(19)대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。Int. Cl.[7]
G06F 9/44
G06F 17/00

(11) 공개번호 10-2005-0056123
(43) 공개일자 2005년06월14일

(21) 출원번호 10-2004-0091708
(22) 출원일자 2004년11월11일

(30) 우선권주장 10/731,597 2003년12월09일 미국(US)

(71) 출원인 마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이

(72) 발명자 존스브라이언엠.
미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 내
선더랜드마크
미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 내
사위키마신
미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 내
리틀로버트에이.
미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 내

(74) 대리인 주성민
백만기
이중희

심사청구 : 없음

(54) 소프트웨어 애플리케이션에서의 네임스페이스 또는 스키마라이브러리 지원을 위한 프로그램 가능한 객체 모델

요약

프로그램 가능한 객체 모델은, 사용자가 XML 스키마 파일 및 관련 XML 기반 리소스를 포함하는 네임스페이스 또는 스키마 라이브러리를 프로그램적으로 액세스 및 사용하도록 하여, 스키마 파일 및 XML 기반 리소스를 하나 이상의 문서와 조합하며, 스키마 파일 및 XML 기반 리소스와 관련된 기능성을 주문 제작한다. 또한, 프로그램 가능한 객체 모델은, 사용자/프로그래머가 이전에 조합된 문서들을 이용하여 조합으로부터 스키마 파일 및 다른 XML 기반 리소스를 제거하도록 한다.

대표도

도 4

색인어

프로그램 가능한 객체 모델, XML, 스키마 파일, 네임스페이스, 스키마 라이브러리

명세서

도면의 간단한 설명

도 1은 본 발명에 대한 예시적인 오퍼레이팅 환경을 제공하는 컴퓨팅 시스템, 결합된 주변 디바이스 및 네트워크 디바이스의 간략한 블록도이다.

도 2는 객체 지향 프로그래밍 모델에 따른 소프트웨어 객체들간의 상호작용을 설명하는 간략한 블록도이다.

도 3은 문서, 첨부 스키마 파일 및 스키마 확인 기능성 모델간의 상호작용을 설명하는 블록도이다.

도 4는 문서, 네임스페이스 또는 스키마 라이브러리 및 제3자 소프트웨어 애플리케이션간의 상호작용을 설명하는 블록도이다.

<도면의 주요 부분에 대한 부호의 설명>

305 : 애플리케이션

400 : 네임스페이스(스키마) 라이브러리

450 : 제3자 프로그램

470 : 애플리케이션 프로그래밍 인터페이스

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

관련 출원

본 출원인의 사건 번호 60001.0263US01/MS303917.1 명칭 "Programmable Object Model for Extensible Markup Language Schema Validation" 의 미국 특허출원 및 본 출원인의 사건 번호 60001.0264US01/MS303918.1 명칭 "Programmable Object Model for Extensible Markup Language Markup in an Application" 의 미국 특허출원은 본 명세서에 참조로서 포함된다.

저작권 공고

본 발명은 프로그램 가능한 객체 모델에 관한 것이다. 특히, 본 발명은 소프트웨어 애플리케이션에서의 네임스페이스(Namespace) 또는 스키마(schema) 라이브러리 지원을 위한 프로그램 가능한 객체 모델에 관한 것이다.

컴퓨터 소프트웨어 애플리케이션들을 이용하여 사용자들은 일, 교육 및 레저를 돕는 다양한 문서들을 생성한다. 예를 들어, 인기 있는 워드 프로세싱 애플리케이션들을 이용하여 사용자들은 편지, 기사, 책, 메모 등을 생성한다. 스프레드시트 애플리케이션들을 이용하여 사용자들은 다양한 영숫자(alphanumeric) 데이터를 저장, 조작, 프린트 및 표시한다. 그러한 애플리케이션들은, 편집, 포맷팅, 프린팅, 계산 및 온 라인과 오프 라인 편집을 포함하는 잘 알려진 많은 능력을 구비한다.

대부분의 컴퓨터 소프트웨어 애플리케이션들은 모든 잠재 사용자에 의해 요구되는 기능성(functionality)을 제공하기 위한 모든 필수적인 프로그래밍을 포함하지는 않는다. 많은 프로그래머들은, 흔히 자신의 프로그램들에 존재하는 애플리케이션의 능력들을 이용하거나, 애플리케이션의 기능성을 주문 제작하며, 그 기능성이 특정 사용자들 또는 동작에 좀더 적합하기를 원한다. 예를 들어, 금융 산업에 종사하는 프로그래머는, 금융 보고서들을 편집하는 금융 애널리스트들로 구성되는 사용자층을 위한 워드 프로세서를 주문 제작하기를 희망할 수도 있다. 최근에는, 확장성 생성 언어(Extensible Markup Language)가, 많은 사용자들을 위한 호환가능한 데이터 포맷으로서 널리 이용되어왔다. XML 기능성의 사용자들은, 흔히 하나 이상의 XML 스키마 파일들 또는 XML 기반의 솔루션들을 사용자에 의해 편집되거나 생성되는 문서에 첨부하거나 조합시킨다. 그러나, 사용자/프로그래머들은 XML 스키마 파일 및 다른 XML 기반의 솔루션 기능성을 소정의 문서에 적용하는데 있어서 제한을 받는데, 이는, 그 사용자/프로그래머가, XML 스키마 파일들 또는 다른 XML 기반의 솔루션들을 포함하는 네임스페이스 또는 스키마 라이브러리에 직접적으로 용이하게 액세스하지 않기 때문이다.

따라서, 본 기술 분야에서는, 사용자/프로그래머가 XML 리소스들의 네임스페이스 또는 스키마 라이브러리에 액세스하도록 하여, 그 리소스들을 주문 제작하거나 조작하여, 소프트웨어 애플리케이션 문서와 함께 XML 기능성의 사용자/프로그래머의 이용을 강화시키기 위한 프로그램 가능한 객체 모델에 대한 필요성이 존재한다. 본 발명은 이러한 사항들을 고려한 것이다.

발명이 이루고자 하는 기술적 과제

본 발명은, 사용자가 XML 스키마 파일 및 관련 XML 기반 리소스를 포함하는 네임스페이스 또는 스키마 라이브러리를 프로그램적으로 액세스 및 사용하도록 하여, 그러한 XML 기반 리소스를 하나 이상의 문서와 조합하며, 그러한 XML 기반 리소스와 관련된 기능성들을 주문 제작하기 위한 방법 및 시스템을 제공한다. 사용자 또는 프로그래머가 네임스페이스 또는 스키마 라이브러리에 액세스하면, 사용자는 XML 스키마 파일들을 관련 문서내 XML 데이터와 프로그램적으로 조합할 수도 있으며, 역으로, 사용자는 문서에 포함된 XML 데이터와의 XML 스키마 파일들의 조합을 검출하여 제거할 수도 있다. 또한, 사용자는 변환 파일들을 문서에 포함된 XML 데이터와 프로그램적으로 조합할 수도 있으며, 문서에 포함된 XML 데이터와 관련된 현존하는 변환 파일들을 검출하여 제거할 수도 있다. 또한, 사용자는 다른 파일들 및 XML 기반의 다른 문서 솔루션들과 관련된 실행가능한 소프트웨어를 문서에 포함된 XML 데이터와 조합할 수도 있다. 부가적으로, 사용자는 XML 기반 솔루션들과 다른 타입의 실행가능한 소프트웨어와의 조합을 문서에 포함된 관련 XML 데이터로부터 검출하여 제거할 수도 있다.

본 발명의 이러한 특징, 장점, 및 양상은 이하에서 개시된 실시예들의 상세한 설명과, 첨부된 도면들과 특허청구범위를 참조함으로써 좀더 명확하게 이해될 수도 있다.

발명의 구성 및 작용

본 발명의 실시예들은, 사용자가 XML 스키마 파일, XML 기반의 솔루션 및 리소스의 네임스페이스/스키마 라이브러리를 프로그램적으로 호출하도록 하여, 그러한 파일, 솔루션 및 리소드를 하나 이상의 문서에 조합하는 것을 제어하기 위한 방법 및 시스템에 관한 것이다. 이러한 실시예들이 결합될 수도 있고, 다른 실시예들이 사용될 수도 있으며, 본 발명의 사상이나 범위를 벗어나지 않는 구조적인 변경이 이루어질 수도 있다. 그러므로, 후속하는 상세한 설명은, 첨부된 특허청구범위 및 그 동등물에 의해 정의되는 본 발명의 의미와 범위를 한정하는 것은 아니다.

