(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2008-0100923
(43) 공개일자 2008년11월21일

(51) Int. Cl.
G06F 17/30 (2006.01) *G06F 17/00* (2006.01)
(21) 출원번호 10-2007-0046967
(22) 출원일자 2007년05월15일
심사청구일자 2007년05월15일

(71) 출원인
충남대학교산학협력단
대전광역시 유성구 궁동 220번지 충남대학교
(72) 발명자
이규철
대전광역시 서구 삼천동 가람아파트 6-1303
유정연
대전광역시 유성구 궁동 다솔아파트 102동 908호
(74) 대리인
권오식, 박창희, 김종관

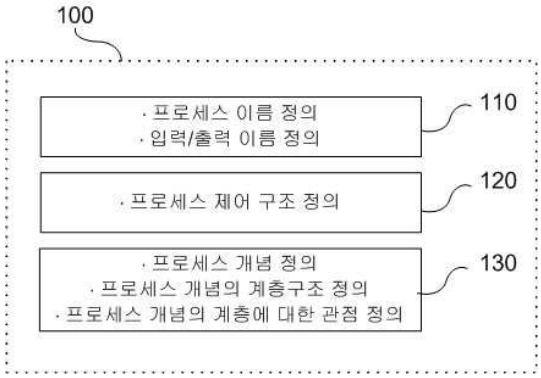
전체 청구항 수 : 총 11 항

(54) 지능적 웹 서비스 발견을 위해 프로세스 정보를 온톨로지로표현하는 방법

(57) 요 약

본 발명은 시맨틱 웹 서비스에서 제안된 발견의 문제점들을 해결하고 지능적 웹 서비스 발견을 제공하기 위해, 프로세스 정보를 온톨로지로 모델링하고 표현하는 방법에 관한 것이다. 상세하게는 서비스를 구성하는 프로세스 정보를 온톨로지로 표현하는 방법에 있어, 온톨로지 언어인 토픽 맵을 이용하며, 프로세스 이름, 입력 이름 또는 출력 이름에 대한 정의를 포함하는 프로세스 시그니처 온톨로지; 프로세스 제어 구조에 대한 정의를 포함하는 프로세스 구조 온톨로지; 프로세스 개념들에 대한 정의를 포함하는 프로세스 개념 온톨로지를 포함하는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 정보를 온톨로지로 표현하는 방법에 관한 것이다.

대 표 도 - 도3



특허청구의 범위

청구항 1

웹 서비스를 구성하는 프로세스 정보를 온톨로지로 표현하는 방법에 있어,

온톨로지 언어인 토픽 맵을 이용하며,

프로세스 이름, 입력 이름 또는 출력 이름에 대한 정의를 포함하는 프로세스 시그니처 온톨로지;

프로세스 제어 구조에 대한 정의를 포함하는 프로세스 구조 온톨로지; 및

프로세스 개념들에 대한 정의, 개념들 간의 관계의 정의, 개념들에 대한 관점의 정의, 개념들 간의 분류 정의 및 분류에 대한 관점의 정의를 포함하는 프로세스 개념 온톨로지;

를 포함하는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지의 표현방법.

청구항 2

제 1항에 있어서,

상기 프로세스 시그니처 온톨로지의 프로세스 이름, 입력 이름 또는 출력 이름은 토픽 맵의 토픽으로 정의되는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지의 표현방법.

청구항 3

제 1항에 있어서,

상기 프로세스 구조 온톨로지에서 순차, 분리, 분리와 결합, 임의-순서, 선택, 조건 또는 반복을 포함하는 프로세스 제어 구조는 토픽 맵의 토픽으로 정의 되는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지의 표현방법.

청구항 4

제 1항에 있어서,

상기 프로세스 시그니처 온톨로지의 프로세스 이름과 입력/출력 이름간의 관계는 연관관계를 이용하여 정의 되는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지의 표현방법.

청구항 5

제 1항에 있어서,

상기 프로세스 구조 온톨로지의 프로세스들 간의 제어구조 표현은 연관관계를 이용하여 정의 되는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지의 표현방법.

청구항 6

제 1항에 있어서,

상기 프로세스 개념 온톨로지에 사용되는 모든 개념들은 토픽 맵의 토픽으로 정의는 특징을 갖는 프로세스 기반 온톨로지의 표현방법.

청구항 7

제 6항에 있어서,

상기 개념 간 관계 또는 개념들 간의 다중 관계는 토픽 맵의 연관관계를 이용하여 표현되는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지의 표현방법.

청구항 8

제 6항에 있어서,

상기 개념에 대한 관점의 정의 및 상기 분류에 대한 관점의 정의는 토픽 맵의 연관관계의 스코프를 사용하여 정의되는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지의 표현방법.

청구항 9

제 1항에 있어서,

상기 프로세스 시그니처 온톨로지 또는 상기 프로세스 구조 온톨로지는 OWL-S에 정의된 온톨로지를 토픽 맵을 이용하여 변환시킨 것을 특징으로 하는 프로세스 기반 온톨로지의 표현방법.

청구항 10

제 9항에 있어서,

상기 OWL-S에 정의된 온톨로지의 변환 시 하기의 규칙을 따르는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지 표현방법.

규칙 1 : OWL-S에 정의된 온톨로지에 사용되는 프로세스를 기술하기 위해 필요한 모든 클래스들은 토픽 맵의 토픽으로 정의됨.

규칙 2 : 특정 클래스의 이름은 그 개념을 정의하는 토픽의 베이스이름으로 정의됨.

규칙 3 : 클래스들 간의 관계, 관계추론, 클래스의 속성 제약은 토픽맵에서 토픽들의 연관관계, 토픽의 역할 또는 스코프, 또는 이들의 조합을 이용하여 정의됨.

규칙 4 : 합집합, 여집합, 교집합, 열거 또는 분리를 포함하는 집합 연산자는 토픽의 역할과 연관관계를 이용하여 정의됨.

규칙 5 : 역관계를 포함한 속성에 대한 추론은 토픽들의 연관관계, 스코프, 베이스이름을 이용하여 정의됨.

청구항 11

제 1항에 있어서,

상기 프로세스 개념 온톨로지에 정의된 한 개 이상의 개념을 토픽의 인스턴스로 표현함으로써, 상기 토픽으로 표현된 프로세스의 의미를 표현하는 것을 특징으로 하는 토픽 맵을 이용한 프로세스 기반 온톨로지의 표현방법.

명 세 서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

<8> 웹 서비스(Web services)는 네트워크 상에서 공동 이용 가능한 기계 간에 상호작용을 제공하기 위해 설계된 소프트웨어 시스템이다. 웹 서비스는 3가지 역할을 중심으로 동작한다. 1) 서비스 제공자(service provider)는 기계가 처리 할 수 있는 형식으로 서비스 정보를 기술하고, 서비스 중재자(service broker)에 서비스를 등록한다. 2) 서비스 요구자(service requester)는 서비스 중재자를 통해 원하는 서비스를 찾는다. 3) 서비스 요구자는 서비스 제공자가 제공하는 서비스와 바인드(bind)하여 서비스를 실행한다. 이러한 특징은 SOA(Service Oriented Arch- itecture) 서비스의 구조를 구현하는 기술로서 활용된다.

<9> 지능적 웹 서비스(intelligent web services)는 웹 서비스를 시맨틱 웹으로 기술함으로써, 웹 서비스와 서비스 인터페이스에 대한 구조화된 지식과 시맨틱 기술을 질의하고 추론하는 것이다. 그러므로, 지능적 웹 서비스 발견(intelligent web services discovery)은 시맨틱을 부여하여 서비스의 정확률(precision)과 재현률(recall)을 향상시킴으로써, 기존의 웹 서비스 발견에 비해 관련된 서비스를 많이 검색하는 것을 말한다.

