ISAAC : 문장분석용 통합시스템 및 사용자 인터페이스

김 곤[†] · 김 민 찬[††] · 배 재 학[†††] · 이 종 혁[††††]

요 약

본 논문에서 소개할 ISAAC(An Interface for Sentence Analysis & Abstraction with Cogitation)은 문장분석용 통합 사용자 인터페이스를 제공한다. 이 시스템에는 문장분석시 필요한 다양한 언어학적 도구와 자원이 통합되어 있다. 문장분석에 가용한 도구와 자원은 대부분 독립적으로 개발·축적된 것들이다. 이들을 활용한 문장분석의 경우, 단계적으로 얻어지는 문장분석 정보들을 문장분석가가 관리, 처리하기에는 어려움이 있다. 이에 본 논문에서는 가용 도구와 자원들을 통합하고, 각 기능들에 대해 사용자 중심의 일관된 인터페이스를 ISAAC이 제공하도록 하였다. 문장분석 처리과정은 총 14단계로 나눌 수 있었다. ISAAC에서는 이 단계들을 독립적인 기능을 가지는 4개의 모듈 - ① 문장의 통사구조 분석, ② 원형어휘 판별, ③ Roget 시소러스 범주정보 검색, ④ OfN(Ontology for Narratives) 범주정보 검색 - 로 처리하게 되어 있다. 따라서, ISAAC을 활용한 문장분석의 경우, 전체 14단계의 처리과정이 4개의 단계로 줄어든다. 이것은 문장분석가의 작업효율을 3.5배 이상 향상시킬 수 있음을 의미한다. 뿐만 아니라, 각 단계별 처리에 필요한 지루한 정보기록 이전작업을 ISAAC이 담당하게 함으로써 문장분석정보의 정확성도 높일 것으로 예상할 수 있다.

ISAAC : An Integrated System with User Interface for Sentence Analysis

Gon Kim[†] · Min-Chan Kim[††] · Jae-Hak J. Bae[†††] · Jong-Hyeok Lee[††††]

ABSTRACT

This paper introduces ISAAC (An Interface for Sentence Analysis & Abstraction with Cogitation) which provides an integrated user interface for sentence analysis. Into ISAAC, the various linguistic tools and resources are integrated. They are necessary for sentence analysis. Most of the tools and resources for sentence analysis are developed and accumulated independently. In the sentence analyzing with these tools and resources, it is difficult for sentence analyst to manage and control information which is taken on each step. In this respect, we have integrated the usable tools and resources, and made ISAAC to provide the consistent user oriented interface to each function. We have been able to divide sentence analysis process into 14 steps. In ISAAC, these steps are processed by four individual modules : ① syntactic analysis of sentence, ② retrieval of a root word, ③ searching category information in Roget's Thesaurus, and ④ searching category information in OfN(Ontology for Narratives). Therefore, in case of sentence analysis with ISAAC, the process of total 14 steps falls into 4 steps. This means that it is able to improve the performance of sentence analyst to the extent 3.5 times or more. Furthermore, ISAAC undertaking tedious transcription needed to process each step, we expect that ISAAC can help the analyst to maintain the accuracy of sentence analysis.

키워드 : 문장분석(Sentence Analysis), 통합 사용자 인터페이스(Integrated User Interface)

1. 서 론

자연어 처리과정에서 문장분석은 가용한 언어학적 도구와 자원을 활용한다. 대부분의 문장분석가들은 개별적인 도구와 자원을 활용하고 있다. 이들간 상호작용을 가능하게 하고 통합된 인터페이스를 제공한다면, 문장분석가는 작업효율을 높일 수 있을 것이고, 분석결과의 정확성도 향상시킬 수 있을 것이다.

1.1 연구동기

문장분석에는 다양한 언어학적 도구와 자원들이 필요하다. 가용한 도구와 자원의 대표적인 예로는 구문분석기나 유의어사전 등을 들 수 있다. 이러한 도구와 자원들은 대부분 독립적으로 개발되고 축적된 것들이다. 이들을 활용하는 문장분석 작업은 자동적인 처리가 가능한 부분도 있으나, 문장분석가가 수작업으로 해결해야 할 부분들도 있다. 이는 문장분석에 필요한 정보들을 관리하고 처리하기에 어려움이 있음을 의미한다. 따라서, 문장분석시 문장분석가의 효

† 종신회원 : 울산대학교 대학원 컴퓨터·정보통신공학부
†† 준 회 원 : 울산대학교 대학원 컴퓨터·정보통신공학부
††† 종신회원 : 울산대학교 컴퓨터·정보통신공학부 교수
†††† 정 회 원 : 포항공과대학교 컴퓨터공학과 교수
논문접수 : 2003년 10월 18일, 심사완료 : 2004년 1월 31일

율을 높이기 위해서는 개별적으로 존재하는 도구와 자원을 통합한 시스템과 연관된 정보를 정확하고 체계적으로 관리해줄 통합 인터페이스가 필요하다.