도면들을 참조하면, 일부 도면에서 비슷한 숫자들이 비슷한 엘리먼트들을 나타내며, 본 발명과 예시적인 오퍼레이팅 환경의 양상이 설명될 것이다. 도 1 및 다음의 논의는, 본 발명이 구현될 수도 있는 컴퓨팅 환경에 적합한 간략하면서도 일반적인 설명을 제공하려는 의도이다. 본 발명은 개인용 컴퓨터의 오퍼레이팅 시스템에서 작동하는 애플리케이션 프로그램과 함께 실행되는 프로그램 모듈들의 일반적인 상황하에서 설명될 것이지만, 본 기술 분야의 숙련자들은, 본 발명이 다른 프로그램 모듈들의 결합에서 구현될 수도 있음을 인식할 것이다.

일반적으로, 프로그램 모듈들은, 루틴, 프로그램, 컴포넌트, 데이터 구조, 및 특별한 태스크를 수행하거나, 특별한 요약 데이터 타입을 구현하는 다른 타입의 구조를 포함한다. 더욱이, 본 기술 분야의 숙련자들은, 본 발명이 소형 디바이스, 멀티프로세서 시스템, 마이크로프로세서 기반 또는 프로그램 가능한 소비자 일렉트로닉스, 미니컴퓨터, 메인프레임 컴퓨터 등을 포함하는 다른 컴퓨터 시스템 구성을 이용하여 실행될 수도 있음을 인지할 것이다. 본 발명은, 태스크들이, 통신 네트워크를 통해 링크된 원격 프로세싱 디바이스들에 의해 수행되는 분산 컴퓨팅 환경에서 또한 실행될 수도 있다. 분산 컴퓨팅 환경에서, 프로그램 모듈들은 로컬 및 원격 메모리 저장 디바이스들에 위치될 수도 있다.

도 1을 참조하여, 본 발명의 다양한 실시예들을 실행하기 위한 개인용 컴퓨터(2)에 대한 컴퓨터 아키텍처를 설명한다. 도 1에 도시한 컴퓨터 아키텍처는, CPU(4)와, RAM(8) 및 ROM(10)을 포함하는 시스템 메모리(6)와, 메모리를 CPU(4)에 연결하는 시스템 버스(12)를 포함하는 종래의 개인용 컴퓨터를 나타낸다. 예를 들어, 스타트 업 동안에 컴퓨터내 엘리먼트들 간의 정보 전송을 돕는 기본 루틴들을 포함하는 기본 입/출력 시스템은 ROM(10)에 저장된다. 개인용 컴퓨터(2)는, 오퍼레이팅 시스템(16)과, 애플리케이션 프로그램(305)과 같은 애플리케이션 프로그램들과, 데이터를 저장하기 위한 대용량 저장 디바이스(14)를 또한 포함한다.

대용량 저장 디바이스(14)는 버스(12)에 연결된 대용량 저장 제어기(도시하지 않음)를 통해 CPU(4)에 연결된다. 대용량 디바이스(14) 및 그와 조합된 컴퓨터 판독가능한 미디어는 비휘발성 저장을 개인용 컴퓨터(2)에 제공한다. 본 명세서에 포함된 컴퓨터 판독가능한 미디어에 대한 설명은, 하드 디스크 또는 CD-ROM 드라이브와 같은 대용량 저장 디바이스를 의미하지만, 본 기술 분야의 숙련자들은, 컴퓨터 판독가능한 미디어는 개인용 컴퓨터(2)에 의해 액세스될 수 있는 임의의 이용가능한 미디어일 수 있음을 인지해야 한다.

예로서, 컴퓨터 판독가능한 미디어는 컴퓨터 저장 미디어 및 통신 미디어를 포함할 수도 있지만, 이에 한정되지는 않는다. 컴퓨터 저장 미디어는, 컴퓨터 판독가능한 명령, 데이터 구조, 프로그램 모듈 또는 다른 데이터와 같은 정보를 저장하기 위한 임의의 방법 또는 기술에서 구현된 휘발성 및 비휘발성, 분리성 및 비분리성 미디어를 포함한다. 컴퓨터 저장 미디어는, RAM, ROM, EPROM, EEPROM, 플래시 메모리 또는 다른 고체 상태 메모리 기술, CD-ROM, DVD, 또는 다른 광저장, 자기 카세트, 자기 테이프, 자기 디스크 저장 또는 다른 자기 저장 디바이스, 또는 컴퓨터에 의해 액세스될 수 있는 소정의 정보를 저장하기 위해 사용될 수 있는 임의의 다른 매체를 포함하지만, 이에 한정되지는 않는다.

본 발명의 다양한 실시예들에 따르면, 개인용 컴퓨터(2)는, 인터넷과 같은 TCP/IP 네트워크(18)를 통해 논리적인 연결들을 원격 컴퓨터들에 이용하는 네트워크 환경에서 작동할 수도 있다. 개인용 컴퓨터(2)는 버스(12)에 연결된 네트워크 인터페이스 유닛(20)을 통해 TCP/IP 네트워크(18)에 연결될 수도 있다. 네트워크 인터페이스 유닛(20)은 다른 타입의 네트워크들과 원격 컴퓨터 시스템들에 연결하기 위해 사용될 수도 있음을 인지해야 한다. 개인용 컴퓨터(2)는, 키보드 또는 마우스(도시하지 않음)를 포함하는 수많은 디바이스들로부터의 입력을 수신하고 처리하기 위한 입/출력 제어기(22)를 또한 포함할 수도 있다. 유사하게, 입/출력 제어기(22)는, 표시 스크린, 프린터, 또는 다른 타입의 출력 디바이스에 출력을 제공할 수도 있다.

간략하게 상술한 바와 같이, 수많은 프로그램 모듈들과 데이터 파일들은, Washington, Redmond에 위치한 MICROSOFT CORPORATION의 WINDOWS XP 오퍼레이팅 시스템과 같은 네트워크형 개인용 컴퓨터의 운영을 제어하는데 적합한 오퍼레이팅 시스템(16)을 포함하는 개인용 컴퓨터(2)의 대용량 저장 디바이스(14)와 RAM(8)에 저장될 수도 있다. 대용량 저장 디바이스(14)와 RAM(8)은 하나 이상의 애플리케이션 프로그램들을 또한 저장할 수도 있다. 특히, 대용량 저장 디바이스(14)와 RAM(8)은 전자 문서(310)를 생성하고 편집하기 위한 애플리케이션 프로그램(305)을 저장할 수도 있다. 예를 들어, 애플리케이션 프로그램(305)은, 워드 프로세싱 애플리케이션 프로그램, 스프레드시트 애플리케이션,

콘택트 애플리케이션 등을 포함할 수도 있다. 다른 타입의 전자 문서들을 생성하고 편집하기 위한 애플리케이션 프로그램들이 본 발명의 다양한 실시예들과 함께 또한 사용될 수도 있다. 도시한 스키마 파일(330)과 네임스페이스/스키마 라이브러리(400)는 이하에서 설명한다.

본 발명의 예시적인 실시예들은 객체 지향 프로그래밍 환경에서의 다른 소프트웨어 객체들간의 통신에 의해 구현된다. 다음에서 본 발명의 실시예들을 설명하기 위하여, 객체 지향 프로그래밍 환경의 컴포넌트들에 대하여 간략하게 설명하기로 한다. 도 2는 객체 지향 프로그래밍 모델에 따른 소프트웨어 객체들간의 상호작용을 설명하는 간략한 블록도이다. 객체 지향 프로그래밍 환경에 따르면, 제1 객체(210)는, 소프트웨어 코드, 실행가능한 방법, 속성, 및 파라미터를 포함할 수도 있다. 유사하게, 제2 객체(220)는, 소프트웨어 코드, 실행가능한 방법, 속성, 및 파라미터를 또한 포함할 수도 있다.

제1 객체(210)는 제2 객체(220)와 통신하여, 메시지 호출(230)을 통해 제2 객체(220)를 호출함으로써 제2 객체(220)로부터 정보 또는 기능성을 얻을 수도 있다. 객체 지향 프로그래밍 환경 기술 분야의 숙련자들에게 알려진 바와 같이, 제1 객체(210)는, 두 개의 다른 소프트웨어 객체들(210,220)이 서로에게서 정보와 기능성을 얻기 위하여 서로 통신하도록 하는 애플리케이션 프로그래밍 인터페이스(API)들을 통해 제2 객체(220)와 통신할 수도 있다. 예를 들어, 제1 객체(210)가 제2 객체(220)에 포함된 방법에 의해 제공되는 기능성을 요구하면, 제1 객체(210)는 메시지 호출(230)을 제2 객체(220)로 전달할 수도 있는데, 그 메시지 호출(230)에서 제1 객체는 그 요구된 방법을 식별하고, 임의의 요구된 파라미터들을 그 식별된 방법을 운영하기 위한 제1 객체에 의해 요구된 제2 객체로 전달한다. 제2 객체(220)가 제1 객체로부터 호출을 수신하면, 제2 객체는, 호출된 방법을 제공된 파라미터들에 기초하여 실행하고, 그 실행된 방법으로부터 얻은 값을 포함하는 리턴 메시지(250)를 제1 객체(210)로 다시 송신한다.