<10> 웹 서비스의 기본 표준들에서 서비스의 기술(description)은 WSDL(Web Service Description Language), 발견(discovery)은 UDDI(Universal Description Discovery and Integration)을 이용한다. 서비스 제공자는 WSDL을 이용하여 서비스를 기술하고, UDDI 레지스트리(registry)에 서비스를 등록한다. 서비스 요구자는 UDDI 레지스트

리를 통해 원하는 서비스를 검색한다. WSDL은 서비스의 구문 정보(syntactic information)만을 기술하고, UDDI는 키워드를 기반으로 서비스를 검색한다.

<11> 그러나, WSDL과 UDDI를 이용한 서비스 검색은 만족하지 않는 검색 결과를 제공한다. 예를 들어, 사용자가 "가격을 입력 받아 구매가 가능한 자동차들을 출력하는 자동차 판매 서비스를 찾아라."라고 요청했다고 가정하자. 이 질의는 가격을 입력(input)받고, 자동차를 출력(output)하는 자동차 판매 서비스를 검색하는 것이다. 그러나, 현재 웹서비스는 서비스의 이름과 설명 정보만을 이용하여 검색하므로 관련 있는 서비스들을 검색하지 못한다.

<12> 시맨틱 웹 서비스(Semantic Web services)는 위와 같은 현재 웹 서비스의 발견을 보완하기 위한 하나의 수단으로 제안되었다. 즉, 시맨틱 웹 서비스는 시맨틱 웹을 이용하여 웹 서비스를 기술한 것으로, 지능적 웹 서비스 발견을 제공하며, 자동화된 조합 및 상호 운용이 가능하도록 한다. 시맨틱 웹을 가능하게 하는 기술은 온톨로지(Ontology)이다. 이는 개념과 그들 간의 관계들로 정의하며, 특정 도메인에 관련된 단어들을 계층적으로 표현하고 이를 확장하여 추론 규칙을 가짐으로써 온톨로지를 표현한다. 온톨로지는 기존의 웹 서비스가 가지는 발견의 문제점을 해결할 수 있다. 앞에서 언급한 질의에 대해 입력/출력(input/output)에 대한 온톨로지를 정의하고 검색하면, "SUV", "Sedan"과 같이 자동차에 관련된 것을 출력(output)하는 자동차 판매 서비스들을 모두 검색할 수 있다.

<13> 시맨틱 웹 서비스 발견은 WSMO(Web Service Modeling Ontology)와OWL-S(OWL-based Web Service Ontology)의 서비스 프로파일(Service Profile)을 이용하여 검색되어 왔다. 그러나, 이 발견은 서비스 이름과 입력/출력에 관련된 온톨로지만을 이용하여 검색한다. 이것은 "배송 후 결제를 하는 도서 판매 서비스를 찾아라."와 같은 질의가 요청될 때, 서비스의 내부 정보를 이용하지 않아 사용자가 원하는 서비스들을 검색하지 못하거나 관련 없는 서비스들이 검색된다.

<14> 시맨틱 웹 서비스 표준으로는 3가지가 있다. 미주를 중심으로 한 OWL-S, 유럽을 중심으로 한 WSMO, 미주와 유럽의 연합으로 제안된 SWSF가 있다.

<15> OWL-S는 웹 서비스의 속성들(properties)과 능력들(capabilities)을 기술하는 온톨로지이다. OWL-S는 도 1과 같이 3가지 온톨로지인 서비스 프로파일(Service Profile), 프로세스 모델(Process Model), 서비스 그라운딩(Service Grounding)으로 구성되어 있다.

<16> 이 온톨로지들은 웹 서비스의 생명 주기에서 서로 다른 활동(activities)에 적용된다. 서비스 프로파일은 서비스 제공자가 서비스를 공표하고, 서비스 요구자가 서비스를 검색하는데 사용된다. 만일, 찾은 서비스가 사용자의 요구 사항에 맞는지를 알려면, 프로세스 모델을 이용하여 검토한 후 서비스를 선택할 수 있다. 이 프로세스 모델은 서비스의 시뮬레이션, 검증, 모니터링, 다중 서비스들과의 조합에 이용된다. 서비스 그라운딩은 서비스 접근에 관련된 서비스 호출 및 상호 운용에 이용된다.

<17> 서비스 프로파일은 서비스를 발견하기 위해 필요한 정보를 제공하는 온톨로지이다. 이 온톨로지는 3가지 종류의 정보로 구성되어 있다. 1) 서비스를 제공하는 사람이나 기관, 2) 서비스의 능력, 3) 서비스의 특성이다. 서비스의 능력은 정보 변화와 상태 변화를 표현한다. 정보 변화는 실행을 위해 요구되는 정보인 input과 실행 결과 정보인 output이다. 상태 변화는 서비스의 성공적인 실행을 위한 조건인 precondition과 서비스 실행에서 기대되는 결과인 effect로 나타낸다. 예를 들어, 도서 판매 서비스에 대한 능력은 도 2과 같이 표현할 수 있다. 이 서비스는 '신용카드 번호'와 '신용 카드 유효 날짜'를 input으로 받아, 실행 결과에 따라 output과 effect를 발생한다. 즉, 도서 판매에 대한 결제가 성공할 경우, 결제 성공 메시지를 출력(output)함과 동시에 구매자에게 도서를 배송(effect) 한다. 그렇지 않을 경우, 결제 실패 메시지를 출력(output)한다. 이 서비스가 실행되기 위해서는 반드시 구매자가 신용카드를 가지고 있어야 한다(precondition).

<18> 서비스의 특징은 서비스의 분류(classification)나 품질(quality) 등을 이용하여 기술할 수 있다. 서비스 프로파일은 UNSPSC(Universal Standard Products and Services Classification)나 NAICS(North American Industry Classification System)와 같이 OWL-S 외부에 정의된 분류를 이용하여 서비스의 분류 정보를 기술한다. 그 외에 추가적인 특징을 기술 할 수 있도록 변수를 정의하고 있다.

<19> 프로세스 모델은 프로세스를 이용하여 서비스의 동작을 상세히 기술한다. 서비스는 웹에서 접근이 가능한 프로그램으로, 하나의 프로세스 또는 여러 개의 하위 프로세스들로 표현 될 수 있다. 사용자는 최상위 프로세스를 이용하여 서비스의 정보를 검색할 수 있다. 이 프로세스 모델은 3종류의 프로세스인 원자 프로세스(Atomic Process), 복합 프로세스(Composite Process), 단순 프로세스(Sim- ple Process)를 이용하여 서비스를 기술한

다.

<20> 원자 프로세스는 가장 작은 단위의 프로세스이다. 이 프로세스는 한 단계로 실행된다. 복합 프로세스는 여러 개의 하위 프로세스들로 구성되어 있는 프로세스이다. 이것은 순차(Sequence), 분리(Split), 분리+결합(Split+Join) 등과 같은 제어 구조(Control Structure)를 이용하여 하위 프로세스들 간의 구조를 정의한다. 단순 프로세스는 호출이 불가능한 프로세스로써, 추상적인 요소로 사용한다. 이 프로세스는 원자 프로세스나 복합 프로세스에 대한 관점을 제공하기 위해 사용된다. 즉, 원자 프로세스의 사용에 대한 구체화된 방법을 제공하거나, 계획과 추론을 위해 복합 프로세스에 대한 단순화된 표현을 제공한다.

<21> WSMO는 e-워크(e-Work)와 e-전자상거래(e-Commerce)에서 서비스들간의 통합 문제를 해결하는 것을 목표로 정의된 개념적인 모델이다[16][17][18]. 이것은 유럽의 SWWS(Semantic Web enabled Web Services), DERI(Digital Enterprise Research Institute), Knowledge Web등의 구성 멤버들의 참가로써 개발되어 왔다. 최근에는 2005년 4월에 버전 1.2를 공표하였다.

<22> 이 모델은 온톨로지, 목적(goal), 웹 서비스, 미디에이터(mediator)와 같은 요소들로 구성되어 있다. 온톨로지는 다른 요소들을 기술하기 위해 필요한 용어와 정형화된 의미(formal semantics)를 제공한다. 목적은 사용자가 찾고자 하는 서비스의 정보들을 정의한다. 웹 서비스는 서비스에 대한 시맨틱 기술이다. 미디에이터는 구성 요소들간의 이질적 문제를 해결하는 연결자 역할을 한다.

