경영정보학연구
제18권 제1호
2008년 3월

유사도 알고리즘을 활용한 시맨틱 프로세스 검색방안*

이 홍 주**, Mark Klein***

Semantic Process Retrieval with Similarity Algorithms

Hong Joo Lee, Mark Klein

One of the roles of the Semantic Web services is to execute dynamic intra-organizational services including the integration and interoperation of business processes. Since different organizations design their processes differently, the retrieval of similar semantic business processes is necessary in order to support inter-organizational collaborations. Most approaches for finding services that have certain features and support certain business processes have relied on some type of logical reasoning and exact matching.

This paper presents our approach of using imprecise matching for expanding results from an exact matching engine to query the OWL(Web Ontology Language) MIT Process Handbook. MIT Process Handbook is an electronic repository of best-practice business processes. The Handbook is intended to help people: (1) redesigning organizational processes, (2) inventing new processes, and (3) sharing ideas about organizational practices.

In order to use the MIT Process Handbook for process retrieval experiments, we had to export it into an OWL-based format. We model the Process Handbook meta-model in OWL and export the processes in the Handbook as instances of the meta-model. Next, we need to find a sizable number of queries and their corresponding correct answers in the Process Handbook. Many previous studies devised artificial dataset composed of randomly generated numbers without real meaning and used subjective ratings for correct answers and similarity values between processes. To generate a semantic-preserving test data set, we create

* 본 논문은 2006년정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구이다 (KRF-2006-214-D00193).
저자들은 Department of Informatics, University of Zurich의 Christoph Kiefer와 Abraham Bernstein의 조언과 실험에 도움주신 것에 감사드립니다.
** 교신저자, 가톨릭대학교 경영학부
*** Principal Research Scientist at the MIT Center for Collective Intelligence

20 variants for each target process that are syntactically different but semantically equivalent using mutation operators. These variants represent the correct answers of the target process.

We devise diverse similarity algorithms based on values of process attributes and structures of business processes. We use simple similarity algorithms for text retrieval such as TF-IDF and Levenshtein edit distance to devise our approaches, and utilize tree edit distance measure because semantic processes are appeared to have a graph structure. Also, we design similarity algorithms considering similarity of process structure such as part process, goal, and exception. Since we can identify relationships between semantic process and its subcomponents, this information can be utilized for calculating similarities between processes. Dice's coefficient and Jaccard similarity measures are utilized to calculate portion of overlaps between processes in diverse ways.

We perform retrieval experiments to compare the performance of the devised similarity algorithms. We measure the retrieval performance in terms of precision, recall and F measure? the harmonic mean of precision and recall. The tree edit distance shows the poorest performance in terms of all measures. TF-IDF and the method incorporating TF-IDF measure and Levenshtein edit distance show better performances than other devised methods. These two measures are focused on similarity between name and descriptions of process.

In addition, we calculate rank correlation coefficient, Kendall's tau b, between the number of process mutations and ranking of similarity values among the mutation sets. In this experiment, similarity measures based on process structure, such as Dice's, Jaccard, and derivatives of these measures, show greater coefficient than measures based on values of process attributes. However, the Lev-TFIDF-JaccardAll measure considering process structure and attributes' values together shows reasonably better performances in these two experiments. For retrieving semantic process, we can think that it's better to consider diverse aspects of process similarity such as process structure and values of process attributes.

We generate semantic process data and its dataset for retrieval experiment from MIT Process Handbook repository. We suggest imprecise query algorithms that expand retrieval results from exact matching engine such as SPARQL, and compare the retrieval performances of the similarity algorithms. For the limitations and future work, we need to perform experiments with other dataset from other domain. And, since there are many similarity values from diverse measures, we may find better ways to identify relevant processes by applying these values simultaneously.

Keywords : Semantic Business Process, Process Retrieval, Similarity, Semantic Web

Ⅰ. 서 론

데이터와 서비스가 의미를 포함하게 되어 조직간 응용프로그램의 연계와 자동화를 활성화할 수 있을 것이라는 시맨틱 웹의 목표를 이루기 위해 학계와 산업계에서 많은 노력과 연구들이 이루어져 왔다. 김학래와 김홍기(2003)은 시맨틱 웹을 "컴퓨터가 정보의 의미를 이해하고 처리할 수 있는 웹"으로 정의하고 있으며, 웹 자원에 대한 메타정보를 기계가 읽고 처리함으로써 웹에 있는 방대한 정보와 지식을 연결하고 활용할 수 있게 되는 것을 뜻한다[Davies *et al*., 2003; 김학래와 김홍기, 2003]. 웹 자원의 메타정보를 표현하기 위한 표준 시맨틱 웹 언어들이 고안되어 왔으며, Resource Description Framework(RDF), Web Ontology Language(OWL)가 활용되고 있다. 시맨틱 웹의 목표를 이루기 위한 어려운 점들도 논의되고 있으나[McCool, 2005], 데이터의 표현에 있어서 시맨틱 웹 기술이 많이 활용되고 있다. 조직내외의 비즈니스 프로세스를 연계하고 상호연동하기 위한 시맨틱 비즈니스 프로세스에 대한 연구도 수행되어 왔다[Ehrig *et al*., 2007; 김학래와 김홍기, 2002; 김형도와 김종우, 2006]. 시맨틱 프로세스는 비즈니스 프로세스를 시맨틱 웹 표준 언어로 표현한 것을 뜻하며, 이로 인해 모호하지 않은 형태로 프로세스가 표현되기 때문에 컴퓨터 추론이 가능하고 프로세스 구성자동화가 가능해진다[Ehrig *et al*., 2007]. 그러나, 조직들이 비즈니스 프로세스를 다양한 방식으로 표현하고 있기 때문에 조직간 협업을 활성화하기 위해서는 연계가능한 유사한 시맨틱 비즈니스 프로세스를 찾는 것이 필요하다.

시맨틱 웹의 발전과 함께 다양한 시맨틱 웹 자원(resource)에서 적합한 정보를 찾기 위한 노력들이 진행되어왔다. 시맨틱 웹 자원이 포함하고 있는 정보를 바탕으로 사용자의 질의어와 매칭되는 자원을 찾아주기 위한 노력들이 있었으며, RDF 기반의 자원을 검색하기 위한 질의 언어로 RQL, RDQL, SPARQL 등이 제시되어 왔다[Haase *et al*., 2004]. 위의 방안들은 URI나 RDF 문장에 포함되어 있는 정보와 RDF 하위그래프 정보를 추출하여 질의어와 매칭되는 정보를 찾는 정확한(exact) 매칭 방안이나 추론방안을 활용하고 있다[Klusch *et al*., 2006]. 정확한 매칭방안이 문제시되는 경우는 같은 의미이나 다른 용어로 표현된 경우와 유사정보를 찾기가 어렵다는 것이며, 이로 인해 정보검색에서 이야기하는 정확도(precision)는 높으나 상기도(recall)는 낮아지는 것이다. 따라서 정확도를 유지하면서 상기도를 높이기 위한 방안으로 유사한(imprecise) 매칭방안을 정확한 매칭과 보완해서 사용하려는 노력들이 있어왔다[Bernstein and Kiefer, 2006; Kiefer *et al*., 2007; Klusch *et al*., 2005; Klusch *et al*., 2006].

시맨틱 프로세스를 표현하고 유사한 프로세스를 검색하는 연구는 현재까지 시맨틱 프로세스의 구조를 제시하고 이 구조에 맞게 인위적으로 생성된 프로세스 데이터를 활용하여 실험을 실시하여 왔으며, 프로세스 구조보다는 프로세스 설명이나 이름과 같은 데이터에 기반하여 유사한 프로세스를 찾아왔다. 시맨틱 웹 언어로 표현된 비즈니스 프로세스는 프로세스에 속해있는 개체들이 어떠한 의미를 갖는 관계인지를 파악할 수 있도록 표현되어 있기 때문에, 시맨틱 프로세스 구조에서도 유사한 비즈니스 프로세스를 찾기 위한 중요한 정보를 얻을 수 있다.