1.2 관련 연구

자연어처리 분야에서는 언어 지식 데이터베이스들을 재사용하고, 문장분석을 위해 활용할 수 있는 언어학적 도구나 자원에 대한 연구가 계속되어 왔다. 이러한 연구 결과들은 언어학적 도구로서 활용할 수 있는 가용자원으로 그 가치가 있다. 그러나, 각 연구의 특성 및 목적에 따라 그 결과들은 개별화되어 있고, 다양성을 띄고 있다. 이들을 통합하는 것은 자원재사용의 효과를 얻을 수 있고, 연구효율을 높일 수 있다. 또한, 지식 데이터베이스의 확장을 통해 부가적인 정보를 얻을 수 있다.

1.2.1 자원 재사용과 도구 통합

가용 자원의 재활용과 그 효율성을 높이기 위한 연구로 GATE(General Architecture for Text Engineering)[12]의 예가 있다. 이 시스템은 문법, 말뭉치사전, 시소러스와 같은 재사용 가능한 자원들을 관리하고 처리한다. GATE는 자원 또는 도구들의 관리에만 목적을 두는 것이 아니라 이들을 위한 응용프로그램을 통하여 사용자에게 보다 다양한 형태와 직관적인 인터페이스를 제공하고 있다[13]. GATE는 이종의 자연어처리 모듈을 제공한다. 이는 컴포넌트 기술의 연구에 기여하며, 컴포넌트의 재사용을 촉진한다[21]. 그러나, GATE는 문서의 정보 및 그 처리에 대한 호환성을 충분히 제공해 주지 못하고 있다. 첫째는, 문서에 관한 정보, 정보의 저장방법, 정보의 검색과 내부 기능간 상호작용 등을 표현하기 위한 호환성의 부족이다. 둘째는, 서로 다른 기능으로부터 사용되거나 생성되는 정보의 비호환성이다.

WhiteBoard[18] 아키텍처는 자연어처리시스템에서 이종의 컴포넌트들을 통합하는 방법에관한 연구이다. WhiteBoard는 각각의 컴포넌트를 관리하는 Manager와 시스템이 원하는 형태로 정보를 제공해주는 Coordinator로 구성되어 있다. WhiteBoard는 아키텍처의 견고성과 정보손실을 줄이는 것을 목적으로 하고 있다. 그러나, 여러 컴포넌트를 위해 각각 새로운 Manager를 작성해야할 뿐만 아니라, Manager의 정보를 시스템이 원하는 형태로 표현해야 하는 Coordinator를 구성해야 한다는 단점이 있다.

자연어처리에서 구문분석의 수행력을 개선하기 위한 한 방법으로 통계적 구문분석을 사용하고 있다. 통계적 구문분석의 향상된 결과를 얻기 위해 Probabilistic Partial Parsing과 Committee-Based Decision Making을 접목하고 있다[23]. 전자의 경우는 확률적으로 신빙성이 있는 부분만을 선택하는 방법이고, 후자는 더 나은 결정을 위하여 서로 다른 구문분석기로부터 얻은 구문분석 정보들을 취합하는 방법이다.

1.2.2 사용자 중심의 인터페이스

사용자 인터페이스의 중요성에 관해서는 사용자 중심의 시스템 즉, HCI(Human Computer Interfaces)[14]에 관한 연구가 있다. HCI의 기본원리는 사용자 편의에 중점을 두고 인터페이스가 설계되어야 한다는 것이다. 사용자 인터페이스 설계를 필요로 하는 작업들은 대개 다양하고 복잡한 단계들로 구성되어 있으며 반복적인 형태를 취하고 있다. 사용자 중심의 인터페이스는 표준화, 그래픽디자인, 기술문서, 성능, 구현시간 등의 균형을 유지하면서 설계되어야 한다. 이러한 개념을 적용하여 설계된 사용자 인터페이스는 반복적인 과정을 단축시키며, 작업의 능률을 향상시킬 수 있다는 장점을 가지고 있다. 그러나 HCI의 구현은 지속적인 컴퓨터 환경의 변화와 요구되는 시스템의 견고성, 다중언어의 지원, 복잡도의 해결, 부분 기능들의 모듈화 등 쉽게 해결하기 어려운 문제점을 안고 있다.