<23> SWSF는 SWSF는 시맨틱 웹 서비스의 개발을 위해 SWSI(Semantic Web Services Initiative)에서 제안한 것으로, SWSF 모델은 SWSL(Semantic Web Services Language)과 SWSO(Semantic Web Services Ontology)를 포함한다. SWSL은 범용 논리 언어로써, 웹 서비스 개념과 기술에 대한 정형화된 특징을 명시하기 위해 사용하고, SWSO는 시맨틱 웹 서비스를 위한 개념적 모델을 정의한다.

<24> FLOWS(First-order Logic Ontology for Web Services)은 SWSL-FOL로 표현된 SWSO로써, 시맨틱 웹 서비스를 선언하고 여러 환경에서 시맨틱 웹 서비스의 추론을 가능하게 한다. FLOWS에서 웹 서비스를 기술은 PSL(Process Specification Language)의 메커니즘을 기반으로 한다. FLOWS의 구조는 OWL-S에서 제안한 것과 매우 유사하다. 이것은 서비스 기술자(Service Descriptor), 프로세스 모델(Process Model), 그라운딩(Grounding)으로 구성되어 있다. 서비스 기술자는 이름, 저자, 연락 정보, URL 등과 같이 서비스를 발견하고 매칭하는데 이용되는 웹 서비스에 관한 기본 정보를 제공한다.

<25> 웹 서비스의 발견 연구는 OWL-S를 기반으로 개발되어 온 Matchmaker 프로젝트와 METEOR-S 프로젝트, WSMO에서 제안한 WSMO 발견 명세서가 있으며, SWSF를 기반으로 한 연구는 아직 개발되어 있지 않고 있다.

<26> 상술한 기존의 연구들은 서비스를 발견하는데 있어 몇 가지 문제점을 가지고 있다.

<27> 첫째, 사용자는 서비스 이름, 입력/출력과 같은 서비스의 외부 정보만을 이용하여 검색한다. 예를 들어, 사용자가 "전화번호를 입력 받아 결제 할 수 있는 도서 판매 서비스를 찾아라."라고 요청했을 때, 기존 연구에서는 전화번호를 input으로 받고, 결제 메시지를 output하는 도서 판매 서비스들이 검색 될 것이다.

<28> 그러나 모든 결과가 사용자가 원하는 서비스는 아니다. 같은 input들과 output을 가진 도서 판매 서비스들이라도, 전화번호가 아이디/패스워드 찾기에 사용되는 서비스 1과, 전화번호가 전화 결제에 사용되는 서비스 2를 구별하지 못하며, 이때, 서비스 2만 사용자의 요구 사항을 만족하는 결과가 된다. 그러므로 사용자가 원하는 서비스만을 검색하기 위해서는 입력/출력과 이들과 관련된 프로세스를 함께 이용하여 검색해야 한다.

<29> 두 번째 문제점은 서비스를 이루는 내부 프로세스들인 내부 행위(internal behavior)를 내포하는 질의를 수행할 수 없다는 것이다. 예를 들어,서비스의 수행 과정에 대한 상세 정보를 알 수 없으므로, "배송 후 결제가 가능한 도서 판매 서비스를 찾아라."와 같은 질의를 수행할 수 없다. 이와 같은 질의는 서비스를 구성하는 프로세스들의 내부적인 동작들을 보아야만 알 수 있다. 그러나 현재 연구들은 서비스 제공자가 정의한 서비스의 기술 정보에 의존하여 검색한다.

<30> 세 번째 문제점은 프로세스의 단순한 개념만을 검색한다는 것이다. 사용자가 "전자 상점을 통해 고객에게 책을 판매하는 프로세스를 찾아라."라고 하였다고 하자. 이 질의는 판매 프로세스에 대해 판매 방법(sell how), 판매 대상(sell to whom), 판매 종류(sell what) 관점에서 정의한 개념들을 이용하여 검색되어야 한다. 즉, 판매에 대해 판매 방법 관점인 전자 상점, 판매 대상(sell to whom) 관점인 고객, 판매 종류(sell what) 관점으로는 책의 개념을 가지는 프로세스를 찾는다. 그러나, 기존의 연구들은 제품에 대해 단순하게 분류한 온톨로지만을 이용하여 검색한다.

발명이 이루고자 하는 기술적 과제

<31> 위와 같은 문제점들을 해결하기 위해 서비스의 내부 정보를 검색할 수 있는 프로세스 정보를 온톨로지로 모델링하고 체계적으로 표현함으로써, 지능적 웹 서비스 발견이 가능하도록 하는 것이 본 발명의 목적이다.

<32> 따라서 본 발명에서는 프로세스 정보를 온톨로지로 모델링하는데 필요한 요소들을 추출하고, 이 온톨로지가 지닌 특성들을 잘 표현할 수 있는 시맨틱 웹 언어인 토픽 맵을 이용하여 표현한 TM-S(Topic Maps for Service) 온톨로지를 제안한다.

발명의 구성 및 작용

<33> 상기의 문제점을 해결하기 위한 본 발명의 프로세스 기반 온톨로지는 온톨로지 언어인 토픽 맵을 이용하며, 프로세스 이름, 입력 이름 또는 출력 이름에 대한 정의를 포함하는 프로세스 시그니처 온톨로지; 프로세스 제어 구조에 대한 정의를 포함하는 프로세스 구조 온톨로지; 프로세스 개념들에 대한 정의, 개념들 간의 관계의 정의, 개념들에 대한 관점의 정의, 개념들 간의 분류 정의 또는 분류에 대한 관점의 정의를 포함하는 프로세스 개념 온톨로지를 포함하는 것을 특징으로 한다.

<34> 이때, 사용되는 기술 용어 및 과학 용어에 있어서 다른 정의가 없다면, 이 발명이 속하는 기술 분야에서 통상의 지식을 가진 자가 통상적으로 이해하고 있는 의미를 가진다.

<35> 또한 다음에 소개되는 실시예들은 당업자에게 본 발명의 사상이 충분히 전달될 수 있도록 하기 위한 예로서 제공되는 것이다. 따라서, 본 발명은 이하 설명되는 실시예들에 한정되지 않고 다른 형태로 구체화될 수도 있다.

<36> 도 3는 본 발명의 프로세스 기반 온톨로지(TM-S 온톨로지)의 개념도이다. 본 발명의 TM-S 온톨로지(100)는 프로세스 시그니처(process signature) 온톨로지(110), 프로세스 구조(process structure) 온톨로지(120), 프로세스 개념(process concept) 온톨로지(130)를 포함한다.

<37> 상기 프로세스 시그니처 온톨로지(110)는 프로세스와 상기 프로세스와 연관되어 있는 요구정보/결과정보인 입력/출력(input/output)에 대한 텍스트 정보를 정의한다. 즉 프로세스 시그니처 온톨로지는 프로세스의 이름과 입력/출력 이름에 대한 정의를 포함한다.

<38> 상기 프로세스 구조 온톨로지(120)는 서비스의 내부 동작(internal behavior)을 기술하기 위한 온톨로지로 순차(sequence), 분리(split), 분리와 결합(split and join), 임의-순서 (and-order), 선택(choice), 조건(if-then-else), 반복(iterate) 같은 프로세스 구조를 표현하기 위해 필요한 제어 구조의 정의를 포함한다.

<39> 상기 프로세스 개념 온톨로지(130)는 프로세스 개념을 다양한 관점에서 정의한 것으로써, 프로세스 개념들에 대한 정의, 개념들 간의 관계의 정의, 개념들에 대한 관점의 정의, 개념들 간의 분류 정의 또는 분류에 대한 관점의 정의를 포함한다.