예를 들어, 본 발명의 실시예들에 의하여, 그리고 이하에서 설명하는 바와 같이, 제1 객체(210)는, 확장성 생성 언어 스키마 확인 객체와 같은 제2 객체에 메시지를 전달하는 제3자 주문 생산된 애플리케이션일 수도 있는데, 그로 인하여 제1 객체는 문서내 특정 XML 엘리먼트의 확인을 요구하는 방법을 식별하고, 그 특정 XML 엘리먼트는 식별된 방법과 함께 제1 객체에 의해 전달된 파라미터이다. 이 예에 따르면, 제1 객체로부터의 호출을 수신하면, 스키마 확인 객체는 식별된 방법을 특정 XML 엘리먼트에 대해 실행하고, 확인된 XML 엘리먼트와 관련된 결과 또는 값의 형태로 메시지를 제1 객체에 리턴한다. 간략하게 상술한 바와 같이, 객체 지향 프로그래밍 환경들의 운영은 본 기술 분야의 숙련자들에게 잘 알려져 있다.

후술되는 바와 같이, 본 발명의 실시예들은 확장성 생성 언어(XML)의 컴포넌트들의 사용, 주문 제작, 및 응용에서의 소프트웨어 객체들의 상호작용을 통해 구현된다. 도 3은 문서, 첨부 스키마 파일, 및 스키마 확인 기능성 모듈간의 상호작용을 설명하는 블록도이다. 본 기술 분야의 숙련자들에게 잘 알려진 바와 같이, 확장성 생성 언어(XML)는, 사용자가 문서내 텍스트 또는 데이터에 적용되는 태그 이름들을 생성하도록 함으로써, 문서내 텍스트와 데이터를 설명하는 방법을 제공하는데, 조합된 태그들에 적용되는 텍스트 또는 데이터를 교대로 정의한다. 예를 들어 도 3을 참조하면, 애플리케이션(305)과 함께 생성된 문서(310)는, XML 태그들(315,320,325)로 표시되는 텍스트를 포함한다. 예를 들어, 텍스트 "Greetings"는 XML 태그 <title>를 이용하여 주석을 단다. 텍스트 "My name is Sarah"는 <body> 태그를 이용하여 주석을 단다. XML에 따르면, <title> 및 <body> 태그들의 크리에이터(creator)는, 그러한 태그들이 적용될 태그들을 설명하기 위한 크리에이터 자신의 태그들을 생성할 수 있다. 그 다음, 임의의 다운스트림 소모 애플리케이션 또는 컴퓨팅 머신이, 텍스트에 적용된 태그들의 정의에 따라서 제공된 명령들인 동안, 그 애플리케이션 또는 컴퓨팅 머신은 그 태그들에 따른 데이터를 이용할 수도 있다. 예를 들어, 다운스트림 애플리케이션이, 그 애플리케이션에 의해 처리된 조항 또는 공개물의 타이틀로서 정의된 텍스트를 추출하도록 프로그램된 경우, 그 애플리케이션은 문서(310)를 파스(parse)하여, 도 3에 도시한 바와 같이, 텍스트 "Greetings"를 추출할 수도 있는데, 이는, 그 텍스트가 태그 <title>을 이용하여 주석을 달기 때문이다. 도 3에 도시한 문서(310)에 대한 특정 XML 태그를 명명하는 크리에이터는 제3자들에 의해 사용될 수도 있는 문서(310)에 포함된 텍스트 또는 데이터에 대한 유용한 설명을 제공하는데, 그 제3자들에게는 텍스트 또는 데이터에 적용된 태그들과 관련된 정의들을 제공한다.

본 발명의 실시예들에 따르면, 문서(310)에 입력된 텍스트와 XML 마크업은, 다양한 다른 파일 포맷들에 따라, 그리고 문서(310)와 함께 생성되는 애플리케이션(305)의 고유 프로그래밍 언어에 따라 저장될 수도 있다. 예를 들어, 텍스트와 XML 마크업은, 워드 프로세싱 애플리케이션, 스프레드시트 애플리케이션 등에 따라 저장될 수도 있다. 대안적으로, 문서(310)에 입력된 텍스트와 XML 마크업은 XML 포맷으로서 저장될 수도 있으며, 그로 인하여 텍스트 또는 데이터, 임의의 적용된 XML 마크업, 및 폰트, 스타일, 문맥 구조 등과 같은 임의의 포맷팅이 XML 표현으로서 저장될 수도 있다. 따라서, XML로서 저장된 데이터를 이해할 수 있는 다운스트림 또는 제3자 애플리케이션들은, XML 표현으로서 저장된 텍스트 또는 데이터를 개방하여 소모할 수도 있다. 문서(310)의 텍스트, XML 마크업, 관련 포맷팅 및 다른 속성들을 XML로서 저장하는 것에 대한 상세한 논의는, "Word Processing Document Stored in a Single XML File that may be Manipulated by Applications that Understanding XML" 제목으로 2002년 6월 28일에 출원된 미국 출원번호 제10/187,060호를 참조하는데, 그 내용은 본 명세서에서 완전하게 설명된 참조로서 포함된다.

텍스트 또는 데이터에 적용된 XML 마크업 엘리먼트들(태그들)에 대한 정의 프레임워크를 제공하기 위하여, 도 3에 도시한 바와 같이, 표시되고 저장된 데이터의 사용자들 및 소비자들이, 문서의 크리에이터에 의해 설계된 그 XML 태그 정의들을 이해하도록 하기 위한 필수적인 정보를 포함하는 XML 스키마 파일들이 생성된다. 본 기술 분야에서 XSD 파일로서 또한 간주되는 각각의 스키마 파일은 바람직하게는, 소정의 스키마 파일에 따라 문서에 적용될 수도 있는 모든 XML 엘리먼트들(태그들)의 리스팅(listing)을 포함한다. 예를 들어, 도 3에 도시한 스키마 파일(330)은 문서(310)에 적용될 수도 있는 어떤 XML 엘리먼트들의 정의를 포함하는 스키마 파일일 수도 있으며, XML 엘리먼트들의 속성, 또는 그 스키마 파일에 따른 XML 엘리먼트들을 이용하여 주석을 달 수도 있는 텍스트 또는 데이터와 관련된 제한 및/또는 규칙을 포함한다. 예를 들어, 도 3에 도시한 스키마 파일(330)을 참조하면, 그 스키마 파일은 네임스페이스 "intro"에 의해 식별되고, <introCard>의 루트 엘리먼트를 포함한다.

스키마 파일(330)에 따르면, <introCard> 엘리먼트는 스키마 파일을 위한 루트 엘리먼트의 역할을 하고, 또한 두 개의 자식 엘리먼트 <title> 및 <body>에 대한 부모 엘리먼트의 역할을 한다. 본 기술 분야의 숙련자들에게 잘 알려진 바와 같이, 수많은 부모 엘리먼트들이 단일 루트 엘리먼트하에서 정의될 수도 있으며, 수많은 자식 엘리먼트들이 각각의 부모 엘리먼트하에서 정의될 수도 있다. 그러나, 전통적으로 소정의 스키마 파일(330)은 하나의 루트 엘리먼트만을 포함한다. 도 3을 참조하면, 스키마 파일(330)은 <title> 및 <body> 엘리먼트 각각에 대한 속성들(340,345)을 또한 포함한다. 속성들(340,345)은, 각각의 엘리먼트들을 문서(310)내 텍스트 또는 데이터에 적용하는 것과 관련된 정의 또는 규칙을 제공할 수

도 있다. 예를 들어, 속성(340)은, <title> 엘리먼트로 주석을 단 텍스트는 길이가 25 문자 보다 짧거나 동일해야 됨을 정의한다. 따라서, 길이 25 문자를 초과하는 텍스트가 <title> 엘리먼트 또는 태그를 이용하여 주석을 달면, 그 텍스트의 주석은, 스키마 파일(330)에 포함된 정의들에 따라 타당하지 않다.