GUI(Graphical User Interface)는 소프트웨어간 상호작용을 수용하도록 설계되어져 왔다. 이는 사용자에게 편의성을 제공하지만, 사용자 관점의 편의성은 소프트웨어의 개발을 복잡하게 한다[14, 20, 22]. 사용자 인터페이스의 유용성평가(Usability Evaluation)[19]는 인터페이스의 설계과정에서 지속적으로 그 중요성이 증대되고 있다. 유용성은 사용자의 만족도를 촉진시키면서, 사용자가 원하는 목적을 효과적이고 효율적으로 성취할 수 있도록 한다. 이러한 유용성 평가에 수반되는 활동은 ① 작업 완료시간이나 오류와 같은 유용성 평가 자료의 취합, ② 이러한 자료를 인터페이스 측면에서 유용성 문제를 식별하기 위한 분석, ③ 문제점을 완화하기 위한 해결이나 개선의 제안이다.

본 논문에서는 관련연구에서 제시된 자원의 재사용 및 기능의 호환성, 이종의 컴포넌트들의 통합, 사용자 편의성 제공 등의 요구사항을 해결하고자, 문장분석용 통합 사용자 인터페이스를 구현하였다. 이를 위하여, 가용한 도구와 자원들을 통합하고 일관된 사용자 인터페이스를 제공하였다. 또한, 시스템 평가를 통하여, 문장분석의 수행력을 높일 수 있음을 보이고, 그 유용성을 확인해 보았다.

1.3 연구내용 및 기여도

본 논문에서 문장분석을 위해 활용하는 언어학적 도구와 자원은 ① 문장의 통사구조를 밝히기 위한 구문분석기로 LGPI+[1, 2, 4, 5], MINIPAR+[1], APPLE PIE[25], ② Roget 시소러스(Roget Thesaurus)[6]의 범주정보, ③ 이야기를 이해하기 위한 온톨러지 OfN(Ontology for Narratives)[3-5]의 정보 등이다.

LGP[7]은 입력문장에 대한 구문구조를 처리한다. 구문분석의 결과는 표식고리(Labeled Link)의 집합으로 문장의 통사구조가 표현된다. 표식고리는 한 쌍의 단어를 연결하며 그것들의 문법적인 기능을 표시한다. LGPI+는 이러한 LGP

구문분석 정보를 기계가독형으로 전환한다. 이는 문장의 주요구성성분들의 위치를 확인하여 문장추상화에 적용할 수 있도록 한다. 이와 더불어, 구문분석의 성공률을 높이고 LGP의 구문분석정보와 상호보완할 수 있도록 구문분석기 MINIPAR(A Minimalist Parser)[15]를 확장한 MINIPAR+와 APPLE PIE를 ISAAC(An Interface for Sentence Analysis & Abstraction with Cogitation)[1, 2]에 추가하였다.

주어진 문장에서 중요정보를 분별하고 이야기를 이해하기 위한 온톨러지로 OfN(Ontology for Narratives)을 사용한다. OfN은 문장추상화 과정에서 추출할 문장구성성분에 대한 선택기준을 제공한다. 온톨러지 OfN은 Roget 시소러스[6]의 범주를 재편성하고, 고유명사 자원[11]과 수사구조(Rhetorical Structure)의 연구결과[10]를 활용하여 구축하였다.

문장에서 나타나는 어휘는 주로 그 원형이 아닌 활용형으로 나타난다. 이러한 활용어휘들은 Roget 시소러스의 색인정보를 검색하는데 사용하기 위하여 어휘의 원형을 찾아주는 원형어휘판별과정을 거친다. 또한, 문장분석을 위한 말뭉치는 DearAbby[9] 상담문에서 발췌한 이야기를 기반으로 하였다.

본 논문에서 구현 및 개선한 문장분석용 통합시스템 및 인터페이스 ISAAC(An Interface for Sentence Analysis & Abstraction with Cogitation)은 문장분석에 필요한 언어학적 도구와 자원들을 통합하고 다양한 문장분석 정보들을 시각적이면서도 체계적으로 관리한다. 즉, 가용한 언어학적 도구와 자원들의 모든 기능들을 유지하면서 쉽고 단일한 사용자 인터페이스를 제공하고 있다. 이를 통하여, 사용자가 주어진 여러 문장들을 좀더 체계적이고 능률적으로 분석할 수 있도록 한다.