본 연구의 목적은 실제 비즈니스 프로세스 구조에 바탕을 두어 시맨틱 프로세스를 표현하고, 효과적인 프로세스 검색을 위해 정확한 매칭방안의 결과를 확장하여 유사한 프로세스를 검색하는 유사도 기반의 검색 알고리즘을 제시하는 것이다. 이를 위해 MIT 프로세스 핸드북(Process Handbook) 프로젝트에서 구축해 놓은 기업들의 프로세스 데이터를 OWL로 표현하였으며, 표현된 비즈니스 프로세스 데이터를 SPARQL과 연계하여 검색하기위한 유사도 알고리즘을 제시하고 이들의 검색성과를 비교한다. 활용한 유사도 알

고리즘들은 프로세스의 구조와 프로세스가 가지고 있는 속성들의 설명이나 이름을 활용하는 기본적인 방안들을 조합하여 생성하였다.

제 Ⅱ장에서 시맨틱 프로세스 표현과 검색에 대한 관련 연구를 소개하며, 제 Ⅲ장에서 본 연구에 활용된 프로세스 핸드북과 실험에 활용된 유사도 기반 검색기법을 소개한다. 제 Ⅳ장에서는 제 Ⅲ장에서 제시한 기법의 검색 성과를 알아보기 위해 수행한 실험과 실험데이터를 소개하며, 각 방안들의 검색성과를 비교한다. 결론으로 제 Ⅴ장에서 실험 성과에 대해 토의하며 실험의 한계와 향후 연구방향에 대해 논의한다.

Ⅱ. 관련 연구

시맨틱 웹 자원의 검색을 위해서는 정확한 매칭기법에 기반을 둔 RQL, RDQL, SPARQL 등과 같은 질의언어들이 제시되어 왔다[Haase *et al.*, 2004]. 이들은 RDF기반으로 표현된 시맨틱 웹 자원이 그래프로 표현될 수 있다는 것을 활용하여, 질의어가 포함된 그래프 구조를 파악하는 데에 중점을 두고 있다. 정확한 매칭 기법에 기반을 두고 있기 때문에 정확한 검색결과를 가져올 수는 있으나, 유사한 자원을 파악하거나 관련 정보를 활용하여 검색성과를 높이는 것은 어려웠다. 따라서, 시맨틱 웹 언어로 표현된 데이터들의 검색성과를 높이기 위해 통계적인 방안에 기반을 둔 유사도 알고리즘을 활용하는 iRDQL[Bernstein and Kiefer, 2006]과 iSPARQL[Kiefer *et al.*, 2007]과 같은 접근방안이 제시되어 왔다. 위의 방안들에서 활용된 유사도 알고리즘으로는 크게 온톨로지 기반[Resnik, 1999], 정보이론 기반[Lin, 1998; Resnik, 1999], 벡터공간모델 기반[Baeza-Yates and Ribeiro-Neto, 1999], 텍스트기반 알고리즘이 활용되어왔다[Bernstein *et al.*, 2005]. 또한 시맨틱 웹 데이터가 그래프 형태로 표현되기 때문에 그래프에 기반을 둔 트리 알고리즘[Sager *et al.*, 2006]이나 그래프기반 유사도 방안들[Hau *et al.*, 2005]도 활용되고 있다.

일반적인 시맨틱 웹 자원이 아니고 웹에서 정보를 제공하거나 프로그램간의 연계를 지원하는 서비스를 표현하기 위해 고안된 시맨틱 웹 서비스를 검색하기 위한 방안에 대해서도 많은 연구들이 진행되어 왔다[Bianchini *et al.*, 2006; Hau *et al.*, 2005; Wang and Stroulia, 2003]. 시맨틱 웹 서비스에 대한 설명은 OWL-S 서비스 프로파일(Profile), 서비스 모델, 서비스 그라운딩(grounding) 정보를 근간으로 하며 로직(Logic) 추론을 통해 서비스가 가능한 웹 서비스를 찾는 방안[Klusch *et al.*, 2005]이나 웹 서비스의 입력 변수와 출력 결과를 바탕으로 두 서비스간의 매칭 정도를 계산하는 방안들이 활용되고 있다[Ouzzani and Bouguettaya, 2004]. 또한 로직 추론을 근간으로 하고 변수들간의 유사도를 계산하여 로직 추론을 보완하는 OWLS-MX방안[Klusch *et al.*, 2005; Klusch *et al.*, 2006]은 loss-of-information, extended Jaccard, Cosine, Jensen-Shannon information divergence와 같은 유사도 알고리즘을 활용하였다.

비즈니스 프로세스 검색에도 검색성과를 향상시키기 위해 유사도 알고리즘을 적용하는 연구들이 진행되어 왔다. Bernstein and Klein[2002]은 MIT 프로세스 핸드북 데이터를 검색하기 위한 Process Query Language(PQL)을 제시하였다. PQL은 정확한 매칭에 기반하여 사용자의 질의와 매칭되는 프로세스를 추출하여준다. PQL을 확장한 연구인 Bernstein *et al.*[2004]에서는 온톨로지를 활용하여 프로세스들간의 유사도를 구해 질의어와 관련된 프로세스를 찾았다. 프로세스들간의 일반화, 특수화 관계를 하나의 노드로 간주하여 전체 프로세스 온톨로지에서 두 개의 프로세스간의 거리가 얼마나 되는지에 기반을 둔 온톨로지 기반 유사도와 정보이론 기반, Levenshtein edit distance[Bernstein *et al.*, 2005], 키워드 기반 유사도 알고리즘을 PQL과 보완 적용하여 검색 성과를 높일 수 있었다. 검색실험에는 MIT 프로세스 핸드북 데이터를 활용하였으며, 실험에 사용

된 프로세스들간의 유사도를 참여 연구진들이 주관적으로 정의하여 질의에 대한 응답집단을 생성하였다.

시맨틱 프로세스들의 검색에도 검색성과를 향상시키기 위해 정확한 매칭과 이를 보완하기 위한 유사도 알고리즘 기반 방식들이 활용되었다[Ehrig *et al.*, 2007; Klusch *et al.*, 2005; Klusch et al., 2006]. Ehrig *et al.*[2007]은 기업의 비즈니스 프로세스를 페트리 네트(Petri net)에 기반을 두어 OWL로 표현하였으며, 표현된 시맨틱 프로세스간의 유사도를 Levenshtein edit distance 같은 표현에 활용된 단어들간의 유사도를 활용하는 방안과, 구조(Syntactic)정보를 활용하는 방안, 그리고 표현된 프로세스의 그래프 구조를 바탕으로 구조적인 유사도를 계산하는 방안들이 활용되었다.

시맨틱 프로세스를 표현하고 검색하는 현재까지의 연구들이 알고리즘의 적용가능성을 예시적으로 보이거나[Ehrig *et al.*, 2007], 인공적으로 생성된 데이터 집합을 활용하거나[Klusch *et al.*, 2005], 연구진들에 의해 프로세스들간의 유사도를 할당하는 방안을 통해 실험집합을 생성하여 왔다[Bernstein *et al.*, 2004].

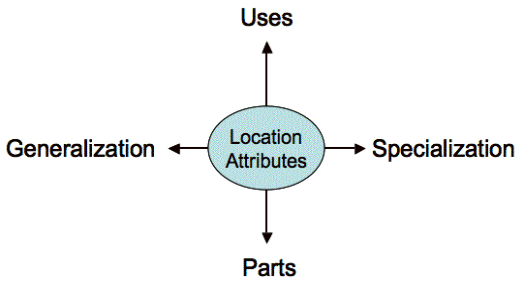
본 연구에서는 실제 기업 비즈니스 프로세스 데이터를 활용하여 실제의 의미가 포함된 시맨틱 프로세스 데이터를 생성하며, 이를 검색 실험에 활용하기 위한 실험 데이터 생성방안을 제시한다. 또한 생성된 비즈니스 프로세스 데이터의 속성에 기반을 둔 유사도 알고리즘을 고안하여 시맨틱 프로세스 검색방안을 제시하고 이들간의 검색성과를 비교하였다.

