시맨틱 웹 어플리케이션을 위한 제약 언어 SWCL의 응용

정균범[a], 김우주[b], 이명진[c]

[abc] *Department of Information and Industrial Engineering, Yonsei University*
134 Shinchon-Dong, Seodaemun-Gu, Seoul, 120-749, Korea
Tel: +82-2-2123-7754, Fax: +82-2-364-7807, E-mail:{leoeternal,wkim,xml}@yonsei.ac.kr

[abc] *연세대학교 공과대학 정보산업공학과*
서울 서대문구 신촌동 134
Tel: +82-2-2123-7754, Fax: +82-2-364-7807, E-mail:{leoeternal,wkim,xml}@yonsei.ac.kr

Abstract

가상기업(virtual enterprise)은 21세기에 고객의 요구가 매우 다양해지고 기술은 점점 더 복잡해지는 상황에서 정보네트워크 기술의 발전을 이용한 새로운 기업간 협력을 통해 극복하려는 경쟁전략의 일환이다. 현재까지의 가상기업에서는 주로 EDI(Electronic Data Interchange)를 이용한 국한된 정보의 교류가 이루어졌다. 그러나 이러한 국한된 정보이외에도 실제 가상조직 내에서는 규칙 또는 제약식 형태의 정보의 교류 또한 필요한 실정이다. 이러한 정보의 교류에 시맨틱 웹 플랫폼을 이용하게 되면 더욱 쉽고 유연한 정보의 전달이 가능하다. 현재 웹 온톨로지 언어인 OWL과 시맨틱 웹 규칙 언어인 SWRL을 통해서 사실 및 규칙의 포맷은 갖추어져 있는 상황이지만 제약식의 표현에 있어서는 뚜렷한 포맷이 정해져 있지 않아 가상조직 내에서의 구성원간의 제약조건의 교류에 문제가 있는 상황이다. 본 논문에서는 가상기업조직에 있어서 각각의 구성 기업들간의 제약식 공유가 이루어 질 수 있도록, 시맨틱 웹 표현의 대표적 언어인 OWL 기반으로 하는 제약식 표현 언어(Semantic Web Constraint Language)를 응용한다. 또한 실제로 이러한 SWCL이 가상기업 문제에 얼마나 효율적으로 적용될 수 있는지 증명하기 위해 실례를 적용해 보고자 한다.

Keywords:

가상조직, SWCL, 제약식, 교류, 시맨틱 웹, OWL

Introduction

오늘날 생산성의 급격한 향상, 기업조직의 거대화/복잡화, 소규모기업의 자원부족에 따른 한계 절감, 가격 · 제품정보 · 서비스 등 다차원적인 무한 경쟁, 시장 환경의 불확실성 증대, 정보기술과 시장의 빠른 변화 등으로 인해 기업들은 변화하지 않을 수 없게 되었다. 그 변화의 한가지 방향으로 등장한 것이 가상기업이다.

가상기업이란 공통의 목적을 위해 상호 보완적인 핵심역량을 보유한 기업간 또는 기업과 개인간에 계약에 의해 일시적으로 가치사슬을 형성하고, 각 조직의 효율성과 효과성을 향상시키는 방향으로 각 기업의 역량을 집중하고 상호지원을 공유하는 협력형태를 말한다.(한국전산원, 1999)

이러한 가상기업에서는 서로의 정보를 공유하는 것이 가장 중요한 것 중 하나라고 볼 수 있다. 현재 각 기업에서 가능한 생산량이나 현재 들어온 수요량과 같은 정보의 공유가 필요한 상황이 벌어진다. 이미 EDI를 이용한 국한된 정보의 교류가 이루어 지고 있으나 이는 표준화 되어있지 않은 특정한 기업간의 정보교류만 가능한 플랫폼이다. 이러한 상황에서 정보들 중 중요한 것은 각각의 제약식 형태로 볼 수 있는 정보들이다. 이러한 제약식 형태의 정보를 서로 공유하여 전체적인 가상기업 관점에서의 최적화가 필요한 것이다.

