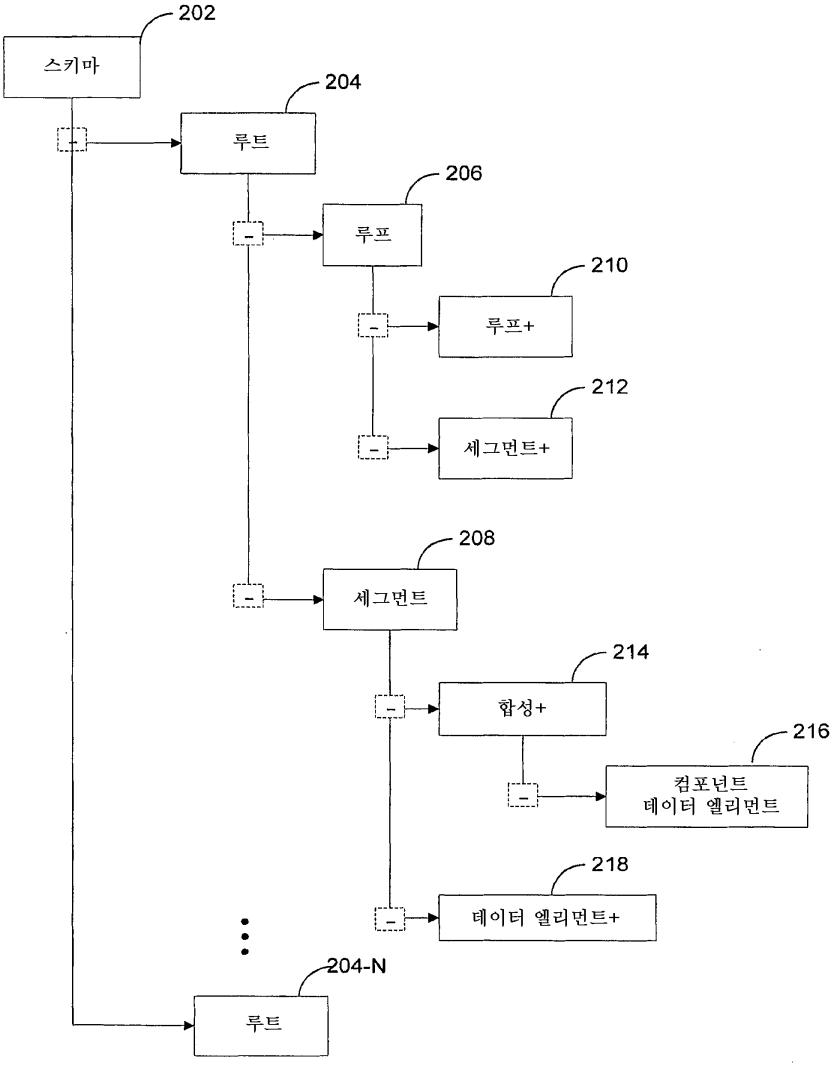
<15> 대응하는 참조부호들은 도면 전체에 걸쳐 대응하는 부분들을 가리킨다.