Ⅲ. 시맨틱 프로세스 표현

3.1 MIT 프로세스 핸드북

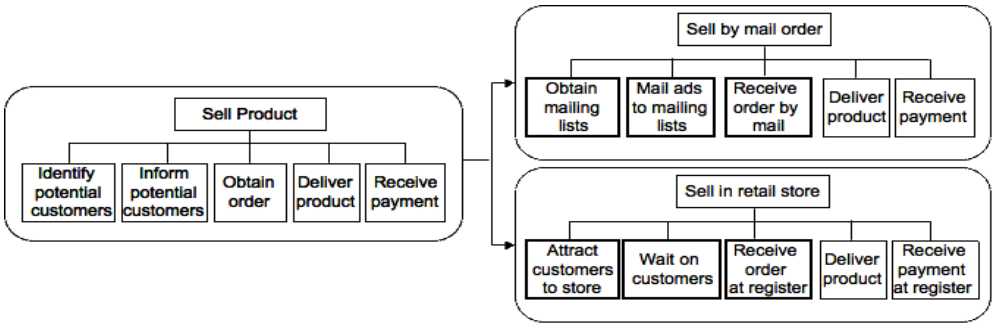
MIT 프로세스 핸드북은 MIT Center for Coordination Science[1]에서 기업의 베스트 프랙티스(best-practice) 비즈니스 프로세스를 표현하고 축적하기 위해 구현한 프레임워크이며 이에 기반을 두어 실제 기업들의 비즈니스 프로세스를 축적하여왔다. 프로세스 핸드북의 목적은 기업의 비즈니스 프로세스 재설계, 새로운 프로세스의 설계, 그리고 기업 프랙티스들에 대한 아이디어를 공유하기 위해 고안되었다[Malone *et al.*, 1999]. 현재까지 약 8천 개의 비즈니스 프로세스를 축적하였으며, 프로세스 검색, 분석과 편집을 위한 도구들을 구현하여 왔다[Malone *et al.*, 2003]. 프로세스 핸드북에서 비즈니스 프로세스는 프로세스 계층관리(specialization), 연관관계(dependency) 관리, 예외관리(exception handling)의 세 가지 개념에 바탕을 두어 표현되고 있다[Klein and Petti, 2006].

계층관리 : 프로세스 핸드북은 아래 <그림 1>처럼 두 가지 차원의 계층을 관리하고 있다. <그림 1>의 수직 축은 세분화(decomposition) 관계를 나타내고, 일반적으로 계층(hierarchy)관리에서 다루어지는 일반화/특수화(generalization/specialization) 관계가 수평축의 관계를 나타내고 있다. 하나의 프로세스가 여러 개의 파트(part)로 구성되었으며, 각 파트가 프로세스에서 활용(use)되고 있다는 것을 나타내는 것이 수직 축이다[Klein and Petti, 2006; Taivalsaari, 1996; Van der Aalst and Basten, 1999].



<그림 1> 프로세스 컴파스(Compass)

1) 현재는 MIT Center for Collective Intelligence로 변경되었으며 관련정보는 http://cci.mit.edu에서 확인할 수 있다.



<그림 2> 특수화 관계에서의 상속 예제[2)]

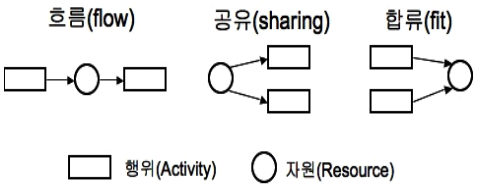
<그림 2>는 일반화/특수화 관계의 예제이다. 일반 프로세스인 "Sell Product"는 "Identify potential customers"와 "Inform potential customers"와 같은 파트로 구성된다. 또한 "Sell Product"는 좀 더 특수한 프로세스인 "Sell by mail order"와 "Sell in retail store"로 나누어 질 수 있다. 특수화된 프로세스는 기본적으로 상위 프로세스의 파트나 속성들을 상속(inheritance)받게 되며, 속성이나 파트를 추가하거나 변경하는 것이 가능하다.

예를 들어, "Sell by mail order" 프로세스에서 "Deliver product"와 "Receive payment" 파트는 수정 없이 상속된 것이지만, "Identify potential customers"는 좀 더 특수한 행위인 "Obtaining mailing lists"로 변경되었다.

프로세스 핸드북에서는 특수화와 관련되어 번들(bundle)과 비교테이블(Trade-off)이라는 두 가지 아이디어를 추가하였다. 번들은 유사한 특수화 프로세스들의 묶음이며, 번들에 속한 프로세스들간의 비교를 쉽게 하기 위하여 각 프로세스의 속성들을 나열한 것이 비교테이블이다[Klein and Petti, 2006].

연관관계관리: 두 번째 주요 개념은 조정(Coordination)이 행위들간의 연관관계를 관리하는 프로세스라고 볼 수 있다는 것이다[Malone *et al.*, 1999]. 모든 연관관계는 공통적으로 활용하는 자원의 흐름을 관리할 수 있는 조정방안을 가질 수 있으며, 이를 통해 연관관계를 가진 행위들을 조정하는 것이 가능하다는 것이다. 프로세스 핸드북은 자원들간의 관계를 연관관계로 정의하고 있으며, <그림 3>과 같이 흐름(flow), 공유(sharing), 합류(fit)의 세 가지 연관관계 유형을 갖고 있다. 흐름은 한 행위의 결과물이 다른 행위에 의해서 활용되는 것이며, 공유는 복수의 행위들이 하나의 자원을 공유하여 활용한다는 것이다. 합류는 복수의 행위들이 하나의 자원을 만들어 내는 것을 뜻한다.

각 연관관계들은 자원 활용을 조정하기 위한 조정방안(coordination mechanism)을 가질 수 있다. 공유는 선입선출 혹은 경매와 같은 방안에 의해 조정될 수 있으며, 흐름은 주기적인 자원전달, 수요에 따른 자원전달, 생성 시 자원전달과 같은 조정방안을 가질 수 있다.



<그림 3> 연관관계[3)]

예외관리: 세 번째 주요개념은 향후에 있을 수

2) Malone *et al.*[2003], p. 16에서 차용하였다.
3) Malone *et al.*[2003], p. 20에서 차용하였다.

<표 1> 프로세스 핸드북 온톨로지

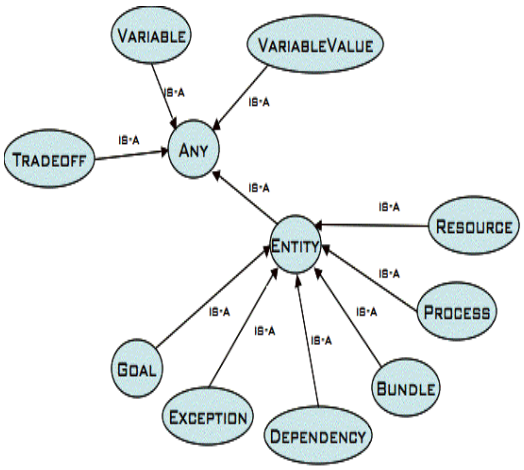
클래스	설명	
Any	관계	hasSpecialization ONLY Process, Entity, hasGeneralization ONLY Process, hasPart ONLY Process
	Domain of	hasException, hasInput, hasOutput, isEnabledBy, isPerformedBy, requires
Entity	상위 클래스	Any
	Domain of	hasBundle, hasGeneralization, hasPart, hasSpecialization, isAssociatedTo
Process	관계	hasSpecialization ONLY Process, Entity, hasGeneralization ONLY Process, hasPart ONLY Process
	Domain of	hasException, hasInput, hasOutput, isEnabledBy, isPerformedBy, requires
Goal	관계	hasSpecialization ONLY Goal, Entity, hasGeneralization ONLY Goal, hasPart ONLY Goal
	Domain of	isAchievedBy
Exception	관계	hasSpecialization ONLY Exception, Entity, hasGeneralization ONLY Exception, hasPart ONLY Exception
	Domain of	Causes, hasViolated, isHandledBy
Bundle	관계	hasSpecialization ONLY Bundle, Entity, hasGeneralization ONLY Bundle, hasPart ONLY Bundle
	Domain of	hasTradeoff
Dependency	관계	hasSpecialization ONLY Dependency, Entity, hasGeneralization ONLY Dependency, hasPart ONLY Dependency
	Domain of	Dependency
Resource	관계	hasSpecialization ONLY Resource, Entity, hasGeneralization ONLY Resource, hasPart ONLY Resource
	Domain of	isConnectedTo
Tradeoff	상위 클래스	Any
	Domain of	hasValue, hasVariables
Variable	상위 클래스	Any
VariableValue	상위 클래스	Any
	Domain of	referTo, regardTo, value

있는 수행 실패 시에 어떻게 프로세스를 다룰 수 있을지 정의하는 것이다[Klein and Dellarocas, 2003; Klein and Petti, 2006]. 모든 프로세스는 목적달성에 실패한 경우에 어떻게 처리할지에 대한 방안들이 정의되어 있다. 다른 모든 프로세스의 속성들처럼 이 예외들도 특수화 관계에서 상속되며, 예외를 처리하기 위한 프로세스와 연계되어 있다. <그림 2>의 'Sell Product'의 파트인 'Inform potential customers'은 예외로 '원하지 않는 고객접촉'을 가질 수 있다.