본 연구에서는 이러한 관점에서 여러가지 자원이나 제약식에 관한 정보의 공유가 특정한 EDI를 기반으로 하여 유연하지 못한 파트너쉽을 가지는 것보다 가상기업의 특성에 따라 여러 기업들이 서로의 조건이 맞는 기업끼리 가상기업을 구성하여 그들간의 정보의 교류가 더욱 효율적으로 이루어질 수 있도록 시맨틱 웹 환경을 도입하고 그에 따라 시맨틱 웹 제약 언어(SWCL)를 가상기업에 응용하여 가상기업 전체 관점에서의 최적화를 이룰 수 있는 방법을 제안하고자 한다.

이에 따라 2장에서는 가상기업에서 제약식 형태의 정보의 실례를 들어 설명하고, 3장에서는 이러한 제약식의 공유를 위해 응용하려 하는 SWCL에 대하여 설명하고자 한다. 4장에서는 이러한 SWCL이

실제 가상기업 문제에서 얼마나 효율적으로 기술될 수 있는지 증명하기 위하여 2장에서 설명한 예제를 더 자세하게 만들어 적용해 보도록 하겠다.

Example of Constraint satisfaction problem in Virtual Enterprise

본 장에서는 가상기업에서 일어날 수 있는 실제 제약식 만족 문제의 한 예를 들어 설명해 보고자 한다.

Virtual Enterprise의 예

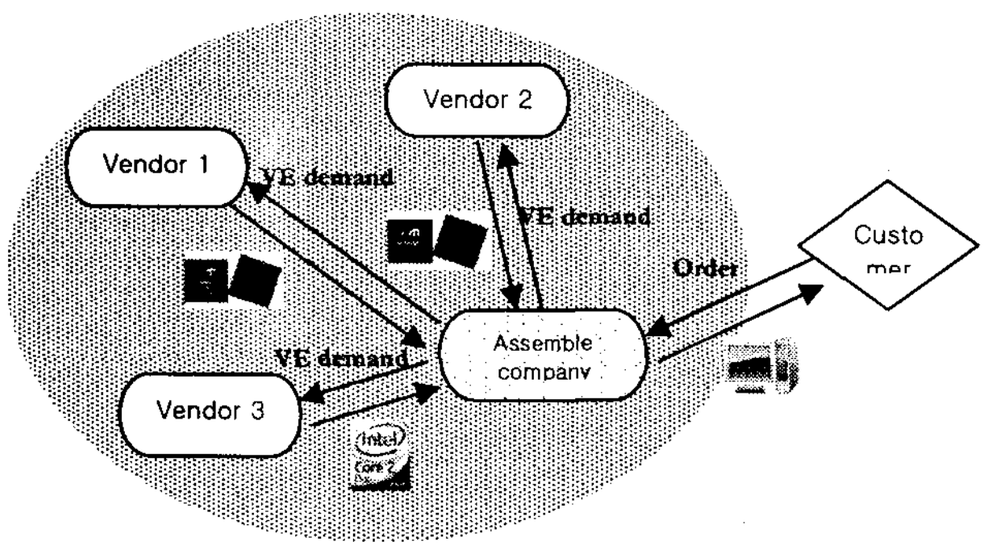

Figure 1 – Sample Virtual Enterprise

위의 그림에서 보듯이 이 예제는 고객이 하나의 컴퓨터 조립업체에 컴퓨터를 주문하면 이 조립업체를 중심으로 여러 개의 부품생산업체가 함께 가상기업을 구성하고 있는 형태이다.
컴퓨터는 CPU와 메모리 두 가지의 부품으로 이루어져 있는 것으로 가정하여 간단하게 만들었고, 부품생산업체 1과 2는 메모리 생산업체, 부품생산업체 3은 CPU 생산업체이다. 고객의 주문내용에 따른 조립업체 및 부품생산업체에서의 상황은 다음과 같다.