도면

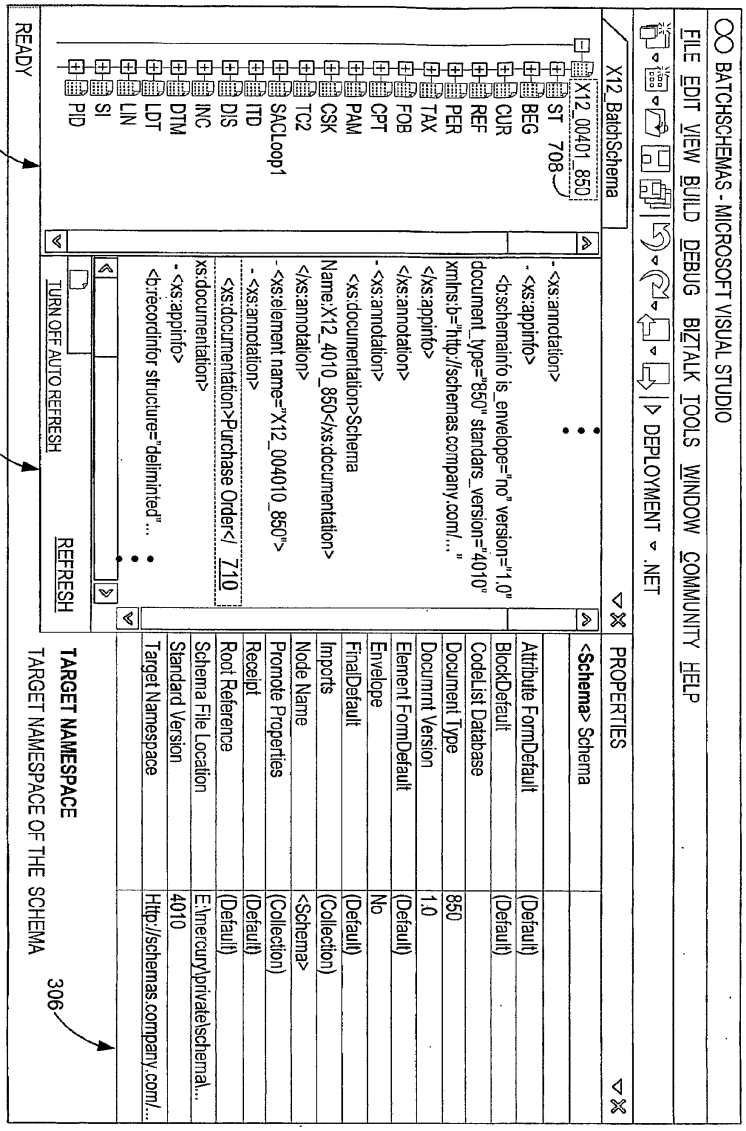
도면1


104-1
스포크
100
104-2
스포크
허브
106
프로세서
112
파서/
시리얼라이저
108
102
EDI 데이터
110
104-N
스포크
104-3
스포크

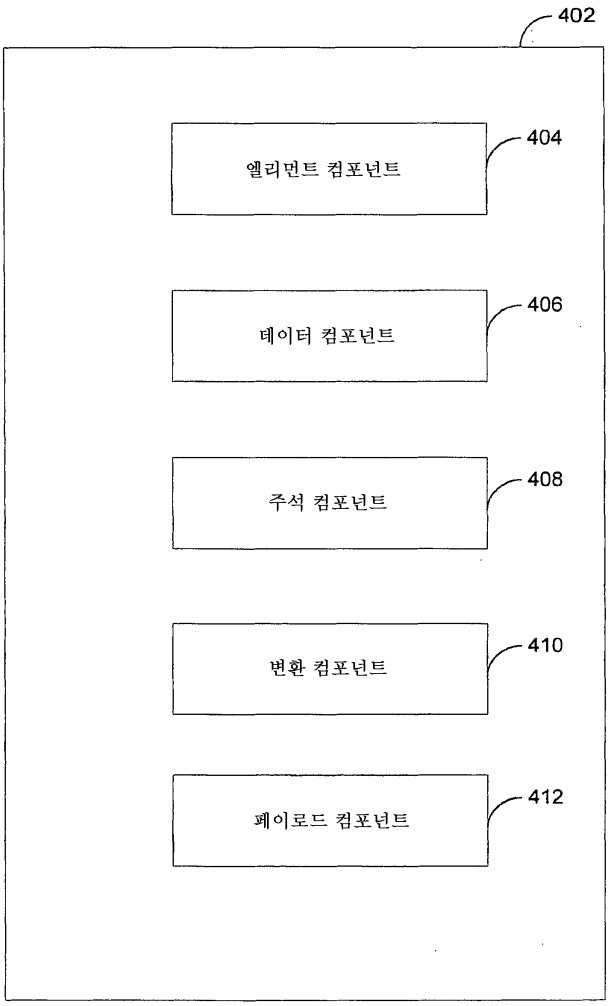
도면2


202
스키마
204
루트
206
루프
210
루프+
212
세그먼트+
208
세그먼트
214
합성+
216
컴포넌트
데이터 엘리먼트
218
데이터 엘리먼트+
204-N
루트