3.2 프로세스 핸드북 온톨로지와 OWL 파일

프로세스 핸드북에 정의된 구조와 개념들에 바탕을 두어 프로세스 핸드북 온톨로지를 설계하였다. <그림 4>가 설계된 프로세스 핸드북 온톨

로지의 개요이며, 온톨로지에 포함된 개체들과 기본적인 연관관계가 표현되어 있다. <표 1>은 각 클래스들이 가지고 있는 계층과 관계(relation)에 대한 설명이다. 설계된 프로세스 핸드북 온톨로지를 메타모델로 삼아, 프로세스 핸드북에 축적된 모든 비즈니스 프로세스들을 OWL로 표현하였다.[4] 프로세스 핸드북에 표현되어 있는 프로세스(process), 번들(bundle), 목적(goal), 예외(exception), 자원(resource), 연관관계(dependency) 그리고 비교테이블(Trade-off) 등 모든 개체들을 OWL 클래스로 표현하였다.



<그림 4> 프로세스 핸드북 온톨로지 구조

Tradeoff, Variable, VariableValue는 특정 프로세스를 표현하는 구성요소가 아니고, 여러 프로세스를 비교하는 목적으로 도입된 개념이다. 따라서, 프로세스를 표현하는데 필요한 Entity 클래스가 아니고 따로 Any라는 클래스의 하위 클래스들로 표현하였다. <그림 5>는 OWL로 표현된 비즈니스 프로세스 "E1024"이며, 하나의 예외(processHandbook: hasException)와 2개의 프로세스 특수화(processHandbook: has Specialization) 그리고 하나의 프로세스 일반화 관계(process Handbook: hasGeneralization)를 가지고 있다.

```
- <rdf:RDF xml:base="http://www.ifi.unizh.ch/ddis/ph/2006/08/E1024.owl">
  - <processHandbook:Process rdf:ID="E1024">
      <processHandbook:name xml:lang="en">Determine cost</processHandbook:name>
    - <processHandbook:description xml:lang="en">
        This is a general activity to determine the cost to the organization of purchase or production.
      </processHandbook:description>
      <processHandbook:hasException
      rdf:resource="http://www.ifi.unizh.ch/ddis/ph/2006/08/E17159.owl#E17159"/>
      <processHandbook:hasSpecialization
      rdf:resource="http://www.ifi.unizh.ch/ddis/ph/2006/08/E6302.owl#E6302"/>
      <processHandbook:hasSpecialization
      rdf:resource="http://www.ifi.unizh.ch/ddis/ph/2006/08/E8007.owl#E8007"/>
      <processHandbook:hasGeneralization
      rdf:resource="http://www.ifi.unizh.ch/ddis/ph/2006/08/E3356.owl#E3356"/>
    </processHandbook:Process>
  </rdf:RDF>
```

<그림 5> OWL로 표현된 비즈니스 프로세스 예제

Ⅳ. 시맨틱 비즈니스 프로세스 검색방안

4.1 유사도 알고리즘

시맨틱 프로세스간의 유사도를 계산하기 위하여 iSPARQL 프레임워크[Kiefer *et al.*, 2007]를 활용하였다. iSPARQL은 SPARQL에 바탕을 두어 유사도 알고리즘을 통해 검색결과를 확장할 수 있도록 구현된 검색엔진이다. iSPARQL에서는 SimPack[Bernstein *et al.*, 2005]에서 구현된 유사도 알고리즘을 모두 활용할 수 있다. 실험 데이터의 검색을 위해 일반적으로 활용할 수 있는 기본 유사도 알고리즘과 실험 데이터의 특성을 고려하여 설계된 유사도 알고리즘을 검색 실험에 활용하였으며, <표 2>에 활용된 알고리즘을 정리하였다. 유사도 알고리즘에서 유사도를 고려하는 기본적인 정보는 프로세스의 이름과 설명에 포함된 키워드(TFIDF, Lev)와 프로세스가 포함하고 있는 개체(Dice와 Jaccard 척도)이다. 전자는 시맨틱 프로세스가 포함하고 있는 문자열을 비교하여 유사도를 계산한다면, 후자는 시맨틱 프로세스가 포함하고 있는 개체가 얼마나 중복되

4) http://www.ifi.unizh.ch/ddis/ph-owl.html에 온톨로지 정보, OWL 파일, 데이터집합 등 관련 정보들이 공유되어 있다.

<표 2> 활용된 유사도 알고리즘

알고리즘			설명
방안	구분	이름	
기본 방안	1	TFIDF	프로세스 설명부분에 포함된 키워드들을 활용하여 두 프로세스간의 TFIDF 값을 계산한다. 미리 계산된 각 키워드들의 빈도에 대한 정보를 활용한다. TFIDF는 전통적으로 활용되는 코사인 척도를 확장한 개념이라고 볼 수 있다[Baeza-Yates and Ribeiro-Neto, 1999].
	2	Lev	프로세스의 이름을 가지고 Levenshtein edit distance를 활용하여 유사도를 계산한다. 이때 유사도는 하나의 문자열에서 다른 문자열로 전환하기 위해 필요한 삽입, 삭제 그리고 변환행위의 횟수를 가지고 계산된다[Levenshtein, 1966].
	3	Dice-Parts	프로세스에 포함된 파트를 비교하여 두 프로세스간의 Dice 유사도를 계산한다. Dice 유사도는 동일한 파트의 갯수를 두 개의 프로세스 파트의 갯수로 나누어 계산된다.
	4	Jaccard-Parts	프로세스에 포함된 파트를 비교하여 두 프로세스간의 Jaccard 유사도를 계산한다. Jaccard 유사도는 코사인 척도와 같이 벡터 공간 모델에 기반을 두고 있다. Jaccard 척도는 Dice 척도와 비교할 때 공통적인 파트의 갯수가 적은 경우에 더욱 낮은 유사도 값을 갖게 된다. 두 개의 벡터 X, Y의 Jaccard 유사도는 아래와 같이 계산된다. (X*Y)/(|X||Y|-(X*Y)), (X*Y)는 두 벡터의 내적이며, |X| = (X*X)^1/2이다.
	5	TreeEdit	프로세스에 포함된 파트들이 트리 형태를 띄고 있기 때문에 이를 이용하여 프로세스들간의 트리 변경 거리(Tree edit distance)[Valiente, 2002]를 계산한다. 트리 변경 거리가 작을 수록 유사한 프로세스로 판단할 수 있다.
설계된 방안	6	Lev-TFIDF	프로세스 이름을 활용한 Levenshtein Level 2 유사도 계산방안과 프로세스 설명을 활용한 TFIDF유사도 계산방안 결과를 조합하여 최종 유사도를 계산하는 방안이다. 두 방식의 유사도를 조합할 때 동일한 가중치가 부여된다. Levenshtein Level 2(Levehnshtein of Levenshtein)는 두 문자열의 문자와 문자를 비교하는 Levenshtein edit distance에 기반을 두고 있으나, 문자와 문자를 비교하는 것이 아니고 문자열에 포함된 단어를 비교하여 일정한 수준이상의 유사도를 가지면 동일한 단어로 판단한다[Hollenstein, 2005]. 두 문자열을 동일하게 만들기 위해서 단어를 삽입, 삭제 혹은 변환하는 행위의 수에 따라서 유사도가 계산된다. 예를 들어, “Programming Language“와 “Semantics of Programming Languages“라는 문자열이 있고, 단어의 동일 여부를 판단하는 기준을 유사도 0.7이라고 한다면 Language와 Languages는 동일한 단어가 된다. 따라서 Semantics와 of를 삽입하는 행위만으로 동일한 단어를 만들 수 있다. 이 경우 Levenshtein edit distance는 2이고, 최악의 경우 4가 된다. 따라서 유사도는 0.5(=1-2/4)값이 된다.
	7	JaccardAll	4번 방안인 Jaccard-Parts는 프로세스가 가지고 있는 파트 프로세스만 고려하여 유사도를 계산하는 것이라면, JaccardAll은 프로세스에 포함된 모든 개체(파트, 예외, 목적, 자원, 인풋과 아웃풋)를 활용하여 Jaccard 유사도를 계산하는 방안이다. 이때 각 개체별 유사도를 합칠때 개체별 가중치는 동일하다.
	8	Weighted JaccardAll (except goal)	프로세스에 포함된 모든 개체를 활용한 Jaccard 유사도 계산 방안이며, 각 개체유형의 수에 따라 매칭 여부에 대한 가중치를 부여하여 최종 유사도를 계산한다. 위의 7번방안에서 개체별 가중치로 파트, 예외, 자원, 인풋, 아웃풋에 0.5, 0.2, 0.1, 0.1, 0.1을 부여하였다.
	9	Lev-TFIDF-JaccardAll	프로세스 이름을 활용한 Levenshtein Level 2 유사도 계산방안, 프로세스 설명을 활용한 TFIDF유사도 계산방안과, 프로세스 개체에 대한 Jaccard 유사도 계산방안의 결과를 조합하여 최종 유사도를 계산하는 방안이다. 최종 유사도 조합시의 가중치는 세 가지방안이 동일한 값을 갖는다.

```
PREFIX ph: <http://www.ifi.unizh.ch/ddis/ph/2006/08/ProcessHandbook.owl #>
PREFIX isparql: <java:ch.unizh.ifi.isparql.query.property.>

SELECT ?process2 ?name2 ?similarity
WHERE {
    ?process 2 ph: name ?name2 .
    ?strategy isparql: name "TFIDF".
    ?strategy isparql: arguments (ph: E16056 ?process2).
    ?strategy isparql: similarity ?similarity.
}ORDER BY DESC (?similarity)
```