Spend / Produce	1st week		2nd week		3rd week		4th week		Capacity
	OWN	VE	OWN	VE	OWN	VE	OWN	VE	
1st week									cap_{1k}
2nd week									cap_{2k}
3rd week									cap_{3k}
4th week									cap_{4k}
Demand	DO_{1k}	dv_{1k}	DO_{2k}	dv_{2k}	DO_{3k}	dv_{3k}	DO_{4k}	dv_{4k}	

Figure 2 – Demand and Capacity in Vendor k

각 부품생산업체 및 조립업체는 주 단위로 생산할 수 있는 Capacity가 정해져 있고, 각 주 별로 얼마만큼씩 필요한가에 해당하는 주문이 고객으로부터 들어오게 되면 그에 따라 각 주에 생산해야 할 수요량이 정해지게 된다. 다만, 각 업체가 이미 다른 곳으로부터 받아둔 주문이 있다면 그 주문량까지 생산해야 할 수요량이 될 것이다. 또한 각 주에 생산된 부품 또는 제품은 일정량의 재고비용을 치르고 재고로써 생산 이후 다른 주의 수요량을 위해 사용될 수 있고, 수요는 앞으로 4주의 수요까지만 예측하여 생산 계획을 세우고자 한다. 그림2에서 DO_{jk} 는 j번째 주에 필요한 업체 k가 자체적으로 이미 다른 곳으로부터 받은 주문에 의한 수요량이고, dv_{jk} 는 j번째 주에 업체 k가 가상기업으로부터 받은 주문에 의한 수요량을 나타낸다. cap_{ik} 는 i번째 주에 생산할 수 있는 업체 k의 가능한 최대 생산량이다. 부품을 생산하여 조립업체에 보내면 조립업체가 조립하는데 1주일이 걸린다. 그러므로 1주차의 수요는 이미 부품생산업체에서 가지고 있던 재고를 바로 조립해서 만든 제품과 조립업체에서 가지고 있던 재고의 합으로써 만족시켜야 한다. 고객, 부품생산업체, 조립업체의 세부적인 제약식을 수식을 통해 표현하면 다음과 같다.

* Constraint from Vendor 1

$$\textit{For all } i, \quad \sum_{j \geq i}^{4} x_{ij1} \leq Cap_{i1} \qquad (1)$$

$$\textit{For all } j, \quad \sum_{i=1}^{j} x_{ij1} \geq DO_{j1} \qquad (2)$$

* Constraints from Vendor 2

$$\textit{For all } i, \quad \sum_{j \geq i}^{4} x_{ij2} \leq Cap_{i2} \qquad (3)$$

$$\textit{For all } j, \quad \sum_{i=1}^{j} x_{ij2} \geq DO_{j2} \qquad (4)$$

* Constraints from Vendor 3

$$\textit{For all } i, \quad \sum_{j \geq i}^{4} x_{ij3} \leq Cap_{i3} \qquad (5)$$

$$\textit{For all } j, \quad \sum_{i=1}^{j} x_{ij3} \geq DO_{j3} \qquad (6)$$

* Constraints from Vendor 4 (조립업체)

$$\textit{For all } i, \quad \sum_{j \geq i}^{4} x_{ij4} \leq Cap_{i4} \qquad (7)$$

$$\textit{For all } j, \quad \sum_{i=1}^{j} x_{ij4} \geq DO_{j4} + dv_{j4} \qquad (8)$$

* Constraints from through Virtual Enterprise

$$\textit{For all } j, \quad \sum_{i=1}^{j} (x_{ij1} + x_{ij2}) \geq DO_{j1} + DO_{j2} + dv_{j1} + dv_{j2} \qquad (9)$$

$$\textit{For all } j, \quad dv_{j1} + dv_{j2} = 2\sum_{i=1}^{j} x_{i(j+1)4} \qquad (10)$$

$$For\ all\ j,\quad \sum_{i=1}^{j} x_{ij3} \geq DO_{j3} + dv_{j3} \qquad (11)$$

$$j = 1,2,3,4,\quad dv_{j3} = \sum_{i=1}^{j+1} x_{i(j+1)4} \qquad (12)$$

$$Inv_4 + 0.5(Inv_1 + Inv_2) \geq dv_{14} \qquad (13)$$

$$Inv_4 + Inv_3 \geq dv_{14} \qquad (14)$$

각각의 업체 내에서 나올 수 있는 제약식들이 있는 반면 전체적인 가상기업 관점으로 가서 각 업체들의 정보가 공유되어야 만들어 지는 제약식들도 존재한다. 참고로 x_{ijk} 는 i번째 주에 생산하여 j번째 주에 소비되는 업체k에서 생산하는 생산량을 의미하고, Inv_k 는 업체k에서의 1주차 이전에 가지고 있는 재고량을 말한다.