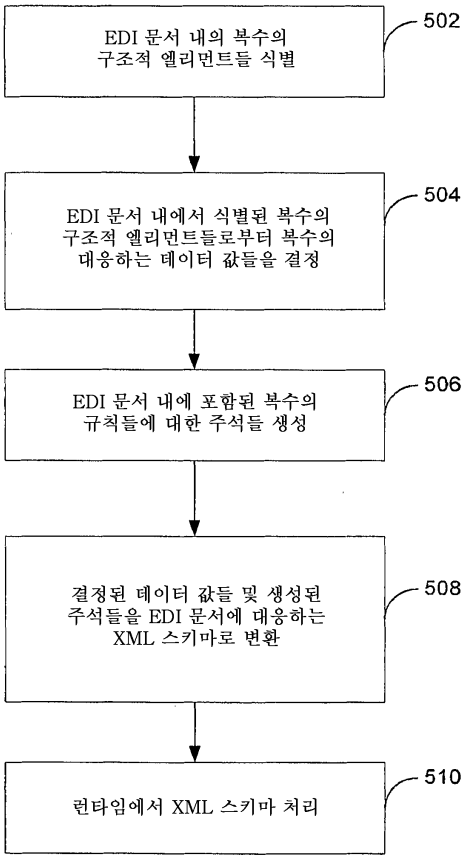
도면3


BATCHSCHEMAS - MICROSOFT VISUAL STUDIO
FILE EDIT VIEW BUILD DEBUG BIZTALK TOOLS WINDOW COMMUNITY HELP
DEPLOYMENT .NET
X12_BatchSchema
X12_00401_850
ST
BEG
CUR
REF
PER
TAX
FOB
CPT
PAM
CSK
TC2
SACLoop1
ITD
DIS
INC
DTM
LDT
LIN
SI
PID
708
PROPERTIES
<Schema> Schema
Attribute FormDefault (Default)
BlockDefault (Default)
CodeList Database
Document Type 850
Document Version 1.0
Element FormDefault (Default)
Envelope No
FinalDefault (Default)
Imports (Collection)
Node Name <Schema>
Promote Properties (Collection)
Receipt (Default)
Root Reference (Default)
Schema File Location E:\mercury\private\schema\...
Standard Version 4010
Target Namespace Http://schemas.company.com/...
- <xs:annotation>
- <xs:appinfo>
<b:schemaInfo is_envelope="no" version="1.0"
document_type="850" standars_version="4010"
xmlns:b="http://schemas.company.com/..."
</xs:appinfo>
</xs:annotation>
- <xs:annotation>
<xs:documentation>Schema
Name:X12_4010_850</xs:documentation>
</xs:annotation>
- <xs:element name="X12_004010_850">
- <xs:annotation>
<xs:documentation>Purchase Order</ Z10
xs:documentation>
- <xs:appinfo>
<b:recordInfor structure="delimited"...
REFRESH
TURN OFF AUTO REFRESH
READY
TARGET NAMESPACE
TARGET NAMESPACE OF THE SCHEMA
302
304
306