2. 활용한 도구와 자원

ISAAC은 문장분석을 위하여 구문분석기 LGPI+와 MINIPAR+, APPLE PIE, 온톨로지 OfN(Ontology for Narratives)을 사용한다. OfN은 로젯 시소러스(Roget's Thesaurus)를 심중사전(Lexicon)으로 삼아 이를 재구성하여 얻은 것이다. 또한, 로젯 시소러스의 색인정보 검색을 위하여 문장 어휘들의 원형을 찾는 원형어휘판별과정을 포함하고 있다. ISAAC의 말뭉치 자원으로는 DearAbby 상담문을 활용한다.

2.1 구문분석기 3개

기존의 ISAAC은 구문분석기로 LGPI를 확장한 LGPI+를 사용하고 있다. 문장분석의 과정에서는 구문분석기가 문장을 처리하지 못하거나 분석의 오류가 발생할 수 있다. 그러한 경우에 또 다른 구문분석기를 활용하여 구문분석의 성공률을 높이고, 서로 다른 구문분석기로부터 얻은 분석정보를 상호보완 할 수 있다. 이를 위하여, ISAAC에 두 개의 구문분석기 MiniPar+와 APPLE PIE를 추가하고, 각 구분분석기의 결과들을 기계가독형으로 전환하여 ISAAC의 기능을 개선하였다.

각 구문분석기의 출력정보를 기계가 가독하기에는 무리가 있다. 따라서, 구문분석기의 출력을 기계가독형으로 전환해 주는 함수가 필요하다. 이를 위해 적용한 입출력 알고리즘은 (그림 1)과 같다.

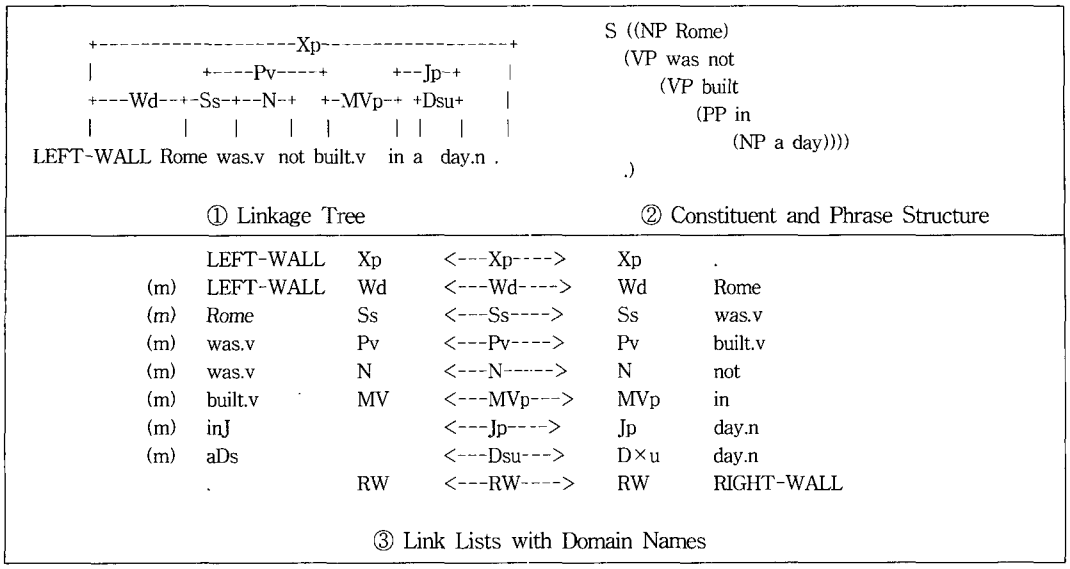
```
Constituent
    1. 입력된 문장의 각 Link정보 추출
    2. if Link_Label = Left-Wall 일 경우 Constitute 4. 수행
       else 해당링크의 Link Flag
    3. Constitute_Link에서 해당 Link정보
    4. 다음 Link정보로 이동(recursive call)
    5. Link Finish Marker 출력
    6. 결과출력
Constitute_Link
    1. if Link_Label 이 Left-Wall 이거나 Right-Wall 일 경우
                Wall_Display
       else
                Word 정보
    2. 선택된 Left Word의 Type 정보 추출
    3. Link의 Right Word 정보
    4. 선택된 Right Word의 Type 정보 추출
Constitute_Phase
    1. repeat 선택된 Node에서 모든 Child Node
    2. 선택된 Node에서 Phase Type
    3. if 선택된 Word가 포함하고 있는 정보가 없을 경우
                Word정보와 문장 내 위치정보, 현재 Phase 종결
       else
                if ChildNode에서 이하Node가 존재하지 않을 경우
                {현재 Phase 종결}
                현 Node의 하위 Node로 이동(recursive call)
                if 최종 Node일 경우 { Phase 종결 }
    4. end repeat
```