<40> 본 발명의 TM-S 온톨로지는 토픽 맵(Topic Maps)을 이용하여 표현되는 특징을 갖는다. 온톨로지 언어 중에 토픽 맵은 지식 구조를 기술하고 정보 자원과의 연관 관계를 기술하여 온톨로지를 정의하는 개념적 모델을 제공한다. 이 모델은 2000년 국제표준화기구인 ISO(the International Organization for Standardization) 표준으로 승인되었다. 최근에는 웹 에서 토픽 맵의 교환을 목적으로, XML(eXtensible Markup Language)을 기반으로 Topic Maps의 모델을 정의한 XTM(XML Topic Maps) 명세서를 발표하였다. 토픽 맵는 3가지의 기본요소로 구성되어 있다. 토픽(Topic), 연관관계(Association), 어커런스(Occurrence)이다. 토픽은 사람, 사물, 개념 등과 같이 어떤 대상을 표현한다. 토픽은 한 개 이상의 토픽 타입과 이름을 가진다. 연관관계는 2개 이상의 토픽들 간에 관계를 정의한다. 연관관계에 참여하는 토픽들은 역할을 가진다. 어커런스는 토픽에 관련된 정보 자원을 정의한다.

<41> 이러한 토픽 맵의 개념 모델은 TM-S 온톨로지의 지식 구조를 잘 표현할 수 있다.

<42> 본 발명의 프로세스 시그니처 온톨로지(110)의 프로세스 이름, 입력 이름 또는 출력 이름은 토픽 맵의 토픽으로 정의되며, 프로세스 구조 온톨로지(120)에서 순차, 분리, 분리와 결합, 임의-순서, 선택, 조건 또는 반복을 포함하는 프로세스 제어 구조도 토픽 맵의 토픽으로 정의 된다.

<43> 또한 프로세스 시그니처 온톨로지(110)의 프로세스 이름과 입력/출력 이름간의 관계는 연관관계를 이용하여 정의되며, 프로세스 구조 온톨로지(120)의 프로세스들 간의 제어구조 표현은 연관관계를 이용하여 정의 된다.

<44> 따라서 토픽은 프로세스와 입력/출력, 프로세스의 제어 구조를 표현하며, 연관관계는 프로세스와 입력/출력, 프

로세스의 제어 구조와 구성 요소들 간의 관계를 표현한다.

<45> 본 발명의 프로세스 개념 온톨로지(130)에 사용되는 모든 개념들 또한 토픽 맵의 토픽으로 정의된다. 상기 프로세스 개념 온톨로지(130)에 정의된 개념 간 관계 또는 개념들 간의 다중 관계는 토픽 맵의 연관관계를 이용하여 표현되며, 상기 개념에 대한 관점의 정의 및 상기 분류에 대한 관점의 정의는 토픽 맵의 연관관계의 스코프를 사용하여 정의된다.

<46> 토픽 맵의 연관관계가 가지는 다중 관계와 스코프(Scope)의 특성은 여러 개의 개념들과의 관계를 정의하고, 이 개념들 간의 다양한 관점을 표현하는데 효율적인 표현 능력을 제공한다.

<47> 프로세스 개념 온톨로지의 일 실시예로 판매 개념에 대한 모델을 도 4에 도시하였다. 도 4의 'Sell'(211), 'Sell via store'(212), 'Sell product'(213), 'Sell via physical store'(214), 'Sell via electric store'(215), 'Sell medicine'(216), 'Sell computer'(217) 및 'Sell book'(218)은 판매 개념을 정의하는 토픽들이며, 점선 화살표(230, 231, 232)는 토픽들 간 계층 관계 및 다중 관계를 표현하는 연관관계를 의미한다. 토픽 간의 계층관계는 슈퍼클래스 - 서브클래스 타입의 연관관계(230)를 사용하여 표현되며, 이때, 상위 개념인 토픽 'Sell'(211)은 슈퍼클래스의 역할을 하며, 하위 개념인 토픽 'Sell via store'(212)는 서브클래스의 역할을 한다. 또한 토픽간의 다중 관계를 가지는 'Sell via store'(212), 'Sell via physical store'(214) 및 'Sell via electric store'(215)는 하나의 연관관계(231)를 사용하여 정의된다. 연관관계(232) 또한 토픽 간 다중관계를 정의한 것이다. 연관관계의 스코프를 이용하여 토픽간의 관계에 대한 관점인 'Sell how'(221)과 'Sell what'(222)가 정의된다. 표 1은 프로세스 개념 온톨로지의 일 실시예인 도 4의 판매 개념을 XTM을 이용하여 표현한 것이다.

<48> (표 1) XTM을 이용한 판매 개념 정의

<49>

```
<association>
   <instanceOf>
      <topicRef xlink:href="#superclass-subclass"/>
  </instanceOf>
  <scope>
      <topicRef xlink:href="ontology.xtm#sell how"/>
  </scope>
  <member>
      <roleSpec>
          <topicRef xlink:href="#superclass"/>
      </roleSpec>
          <topicRef xlink:href="#sell"/>
  </member>
  <member>
      <roleSpec>
          <topicRef xlink:href="#subclass"/>
      </roleSpec>
          <topicRef xlink:href="#sell via store"/>
  </member>
</association>
<association>
   <instanceOf>
      <topicRef xlink:href="#superclass-subclass"/>
  </instanceOf>
  <member>
      <roleSpec>
          <topicRef xlink:href="#superclass"/>
      </roleSpec>
          <topicRef xlink:href="#sell via store"/>
  </member>
  <member>
      <roleSpec>
          <topicRef xlink:href="#subclass"/>
      </roleSpec>
          <topicRef xlink:href="#sell via physical store"/>
  </member>
<member>
      <roleSpec>
          <topicRef xlink:href="#subclass"/>
      </roleSpec>
          <topicRef xlink:href="#sell via electronic store"/>
  </member>
</association>
```

<50> 웹 서비스를 구성하는 모든 프로세스는 상기 프로세스 개념 온톨로지(130)에 정의된 한 개 이상의 개념들을 이용하여 프로세스의 의미를 표현할 수 있다. 프로세스는 토픽으로 표현되므로, 상기 프로세스 개념 온톨로지(130)에 정의된 한 개 이상의 개념들을 토픽의 인스턴스(instance)로 표현함으로써, 토픽으로 표현된 프로세스의 의미를 표현할 수 있다.

<51> 일예로 도 5a에 도서 판매 서비스인 도서 판매 프로세스(300)를 도시하였다. 도 5a에 도시된 구들은 각각 하나의 프로세스인 원자 프로세스를 의미한다. 도서 판매 서비스는 복합 프로세스인 도서 판매 프로세스(300)로 표

현되고, 프로세스(310)는 지불 프로세스로 복합 프로세스이며, 프로세스(311)는 신용 카드 지불 프로세스로 원자 프로세스이다. 도 5a에서 알 수 있듯이 도서 판매 서비스는 여러 원자 프로세스와 복합 프로세스로 구성된 복합 프로세스이다. 도 5a에 도시된 도서 판매 프로세스(300)와 도서 판매 프로세스를 구성하는 하부 프로세스들은 프로세스 개념 온톨로지(130)에 정의된 한 개 이상의 개념들을 이용하여 각각의 프로세스의 의미가 표현된다. 일 실시예로 도 5a의 도서 판매 프로세스(300)와 지불에 관련된 복합 프로세스(310)를 구성하는 신용 카드 지불 프로세스인 원자 프로세스(311)의 의미를 표현 할 수 있는 도서 판매에 관련된 프로세스 개념 온톨로지를 도 5b에 도시하였다. 도 5b 또한 도 4의 일예와 마찬가지로 사각형은 토픽을 의미하며, 타원은 연관관계의 스코프를 의미하고, 화살표는 토픽 간 연관관계를 의미한다. 도서 판매 프로세스(300)는 판매에 관한 것이므로 토픽 'Sell'을 이용한다. 도서 판매 프로세스(300)의 의미는 도 5b의 토픽'Sell via electronic store'과 토픽 'Sell book'를 이용하여 표현되며, 카드 결제 프로세스인 원자 프로세스(311)의 의미는 토픽 'Pay'를 이용하며 토픽'Pay using credit card'와 토픽 'Pay for inputs'를 이용하여 표현된다. 표 2은 XTM을 이용하여 토픽 'SellBookProcess'의 의미를 표현한 일 실시예이다.