Semantic Web Constraint Language (SWCL)

앞장에서 소개한 바와 같이 가상기업에서의 생산 계획 의사 결정과 관련된 문제 해결 과정에 있어서 제약식을 공유가 꼭 필요하다. 본 장에서는 제약식의 공유에 필요한 표준 포맷으로써 SWCL의 abstract syntax에 대하여 설명하고, 간단한 예제를 통해 SWCL 표현을 보다 잘 이해하고자 한다.

SWCL abstract Syntax와 예제

현재의 시맨틱 웹에서 로직을 기술하는 언어 외에, 데이터 모델과의 연결을 통한 제약식 표현 언어에는 한계가 있다. 본 절에서는 이에 따라 제안된 SWCL을 설명하며, 이러한 Syntax를 통하여 기존의 정보로 구성된 OWL 상의 컨셉이나 속성과 같은 데이터들을 제약식으로 어떻게 연결하여 표현할 수 있을 지를 살펴보고자 한다. 아래 예제는 OWL에 기술된 클래스와 프로퍼티를 통하여 표현할 수 있는 제약식을 설명하는 것이다.

OWL 상에 'Country' 와 'Province' 두 개의 클래스와 프로퍼티 'hasPart', 'populationValue' 가 있다. 특히, 'hasPart' 프로퍼티는 'Country' 와 'Province' 에 있어서 part-of 포함관계를 의미한다. 또한 'populationValue' 는 그 지역에서의 거주자 수를 표현하는 것이다. 이러한 내용을 OWL document로 기술하면 아래와 같다.

```
<owl:Class rdf:ID="Province"/>
<owl:Class rdf:ID="Country"/>
<owl:ObjectProperty rdf:ID="partOf">
  <rdfs:domain rdf:resource="#Province"/>
  <rdfs:range rdf:resource="#Country"/>
  <owl:inverseOf rdf:resource="#hasPart"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="hasPart">
  <rdfs:domain rdf:resource="#Country"/>
  <rdfs:range rdf:resource="#Province"/>
  <owl:inverseOf rdf:resource="#partOf"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID="populationValue">
  <rdfs:range rdf:resource="&xsd;positiveInteger"/>
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Country"/>
        <owl:Class rdf:about="#Province"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
</owl:DatatypeProperty>
```

위의 내용을 바탕으로 볼 때, Country의 인구수는 각 Province 들에서의 인구 수의 합과 같다는 것을 명백하게 고려해 볼 수 있다. 이러한 계산 원리를 통해 표현할 수 있는 제약식을 수식화 하여 기술하면 아래와 같다.

$$For\ all\quad y \in \text{Country},$$

$$\sum_{x \in y.hasPart} x.populationValue = y.populationValue \qquad (15)$$

OWL 및 SWRL 만으로는 위와 같은 계산과 관련된 수학적 조건식을 표현하는 것이 불가능하기 때문에, SWCL과 같은 제약식 표현 언어를 사용하고자 하는 것이다. SWCL을 통하여 수학적 표현을 기술하고 이를 시맨틱 웹으로 하여금 이해할 수 있도록 하는 것이 주된 목적이라 하겠다. 또한 OWL에 기술된 클래스 및 프로퍼티를 통하여 수식 표현을 데이터와 연결하는 의미에서 볼 때, SWCL은 OWL의 확장으로 이해할 수 있다.

다음은 SWCL을 구성하는 Syntax를 EBNF 형태로 정의한 것이다.

axiom ::= constraint

constraint axiom 은 qualifier, LHS(lef-hand side), operator, RHS(right-hand side) 네 가지로 구성되어 있다. 각각의 constraint를 구별하기 위해 URIreference를 필요에 따라 기술할 수 있을 것이다.

constraint ::= 'Constraint(' **[URIreference] { qualifier } LHS operator RHS** ')'

qualifier ::= 'Qualifier(' **variable** | **description** ')'

variable ::= 'Variable(' **description** ')'

LHS ::= 'LHS(' **termBlock { termBlock }** ')'

operator ::= 'equal' | 'notEqual' | 'lessThan' | 'lessThanOrEqual' | 'greaterThan' | 'greaterThanOrEqual'