<그림 6> iSPARQL 검색 질의

<표 3> 프로세스 변형방안

변형부분	변형방안 이름	설 명
프로세스 (파트)	STEPSPLIT	하나의파트를 두 개의 형제 파트로 변형
	STEPCHILD	하나의파트를 부모/자식 파트 관계로 변형
	STEPMERGESIB	임의로 선정된 두 개의 형제 파트를 하나로병합
	STEPMERGEPARENT	하나의 파트를 이의 부모 파트와 병합
	STEPDELETE	하나의 파트 삭제
표기내용	NAMEDELETE	프로세스 이름 중 하나의 단어 삭제
	DESCRIPTIONDELETE	프로세스 설명 중 하나의 단어 삭제

는지를 파악하여 유사도를 계산하기 때문에 시맨틱 프로세스의 구조정보를 고려하는 것이라고 볼 수 있다. 설계된 방안들은 기본방안들을 조합하여 활용하거나 프로세스가 여러 유형의 개체들을 포함하고 있기 때문에 이를 고려한 가중치 계산방안이 활용되었다.

위의 유사도 알고리즘은 iSPARQL 프레임워크에 구현되었으며 <그림 6>과 같은 방식으로 질의(Query)되어 두 프로세스간의 유사도를 계산한다. <그림 6>은 TFIDF 방안에 의해 두 프로세스간의 유사도를 구하는 예이다.

4.2 실험

4.2.1 실험 데이터 및 실험 방안

3.2절에서 생성된 프로세스 데이터를 가지고 프로세스 검색 실험을 수행하기 위해서는 검색하기 위한 질의 프로세스와 이에 대응되어 검색되어야만 하는 응답 프로세스들을 정의하여야 한다. 관련연구에서 언급한 것처럼 현재까지는 연구진들에 의해 분류되거나 인공적으로 생성된 의미 없는 프로세스들을 가지고 실험을 수행하였다. 본 연구에서는 의미가 보존되면서(semantic-preserving) 프로세스를 변형(process mutation)시키는 방안을 통해 실험 집합을 구성하는 방안을 활용하였다. 우리는 프로세스 핸드북에서 특수화나 일반화 관계가 구성되어 있는 프로세스들 중에서 80개의 프로세스를 선정하여서 이를 목표집합(target set)으로 삼았다. 목표집합에 속한 프로세스 마다 구조적으로는 상이하지만 의미적(semantic)으로는 동일한 20개의 돌연변이 프로세스를 생성하였다. 이 돌연변이들이 목표집합에 속한 프로세스와 유사한 프로세스를 검색

하였을 때, 검출되어야 하는 정확한 응답집합이다. 목표 프로세스 별로 응답집합에 속한 프로세스들을 제외한 다른 프로세스들이 검색되는 경우에는 부정확한 결과로 계산하였다.

각 돌연변이들은 의미를 보존하는 변형방안들에 의해서 원래의 목표 프로세스에서 변화해 나가며, 응답집합에는 한번 변형한 경우부터 20번 변형한 것까지의 돌연변이 프로세스들이 포함되었다. 응답집합 생성에 활용된 변형방안들은 다음 <표 3>과 같다.

변형방안들은 프로세스를 모델링하는 다양한 다른 방안들에 바탕을 두어 선택되었다. 만약 우리가 식당의 비즈니스 프로세스를 모델링하는 경우에 패스트푸드 식당의 경우 '주문'과 '계산'을 하나의 행위로 표현할 수도 있고 다른 식당의 경우에는 나누어서 표현할 수도 있다. 위 두 프로세스는 표현양식은 좀 다를지라도 식당에서 주문과 계산이라는 의미를 모두 포함하고 있다. STEPMERGESIB가 위의 경우에 해당하는 변형방안이며 반대로 STEPSPLIT이 하나의 파트를 두 개의 파트로 나누는 변형방안이다. 물론 위의 변형방안에 상당한 임의(random)요소가 작용하고 있다. 연구자에 의해서 의미에 기반을 두어 파트가 나누어 지거나 합쳐진 것이 아니고 프로그램에 의해 임의로 선정된 파트들이 나누어 지거나 합쳐지고 일부 설명부분이 삭제되었기 때문이다. 따라서, 목표 프로세스의 응답집합에 속한 돌연변이들은 사람들이 의미에 기반을 두고 생성한 돌연변이와는 차이가 있을 수 있다. 그러나 이러한 변형방안들이 큰 규모의 실제 비즈니스 프로세스를 가지고 테스트 데이터를 생성하는데 유용하리라고 믿는다.

<그림 7>은 목표 프로세스(E3694)와 이로부터 하나의 돌연변이 프로세스(E-TJ78H4-1)가 생성되는 과정을 기술해 놓은 것이다. 생성될 돌연변이 프로세스(E-TJ78H4-1)가 목표 프로세스(E3694)와 특수화 관계를 갖는 것으로 돌연변이 생성과정을 시작한다. 순차적으로 여러 개의 변형방안들

목표 프로세스

```
<?xml version = "1.0"?>
 ..
<processHandbook: Process rdf: ID = "E3694">
<processHandbook: name xml: lang = "en">
 Buy</processHandbook: name>
..
<processHandbook: hasException rdf: resource="&ph;
 E17159.owl#E17159"/>
<processHandbook: hasPart rdf: resource = "&ph;
 E2232.owl#E2232"/>
<processHandbook: hasPart rdf: resource = "&ph;
 E5934.owl#E5934"/>
<processHandbook: hasPart rdf: resource = "&phE606.
 owl#E606"/>
<processHandbook: hasPart rdf: resource = "&ph;
 E2751.owl#E2751"/>
<processHandbook: hasPart rdf: resource = "&ph;
 E4723.owl#E4723"/>
<processHandbook: hasPart rdf: resource = "&ph;
 E2641.owl#E2641"/>
<processHandbook: hasPart rdf: resource = "&ph;
 E1212.owl#E1212"/>
<processHandbook: hasBundle rdf: resource = "&ph;
 E6291.owl#E6291"/>
<processHandbook: hasBundle rdf: resource = "&ph;
 E14573.owl#E14573"/>
<processHandbook: hasBundle rdf: resource = "&ph;
 E14568.owl#E14568"/>
<processHandbook: hasBundle rdf: resource = "&ph;
 E16163.owl#E16163"/>
<processHandbook: hasBundle rdf: resource = "&ph;
 E4835.owl#E4835"/>
<processHandbook: hasBundle rdf: resource = "&ph;
 E4834.owl#E4834"/>
<processHandbook: hasBundle rdf: resource = "&ph;
 E1434.owl#E1434"/>
<processHandbook: hasGeneralization rdf: resource =
 "&ph; E8219.owl#E8219"/>
<processHandbook: hasGeneralization rdf: resource =
 "&ph; E8114.owl#E8114"/>
</processHandbook: Process>

</rdf: RDF>
```