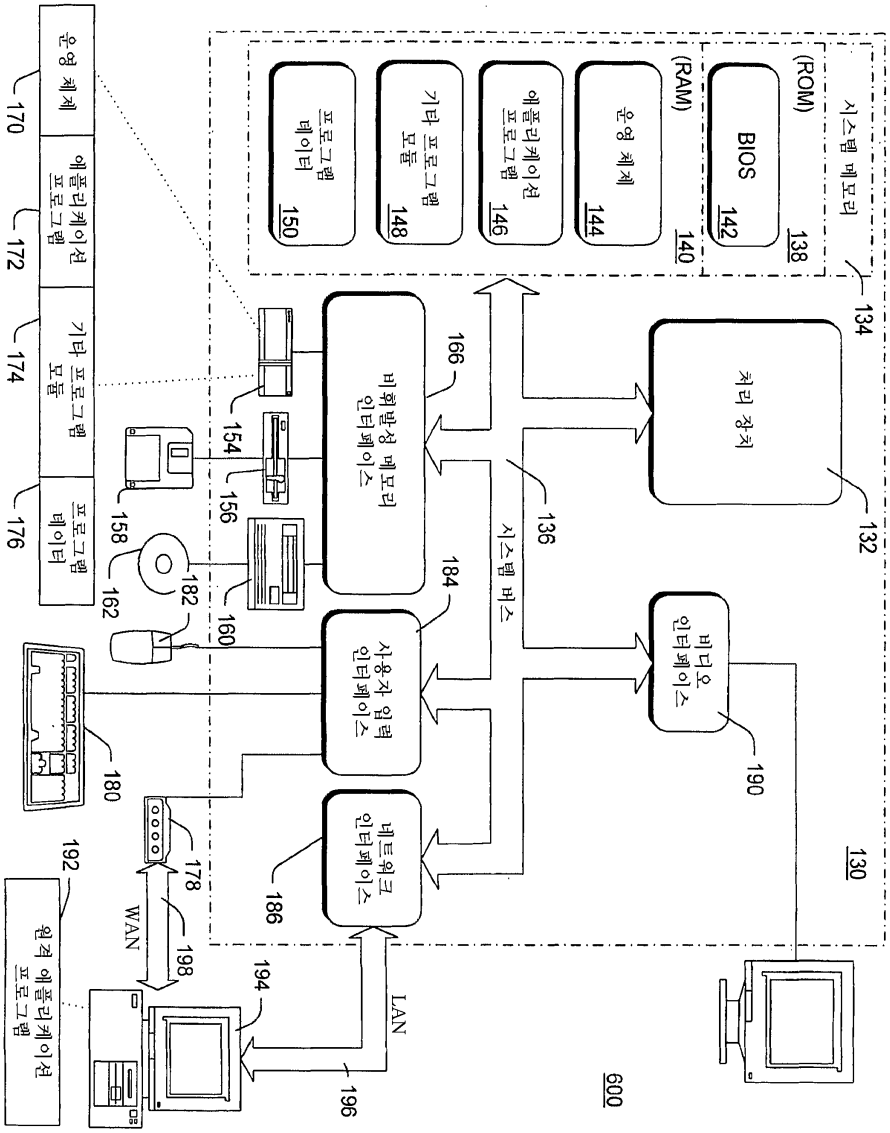
도면4


402
엘리먼트 컴포넌트
404
데이터 컴포넌트
406
주석 컴포넌트
408
변환 컴포넌트
410
페이로드 컴포넌트
412

도면5


EDI 문서 내의 복수의 구조적 엘리먼트들 식별
502
EDI 문서 내에서 식별된 복수의 구조적 엘리먼트들로부터 복수의 대응하는 데이터 값들을 결정
504
EDI 문서 내에 포함된 복수의 규칙들에 대한 주석들 생성
506
결정된 데이터 값들 및 생성된 주석들을 EDI 문서에 대응하는 XML 스키마로 변환
508
런타임에서 XML 스키마 처리
510

도면6


시스템 메모리
(ROM) 138
BIOS 142
(RAM) 140
운영 체제 144
애플리케이션 프로그램 146
기타 프로그램 모듈 148
프로그램 데이터 150
134
처리 장치
132
비휘발성 메모리 인터페이스
166
시스템 버스
136
비디오 인터페이스
190
사용자 입력 인터페이스
184
네트워크 인터페이스
186
130
600
154
156
158
160
162
182
180
178
WAN
198
192
원격 애플리케이션 프로그램
194
LAN
196
운영 체제
170
애플리케이션 프로그램
172
기타 프로그램 모듈
174
프로그램 데이터
176