RHS ::= 'RHS(' **termBlock { termBlock }** ')'

termBlock ::= 'TermBlock(' **sign [aggregateOperator] { parameter } factor { factor }** ')'

sign ::= '+' | '-'

aggregateOperator := 'Sigma' | 'Production'

factor ::= 'Factor(' **classID individualID datavaluedPropertyID** ')'

parameter ::= 'Parameter(' **variable** | **description** ')'

앞의 제약식 (1)번 예제를 통하여 살펴보면, operator '=' 를 기준으로 x.populationValue가 LHS로 term block이 되며, y.populationValue는 RHS로써 또 하나의 term block 으로 각각 기술될 수 있다. 여기서 qualifier는 Country 이다. 한편, LHS는 하나의 termBlock을 갖는데, 이는 sign '+', aggregateOperator 'Sigma', parameter Country.hasPart, factor Province.populationValue가 각 구성 요소가 된다. 여기서 aggreateOperator 와 parameter는 문제별 수리 계산에 따라 제약식에서 사용되지 않는 경우도 있을 것이다.

이상의 구성을 기반으로 RHS 부분을 이해하는 것 역시 마찬가지로 sign '+' 와 factor Country.populationValue로 termblock이 구성된 것이다.

이와 같은 제약식을 통한 지식이 문제 풀이에 직접적으로 활용되기 위해서는 의사 결정 모델링을 필요로 하게 된다. 따라서, 위에서 표현한 항목들을 통하여 다음과 같은 목적 함수 기술을 위한 부분이 필요할 것이다. 다음은 의사 결정과 관련한 최적화 모델을 표현하는 syntax인데 이는 크게 두 부분 목적 함수(objective)와 constraint 부분(subjectTo)으로 구성된다.

optModel ::= 'OptModel(' **objective subjectTo** ')'

objective ::= 'Objective(' **optimizationInstruction objectiveTerm** ')'

subjectTo ::= 'SubjectTo(' **{ constraint }** ')'

optimizationInstruction ::= 'Minimize' | 'Maximize'

objectiveTerm ::= datavaluedPropertyID | termBlock

앞서 설명한 듯이 OWL 상의 데이터와의 연결에 있어서, 이를 고려한 제약식 (1) 번 예제를 SWCL 문서로 나타내면 아래와 같다.

```
<swcl:Constraint rdf:ID="numberOfPopulation">
 <swcl:qualifier>
  <swcl:Variable rdf:id="y">
   <swcl:bindingClass rdf:resource="#Country"/>
  </swcl:Variable>
 </swcl:qualifier>
 <swcl:hasLHS>
  <swcl:TermBlock rdf:ID="termBlock_1">
<swcl:sign rdf:resource="&swcl;plus"/>
<swcl:aggregateOperator
      rdf:resource="&swcl;Sigma"/>
<swcl:parameter>
 <swcl:Variable rdf:id="x">
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty rdf:resource="#partOf"/>
    <owl:hasValue rdf:resource="#y"/>
   </owl:Restriction>
  </rdfs:subClassOf>
 </swcl:Variable>
</swcl:parameter>
<swcl:factor>
 <swcl:FactorAtom>
  <swcl:bindingClass rdf:resource="#x"/>
  <swcl:bindingDatatypeProperty
      rdf:resource="#populationValue"/>
 <swcl:FactorAtorm>
</swcl:factor>
```

```
  </swcl:TermBlock>
</swcl:hasLHS>
<swcl:hasOperator rdf:resource="&swcl;equal"/>
<swcl:hasRHS>
  <swcl:TermBlock rdf:ID="termBlock_2">
    <swcl:sign rdf:resource="&swcl;plus"/>
    <swcl:factor>
      <swcl:FactorAtom>
        <swcl:bindingClass rdf:resource="#y"/>
        <swcl:bindingDatatypeProperty
            rdf:resource="#populationValue"/>
      </swcl:FactorAtom>
    </swcl:factor>
  </swcl:TermBlock>
 </swcl:hasRHS>
</swcl:Constraint>
```

Apply SWCL to Virtual Enterprise Problem

본 장에서는 2장에서의 가상기업 예제를 더 구체화 시켜 SWCL에 적용하여 실제로 얼마나 제대로 적용될 수 있는지를 알아보도록 하겠다.