(그림 1) LGPI 구문분석 결과의 기계가독형 변환 알고리즘

2.1.1 LGPI+

LGPI+는 LGPI(Link Grammar Parser Interface[8])를 확장시킨 것이다. LGPI는 Link Grammar Parser[7]에 대한 SWI-Prolog API를 제공한다. LGP는 6만 어형을 수록한 사전을 내장하고 있으며 다양한 구문구조를 처리할 수 있다. 이 사전은 필요에 따라 확장이 가능하다. 입력문장에 대한 LGP 구문분석 결과는, 표식 고리(Labeled Link)의 집합으로 문장의 통사구조가 표현된다. 표식 고리는 한 쌍의 단어를 연결함과 아울러 그것들의 문법적인 기능을 표시한다.

(그림 2)의 구문분석 결과에서 ① *Linkage Tree*는 문장구성성분들간의 관계를 표식고리로써 나타내고 있다. ② *Constituent and Phrase Structure*는 주어진 문장을 구단위로 분석하고, 명사구, 동사구, 전치사구와 같은 형식의 문장 구성성분들을 보여준다. ③ *Link Lists with Domain Names*는 적용된 표식고리의 영역(Domain) 유형에 따라,



(그림 2) 예문의 구문분석 결과 : LGPI

문장에서 나타나는 단어간 표식고리를 명기한다. (그림 2)에서 확인되는 각 표식고리의 역할은 다음과 같다.

① *Xp*는 문장의 마침표와 좌벽을 연결시킨다.
② *Pv*는 수동문에서 Be동사와 과거분사를 연결시킨다.
③ *Jp*는 전치사와 그것의 복수형 목적어를 연결시킨다.
④ *Wd*는 평서문에서 주부를 좌벽에 연결시킨다.
⑤ *Ss*는 단수명사를 단수동사형과 연결시킨다.
⑥ *N*은 not과 선행동사를 연결시킨다.
⑦ *MVp*는 동사를 전치사와 연결시킨다.
⑧ *Dsu*는 부정관사와 명사를 연결시킨다.

(그림 3)은 위 예문에 대한 LGPI+의 출력결과이다. LGPI+는 LGP의 구문분석 결과를 기계가독형으로 바꾸어 주며, 문장구성성분들 간의 연결유형은 문장의 각 단어에 대한 색인번호로 그 대상을 지정한다.

```
linkage(010129,
[link([], connection(8-9, rw, '.'(_g110), right-wall(_g111))),
link([m], connection(6-7, ds, a(_g112), day(n))),
link([m], connection(5-7, j, in(_g113), day(n))),
link([m], connection(4-5, mv, built(v), in(_g114))),
link([m], connection(2-3, n, was(v), not(_g115))),
link([m], connection(2-4, pv, was(v), built(v))),
link([m], connection(1-2, ss, rome(_g116), was(v))),
link([m], connection(0-1, wd, left-wall(_g117), rome(_g118))),
link([], connection(0-8, xp, left-wall(_g119), '.'(_g120)))],
s(np(rome/1), vp(was/2, not/3, vp(built/4, pp(in/5, np(a/6,
day/7)))), '.'/8)
).
```

(그림 3) 예문의 구문분석 결과 : LGPI+

(그림 3)에서 7번째 줄은 "*was*"와 "*built*"간의 연결관계를 보여준다. *2-4*는 문장에서 두 단어가 나타나는 순서로 "*was*"가 2번째, "*built*"가 4번째에 나타남을 보여준다. *Pv*는 표식고리로 Be동사와 과거분사와의 연결유형이라는 걸 나타낸다. 그리고 *(v)*는 해당어휘가 문장에서 동사로 사용됨을 나타낸다. LGPI+는 문장에서 나타나는 문장구성성분간의 연결관계와 함께 각 구성성분들을 구(Phrase) 단위로 묶어 전체적인 구조를 보여준다.