속성들로서의 그러한 정의들 또는 규칙들을 XML 엘리먼트들에 적용함으로써, 스키마의 크리에이터는, 소정의 스키마 파일과 관련된 문서에 포함된 데이터의 구조를 설명할 수도 있다. 예를 들어, 지원 문서에 적용된 XML 마크업을 정의하기 위한 스키마 파일(330)의 크리에이터가, 그 지원 문서의 경력 섹션에 네 개 이하의 현재 또는 과거의 직업을 포함하기를 희망한다면, 스키마 파일(330)의 크리에이터는, 예를 들어 네 개 이하의 현재 또는 과거의 직업이 <experience> 태그들 사이에 순서대로 입력되어, 그 경력 텍스트가 스키마 파일(330)에 따라 타당할 수도 있음을 허용하는 <experience> 엘리먼트의 속성을 정의할 수도 있다. 본 기술 분야의 숙련자들에게 잘 알려진 바와 같이, 스키마 파일(330)은 소정의 문서(310)와 첨부되거나 조합될 수도 있으며, 첨부된 스키마 파일에서 정의된 허용가능한 XML 마크업을 문서(310)에 적용한다. 하나의 실시예에 따르면, 첨부되거나 조합된 스키마 파일(330)의 XML 엘리먼트들로 표시된 문서(310)는, 첨부되거나 조합된 스키마 파일(330)을 식별하는 네임스페이스와 관련된 균일 리소스 식별자(URI)를 지적함으로써, 첨부되거나 조합된 스키마 파일을 지적할 수도 있다.

본 발명의 실시예들에 따르면, 문서(310)는 복수의 첨부 스키마 파일을 구비할 수도 있다. 즉, 문서(310)의 크리에이터는, 하나 이상의 스키마 파일로부터의 XML 마크업의 주석을 위한 프레임워크를 제공하기 위하여, 하나 이상의 스키마 파일(330)을 문서(310)와 조합하거나 첨부할 수도 있다. 예를 들어, 문서(310)는 금융 데이터와 관련된 텍스트 또는 데이터를 포함할 수도 있다. 문서(310)의 크리에이터는, XML 마크업을 포함하는 XML 스키마 파일(330)들을 복수의 금융 제도와 관련된 정의들과 조합하기를 희망할 수도 있다. 따라서, 문서(310)의 크리에이터는 하나 이상의 금융 제도로부터의 XML 스키마 파일(330)들을 문서(310)와 조합할 수도 있다. 마찬가지로, 소정의 XML 스키마 파일(330)은, 금융 데이터를 소정의 포맷으로 배치하기 위한 템플릿(template)과 같은 특정 문서 구조와 조합될 수도 있다.

본 발명의 실시예들에 따르면, XML 스키마 파일들 및 관련 문서 솔루션들의 집합은, 문서(310)로부터 개별적으로 위치된 네임스페이스 또는 스키마 라이브러리에서 유지될 수도 있다. 문서(310)는, 그 문서(310)와 첨부되거나 조합된 하나 이상의 스키마 파일과 관련된 네임스페이스 또는 스키마 라이브러리내 URI들에 대한 포인터들을 교대로 포함할 수도 있다. 문서(310)는 하나 이상의 관련 스키마 파일로부터 정보를 요구하기 때문에, 그 문서(310)는 네임스페이스 또는 스키마 라이브러리를 지적하여, 그 요구된 스키마 정의들을 얻는다. 네임스페이스 또는 스키마 라이브러리들의 운영에 대한 상세한 논의는, "System and Method for Providing Namespace Related Information" 제목으로 2002년 6월 27일에 출원된 미국 출원번호 제10/184,190호 및 "System and Method for Obtaining and Using Namespace Related Information for Opening XML Documents" 제목으로 2002년 6월 27일에 출원된 미국 출원번호 제10/185,940호를 참조하는데, 두 개의 미국 출원의 내용은 본 명세서에 완전하게 설명된 참조로서 포함된다. XML 스키마 파일들 및 관련 솔루션들과 같은 소프트웨어 컴포넌트들을 네임스페이스 또는 스키마 라이브러리로부터 다운로드하기 위한 메커니즘에 대한 상세한 논의는, "Mechanism for Downloading Software Components from a Remote Source for Use by a Local Software Application" 제목으로 2002년 6월 5일에 출원된 미국 출원번호 제10/164,260호를 참조한다.

다시 도 3을 참조하여, 상술한 바와 같이, 문서(310)에 적용된 XML 마크업을 그 문서(310)에 첨부되거나 조합된 XML 스키마 파일(330)과 대조하여 확인하기 위한 스키마 확인 기능성 모듈(350)을 설명한다. 상술한 바와 같이, 스키마 파일(330)은 허용가능한 XML 엘리먼트들 및 관련 속성들을 설정하고, 관련 스키마 파일(330)로부터의 XML 마크업을 이용하여 문서(310)의 타당한 주석을 위한 규칙들을 정의한다. 예를 들어, 스키마 파일(330)에 도시한 바와 같이, 두 개의 자식 엘리먼트 <title> 및 <body>는, 루트 또는 부모 엘리먼트 <introCard> 하에서 정의된다. 자식 엘리먼트 <title> 및 <body>와 관련된 텍스트의 허용가능한 스트링(string) 길이를 정의하는 속성들(340,345)이 또한 설명된다. 상술한 바와 같이, 사용자가 스키마 파일(330)에 포함된 XML 마크업 정의들을 위반하면서, 문서에 첨부되거나 조합된 스키마 파일(330)로부터의 XML 마크업을 이용하여 문서(310)에 주석을 달려고 시도하면, 무효가 되거나 에러 상태가 된다. 예를 들어, 사용자가 25 문자를 초과하는 타이틀 스트링을 입력하려고 시도하면, 그 텍스트 엔트리는 스키마 파일(330)의 <title> 엘리먼트의 최대 문자 길이 속성을 위반한다. 문서(310)에 적용된 XML 마크업을 관련 스키마 파일(330)과 대조하여 확인하기 위하여, 스키마 확인 모듈(350)이 사용된다. 스키마 확인 모듈(350)은, XML 마크업 및 관련 텍스트가 문서(310)에 입력될 때, 문서(310)에 입력된 XML 마크업 및 관련 텍스트를 조합되거나 첨부된 XML 스키마 파일(330)과 대조하여 비교하기 위해 충분한 컴퓨터 실행가능한 명령들을 포함하는 소프트웨어 모듈임을 본 기술 분야의 숙련자들은 이해해야 한다.

본 발명의 실시예들에 따르면, 스키마 확인 모듈(350)은, 문서(310)에 적용된 각각의 XML 마크업 엘리먼트 및 관련 텍스트 또는 데이터를 첨부되거나 조합된 스키마 파일(330)과 대조하여 비교하며, 각각의 엘리먼트 및 관련 텍스트 또는 데이터가, 첨부된 스키마 파일(330)에 의해 설정된 규칙들 및 정의들을 따르는지를 결정한다. 예를 들어, 사용자가 <title> 엘리먼트(320)에 의해 주석을 단 25 문자를 초과하는 문자 스트링을 입력하려고 시도하면, 스키마 확인 모듈은, 그 텍스트 스트링을 첨부된 스키마 파일(330)의 그 텍스트 스트링 속성(340)과 대조하여 비교하며, 사용자에 의해 입력된 그 텍스트 스트링이 최대 허용가능한 텍스트 스트링 길이를 초과함을 결정한다. 따라서, 에러 메시지 또는 다이얼로그가 사용자에게 제공되어, 사용자에 의해 입력되고 있는 텍스트 스트링은 첨부된 스키마 파일(330)에 따른 최대 허용가능한 문자 길이를 초과함을 그 사용자에게 경고할 것이다. 마찬가지로, 사용자가 XML 마크업 엘리먼트를 <title> 및 <body> 엘리먼트들 사이에 추가하려고 시도하면, 스키마 확인 모듈(350)은, 사용자에 의해 적용된 XML 마크업 엘리먼트는 첨부된 스키마 파일(330)에 따른 <title> 및 <body> 엘리먼트들 사이에 허용된 타당한 엘리먼트가 아님을 결정할 것이다. 따라서, 스키마 확인 모듈(350)은 에러 메시지 또는 다이얼로그를 생성하여, 부당한 XML 마크업의 사용을 그 사용자에게 경고한다.