<52> (표 2)

<53>

<54>

```
<topic id="SellBookProcess">
   <instanceOf>
      <topicRef xlink:href="Ontology.xtm#SellviaElectronicStore"/>
  </instanceOf>
  <instanceOf>
      <topicRef xlink:href="Ontology.xtm#SellComputerBook"/>
  </instanceOf>
     …
<baseName>
      <baseNameString>Sell Book Process</baseNameString>
   </baseName>
 </topic>
```

<55> 본 발명의 TM-S 온톨로지(100)는 효율적인 온톨로지를 구축하기 위해, 이미 구축된 온톨로지들은 활용하고 추가가 필요한 요소들은 새로이 온톨로지를 구축한다.

<56> 프로세스 시그니처 온톨로지(110)와 프로세스 구조 온톨로지(120)에서 대부분의 구성 요소들은 기존의 OWL-S 프로세스 모델과 SWSO의 FLOWS 프로세스 모델에 정의되어 있다. SWSO의 FLOWS는 OWL-S를 기반으로 확장한 것이다. 현재, OWL-S는 다른 표준들보다 완성된 온톨로지를 제공하나, SWSO의 FLOWS는 아직 개발 초기 단계에 있다. 그러므로, 본 발명에서는 OWL-S에서 제공하는 온톨로지를 기반으로 모델링을 하였다.

<57> 프로세스 시그니처 온톨로지(110)의 프로세스 이름은 OWL-S 프로세스 모델에 정의된 프로세스 이름과 매핑이 된다. 프로세스 구조 온톨로지(120)의 제어 구조 정보는 OWL-S 프로세스 모델의 7가지 제어 구조 형태와 매핑된다. 그러나, OWL-S 프로세스 모델에 정의되어 있지 않은 입력/출력 이름은 새로이 추가된다.

<58> 프로세스 개념 온톨로지(130)에 대한 정의는 Process Handbook 프로젝트의 형태 구조를 이용하여 모델링 된다. 즉, 개념간의 계층적인 분류는 Process Hand-book의 인덱스 구조와 같은 형태로 모델링되며, 분류에 대한 관점은 관련된 항목들을 그룹화하는 번들과 같은 형태로 모델링 될 수 있다.

<59> 상세하게는 본 발명의 프로세스 시그니처 온톨로지(110) 또는 상기 프로세스 구조(120) 온톨로지는 OWL-S에 정의된 온톨로지를 토픽 맵을 이용하여 변환시켜 구축될 수 있으며, 하기의 규칙을 이용하여 변환된다.

<60> 규칙 1 : OWL-S에 정의된 온톨로지에 사용되는 프로세스를 기술하기 위해 필요한 모든 클래스들은 토픽 맵의 토픽으로 정의됨.

<61> 규칙 2 : 특정 클래스의 이름은 그 개념을 정의하는 토픽의 베이스이름으로 정의됨.

<62> 규칙 3 : 클래스들 간의 관계, 관계추론, 클래스의 속성 제약은 토픽맵에서 토픽들의 연관관계, 토픽의 역할 또

는 스코프, 또는 이들의 조합을 이용하여 정의됨.

<63> 규칙 4 : 합집합, 여집합, 교집합, 열거 또는 분리를 포함하는 집합 연산자는 토픽의 역할과 연관관계를 이용하여 정의됨.

<64> 규칙 5 : 역관계를 포함한 속성에 대한 추론은 토픽들의 연관관계, 스코프, 베이스이름을 이용하여 정의됨.

<65> 상기의 규칙들을 이용하여 웹 서비스를 위한 온톨로지 웹 언어(OWL-based web Service, 이하 OWL-S)의 프로세스 모델 온톨로지를 TM-S 온톨로지로 변환시키는 방법을 도 6에 도시하였다.

<66> OWL-S 프로세스 모델 온톨로지(410)를 본 발명의 TM-S 온톨로지로 변환하기 위해서는 OWL-S 프로세스 모델 온톨로지(410)에 정의된 정보들을 동등한 의미를 가지도록 TM-S 온톨로지로 변환시켜야 한다.

<67> OWL-S 프로세스 모델 온톨로지(410)에 정의된 정보들은 클래스에 해당하는 프로세스 정의, 데이터 타입 속성에 해당하는 프로세스 이름, 클래스에 해당하는 입력/출력 정의, 객체 속성인 프로세스와 입력간의 관계, 추론 속성인 입력과 파라미터의 관계 정의, 원자 프로세스와 복합 프로세스간의 관계정의, 클래스에 해당하는 제어 구조 정의, 집합연산자에 해당하는 제어 구조의 종류, 온톨로지 매핑에 해당하는 제어 구조의 재정의, 관계 제약에 해당하는 Sequence, Split 등의 구성요소가 있다.

<68> 상기 프로세스 정의와 상기 입력/출력 정의는 규칙 1을 이용하여 프로세스 시그니처 온톨로지(420)로 변환되고 상기 프로세스 이름은 규칙 2를 이용하여 프로세스 시그니처 온톨로지(420)로 변환된다. OWL-S 프로세스 모델 온톨로지(410)에서 클래스나 객체간의 관계는 속성으로 표현된다. 상기 속성은 연관관계를 이용하여 동일하게 변환될 수 있으므로, 상기 프로세스와 입력간의 관계와 상기 입력과 파라미터의 관계 정의는 규칙 3을 이용하여 프로세스 시그니처 온톨로지(420)로 변환된다. 상기 속성을 연관관계를 이용하여 변환시키는 일 실시예로 OWL-S 프로세스 모델에서 'Process'가 process를 나타내는 변수이고 'Input'이 파라미터를 나타내는 변수이고, 상기 'Process'가 상기 'Input'을 가질 수 있다는 속성은 규칙 3을 사용하여 hasInput(process:Process,input:Input)으로 변환된다. 상기 원자프로세스와 복합프로세스간의 관계 정의는 규칙 3을 이용하여 프로세스 구조 온톨로지(430)로 변환되고, 상기 제어 구조의 정의는 규칙 1을 이용하여 프로세스 구조 온톨로지(430)로 변환된다. 상기 제어 구조의 종류는 집합연산자에 해당하므로 규칙 4를 이용하여 프로세스 구조 온톨로지(430)로 변환된다. 상기 제어 구조의 재정의 와 상기 Sequence, Split등의 구성 요소는 규칙 3을 이용하여 프로세스 구조 온톨로지(430)로 변환된다.

<69> 표 3는 XTM을 이용하여 'SellBookProcess'라는 프로세스를 정의하고, 프로세스의 이름을 기술한 규칙 1과 규칙 2의 일 실시예이다.

<70> (표 3) XTM을 이용한 'SellBookProcess' 프로세스의 정의

<71>
```
<topic id="SellBookProcess">
  <instanceOf>
      <topicRef xlink:href="Process.xtm#CompositeProcess"/>
  </instanceOf>
  …
  <baseName>
<baseNameString>도서판매프로세스</baseNameString>
  </baseName>
  <occurrence>
<instanceOfc>
        <topicRef xlink:href="#Description"/>
</instanceOf>
   <resourceData> This process is the case of selling book.</resourceData>
  </occurrence>
</topic>
```

<72> 하기의 실시예들은 규칙 1 내지 규칙 4를 이용하여 OWL-S 프로세스 모델 온톨로지를 본 발명의 TM-S 온톨로지로 변환하는 실시예들이다.