돌연변이 과정

applying STEPMERGESIB to E-TJ78H4-1applying STEPDELETE to E2232applying STEPMERGEPARENT to E16170applying STEPCHILD to E606applying STEPDELETE to E16172applying STEPMERGEPARENT to E-TJ78H4-2applying STEPSPLIT to E-TJ78H4-3applying DESCRIPTIONDELETE to E-TJ78H4-7applying STEPSPLIT to E2751applying STEPDELETE to

E-TJ78H4-7applying STEPDELETE to E-TJ78H4-6applying DESCRIPTIONDELETE to E-TJ78H4-10applying STEPSPLIT to E-TJ78H4-9applying DESCRIPTIONDELETE to E-TJ78H4-10==> mutant E-TJ78H4-1 of process E3694

돌연변이 프로세스

```
<?xml version = 1.0“?>
 ..
<processHandbook: Process rdf: ID=“E-TJ78H4-1“>
<processHandbook: name xml: lang=“en“>Buy Place order Manage suppliers Evaluate suppliers </processHandbook:name>
..
<processHandbook: hasPart rdf: resource = “&ph; E-TJ78H4-12.owl#E-TJ78H4-12“/>
<processHandbook: hasPart rdf: resource = “&ph; E-TJ78H4-11.owl#E-TJ78H4-11“/>
<processHandbook: hasPart rdf: resource = “&ph; E-TJ78H4-8.owl#E-TJ78H4-8“/>
<processHandbook: hasPart rdf: resource = “&ph; E-TJ78H4-5.owl#E-TJ78H4-5“/>
<processHandbook: hasPart rdf: resource = “&ph; E16168.owl#E16168“/>
<processHandbook: hasPart rdf: resource = “&ph; E-TJ78H4-10.owl#E-TJ78H4-10“/>
<processHandbook: hasException rdf: resource = “&ph; E17159.owl#E17159“/>
<processHandbook: hasGeneralization rdf: resource = “&ph; E3694.owl#E3694“/>
</processHandbook: Process>

</rdf: RDF>
```

<그림 7> 프로세스 변형 예제

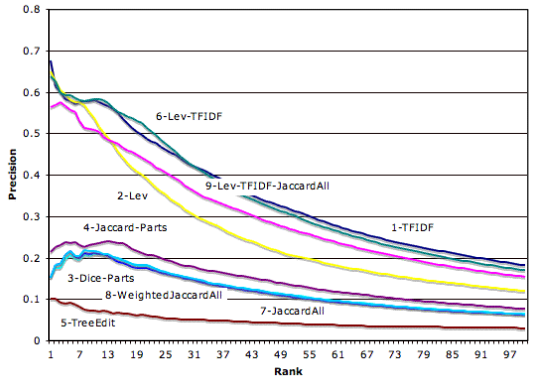
이 적용되어 하나의 돌연변이 프로세스를 생성하며, 이 경우에 처음으로 적용된 돌연변이 방안은 두 개의 임의의 파트를 선정하여 E-TJ78H4-1로 병합하는 것이다. 그 다음은 E2232라는 파트를 삭제하는 것이다. 이 예제에서는 14개의 변형 방안들이 적용되었으며, 변형방안 적용 횟수가 변형 정도를 나타낸다. E-TJ78H4-X 프로세스들은 돌연변이 생성과정에서 파생되는 프로세스들이다.

앞서 언급한 대로 목표집합에 속한 프로세스마다 20개의 돌연변이 프로세스를 생성하였다. 총 27,953개의 프로세스와 80개의 목표집합 프로세스의 돌연변이 1,600개를 대상으로 <표 2>에 제시한 유사도 알고리즘을 활용하여 각 목표집합에 속한 프로세스와의 유사도를 계산하였다. 각 유사도 알고리즘 별로 유사도가 가장 높은 프로세스부터 100번째로 높은 프로세스까지를 점검하면서 이중에 몇 개의 돌연변이 프로세스가 포함되어 있는지를 파악하였다.

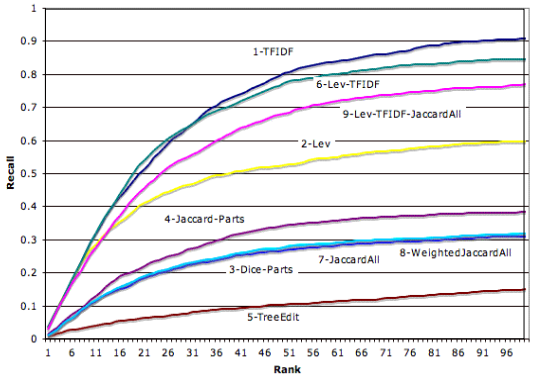
성과 측정기준으로는 정확도(precision), 상기도(recall), 조화평균(Harmonic mean)척도를 활용하여 각 척도 별로 측정하였다. 정확도는 (검색된 데이터 중 관련 데이터 수/검색된 데이터 수)로 계산되며, 상기도는 (검색된 데이터 중 관련 데이터 수/전체 관련 데이터 수)로 계산된다. 하나의 목표 프로세스에 대한 전체 관련 데이터 수는 돌연변이 프로세스의 수인 20개이다. 일반적으로 정확도와 상기도는 정확도가 증가하면 상기도가 하락하고, 상기도가 증가하면 정확도가 하락하는 역의 관계에 있기 때문에, 두 지표를 조합하여 하나의 성과지표로 삼기도 한다[Baeza-Yates and Ribeiro-Neto, 1999]. 조화평균의 값은 P가 정확도, r이 상기도라고 할때 $F = 2/(1/r + 1/P)$와 같이 계산된다. 조화평균은 0과 1사이의 값을 가지며 0은 관련 문서가 검색되지 않은 경우이며, 모든 문서가 검색된 경우에는 1의 값을 갖는다.

4.2.2 실험 결과

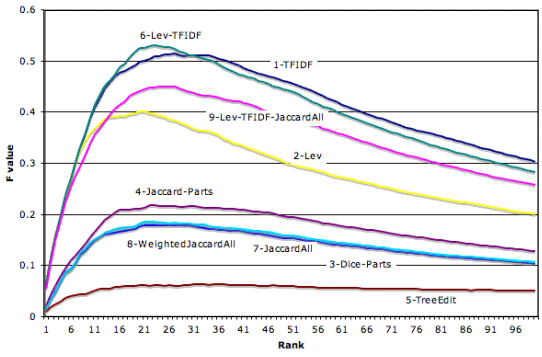
<그림 8>, <그림 9>, <그림 10>은 실험방안들의 정확도, 상기도, 조화평균 값을 보여준다. 각 그림의 가로축은 유사도 알고리즘에 의해 계산된 유사도 값의 순위(rank)를 나타내며, 유사도가 높을 수록 낮은 순위 값을 갖는다. 즉 전체 프로세스들 중에서 해당 프로세스가 몇 번째로 유사하다고 판단하였는지에 대한 순위이다. 각 알고



<그림 8> 실험방안별 평균 정확도



<그림 9> 실험방안별 평균 상기도(Recall)



<그림 10> 실험방안별 조화평균

리즘들이 80개의 목표 프로세스와 관련된 프로세스들을 찾는 경우의 평균 정확도, 평균 상기도, 평균 조화평균이 <그림 7, 8, 9>의 세로축을 나타낸다.