네임스페이스/스키마 라이브러리를 위한 프로그램 가능한 객체 모델

도 3을 참조하여 상술한 바와 같이, 확장성 생성 언어(XML) 마크업을 문서(310)에 적용하기 위한 정의 및 규칙 지향 프레임워크를 제공하기 위하여, 하나 이상의 스키마 파일(330)이 문서와 조합되거나 첨부되어, 소정의 스키마 파일(330)에 대응하는 XML 마크업 엘리먼트들을 문서(310)에 적용하도록 하는 정의들 및 규칙들을 설정할 수도 있다. 상술한 바와 같이, 복수의 XML 스키마 파일 및 다른 문서 솔루션, 예를 들어 사전구성된 템플릿들이 단일 XML 문서(310)에 첨부되거나 조합될 수도 있다. 더욱이, 상술한 바와 같이, 네임스페이스 확인에 의해 식별된 수많은 다른 XML 스키마 파일들 및 수많은 문서 솔루션들은 문서(310)와는 별개인 네임스페이스 또는 스키마 라이브러리에 저장될 수도 있다. 본 발명의 실시예

들에 따르면, 사용자는, 하나 이상의 문서(310)와 관련된 네임스페이스 또는 스키마 라이브러리를 프로그램적으로 호출하여, 스키마 파일 네임스페이스 및 관련 정의, 규칙, 리소스, 및 네임스페이스 또는 스키마 라이브러리에 포함된 다양한 네임스페이스 식별자들과 관련된 솔루션을 주문 제작하거나 조작하도록 허용된다.

도 4는, 문서(310), 네임스페이스 또는 스키마 라이브러리(400) 및 제3자 애플리케이션(450)간의 상호작용을 설명하는 블록도이다. 본 발명의 실시예들에 따르면, 사용자들은 일련의 객체 지향 메시지 호출 또는 애플리케이션 프로그래밍 인터페이스(470)를 통해 네임스페이스 라이브러리(400)를 프로그램적으로 호출하여, 각각의 스키마 파일들(410,430) 또는 네임스페이스 라이브러리(400)에서 식별된 스키마 파일들과 관련된 리소스들(420,440)의 콘텐츠 또는 운영을 수정할 수도 있다. 사용자는 일련의 객체 지향 메시지 호출을 통해 애플리케이션(305) 또는 제3자 프로그램(450)으로부터의 네임스페이스 라이브러리와 통신할 수도 있으며, 그 제3자 프로그램은, C, C+ + , C#, Visual Basic 등의 다양한 프로그래밍 언어들을 이용하여 개발될 수도 있다.

일련의 애플리케이션 프로그래밍 인터페이스(470)를 통해 네임스페이스 라이브러리에 액세스함으로써, 사용자는 하나 이상의 추가 XML 스키마 파일 또는 네임스페이스를 XML 데이터와 프로그램적으로 조합할 수도 있으며, 역으로, 사용자는, 하나 이상의 XML 스키마 파일과, 문서(310)에 적용된 XML 데이터 또는 마크업 사이에 현존하는 조합들을 검출하여 제거할 수도 있다. 또한, 사용자는 외형 정보 언어 변환(XSLT)을 문서에 적용된 XML 데이터와 프로그램적으로 조합할 수도 있으며, 역으로, 사용자는 문서(310)에 적용된 XML 데이터와의 조합으로부터 현존하는 XSLT 변환들을 검출하여 제거할 수도 있다. 더욱이, 사용자는 다른 파일들 및 실행가능한 소프트웨어 애플리케이션들을 문서(310)에 적용된 XML 데이터와 프로그램적으로 조합할 수도 있으며, 다른 소프트웨어 애플리케이션들 및 파일들과 XML 데이터와의 현존하는 조합들을 검출하여 제거할 수도 있다.

예를 들어, 네임스페이스 라이브러리(400)내에 도시한 네임스페이스(430)는 지원 문서 템플릿을 위한 사전포맷된 구조로 구성된 솔루션을 포함할 수도 있다. 그 솔루션이 문서(310)에 적용될 때, 지원 템플릿 문서의 크리에이터에 의해 설계된 관련 스키마 정의 및 규칙은, 문서(310)에 입력된 XML 마크업 및 관련 텍스트에 적용될 것이다. 지원 문서 템플릿과 관련된 스키마 파일이, 지원 문서의 경력 섹션에 적어도 세 개의 과거 또는 현재의 직업 설명을 구비해야 함을 요구한다면, 적어도 세 개의 직업 설명이 경력 섹션에 후속 사용자에 의해 순서대로 입력되어야 한다는 스키마 정의가 문서(310)에 적용되며, 그 XML 문서(310)는 스키마 확인 모듈(350)에 의해 확인될 것이다. 계속되는 이 예에서, 그러한 지원 문서 템플릿 스키마 파일이 문서(310)와 조합되고, 사용자가 스키마 파일과 문서(310)와의 조합을 제거하기를 희망한다면, 그 사용자는, 객체 지향 메시지 호출을 제공된 애플리케이션 프로그래밍 인터페이스를 이용하여 네임스페이스 라이브러리(400) 또는 애플리케이션(305)으로 송신함으로써 제3자 프로그램으로부터 프로그램적으로 행하여, 문서(310)로부터 지원 문서 템플릿 스키마 파일의 조합을 제거하도록 지시할 수도 있다.

다음은, 상술한 바와 같이 사용자가 네임스페이스 라이브러리(400)에 프로그램적으로 액세스하도록 하는 객체 지향 메시지 호출 또는 애플리케이션 프로그래밍 인터페이스를 포함하는 객체들 및 관련 속성들의 설명이다. 이하에서 설정된 각각의 객체 및 관련 속성은, 그 객체 또는 관련 속성의 운영 및 기능성의 설명이다.

애플리케이션 객체

다음은 객체의 방법들 및 속성들이다.

.XMLNamespaces property

XMLNamespaces 집합에 대한 판독용 포인터로서 애플리케이션에 이용가능한 네임스페이스 라이브러리를 나타낸다.

XMLNamespaces collection object - XMLNamespace 객체들에 액세스를 제공하는 객체. 네임스페이스 라이브러리를 나타낸다. 집합내 각각의 XMLNamespace 객체는 네임스페이스 라이브러리내 하나의 유일한 네임스페이스를 나타낸다. 다음은 본 객체의 방법들 및 속성들이다.

.Add() method

집합에 새로운 XMLNamespace 객체를 생성하여 추가하는 방법. 네임스페이스 라이브러리에 새로운 네임스페이스를 등록하기 위해 사용된다. 새로운 XMLNamespace 객체를 리턴시킨다. 다음의 파라미터들을 허용할 수 있다.

Path - 네임스페이스를 위한 스키마 파일에 대한 포인터. 본 포인터는 스트링으로서 표현된 파일 경로일 수 있다.

NamespaceURI - 스키마를 나타내는 네임스페이스의 URI. 본 URI는 텍스트 스트링일 수 있다.

Alias - 프로그래머가 지정할 수도 있는 네임스페이스를 위한 대안적인(사용자 친화적인) 이름을 나타내는 텍스트 스트링.

InstallForAllUsers - 네임스페이스 라이브러리내 새로운 네임스페이스가 컴퓨터의 모든 사용자들 또는 현재의 사용자에게만 이용가능한 것인지를 나타내는 플래그.

.Application property

본 객체 모델의 애플리케이션을 나타내는 애플리케이션 객체에 대한 판독용 포인터.

.Count property

네임스페이스 라이브러리내 등록된 네임스페이스들의 개수를 리턴시키는 판독용 포인터. 본 속성은 XMLNamespaces 집합내 XMLNamespace 객체들의 전체 개수와 동일하다.

.Creator property

본 객체의 크리에이터에 대한 판독용 포인터.

.InstallManifest() method

네임스페이스들을 네임스페이스 라이브러리에 등록하는 솔루션 매니페스트들을 인스톨하기 위한 방법. 다음의 파라미터들을 허용할 수 있다.

Path - 매니페스트를 위한 매니페스트 파일에 대한 포인터. 본 포인터는 텍스트 스트링으로서 표현된 파일 경로일 수 있다.

InstallForAllUsers - 매니페스트에 의해 네임스페이스 라이브러리에 인스톨된 새로운 네임스페이스들이 컴퓨터의 모든 사용자들 또는 현재의 사용자에게만 이용가능한 것인지를 나타내는 플래그.

.Item() method

숫자 인덱스 또는 탐색 키워드를 이용하여 이 집합의 각 멤버들을 액세스하기 위한 방법. 본 방법은 다음의 파라미터들을 허용할 수 있다.

Index - 요구된 XMLNamespace 객체의 네임스페이스 라이브러리내 위치를 나타내는 숫자. 본 인덱스는 에일리어스 또는 요구된 네임스페이스의 URI를 나타내는 텍스트 스트링일 수 있다.