Virtual Enterprise Problem 예제 (Specific)

2장에서 설명한 가상기업의 예제에서 구체적으로 필요한 requirement와 capability들에 대한 가정은 다음과 같다.

*** Customer requirements**

- Product : Personal Computer
- BOM (Bill of Material)

  PC 1EA – consist of 2EA Memory, 1EA CPU

- Demand of PC

  1st week : 50, 2nd week : 40, 3rd week : 55,

  4th week : 60, 5th week : 50

*** Vendor 1 capabilities (Memory vendor)**

- Goods in stock : 30
- Capacity - 1st week : 70, 2nd week : 60,

  3rd week : 60, 4th week : 40

- Already accepted order - 1st week : 20, 3rd week : 10
- Price – 50$/EA, Stock cost – 5$/EA

*** Vendor 2 capabilities (Memory vendor)**

- Goods in stock : 40
- Capacity - 1st week : 50, 2nd week : 80,

  3rd week : 50, 4th week : 50

- Already accepted order - 1st week : 10, 2nd week : 10
- Price – 40$/EA, Stock cost – 10$/EA

*** Vendor 3 capabilities (CPU vendor)**

- Goods in stock : 70
- Capacity – 1st week : 40, 2nd week : 70,

  3rd week : 65, 4th week : 60

- Already accepted order – 2nd week : 10, 4th week : 10
- Price – 70$/EA, Stock cost – 5$/EA

*** Vendor 4 capabilities (Assemble company)**

- Spending time to assemble - 1 week
- Goods in stock : 10
- Capacity - 1st week : 40, 2nd week : 50, 3rd week : 70,

  4th week : 40, 5th week : 50

- Already accepted order - 2nd week : 20, 3rd week : 10
- Price – 30$/EA, Stock cost – 10$/EA

BOM에 의해서 PC 1대는 2개의 메모리와 1개의 CPU로 구성되어있는 것으로 가정하였고, 고객으로부터 받은 각 주별 수요는 5주까지 각각 정해져 있다. 조립업체에서 PC를 조립하는데 1주일이 걸리기 때문에 5주까지의 수요를 처리하는데는 각 부품업체들의 4주까지의 Capacity까지가 필요하게 된다. 또한 1주의 수요를 만족시키기 위해 그 전에 있었던 부품의 재고를 구체적으로 만들어 보았다.

이러한 구체적인 가상기업 예제로부터 먼저 온톨로지를 구성하게 되면 다음과 같이 구성할 수 있다.

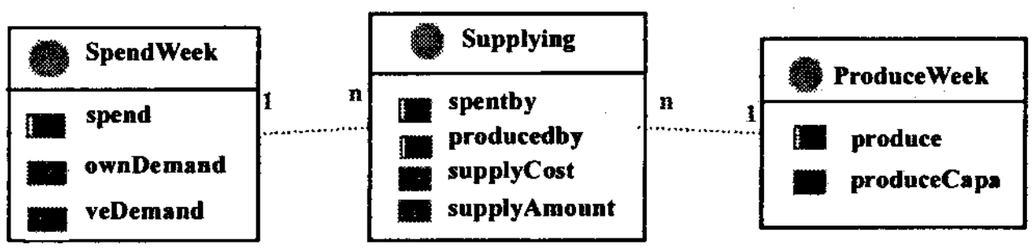

Figure 3 – Virtual Enterprise Ontology

2장의 Figure 2에서 본 바와 같이 각 업체별로 Transportation 문제의 형태를 띠는 표를 하나씩 만들어 낼 수 있다. 이를 온톨로지로 구성한 것이 Figure 3이다. SpendWeek에는 업체에서 미리 다른 곳으로부터 받은 주문에 의한 수요인 ownDemand와 가상기업으로부터 받은 주문에 의한 수요인 veDemand의 두가지 데이터 타입 프로퍼티가 존재하게 되고 ProduceWeek에는 업체의 주당

생산가능 Capacity 데이터 타입 프로퍼티가 존재한다. 각 SpendWeek과 ProduceWeek의 조합별로 공급가격인 supplyCost와 실제 공급하게 되는 양인 supplyAmount의 두 개의 데이터 타입 프로퍼티가 있는 Supplying이라는 클래스가 존재하게 된다.