<73> 규칙 3의 일 실시예로 OWL-S의'Process' 와 'Input'간의 관계정의를 표 4에 나타내었으며, 이를 본 발명의 TM-S

온톨로지로 변환시킨 것을 표 4' 에 나타내었다. 토픽들의 관계는 P(Rx:X,Ry:Y)로 표현되는데, X 및 Y는 연관관계에 참여하는 토픽들이며, P는 토픽 X와 토픽 Y의 연관관계이며, Rx는 연관관계에서 토픽 X의 역할을 나타내며, Ry는 연관관계에서 토픽 Y의 역할을 나타낸다. 표 4'의 실시예에서 P는 'hasinput'이며, 관련 토픽들은 'Process' 및 'Input'이며, 관련 토픽들의 역할은 각각 'process' 와 'input'이 된다.

<74> (표 4)OWL-S의 프로세스와 input간의 관계 정의

<75>

```
<owl:ObjectProperty rdf:ID="hasInput">
  …
  <rdfs:domain rdf:resource="#Process" />
 <rdfs:range rdf:resource="#Input" />
</owl:ObjectProperty>
```

<76> (표 4') TM-S에서 프로세스와 input간의 관계 정의

<77>

```
<association>
  <instanceOf>
      <topicRef xlink:href="Process.xtm#hasInput"/>
  </instanceOf>
  <member>
<roleSpec>
         <topicRef xlink:href="#process"/>
</roleSpec>
      <topicRef xlink:href="#Process"/>
  </member>
  <member>
<roleSpec>
        <topicRef xlink:href="#input"/>
</roleSpec>
        <topicRef xlink:href="#Input"/>
  </member>
</association>
```

<78> 규칙 3의 또 다른 일 실시예로 OWL-S의 파라미터와 입력간의 관계 정의를 표 5에 나타내었으며, 이를 본 발명의 TM-S 온톨로지로 변환시킨 것을 표 5' 에 나타내었다.

<79> 토픽들 간 관계에는 추론이 포함되며, 상기의 추론 또한 토픽 맵의 연관관계를 이용하여 정의된다. 토픽들 간의 상하위 관계를 추론하기 위해 슈퍼클래스 - 서브클래스(superclass - subclass) 또는 슈퍼프로퍼티 - 서브프로퍼티(superproperty - subproperty)의 연관관계를 이용하여 토픽 간의 계층구조를 표현한다. 상기의 연관관계를 이용하여 토픽 X가 Y와 P의 관계를 가지면서 토픽 Y가 Z와 P의 관계를 가지면, 토픽 X와 Z는 P의 관계를 가지는 것으로 추론할 수 있으며 $P(Rx:X,Ry:Y) \wedge P(Ry:Y,Rz:Z) \Rightarrow P(Rx:X,Rz:Z)$로 표현된다. 표 5'은 파라미터(Parameter)와 입력(Input)간의 계층구조에 의한 추론의 일 실시예이다. 상기의 실시예에서 수퍼클래스 - 서브클래스 연관관계를 이용하여 파라미터는 변수(Variable)의 하위 클래스(subclass)라는 계층구조와 입력은 파라미터의 하위 클래스라는 계층구조를 표현한다. 따라서 입력은 변수의 하위 클래스에 속함을 추론할 수 있다.

<80> (표 5) OWL-S의 파라미터와 입력간의 관계 정의

<81>

```
<owl:Class rdf:about="#Parameter">
 <rdfs:subClassOf rdf:resource="&swrl;#Variable"/>
</owl:Class>
<owl:Class rdf:ID="Input">
 <rdfs:subClassOf rdf:resource="#Parameter"/>
</owl:Class>
```

<82> (표 5') TM-S에서 파라미터와 입력간의 관계 정의

<83>

```
<association>
  <instanceOf>
      <topicRef xlink:href= "Process.xtm#superclass-subclass"/>
  </instanceOf>
  <member>
      <roleSpec>
         <topicRef xlink:href="#superclass"/>
      </roleSpec>
         <topicRef xlink:href="#Variable"/>
  </member>
  <member>
      <roleSpec>
         <topicRef xlink:href="#subclass"/>
      </roleSpec>
         <topicRef xlink:href="#Parameter"/>
  </member>
</association>
<association>
  <instanceOf>
      <topicRef xlink:href="Process.xtm#superclass-subclass"/>
  </instanceOf>
  <member>
      <roleSpec>
         <topicRef xlink:href="#superclass"/>
      </roleSpec>
         <topicRef xlink:href="#Parameter"/>
  </member>
  <member>
      <roleSpec>
         <topicRef xlink:href="#subclass"/>
      </roleSpec>
         <topicRef xlink:href="#Input"/>
  </member>
</association>
```

<84> 규칙 3의 또 다른 일 실시예로 OWL-S의 프로파일과 프로세스의 관계 정의를 표 6에 나타내었으며, 이를 본 발명의 TM-S 온톨로지로 변환시킨 것을 표 6' 에 나타내었다. 토픽 간의 추론에 필요한 또 다른 속성으로 기능적 속성(FunctionalProperty), 역기능적 속성(InverseFunctionalProperty), 대칭적 속성(SymmetricProperty)가 있다. 기능적 속성은 단일한 값을 가지는 속성을 정의하는 것으로 토픽 맵의 스키마(schema)언어로 표현되며,

역기능적 속성 또한 토픽 맵의 스키마 언어로 표현된다. 대칭적 속성은 토픽 간 관계의 대칭성을 정의한 것으로, 토픽 X에서 토픽 Y로 P의 관계를 가지면 토픽 Y에서 토픽 X로 P의 관계를 가짐을 의미하며 P(Rx:X,Ry:Y) ⇒ P(Ry:Y,Rx:X)로 표현된다. 표 6'의 실시예에서 토픽'Profile'과 토픽 'Process'는 'has_process'의 연관관계를 가지며 'min'과 'max'를 이용하여 값을 제한한다.

<85> (표 6) OWL-S의 프로파일과 프로세스의 관계 정의

<86>

```
<owl:ObjectProperty rdf:ID="has_process">
  <rdfs:domain rdf:resource="#Profile" />
 <rdfs:range rdf:resource="#Process" />
</owl:ObjectProperty>
<owl:FunctionalProperty rdf:about ="#has_process"/>
```

<87> (표 6') TM-S에서 프로파일과 프로세스의 관계 정의

<88>

```
<association>
  <instanceOf>
      <topicRef xlink:href="Process.xtm#has_process"/>
  </instanceOf>
  <role  min="1" max="inf">
    <instanceOf>
      <internalTopicRef  href="#profile"/>
    </instanceOf>
   <player>
      <internalTopicRef  href="#Profile"/>
   </player>
  </role>
<role  min="1"  max="1" >
 <instanceOf>
      <internalTopicRef  href="#process"/>
    </instanceOf>
   <player>
      <internalTopicRef  href="#Process"/>
   </player>
  </role>
</association>
```

<89>

<90> 규칙 3의 또 다른 일 실시예로 OWL-S의 순차 구조에 대한 구성 요소 정의를 표 7에 나타내었으며, 이를 본 발명의 TM-S 온톨로지로 변환시킨 것을 표 7' 에 나타내었다. 순차 구조에 대한 구성 요소 정의 토픽에 대한 제약 또한 토픽 맵의 연관관계를 이용하여 정의된다. 상술한 바와 같이, 토픽들 간의 연관관계는 P(Rx:X,Ry:Y)로 표현되는데, P는 연관관계 타입이며, X 및 Y는 연관관계에 참여하는 토픽들이며, Rx는 연관관계에서 토픽 x의 역할을 나타내며, Ry는 연관관계에서 토픽 Y의 역할을 나타낸다. 토픽의 특정 개념에 대한 제약은 연관관계에 참여하는 토픽 Y와 역할인 Ry를 이용하여 표현된다. 토픽 Y 대신 Y의 값인 C를 정의하고, 역할인 Ry에는 제약 사항을 정의한다. 표 7'는 토픽 'Sequence'가 'ControlConstructList'를 구성요소로 가지는며, P는 'Component'이고, X는 'Sequence'이며, C는 'ControlConstructList'이다.