2.1.2 MINIPAR+

MINIPAR는 적용범위가 방대한 구문분석기로 초당 300단어를 처리한다. MINIPAR는 어휘사전으로 WordNet을 활용하고 있으며, 각 단어간 관계를 13만개의 엔트리로 표현한다. 구문분석결과는 head와 modifier를 이용하여 문장의 통사구조를 표현하다. SUSANNE Corpus[7]로 평가하였을 때, MINIPAR는 단어들의 의존관계에 관하여 88%의 정확도와 80%의 재현율을 보인다.

```
(
E0 (() fin C * )
1 (Rome ~ N 4 s (gov "build in"))
2 (was be be 4 be (gov "build in"))
3 (not ~ A 2 neg (gov be))
4 (built "build in" V E0 i (gov fin))
5 (in ~ U 4 lex-mod (gov "build in"))
E2 () RomeN 4 obj (gov "buildin") (antecedent 1))
6 (a ~ Det 7 det (gov day))
7 (day ~ N 4 mod (gov "build in"))
)
```

(그림 4) 예문의 구문분석 결과 : MINIPAR

(그림 4)는 MINIPAR를 통하여 예문의 구문분석 결과를 확인한 것이다. 여기에는 각 단어들이 문장에서 나타나는

순서와 함께 단어에 대한 어휘정보 및 품사정보를 보여준다. (그림 4)에서 *"was"*는 문장에서 두 번째로 나타난다. *"was"*의 정보 중 첫 번째 *"be"*는 원형어휘를, 두 번째 *"be"*는 Be동사임을 나타낸다. *4*는 연결관계에 있는 단어가 문장에서 나타나는 순서를 의미하고, *"be"*는 두 단어간의 관계를 나타낸다. *(gov "build in")*는 의존적 관계에 있는 단어의 원형을 보여준다.

```
linkage(010129,
link(connection(1-4, s, Rome(N), build(V))),
link(connection(2-4, be, was(be), build(V))),
link(connection(3-2, neg, not(A), be(be))),
link(connection(5-6, det, a(Det), day(N))),
link(connection(6-4, obj2, day(N), build(V))),
S(i(built/4, s(Rome/1), be(was/2, neg(not/3)),
  obj2(day/6,
det(a/5)))).
```

(그림 5) 예문의 구문분석 결과 : MINIPAR+

(그림 5)에서 강조된 2번째 줄은 *"Rome"*과 *"built"*의 연결관계를 보여준다. *1-4*는 문장에서 두 단어가 나타나는 순서로 *"Rome"*이 첫 번째, *"built"*가 네 번째에 나타남을 보여준다. 또한 각 단어에는 품사정보가 명기되어 있다. *"Rome(N)"*의 경우에는 명사임을 나타낸다. MINIPAR에서는 단어의 원형으로 단어간 연결관계를 나타낸다. (그림 5)의 마지막에는 문장 구성성분들을 구(Phrase) 단위로 묶어 나타내고 있다.

2.1.3 APPLE PIE

APPLE PIE는 PTB(Penn TreeBank[27])에서 제공하는 Grammar와 Dictionary를 가지고 있는 말뭉치 기반의 Chart Parser이다. 이것은 최적 우선 탐색 기법으로 파싱의 진행과정을 모두 기록하면서 가장 적합한 Parse Tree를 찾는 상향식 구문분석기이다. PTB는 4백 5십만개에 달하는 영어 단어에 대한 태그 정보를 가지는 말뭉치이다. (그림 6)은 APPLE PIE를 이용하여 주어진 문장의 구문분석 과정에서 탐색한 트리의 모든 경로를 나타낸다.

```
[ 0, 1] :      0(  0)  -{   -1}→  =S=
[ 1, 2] : NNPX(  0)  -{   -1}→  rome
[ 1, 2] :   NPL( 28)  -{40846}→  NNPX[1 2]
[ 1, 2] :    SS( 82)  -{25363}→  NPL[1 2]
[ 1, 2] :    NP( 69)  -{17837}→  NPL[1 2]
[ 1, 2] :     S(114)  -{ 3259}→  NPL[1 2]
[ 1, 3] :   NPL( 96)  -{41251}→  NNPX[1 2] VBX[2 3]
[ 1, 3] :    SS( 62)  -{28334}→  NPL[1 2]  VBX[2 3]
[ 1, 3] :    NP(122)  -{21309}→  NPL[1 2]  VBX[2 3]
[ 1, 3] :     S( 92)  -{ 5358}→  NPL[1 2]  VBX[2 3]
                                 ...
```

(그림 6) Parse Tree 탐색 경로

```
(S (NPL Rome) (VP was not (VP built (PP in (NPL a day))))
  -PERIOD-)
```

(그림 7) 예문의 구문분석 결과 : APPLE PIE

(그림 7)은 (그림 6)에서 탐색된 경로를 바탕으로 문장을 구(Phrase) 구조로 표현한다. 그림에서 NPL은 그 하부에 NP(Noun Phrase)를 가지고 있지 않는 최하위의 NP이다. SS는 최상위 문장이 아닌 sub S(Sentence)를 나타낸다.