.Parent property

집합의 부모 객체를 리턴시키는 판독용 속성. 본 속성은 XMLNamespaces 집합이 액세스되는 애플리케이션으로 포인터를 리턴시킨다.

XMLNamespace object - 네임스페이스 라이브러리내 각각의 네임스페이스 엔트리를 나타내는 객체(및 XMLNamespaces 집합내 각각의 항목). 다음은 본 객체의 방법들 및 속성들이다.

.Alias property

프로그래머가 네임스페이스와 조합하는 에일리어스를 제어하기 위한 속성. 다음의 파라미터를 지원할 수 있다.

AllUsers - 에일리어스가 모든 사용자들 또는 현재의 사용자에게만 이용가능한 것인지를 나타내는 플래그.

.Application property

본 객체 모델의 애플리케이션을 나타내는 애플리케이션 객체에 대한 판독용 포인터.

.AttachToDocument() method

본 객체에 의해 표현된 네임스페이스의 스키마를 선택된 문서에 첨부하기 위한 방법. 다음의 파라미터들을 지원한다.

Document - 스키마가 첨부되어야 하는 문서에 대한 포인터.

.Creator property

본 객체의 크리에이터에 대한 판독용 포인터.

.DefaultTransform property

네임스페이스와 관련된 디폴트 XSLT 변환을 지적하는 속성. 다음의 파라미터를 지원할 수 있다.

AllUsers - 디폴트 변환 세팅이 머신의 모든 사용자들 또는 현재의 사용자에게만 영향을 끼치는 것인지를 나타내는 플래그.

.Delete() method

집합으로부터 XMLNamespace 객체를 제거하고 파괴하기 위한 방법으로서, 본 객체에 의해 표현된 네임스페이스 조합을 네임스페이스 라이브러리로부터 효과적으로 제거하기 위한 방법.

.Location property

XMLNamespace 객체에 의해 표현된 네임스페이스와 관련된 스키마의 위치를 제어하는 판독용 속성. 다음의 파라미터를 지원할 수 있다.

AllUsers - 스키마 위치 세팅이 머신의 모든 사용자들 또는 현재의 사용자에게만 영향을 끼지는 것인지를 나타내는 플래그.

.Parent property

XMLNamespace 객체의 부모 객체를 리턴시키는 판독용 속성. 본 속성은 객체가 멤버인 XMLNamespaces 집합으로 포인터를 리턴시킨다.

.URI property

본 객체에 의해 표현된 네임스페이스의 URI를 리턴시키는 판독용 포인터.

.XSLTransforms property

본 객체에 의해 표현된 네임스페이스와 관련된 XSLT 변환들을 나타내는 XSLTransforms 집합에 대한 판독용 포인터.

XSLTransforms object - XSLTransforms 객체들에 액세스를 제공하는 객체로서, 각각의 XSLTransforms 객체는 네임스페이스 라이브러리내 네임스페이스와 관련된 하나의 유일한 XSLT 변환을 나타낸다. 다음은 본 객체의 방법들 및 속성들이다.

.Add() method

집합에 새로운 XSLTransform 객체를 생성하여 추가하기 위한 방법. 새로운 XSLT 변환을 네임스페이스 라이브러리내 네임스페이스와 조합하기 위해 사용된다. 새로운 XSLTransform 객체를 리턴시킨다. 다음의 파라미터들을 허용할 수 있다.

Location - XSLT 파일에 대한 포인터; 텍스트 스트링으로서 표현된 파일 경로일 수 있다.

Alias - 프로그래머가 지정할 수도 있는 XSLT 변환을 위한 대안적인(사용자 친화적인) 이름을 나타내는 텍스트 스트링.

InstallForAllUsers - 네임스페이스 라이브러리내 새로운 네임스페이스가 컴퓨터의 모든 사용자들 또는 현재의 사용자에게만 이용가능한 것인지를 나타내는 플래그.

.Application property

본 객체 모델의 애플리케이션을 나타내는 애플리케이션 객체에 대한 판독용 포인터.

.Count property

네임스페이스 라이브러리내 소정의 네임스페이스를 위해 등록된 XSLT 변환들의 개수를 리턴시키는 판독용 속성. XSLTransforms 집합내 XSLTransform 객체들의 전체 개수와 동일하다.

.Creator property

본 객체의 크리에이터에 대한 판독용 포인터.

.Item() method

숫자 인덱스 또는 탐색 키워드를 이용하여 집합의 각 멤버들을 액세스하기 위한 방법. 다음의 파라미터들을 허용할 수 있다.

Index - 요구되는 XSLTransform 객체의 네임스페이스 라이브러리내 위치를 나타내는 숫자. 본 인덱스는 요구된 XSL 변환의 에일리어스를 나타내는 텍스트 스트링일 수 있다.

.Parent property

집합의 부모 객체를 리턴시키는 판독용 속성. 본 속성은 XSLTransforms 집합이 액세스되는 애플리케이션으로 포인터를 리턴시킨다.

XSLTransform object - 네임스페이스 라이브러리내 네임스페이스와 관련된 XSLT 변환을 나타내는 객체. 다음은 본 객체의 방법들 및 속성들이다.

.Alias property

프로그래머가 네임스페이스 라이브러리내 XSLT 변환과 조합하는 에일리어스를 제어하기 위한 속성. 다음의 파라미터를 지원할 수 있다.

AllUsers - 에일리어스가 모든 사용자들 또는 현재의 사용자에게만 이용가능한 것인지를 나타내는 플래그.

.Application property

본 객체 모델의 애플리케이션을 나타내는 애플리케이션 객체에 대한 판독용 포인터.

.Creator property

본 객체의 크리에이터에 대한 판독용 포인터.

.Delete() method

집합으로부터 XSLTransform 객체를 제거하고 파괴하기 위한 방법으로서, XSLT 변환과, 네임스페이스 라이브러리내 네임스페이스간의 조합을 효과적으로 제거하기 위한 방법.

.Location property

소정의 네임스페이스와 관련되고 XSLTransform 객체에 의해 표현된 XSLT 변환의 위치를 제어하는 판독용 속성. 다음의 파라미터를 지원할 수 있다.

AllUsers - XSLT 변환 위치 세팅이 머신의 모든 사용자들 또는 현재의 사용자에게만 영향을 끼지는 것인지를 나타내는 플래그.

.Parent property

XSLTransform 객체의 부모 객체를 리턴시키는 판독용 속성. 본 속성은 객체가 멤버인 XSLTransforms 집합으로 포인터를 리턴시킨다.

상술한 바와 같이, 사용자가 확장성 생성 언어 네임스페이스 또는 스키마 라이브러리에서 식별된 리소스들을 프로그램적으로 호출하도록 하여, 네임스페이스 또는 스키마 라이브러리에서 식별되거나 포함된 리소스들을 하나 이상의 관련 문서와 조합하는 것을 주문 제작하거나 수정하는 방법 및 시스템이 제공된다. 본 기술 분야의 숙련자들은, 본 발명의 범위 또는 사상을 벗어나지 않는 다양한 수정물 또는 변형물들이 본 발명에서 이루어질 수도 있음을 인식할 것이다. 본 기술 분야의 숙련자들은, 본 명세서에서 기재된 본 발명의 설명 및 실행을 고려한 다른 실시예들을 인식할 것이다.

발명의 효과

사용자 또는 프로그래머가 네임스페이스 또는 스키마 라이브러리에 액세스하면, 사용자는 XML 스키마 파일들을 관련 문서내 XML 데이터와 프로그램적으로 조합할 수도 있으며, 역으로, 사용자는 문서에 포함된 XML 데이터와의 XML 스키마 파일들의 조합을 검출하여 제거할 수도 있다. 또한, 사용자는 변환 파일들을 문서에 포함된 XML 데이터와 프로그램적으로 조합할 수도 있으며, 문서에 포함된 XML 데이터와 관련된 현존하는 변환 파일들을 검출하여 제거할 수도 있다. 또한, 사용자는 다른 파일들 및 XML 기반의 다른 문서 솔루션들과 관련된 실행가능한 소프트웨어를 문서에 포함된 XML 데이터와 조합할 수도 있다. 부가적으로, 사용자는 XML 기반 솔루션들과 다른 타입의 실행가능한 소프트웨어와의 조합을 문서에 포함된 관련 XML 데이터로부터 검출하여 제거할 수도 있다.

(57) 청구의 범위

청구항 1.