<91> (표 7) OWL-S의 순차 구조에 대한 구성 요소 정의

<92>

```
<owl:Class rdf:ID="Sequence">
 …
  <owl:Restriction>
 <owl:onProperty rdf:resource="#components" />
 <owl:allValuesFrom rdf:resource="#ControlConstructList" />
 </owl:Restriction>
 …
</owl:Class>
```

<93> (표 7')TM-S에서 순차 구조에 대한 구성 요소 정의

<94>

```
<association>
  <instanceOf>
    <topicRef xlink:href="Process.xtm#components"/>
  </instanceOf>
  <member>
<roleSpec>
    <topicRef xlink:href="#structure"/>
    </roleSpec>
    <topicRef xlink:href="#Sequence"/>
  </member>
  <member>
<roleSpec>
    <topicRef xlink:href="#allValuesFrom"/>
    </roleSpec>
    <topicRef xlink:href="#ControlConstructList"/>
  </member>
</association>
```

<95> 규칙 4의 일 실시예로 OWL-S의 제어 구조의 구성 요소 정의를 표 8에 나타내었으며, 이를 본 발명의 TM-S 온톨로지로 변환시킨 것을 표 8' 에 나타내었다.

<96> 집합 연산자는 이미 정의된 토픽들을 이용하여 새로운 토픽을 만드는 역할을 한다. 이는 객체지향 모델에서 기정의된 클래스를 이용하여 새로운 클래스를 만드는 것과 동일하다. 집합 연산자는 합집합, 여집합, 교집합, 열거 또는 분리를 포함하는 연산자로 상기 집합 연산자는 토픽 맵의 토픽과 연관관계를 이용하여 정의된다. 토픽 맵은 다중 관계에 대한 정의가 가능하므로 하나 이상의 토픽들이 연관관계에 참여할 수 있다. 또한 연관관계에 참여하는 토픽들이 서로 다른 역할을 가지고 연관관계에 참여할 수 있다. 따라서 집합 연산자에서 구성요소의 역할을 하는 토픽 Y_1, Y_2, Y_3, … , Y_n을 연관관계에 참여하는 토픽들로 정의하고, 연관관계를 이용하여 집합 연산자를 정의한다. 표 8'에 정의한 바와 같이 'ControlConstruct'토픽이 'Sequence', 'Split', 'Split-Join', 'Any-Order'의 토픽으로 구성될 때, UnionOf(controlconstruct:ControlConstruct,component:Sequence,component: Split, component:Split-Join, component:Any-Order)로 표현된다.

<97> (표 8) OWL-S의 제어 구조의 구성 요소 정의

<98>

```
<owl:Class rdf:ID="ControlConstruct" />
 <owl:ObjectProperty rdf:ID="components">
  <rdfs:domain>
 <owl:Class>
 <owl:unionOf rdf:parseType="Collection">
 <owl:Class rdf:about="#Sequence" />
 <owl:Class rdf:about="#Split" />
 <owl:Class rdf:about="#Split-Join" />
 <owl:Class rdf:about="#Any-Order" />
 <owl:Class rdf:about="#Choice" />
 </owl:unionOf>
 </owl:Class>
 </rdfs:domain>
 </owl:ObjectProperty>
</owl:Class>
```

<99> (표 8') TM-S에서 제어 구조의 구성 요소 정의

<100>

```
<association>
   <instanceOf>
      <topicRef xlink:href="#unionOf"/>
  </instanceOf>
  <member>
<roleSpec>
<topicRef xlink:href="#ControlConstruct"/>
     </roleSpec>
     <topicRef xlink:href="# ControlConstruct "/>
  </member>
  <member>
<roleSpec>
<topicRef xlink:href="#component"/>
    </roleSpec>
    <topicRef xlink:href="#Sequence"/>
  </member>
<member>
<roleSpec>
<topicRef xlink:href="#component"/>
    </roleSpec>
    <topicRef xlink:href="#Split"/>
  </member>
….
</association>
```

<101> 규칙 5의 일 실시예로 속성에 대한 추론(표 9)을 본 발명의 TM-S 온톨로지로 변환시킨 것을 표9' 에 나타내었다.

<102> OWL에서 inverseOf 속성은 속성에 대한 추론을 제공하는 방법 중에 하나이며, 속성의 역을 정의한다. 만일 P2가 P1에 대한 역 관계일 경우, 모든 X, Y에 대해 P1(X, Y) 이면 P2(Y, X) 가 성립되는 것을 추론할 수 있다. OWL에

서는 단 방향 관계 정의만 가능하므로, 표 9와 같이 inverseOf를 이용하여 P1에 대한 역인 P2를 정의해야 한다.

<103> 토픽 맵은 다방향 관계에 대한 정의가 가능하다. 이것은 연관관계에 참여하는 토픽의 역할을 이용하여 정의한다. P1(Rx:X, Ry:Y)가 Rx에서 Ry로의 관계를 가진다면, 역관계는 Ry에서 Rx로의 관계를 의미한다. 관계 명칭은 연관관계 P1을 정의한 토픽 P1의 베이스이름에 정의한다. 토픽 P1의 <scope>이 Rx인 <basena me>에 P1을 정의한다.Topic P1의 <scope>이 Ry인 <basename>에 P2를 정의한다.

<104> (표 9) OWL-S에서 단순 프로세스와 복합 프로세스의 관계 표현

<105>
```
<owl:ObjectProperty rdf:ID="collapsesTo">
<rdfs:domain rdf:resource="#CompositeProcess" />
 <rdfs:range rdf:resource="#SimpleProcess" />
 <owl:inverseOf rdf:resource="#expandesTo" />
 </owl:ObjectProperty>
```

<106> (표 9') TM-S에서 TM-S에서 단순 프로세스와 복합 프로세스의 관계 표현

<107>
```
<topic id="collapsesTo">
  …..
  <baseName>
      <scope>
        <topicRef xlink:href="#SimpleProcess" />
      </scope>
   <baseNameString>collapsesTo</baseNameString>
   </baseName>
   <baseName>
     <scope>
       <topicRef xlink:href="#CompositeProcess" />
     </scope>
       <baseNameString>expandesTo</baseNameString>
     </baseName>
</topic>
<association>
   <instanceOf>
     <topicRef xlink:href="#collapsesTo"/>
  </instanceOf>
  <member>
<roleSpec>
    <topicRef xlink:href="#simpleProcess"/>
    </roleSpec>
    <topicRef xlink:href="#SimpleProcess"/>
  </member>
  <member>
<roleSpec>
    <topicRef xlink:href="#compositeProcess"/>
    </roleSpec>
    <topicRef xlink:href="#CompositeProcess"/>
  </member>
</association>
```

<108> 본 발명에 대한 상기의 설명과 실시예들을 통해 서비스의 외부 정보만을 사용하여 서비스를 검색하는 문제점을 프로세스 이름과 상기 프로세스와 연관된 입력/출력 이름에 대한 정의를 포함하는 프로세스 시그니처 온톨로지를 사용하여 해결할 수 있으며, 프로세스 제어 구조에 대한 정의를 포함하는 프로세스 구조 온톨로지를 이용하여 서비스의 내부 구조인 수행 과정에 대한 상세한 정보를 얻을 수 있으며, 프로세스의 단순 개념만을 검색하던

기존의 한계를 프로세스 개념 온톨로지를 이용하여 해결 할 수 있다.

발명의 효과

<109> 이상에서 설명한 바와 같이, 본 발명에 의한 프로세스 정보를 표현한 온톨로지는 서비스를 구성하는 프로세스들의 구조 및 개념들을 제공함으로써 지능적 웹 서비스 발견에서 요구되는 서비스 발견의 정확률과 재현률을 높이는 효과를 보여주고 있다. 기존의 웹 서비스 발견에 있어, 서비스의 외부 정보만을 사용하여 서비스를 검색하는 한계를 프로세스 시그니처 온톨로지를 사용하여 해결할 수 있으며, 프로세스 구조 온톨로지를 이용하여 서비스의 내부 구조인 수행 과정에 대한 웹 서비스 발견이 가능해지며, 프로세스 개념 온톨로지를 이용하여 웹 서비스의 요구 시 개인적으로 정확한 의미 전달을 가능하게 하는 풍부하고 복잡한 표현이 사용 될 수 있다. 그리고, 시맨틱 웹 언어인 토픽 맵은 프로세스 정보에 대한 온톨로지 지식을 체계적으로 표현과 구축이 가능하도록 하고 있다.