2.2 Roget 시소러스와 OfN(Ontology for Narratives)

Roget 시소러스는 총 6개의 의미 분류에 기초한 강(Class)으로 구성되어 있다. 각 분류는 하부에 부(Division), 과(Section) 등의 계층구조로 세분화 되어있다. 각 계층은 저마다의 표제정보를 가지고 있으며 계층구조의 말단에는 총 1044개의 범주가 존재한다. 각 범주에는 품사별 유의어 목록이 나열되어 있다.

문장에서 중요정보를 분별하고 이야기를 이해하기 위한 온톨러지(Ontology, 존재론) OfN은 다음의 7가지 범주로 구성된다 : 등장인물(Character), 심상(Affect State), 사건(Event), 상태(State), 시간과 공간의 변화(Delta-{Time, Space}), 담화표지(Discourse Marker). 설정된 OfN을 구축하기 위해서 먼저 Roget 시소러스의 범주를 심상, 시간과 공간, 사건, 그리고 상태 등으로 재편성하였다. 등장인물 유형에 속하는 어휘들은 고유명사 자원[11]을 이용하여 선정하였다. 담화표지의 경우는 수사구조의 연구결과[10]를 활용하였다. 이와는 달리 시공의 변화는 구문분석 후 문장 구성성분간의 상호작용에 의하여 확인되는 유형인 바, 그 기본유형은 시간과 공간이다.

2.3 어형변화 처리

문장의 통사구조 분석을 통하여 문장을 구성하는 어휘들을 분리한다. 문장의 어휘들은 주로 그 원형이 아닌 복수형, 과거형, 불규칙 동사와 같은 활용형으로 나타난다. 이러한 활용형들은 어형변화 처리를 통하여 그 원형을 찾는다. 원형어휘는 Roget 시소러스의 색인정보 검색에 사용된다. 문장의 각 어휘에 대한 원형을 찾아 Roget 시소러스의 범주정보를 구하는 것은 그 어휘에 대한 OfN 범주정보를 찾기 위함이다.

2.4 이야기 말뭉치 : DearAbby

ISAAC은 말뭉치로서 Dearabby[7]에서 발췌한 상담문을 저장하고 있다. DearAbby는 전 세계적으로 상당히 잘 알려져 있는 인생상담 기고란의 이름이다. 상담 이야기는 건강, 금전, 애정, 가족갈등, 대인관계, 학교생활, 인생진로 등에

관한 내용으로, 글의 유형은 개인적인 경험에 속한다.

3. 문장분석용 통합시스템 및 사용자 인터페이스

ISAAC에는 여러 가지 언어학적 자원과 도구들이 통합되어있다. ISAAC은 다음의 구성요소들로 이루어져 있다. ① 구문분석기가 밝혀낸 문장의 통사구조, ② Roget 시소러스의 범주정보, ③ 그 범주를 재편성한 온톨러지 OfN 정보, 그리고 ④ 번역정보 등이 그것이다.

3.1 문장분석 과정

문장분석용 통합 사용자 인터페이스 ISAAC의 처리과정을 간략하게 살펴보면 다음과 같다. 저장되어 있는 말뭉치에서 한 문장을 선택하면 원문과 번역문이 표시된다. 그리고 원문을 구문분석기로 처리하여 저장된 결과를 표시한다. 다음으로 분리된 어휘들의 어형을 판별, 처리하여 Roget 시소러스 범주 색인정보를 찾는다. 이 색인정보에서 추출한 Roget 시소러스 범주에 대한 OfN 범주정보를 결정한다. 이러한 과정을 도식하면 (그림 8)과 같다.

ISAAC의 문장분석 처리과정은 ① 문장의 의미파악을 위한 '문장의 통사구조 분석단계', ② Roget 시소러스와 OfN 검색을 위한 '단어의 원형어휘 판별단계', ③ 원형어휘에 대한 'Roget 시소러스 범주정보 추출단계', ④ 원형어휘에 대한 'OfN 범주정보 추출단계'로 나누어진다.