이를 바탕으로 Vendor 1의 제약식을 다시 모델링 해 보면 다음과 같다.

For all z∈ ProduceWeek,

$$z.produceC\ apa \geq \sum_{w \in produce} w.sup\ plyAmount \quad (16)$$

For all x∈ SpendWeek,

$$x.ownDeman\ d \geq \sum_{v \in spend} v.sup\ plyAmount \quad (17)$$

이러한 온톨로지를 기반으로 한 제약식은 이 온톨로지를 표현해 놓은 OWL파일을 기반으로 SWCL로 연계되어 표현될 수 있다. 다음은 식 16을 SWCL로 표현한 것이다.

```
<swcl:Constraint rdf:ID= "Vendor_1">
<swcl:hasLHS rdf:parseType="Collection">
 <swcl:TermBlock rdf:ID="individualSupply">
  <swcl:Sign
     rdf:resource="http://iwec.yonsei.ac.kr/swcl#plus" />
   <swcl:Factor rdf:parseType="Collection">
     <swcl:FactorAtom>
       <swcl:Variable rdf:ID="z">
        <swcl:bindingClass
                    rdf:resource="#ProduceWeek" />
       </swcl:Variable>
       <swcl:bindingDatatypeProperty
                     rdf:resource="#produceCapa" />
     </swcl:FactorAtom>
   </swcl:Factor>
  </swcl:TermBlock>
</swcl:hasLHS>
<swcl:hasOperator
    rdf:resource="http://iwec.yonsei.ac.kr/swcl#greaterTh
    anOrEqual" />
<swcl:hasRHS rdf:parseType="Collection">
 <swcl:TermBlock rdf:ID="sumOfAmount_1">
    <swcl:Sign
    rdf:resource="http://iwec.yonsei.ac.kr/swcl#plus" />
<swcl:aggregateOperator
rdf:resource="http://iwec.yonsei.ac.kr/swcl#sigma" />
<swcl:Parameter>
 <swcl:Variable rdf:ID="w">
  <swcl:bindingClass rdf:resource= "#supplying" />
 </swcl:Variable>
</swcl:Parameter>
<swcl:Factor rdf:parseType="Collection">
 <swcl:FactorAtom>
  <swcl:bindingClass rdf:resource="#w" />
  <swcl:bindingDatatypeProperty
     rdf:resource="#supplyAmount" />
 </swcl:FactorAtom>
</swcl:Factor>
</swcl:TermBlock>
</swcl:hasRHS>
</swcl:Constraint>
```

이처럼 가상기업에서의 각각의 제약식은 OWL 온톨로지에 표현된 각각의 정보들과 그들간의 관계들을 기반으로 SWCL과 연계되어 표현이 쉽게 가능하다. 이렇게 표현된 SWCL문서를 실제 제약식 만족 문제 Solver인 Prolog의 문법으로 변환해 주면 Solver가 최적해를 돌려주게 된다. 위의 SWCL 표현을 Prolog문법으로 바꾸면 다음과 같다.

```
%%
myGoal :-
  myProb(L, TC),
  fd_minimize(fd_labeling(L), TC),
  write_sol(L), nl,
  write_fdvar('TC',TC), nl.
myProb(L, TC) :-
  fd_domain([X111, X121, X131, X141], 0, 70),
  fd_domain([X221, X231, X241], 0, 60),
  fd_domain([X331, X341], 0, 60),
  fd_domain([X441], 0, 40),
  %
  TC1 #= 50 * X111 + 55 * X121 + 60 * X131 + 65 *
        X141,
  TC2 #= 50 * X221 + 55 * X231 + 60 * X241,
  TC3 #= 50 * X331 + 55 * X341,
  TC4 #= 50 * X441,
```

```
%
20+dv11 #<= X111,
dv21 #<= X121 + X221,
10+dv31 #<= X131 + X231 + X331,
dv41 #<= X141 + X241 + X341 + X441,
70 #>= X111 + X121 + X131 + X141,
60 #>= X221 + X231 + X241,
60 #>= X331 + X341,
40 #>= X441,
```