평균 정확도는 Tree edit 거리를 활용한 5번째 방안이 가장 낮았으며, 1번째 방안인 TFIDF와 Levenshtein edit distance와 TFIDF를 조합한 6번째 방안이 가장 높은 정확도를 보였다. 프로세스의 파트를 이용한 Jaccard 유사도, Dice 유사도를 활용하는 방안들 보다는 프로세스에 대한 설명에 근거한 Levenshtein edit distance를 활용하는 1, 2, 6, 9번째 방안이 전반적으로 높은 정확도를 보였다. 이는 프로세스 돌연변이가 주로 프로세스 구조를 변화시키는 경우가 많으며 이름이나 설명부분을 변화시키는 것이 적기 때문인 것으로 보인다. 그러나 이러한 프로세스 구조와 설명 부분을 모두 고려하는 9번째 방안이 비교적 높은 정확도를 보였다.

실험방안별 상기도를 비교하여 정리한 <그림 9>에서도 평균 정확도를 비교한 <그림 7>과 유사한 결과를 얻었다. Tree edit 거리를 활용한 5번째 방안이 가장 낮은 값을 나타냈으며, TFIDF를 활용하는 1, 6, 9번째 방안이 높은 상기도를 보였다.

<그림 8>과 <그림 9>에 표기된 각 방안들의 정확도와 상기도를 조합하여 계산한 조화평균의 값이 <그림 10>이다.

프로세스의 이름과 설명부분의 유사도를 고려하는 1과 6번째 방안이 가장 높은 값을 나타냈으며, Tree edit 거리 척도를 활용한 5번째 방안이 가장 낮은 값을 보였다. 프로세스 구조와 설명을 모두 고려하는 9번째 방안도 좋은 성과를 보였다.

또한, 돌연변이 프로세스들은 프로세스 변형 정도에 따라 1부터 20까지의 돌연변이 정도 값을 가지고 있다. 적은 수의 프로세스 변형이 이루어진 돌연변이 프로세스가 구조상으로는 좀더 원래 프로세스와 유사하다고 볼 수 있다. 따라서, 돌연변이 정도 순위와 각 유사도 알고리즘에 의해 판단된 유사도 순위를 비교할 수 있다. 동일한 수만큼 변형된 프로세스와 동일한 유사도 값을 가진 프로세스들이 있기 때문에 동일 순위(tie)를 고려한 순위상관계수인 Kendall's tau b 척도를

활용하여 유사도 알고리즘에 의한 유사도 순위와 프로세스 변형정도간의 상관관계를 측정하였으며, 그 결과는 <표 4>와 같다.

<표 4> 순위상관계수

실험방안	Pearson	Kendall's tau b
1-TFIDF	0.439*	0.325*
2-Lev	0.343*	0.273*
3-Dice-Parts	0.653*	0.562*
4-Jaccard-Parts	0.659*	0.524*
5-TreeEdit	0.389*	0.351*
6-Lev-TFIDF	0.427*	0.317*
7-JaccardAll	0.680*	0.568*
8-WeightedJaccardAll	0.674*	0.563*
9-Lev-TFIDF-JaccardAll	0.568*	0.438*

* 유의수준 0.01에서 유의.

조화평균 지표와 다르게 돌연변이 정도와 유사도 순위간의 관계에서는 프로세스 구조의 다양한 부분의 유사도를 평가하는 7, 8번째 방안이 그렇지 않은 다른 방안들에 비해서 높은 순위상관계수 값을 나타냈으며, 프로세스 구조간의 유사도를 고려하는 3, 4번째 알고리즘도 7, 8번째 방안과 거의 유사한 결과를 나타내었다. 프로세스의 이름과 설명간의 유사도를 고려하는 1, 2, 5, 6번 방안은 상대적으로 낮은 성과를 보였으며, 프로세스 구조와 설명을 모두 고려하는 9번째 방안은 두 가지 방안유형들의 중간 값을 보였다.

V. 토의 및 결론

5.1 토의

정확한 매칭방안에 기반을 두고 있는 SPARQL의 검색결과를 확장하기 위하여 우선 SPARQL로 검색된 프로세스들을 찾고, 각 프로세스와 유사하지만 정확한 매칭방안에 의해서는 검색되지 않는 다른 프로세스들을 찾는 방식을 활용하였다. 유사한 프로세스를 찾기 위하여 활용된 프로세스의 정보는 프로세스 계층정보 및 속성정보와 같은 구조정보와 프로세스 이름, 설명과 같은 속성이 가지고 있는 값을 활용하였다. 일반적으로 검색에 활용되는 기본방안들과 이들을 혼합하여 활용하는 방안들을 설계하였으며, iSPARQL을 통하여 각 방안들의 검색 성과를 비교하였다.

검색성과를 비교하는 정확도, 상기도, 조화평균 척도에서는 프로세스의 속성값을 활용하는 방안들이 좋은 성과를 보였으며, 돌연변이 프로세스의 돌연변이 정도와 유사도 순위간의 상관관계를 측정한 실험에서는 프로세스의 구조정보를 활용한 방안들이 높은 성과를 보였다. 실험데이터인 돌연변이 프로세스를 만드는 과정에서 속성값을 변형시키는 정도보다 구조정보를 변형시키는 경우가 더 많이 활용되었기 때문에 유사도를 측정함에 있어서는 비교적 덜 변형된 속성값 정보를 활용하는 것이 좋은 성과를 보였다고 판단된다. 또한, 속성값 정보와 구조정보를 동시에 고려하는 유사도 척도인 9번째 방안이 검색성과와 돌연변이 정도 측정에서 모두 좋은 값을 보였다. 이는 프로세스의 유사도를 바라보는 측면이 다양할 수 있기 때문에, 다양한 측면을 고려한 유사도 알고리즘을 설계하는 것이 유사 프로세스 검색에서 유용하게 활용될 수 있음을 보였다고 할 수 있다.

5.2 결론 및 향후 연구 방향

본 연구에서는 MIT 프로세스 핸드북에 표현되어 있는 실제 비즈니스 프로세스와 이들의 구조에 바탕을 두어 시맨틱 프로세스를 표현하였다. 시맨틱 웹으로 표현된 자원을 검색하기 위한 SPARQL의 검색결과를 확장하기 위하여 프로세스간의 유사도에 기반을 둔 검색방안들을 제시하였으며, 다양한 유사도 척도에 기반을 둔 검색결과 확장방안의 성과를 비교하였다. 또한, 유사도 척도를 활용하여 시맨틱 프로세스를 검색하

기 위한 데이터 집합을 생성하기 위하여 의미를 보존한 상태로 구조와 내용을 변이시키는 방안을 통하여 목표 프로세스의 돌연변이들을 생성하였다. 이를 통해 프로세스의 다양한 요소인 프로세스 구조와 설명을 모두 고려하는 유사도 척도를 활용하는 것이 의미상 가까운 프로세스와 구조가 가까운 프로세스를 검색하기 위한 유용한 유사도 척도가 될 수 있다는 것을 보였다. 프로세스 검색 실험에 있어서도 정확도나 상기도에 기반을 둔 검색(retrieval)기반 성과와 유사도 순위 상관관계를 고려하는 실험을 수행하여 다양한 관점에서 유사도를 평가할 수 있도록 하였다.

이러한 연구결과는 기업들이 비즈니스 프로세스를 활용하는 웹 서비스를 찾거나(matchmaking), 기업의 비즈니스 프로세스와 연계될 수 있는 서비스를 찾아내는 데에 활용될 수 있을 것이다. 비즈니스 프로세스가 시맨틱 웹 언어로 표현된 경우에는 이에 포함된 설명정보뿐만이 아니라 구조정보도 유사도를 평가하는데 중요한 역할을 하므로, 연계할 비즈니스 프로세스를 찾는 경우에 설명정보와 구조정보를 모두 활용하는 것이 필요하다. 이를 통해 기업간 비즈니스 프로세스 연계 자동화에 기여할 수 있으며, 비즈니스 프로세스 재설계나 비즈니스 프로세스 관리에서 연관있는 프로세스를 발견하고 참조하는데 활용될 수 있다[Klein and Petti, 2006].