<110> 본 발명은 지능적 웹 서비스를 위한 온톨로지 모델링 연구에 활용될 수 있다. 또한, 서비스의 자동화를 위한 발견 시스템은 물론, 서비스의 조합, 통합 및 상호 운용을 위한 시스템에 기본이 되는 연구로 활용될 수 있다.

도면의 간단한 설명

<1> 도 1은 OWL-S의 온톨로지 구조이며,

<2> 도 2는 OWL-S의 온톨로지에서 도서 판매 서비스의 능력을 도시한 예이며,

<3> 도 3은 본 발명의 프로세스 기반 온톨로지(TM-S 온톨로지)의 개념도이며,

<4> 도 4는 프로세스 개념 온톨로지의 일 실시예로 판매 개념에 대한 개념 온톨로지의 일 예이며,

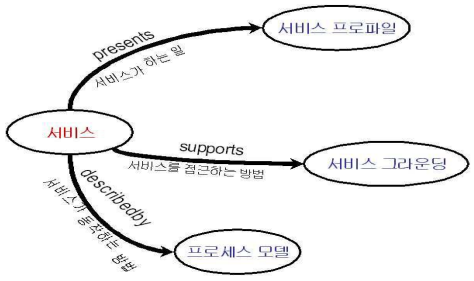
<5> 도 5a는 도서 판매 서비스인 도서 판매 프로세스를 도시한 일 예이며,

<6> 도 5b는 도서 판매에 관련된 프로세스 개념 온톨로지의 일 예이며,

<7> 도 6은 OWL-S 프로세스 모델 온톨로지를 본 발명의 TM-S 온톨로지로 변환하는 일 예이다.

도면

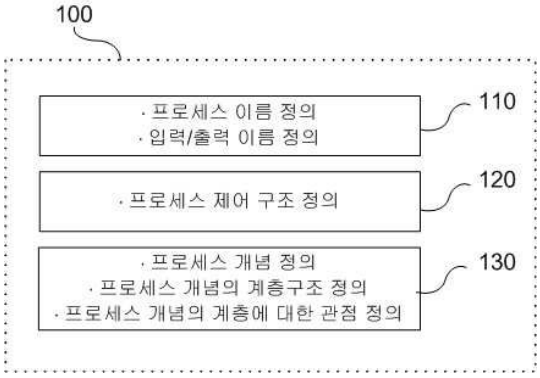
도면1



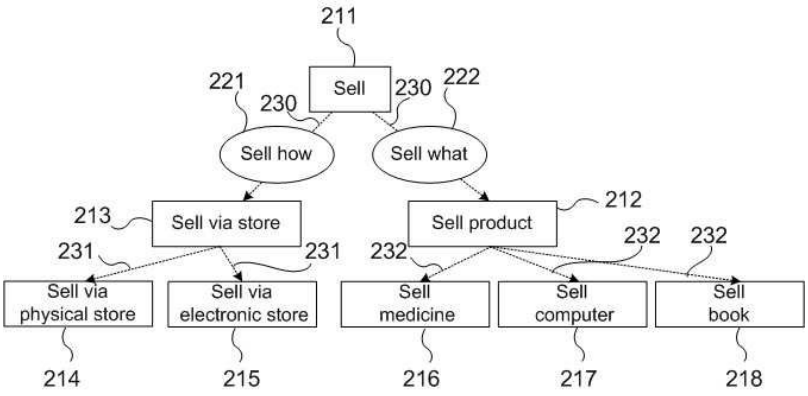
도면2



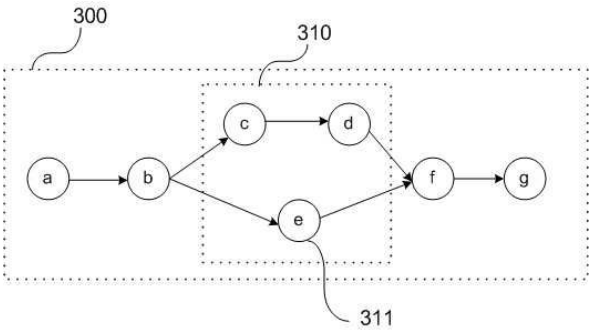
도면3


100
· 프로세스 이름 정의
· 입력/출력 이름 정의
110
· 프로세스 제어 구조 정의
120
· 프로세스 개념 정의
· 프로세스 개념의 계층구조 정의
· 프로세스 개념의 계층에 대한 관점 정의
130

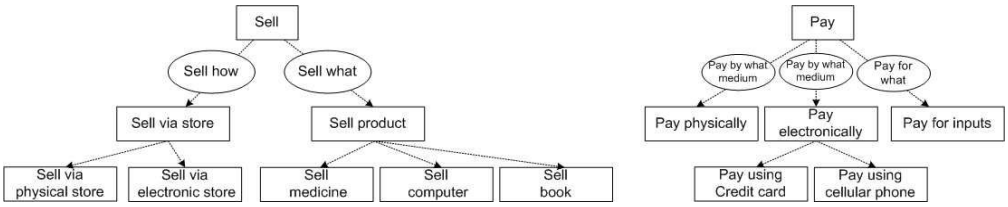
도면4


211
Sell
221
230
230
222
Sell how
Sell what
213
Sell via store
Sell product
212
231
231
232
232
232
Sell via physical store
Sell via electronic store
Sell medicine
Sell computer
Sell book
214
215
216
217
218

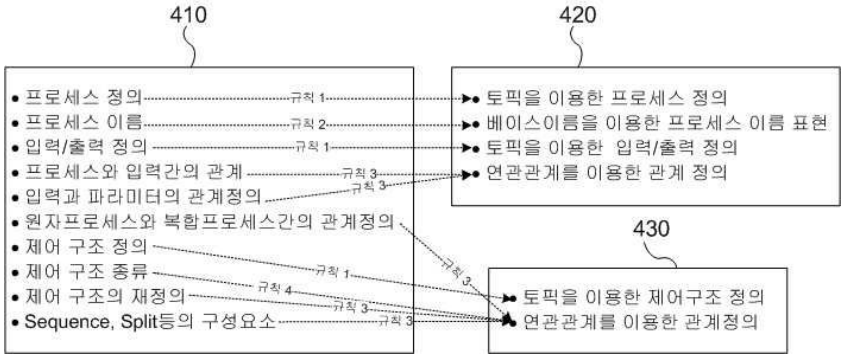
도면5a


300
310
a
b
c
d
e
f
g
311

도면5b


Sell
Sell how
Sell what
Sell via store
Sell product
Sell via physical store
Sell via electronic store
Sell medicine
Sell computer
Sell book
Pay
Pay by what medium
Pay by what medium
Pay for what
Pay physically
Pay electronically
Pay for inputs
Pay using Credit card
Pay using cellular phone

도면6


410
420
430
• 프로세스 정의
• 프로세스 이름
• 입력/출력 정의
• 프로세스와 입력간의 관계
• 입력과 파라미터의 관계정의
• 원자프로세스와 복합프로세스간의 관계정의
• 제어 구조 정의
• 제어 구조 종류
• 제어 구조의 재정의
• Sequence, Split등의 구성요소
규칙 1
규칙 2
규칙 1
규칙 3
규칙 3
규칙 3
규칙 1
규칙 4
규칙 3
규칙 3
• 토픽을 이용한 프로세스 정의
• 베이스이름을 이용한 프로세스 이름 표현
• 토픽을 이용한 입력/출력 정의
• 연관관계를 이용한 관계 정의
• 토픽을 이용한 제어구조 정의
• 연관관계를 이용한 관계정의