확장성 생성 언어(XML) 스키마 라이브러리의 리소스들을 액세스하기 위한 프로그램 가능한 객체 모델로서,

사용자가 XML 스키마 라이브러리에서 식별된 리소스들을 프로그램적으로 액세스하도록 하기 위한 애플리케이션 프로그래밍 인터페이스를 포함하고,

상기 애플리케이션 프로그래밍 인터페이스는 문서에 적용된 XML 마크업에 대한 XML 스키마 파일의 조합을 요구하기 위한 메시지 호출을 포함하며,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 XML 스키마 파일의 조합에 응답하여 상기 XML 스키마 라이브러리로부터 리턴 값을 수신하도록 작동하는 프로그램 가능한 객체 모델.

청구항 2.

제1항에 있어서,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 XML 스키마 파일의 조합을 제거하도록 요구하기 위한 메시지 호출을 더 포함하고,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 XML 스키마 파일의 조합의 제거에 응답하여 상기 XML 스키마 라이브러리로부터 리턴 값을 수신하도록 작동하는 프로그램 가능한 객체 모델.

청구항 3.

확장성 생성 언어(XML) 스키마 라이브러리의 리소스들을 액세스하기 위한 프로그램 가능한 객체 모델로서,

사용자가 XML 스키마 라이브러리에서 식별된 리소스들을 프로그램적으로 액세스하도록 하기 위한 애플리케이션 프로그래밍 인터페이스를 포함하고,

상기 애플리케이션 프로그래밍 인터페이스는 문서에 적용된 XML 마크업에 대한 XSLT 변환의 조합을 요구하기 위한 메시지 호출을 포함하며,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 XSLT 변환의 조합에 응답하여 상기 XML 스키마 라이브러리로부터 리턴 값을 수신하도록 작동하는 프로그램 가능한 객체 모델.

청구항 4.

제3항에 있어서,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 XSLT 변환의 조합을 제거하도록 요구하기 위한 메시지 호출을 더 포함하고,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 XSLT 변환의 조합의 제거에 응답하여 상기 XML 스키마 라이브러리로부터 리턴 값을 수신하도록 작동하는 프로그램 가능한 객체 모델.

청구항 5.

확장성 생성 언어(XML) 스키마 라이브러리의 리소스들을 액세스하기 위한 프로그램 가능한 객체 모델로서,

사용자가 XML 스키마 라이브러리에서 식별된 리소스들을 프로그램적으로 액세스하도록 하기 위한 애플리케이션 프로그래밍 인터페이스를 포함하고,

상기 애플리케이션 프로그래밍 인터페이스는 문서에 적용된 XML 마크업에 대한 하나 이상의 XML 기반 리소스들의 조합을 요구하기 위한 메시지 호출을 포함하며,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 하나 이상의 XML 기반 리소스들의 조합에 응답하여 상기 XML 스키마 라이브러리로부터 리턴 값을 수신하도록 작동하는 프로그램 가능한 객체 모델.

청구항 6.

제5항에 있어서,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 XML 스키마 파일의 조합을 제거하도록 요구하기 위한 메시지 호출을 더 포함하고,

상기 애플리케이션 프로그래밍 인터페이스는 상기 문서에 적용된 상기 XML 마크업에 대한 상기 XML 스키마 파일의 조합의 제거에 응답하여 상기 XML 스키마 라이브러리로부터 리턴 값을 수신하도록 작동하는 프로그램 가능한 객체 모델.

청구항 7.

확장성 생성 언어(XML) 스키마 라이브러리의 리소스들을 액세스하기 위한 프로그램 가능한 객체 모델로서,

객체 지향 메시지 호출을 통해 상기 XML 스키마 라이브러리를 호출하고,

객체 속성 - 상기 객체 속성은 상기 XML 스키마 라이브러리에서 식별된 기능성과 관련되는 소프트웨어 객체와 관련됨 - 을 상기 XML 스키마 라이브러리로 전달하며,

상기 메시지 호출 및 상기 XML 스키마 라이브러리에 전달된 상기 객체 속성에 응답하여, 상기 XML 스키마 라이브러리에 전달된 상기 객체 속성과 관련되는 상기 XML 스키마 라이브러리에서 식별된 상기 기능성에 대한 액세스를 수신하는 프로그램 가능한 객체 모델.

청구항 8.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 파일로 전달하는 것은 새로운 XML 네임스페이스를 생성하여 상기 새로운 XML 네임스페이스를 XML 네임스페이스들의 집합에 추가하기 위한 방법 속성을 전달하는 것을 포함하며,

상기 새로운 XML 네임스페이스와 관련된 스키마 파일에 대한 경로 및 상기 새로운 XML 네임스페이스를 위한 균일 리소스 식별자는 상기 방법 속성의 파라미터들로서 상기 XML 스키마 라이브러리에 전달되는 프로그램 가능한 객체 모델.

청구항 9.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 XML 네임스페이스들을 상기 XML 스키마 라이브러리에 등록하기 위한 솔루션 매니페스트들을 인스톨하기 위한 방법 속성을 전달하는 것을 포함하는 프로그램 가능한 객체 모델.

청구항 10.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 숫자 인덱스 또는 탐색 키워드를 이용하여 XML 리소스들의 집합으로부터 각각의 XML 리소스를 액세스하기 위한 방법 속성을 전달하는 것을 포함하며,

상기 각각의 XML 리소스와 관련된 상기 숫자 인덱스는 파라미터로서 상기 방법 속성과 함께 전달되는 프로그램 가능한 객체 모델.

청구항 11.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 상기 XML 스키마 라이브러리에서 식별된 특정 네임스페이스와 관련되는 에일리어스 이름을 제어하기 위한 객체 속성을 전달하는 것을 포함하는 프로그램 가능한 객체 모델.

청구항 12.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 특정 XML 네임스페이스의 특정 XML 스키마 파일을 특정 문서에 첨부하기 위한 방법 속성을 전달하는 것을 포함하며,

상기 특정 문서에 대한 포인터는 상기 방법 속성의 파라미터로서 상기 XML 스키마 라이브러리에 전달되는 프로그램 가능한 객체 모델.

청구항 13.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 특정 네임스페이스와 관련되는 디폴트 XSLT 변환을 지적하는 객체 속성을 전달하는 것을 포함하는 프로그램 가능한 객체 모델.

청구항 14.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 네임스페이스 객체들의 집합으로부터 XML 네임스페이스 객체를 제거하기 위한 방법 속성을 전달하는 것을 포함하는 프로그램 가능한 객체 모델.

청구항 15.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 새로운 XSLT 변환을 생성하여 상기 새로운 XSLT 변환을 XSLT 변환들의 집합에 추가하기 위한 방법 속성을 전달하는 것을 포함하며,

상기 새로운 XSLT 변환에 대한 포인터는 상기 방법 속성에 대한 파라미터로서 상기 XML 스키마 라이브러리에 전달되는 프로그램 가능한 객체 모델.

청구항 16.

제7항에 있어서,

숫자 인덱스 또는 탐색 키워드를 이용하여 XSLT 변환들의 집합에 포함된 각각의 XSLT 변환을 액세스하기 위한 방법 속성을 더 포함하며, 요구된 XSLT 변환의 상기 XML 스키마 라이브러리에서의 위치를 나타내는 상기 숫자 인덱스는 파라미터로서 상기 방법 속성과 함께 상기 XML 스키마 라이브러리에 전달되는 프로그램 가능한 객체 모델.

청구항 17.

제7항에 있어서,

상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 상기 XML 스키마 라이브러리에서 식별된 XSLT 변환과 관련되는 에일리어스 이름을 제어하기 위한 객체 속성을 전달하는 것을 포함하는 프로그램 가능한 객체 모델.