연구의 한계로는 하나의 데이터 집합을 대상으로 유사도 알고리즘의 성과를 비교하였기 때문에 연구성과를 일반화하기에는 어려움이 따르며, 다른 분야의 데이터 집합을 활용하여 유사도 알고리즘의 성과를 비교하여야 할 필요가 있다. 또한 생성된 실험 데이터를 가지고 성과를 비교하는 검색실험의 기본 가정은 다른 프로세스들보다 목표프로세스의 돌연변이들이 목표프로세스와 가장 유사하리라는 것이다. 하지만 프로세스에 따라서는 다른 프로세스들이 더욱 유사할 수 있으며, 돌연변이 정도에 따라서 돌연변이 프로세스 보다 목표 프로세스에 더욱 유사한 프로세스가 존재할 수 있다. 다른 프로세스들을 고려한 좀 더 엄밀한 실험 데이터 생성이 필요하다. 향후 연구 방향으로는 다양한 유사도 척도에서 얻어지는 유사도 결과들을 활용하는 것이 하나의 유사도 척도를 활용하는 방안보다 더 좋은 검색 성과를 얻을 수 있을 것이다. 따라서 여러 알고리즘에 의한 유사도 결과를 동시에 고려할 수 있는 검색방안에 대한 연구가 필요하다.

<참 고 문 헌>

[1] 김학래, 김홍기, “시맨틱 웹 기반의 e-비즈니스 상호운용성,” *한국경영정보학회 춘계학술대회,* 2002, pp. 311-319.

[2] 김학래, 김홍기, “유비쿼터스 서비스를 위한 시맨틱 웹 기술,” *한국경영정보학회 추계학술대회,* 2003, pp. 31-35.

[3] 김형도, 김종우, “UML기반의 기업간 비즈니스 프로세스 명세 모델링,” *Journal of Information Technology Applications & Management,* Vol. 13, No. 4, 2006, pp. 71-88.

[4] Baeza-Yates, R. and Ribeiro-Neto, B., Modern Information Retrieval, ACM Press, New York, 1999.

[5] Bernstein, A., Kaufmann, E., Buerki, C., and Klein, M., "Object Simialrity in Ontologies: A Foundation for Business Intelligence Systems and High-Performance Retrieval," *Proceedings of Twenty-Fifth International Conference on Information Systems*, 2004, pp. 741-756.

[6] Bernstein, A., Kaufmann, E., Kiefer, C., and Bürki, C., *SimPack: A Generic Java*

Library for Similarity Measures in Ontologies, Technical Report, Department of Informatics, University of Zurich, 2005.

[7] Bernstein, A. and Kiefer, C., "Imprecise RDQL: Towards Generic Retrieval in Ontologies Using Similarity Joins," *Proceedings of SAC'06*, Dijon, France, 2006, ACM, pp. 1684-1689.

[8] Bernstein, A. and Klein, M., "Towards High-Precision Service Retrieval," *Proceedings of the 1st International Semantic Web Conference on The Semantic Web (ISWC'02)*, London, UK, 2002, Springer-Verlag, pp. 84-101.

[9] Bianchini, D., Antonellis, V.D., Pernici, B., and Plebani, P., "Ontology-based methodology for e-service discovery," *Information Systems*, Vol. 31, 2006, pp. 361-380.

[10] Davies, J., Fensel, D. and Harmelen, F.V., ed., *Towards the Semantic Web: ontology-driven knowledge management,* West Sussex, England: John Wiley and Sons Ltd, 2003.

[11] Ehrig, M., Koschmider, A. and Oberweis, A., "Measuring Similarity between Semantic Business Process Models," *Proceedings of the 4th Asia-Pacific Conference on Conceptual Modelling (APCCM'07)*, Ballarat, Victoria, Australia, 2007, pp. 71-80.

[12] Haase, P., Broekstra, J., Eberhart, A. and Volz, R., "A Comparison of RDF Query Languages," *Proceedings of ISWC*, 2004, pp. 502-517.

[13] Hau, J., Lee, W., and Darlington, J., "A Semantic Similarity Measure for Semantic Web Services," *Proceedings of WWW2005*, Chiba, Japan, 2005.

[14] Hollenstein, S., *XQuery Similarity Joins,* University of Zurich, 2005.

[15] Kiefer, C., Bernstein, A., and Stocker, M., "The Fundamentals of iSPARQL-A Virtual Triple Approach For Similarity-Based Semantic Web Tasks," *Proceedings of Proceedings of the 6th International Semantic Web Conference (ISWC)*, 2007.

[16] Klein, M. and Dellarocas, C., "Designing Robust Business Processes," in Thomas W. Malone, Kevin Crowston, and Gerorge A. Herman, ed., Organizing Business Knowledge: The MIT Process Handbook, MIT Press, Cambridge, Massachusetts, USA, 2003, pp. 423-439.

[17] Klein, M. and Petti, C., "A Handbook-Based Methodology for Redesigning Business Processes," *Knowledge and Process Management,* Vol. 13, No. 2, 2006, pp. 108-119.

[18] Klusch, M., Fries, B., Khalid, M. and Sycara, K., "OWLS-MX: Hybrid OWL-S Service Matchmaking," *Proceedings of AAAI '05*, 2005.

[19] Klusch, M., Fries, B. and Sycara, K., "Automated Semantic Web Service Discovery with OWLS-MX," *Proceedings of AAMAS 2006*, Hakodate, Hokkaido, Japan, 2006.

[20] Levenshtein, V.I., "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, Vol. 10, 1966, pp. 707-710.

[21] Lin, D., "An Information-Theoretic Definition of Similarity," *Proceedings of the Fifth International Conference on Machine Learning (ICML '98)*, Madison, WI, 1998.

[22] Malone, T.W., Crowston, K. and Herman, G., ed., *Orgznizing Business Knoweldge: The MIT Process Handbook,* Cambridge, Massachusetts, USA: MIT Press, 2003.

[23] Malone, T.W., Crowston, K., Lee, J. and

Pentlad, B., "Tools for inventing organizations: Toward a handbook of organizational processes," *Management Science*, Vol. 45, No. 3, 1999, pp. 425-443.

[24] McCool, R., "Rethinking the Semantic Web, Part 1," *IEEE INTERNET COMPUTING*, Vol. 9, No. 6, 2005, pp. 86-88.

[25] Ouzzani, M. and Bouguettaya, A., "Efficient Access to Web Services," *IEEE Internet Computing*, Vol. 8, No. 2, 2004, pp. 34-44.

[26] Resnik, P., "Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language," *Journal of Artificial Intelligence Research*, Vol. 11, 1999, pp. 95-130.

[27] Sager, T., Bernstein, A., Pinzger, M. and Kiefer, C., "Detecting Similar Java Classes Using Tree Algorithms," *Proceedings of the 2006 International Workshop on Mining Software Repositories(MSR'06)*, Shanghai, China, 2006.

[28] Taivalsaari, A., "On the notion of inheritance," *ACM Computing Surveys*, Vol. 28, No. 3, 1996, pp. 438-479.

[29] Valiente, G., Algorithms on Trees and Graphs, Springer-Verlag, Berlin, 2002.

[30] Van der Aalst, W.M.P. and Basten, T., *Inheritance of Workflows: An approach to tackling problems related to change,* Technical report, Eindhoven University of Technology, 1999.

[31] Wang, Y. and Stroulia, E., "Semantic Structure Matching for Assessing Web-Service Similarity," *Proceedings of 1st International Conference on Service Oriented Computing,* Trento, Italy, 2003, pp. 194-207.

◆ 저자소개 ◆

이홍주 (Lee, Hong Joo)
가톨릭대학교 경영학부 전임강사로 재직 중이다. KAIST 산업경영학과에서 이학사, KAIST 테크노경영대학원에서 공학석사, 공학박사를 취득하였다. 2006년 Center for Collective Intelligene, MIT에서 Post doctoral fellow로 재직하였다. 주요 관심분야는 전자상거래, 개인화추천, 정보검색, 시맨틱 웹, 의사결정지원시스템, 데이터마이닝 응용 등이다.

Dr. Mark Klein (http://cci.mit.edu/klein/) is a Principal Research Scientist at the MIT Center for Collective Intelligence. His research focuses on understanding the cross-cutting fundamentals of coordination and applying these insights to help create better human organizations and software systems. He has made contributions in the areas of computer-supported conflict management for collaborative design, design rationale capture,business process re-design, exception handling in workflow and multi-agent software systems, service discovery, negotiation algorithms, understanding and resolving 'emergent' dysfunctions in distributed systems and, more recently, 'collective intelligence' systems to help people collaboratively solve complex problems like global warming.

◈ 이 논문은 2007년 09월 10일 접수하여 1차 수정을 거쳐 2008년 02월 21일 게재확정되었습니다.