실질적인 숫자가 들어간 부분은 OWL파일로부터 얻어지고 제약식 관련 부분은 SWCL문서로부터 변환되게 된다. 이러한 변환을 거쳐 만들어진 Prolog파일로써 제약식 만족 문제를 풀어 가상기업 전체적인 관점에서의 최적을 만들어 주는 각 업체들의 생산량을 알 수 있게 된다. 이로써 가상기업에서의 문제에 대해서 시맨틱 웹 환경을 적용하고 이것을 기반으로 SWCL을 이용한 정보의 공유를 이루어 마지막 Solver를 통해 최적화 문제를 푸는 단계까지 적용 가능하다는 사실을 입증하였다.

Conclusion

본 연구에서는 가상기업과 관련된 상황에서 서로의 정보를 편리하게 공유할 수 있도록 시맨틱 웹 환경을 제안하고 이를 기반으로 가상기업 내 각 기업들 간에 실제 공유되어야 하는 제약식들을 표현해 보았다. 이러한 맥락에서 제약식들을 SWCL로 표현하고 이를 기반으로 제약식을 하나로 모아 하나의 제약식 만족 문제로 만들어 실제 최적해를 구할 수 있는 기반이 될 수 있도록 하였다. 또한 실제 가상기업 예제를 SWCL로 표현해 보고 이를 Solver인 Prolog문법으로도 변환하여 실질적인 적용이 가능하다는 사실을 보였다. 가상기업 내 다양한 기업들의 정보들을 기반으로 가상기업 전체의 이익을 최대화 할 수 있는 조합을 가상기업에 제공할 수 있도록 하는 가상기업 의사 결정 모델링을 통하여 보다 효율적인 생산을 할 수 있도록 하였다.

한편 현재 SWCL로 표현되어있는 제약식 표현들을 실제 제약식 만족 문제를 풀기 위해 Prolog문법으로 바꾸는 데 있어 다소 불편함이 따르고 있으므로, SWCL문법으로 쓰여진 SWCL문서를 CSP Solver인 Prolog문법으로 자동으로 변환시켜주는 번역기 형태의 프로그램의 개발에 관한 연구가 추후 진행될 수 있을 것이다.

References

[1] http://www.w3.org/TR/owl-features/

[2] http://www.w3.org/Submission/SWRL/

[3] http://www.w3.org/Submission/SWSF-SWSL/

[4] http://www.w3.org/Submission/WRL/

[5] http://www.w3.org/2005/rules/wg

[6] http://iswc2003.semanticweb.org/invitedtalks.html

[7] http://iwec.yonsei.ac.kr/swcl/shoppingproblem.swcl

[8] Cesarano, C., d'Acierno, A., Picariello, A. (2003). "An Intelligent Search Agent System for Semantic Information Retrieval on the Internet," *Proceedings of the 5th ACM International Workshop on Web Information and Data Management*, pp. 111-117.

[9] Bangyong, L., Jie, T., Juanzi, L. (2005). "Association Search in Semantic Web: Search + Inference," *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pp. 992-993.

[10] Bazaraa, M.S., Jarvis, J.J. (1977). *Linear Programming and Network Flows*. Canada: John Wiley & Sons.

[11] Nemhauser, G.L., Wolsey, L.A. (1988). *Integer and Combinatorial Optimization*. Canada: John Wiley & Sons.

[12] Yeom, K., Lee, J.K. (1996). Logical Representation of Integer Programming Models, *Decision Support Systems 18,* pp. 227-251.

[13] Simon, H.A. (1960). *The New Science of Management Decision*. New York: Harper & Row.

[14] O'Keefe, R.M., McEachern, T. (1998). "Web-Based Customer Decision Support Systems," *Communications of the ACM 41(3).*

[15] http://www.w3.org/TR/rdf-sparql-query/

[16] Berners-Lee, T. (2001). *The Semantic Web*, Scientific American, Vol. 501.

[17] 김우주, 윤숙희, 이명진 (2006, 11). "시맨틱 웹 제약 언어 기반 쇼핑 의사 결정 지원 시스템 개발," *한국지능정보시스템학회, 2006 추계 학술대회 논문집*, pp. 113-121.