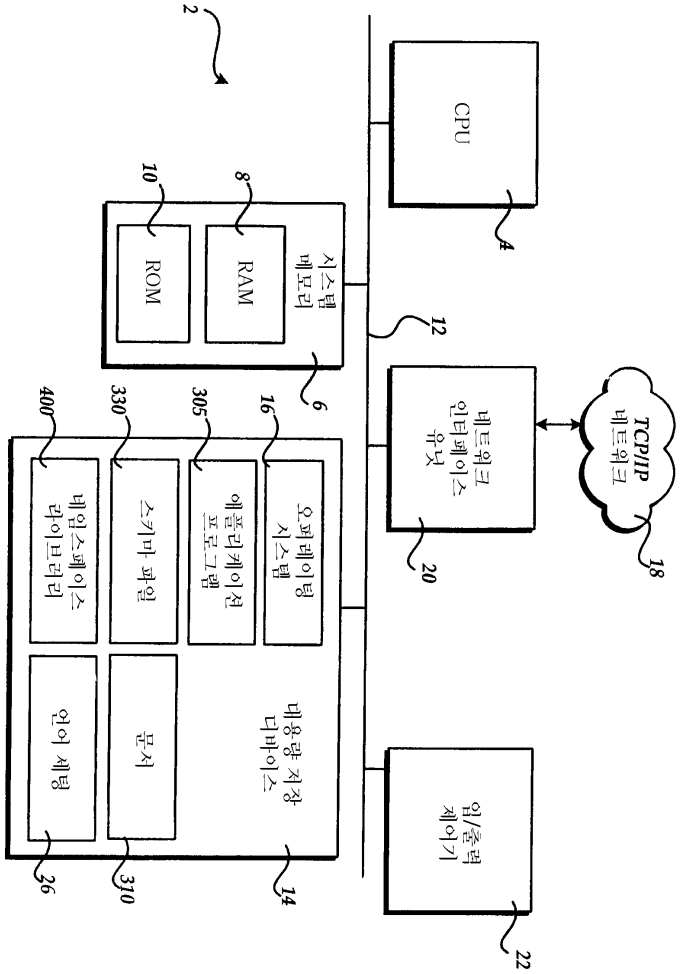
청구항 18.

제7항에 있어서,

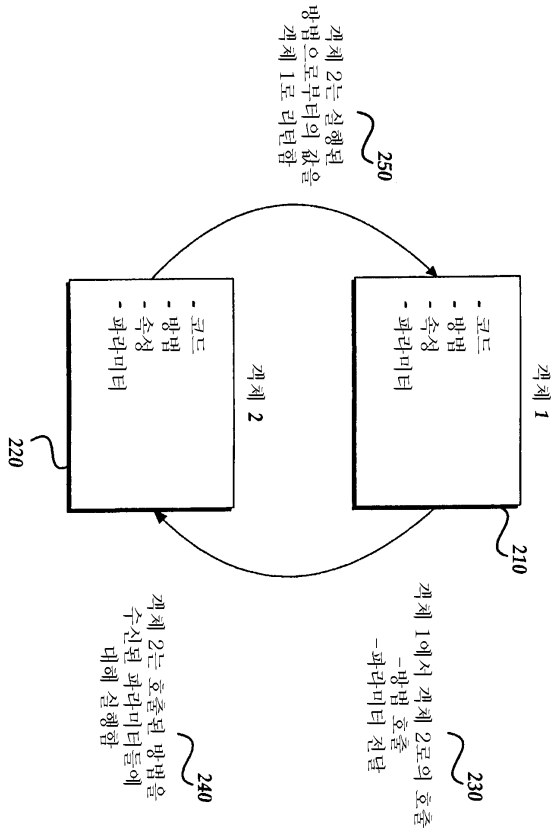
상기 객체 속성을 상기 XML 스키마 라이브러리로 전달하는 것은 XSLT 변환들의 집합으로부터 XSLT 변환을 제거하기 위한 방법 속성을 전달하는 것을 포함하는 프로그램 가능한 객체 모델.

도면

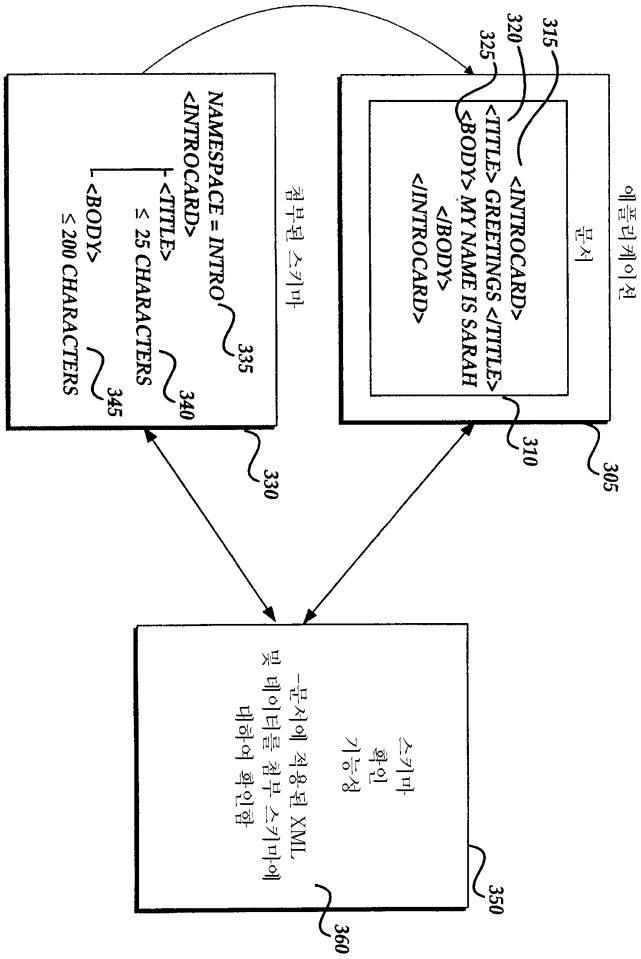
도면1



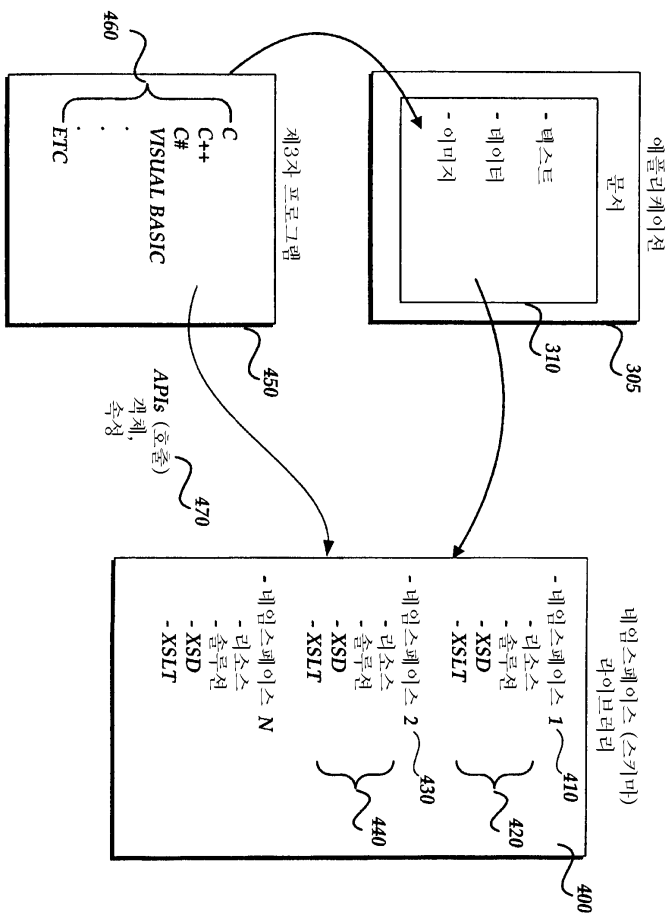
도면2


객체 1
- 코드
- 방법
- 속성
- 파라미터
210
객체 2
- 코드
- 방법
- 속성
- 파라미터
220
객체 1에서 객체 2로의 호출
- 방법 호출
- 파라미터 전달
230
객체 2는 호출된 방법을
수신된 파라미터들에
대해 실행함
240
객체 2는 실행된
방법으로부터의 값을
객체 1로 리턴함
250

도면3


315
320
325
<INTROCARD>
<TITLE> GREETINGS </TITLE>
<BODY> MY NAME IS SARAH
</BODY>
</INTROCARD>
문서
애플리케이션
310
305
NAMESPACE = INTRO
<INTROCARD>
<TITLE>
≤ 25 CHARACTERS
<BODY>
≤ 200 CHARACTERS
335
340
345
첨부된 스키마
330
스키마
확인
기능성
-문서에 적용된 XML
및 데이터를 첨부 스키마에
대하여 확인함
350
360

도면4


에플리케이션
문서
- 텍스트
- 데이터
- 이미지
305
310
제3자 프로그램
C
C++
C#
VISUAL BASIC
ETC
460
450
APIs (호출)
객체,
속성
470
네임스페이스 (스키마)
라이브러리
- 네임스페이스 1
- 리소스
- 솔루션
- XSD
- XSLT
410
420
- 네임스페이스 2
- 리소스
- 솔루션
- XSD
- XSLT
430
440
- 네임스페이스 N
- 리소스
- 솔루션
- XSD
- XSLT
400