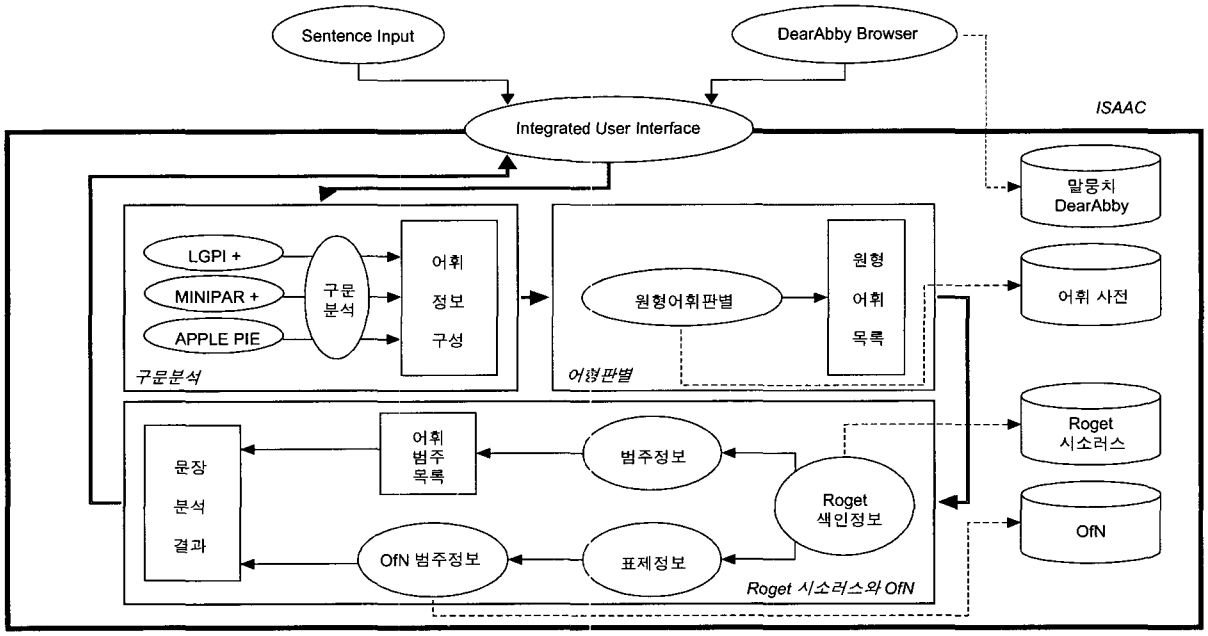
*문장의 통사구조 분석단계*는 문장의 의미분석을 위한 선행단계로 문장을 구성하는 어휘들의 정보를 얻고 문장 구성성분들 간의 문법적인 관계를 파악하는 과정이다. 이를 위해, 구문분석기인 LGP와 LGPI+, MiniPar+, APPLE PIE를 이용한다. 구문분석 결과는 기계 가독형으로 문장의 통사구조를 나타낸다. 이 과정에서 오류가 존재할 경우에는 LGP의 결과를 토대로 문장의 정규화 및 LPG에 수록된 어형사전을 참조하여 단어대용 과정을 거친다.

*원형어휘 판별단계*는 단어에 대한 Roget 시소러스정보를 검색, 추출하기 위하여 그 원형을 찾는 과정이다. ISAAC은 단어의 원형을 찾기 위해 과거형, 복수형, 불규칙 어휘 사전을 구축하고, 해당 단어의 원형어휘를 우선 검색한다. 검색된 원형어휘로 Roget 시소러스 범주정보를 얻는다. 이것을 바탕으로 해당 원형어휘에 대한 OfN 범주정보를 추출한다.

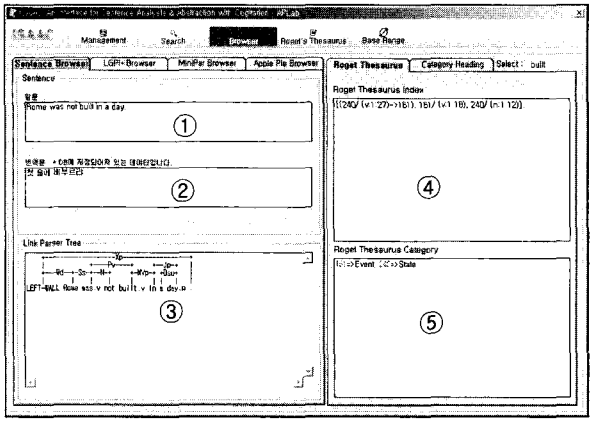
3.2 통합 인터페이스의 구성

ISAAC은 문장 탐색기, 문장분석기, 로젯 시소러스 및 온톨러지 OfN 검색기 등으로 이루어져 있다. (그림 9)는 구현한 통합 사용자 인터페이스 ISAAC의 실행화면이다.

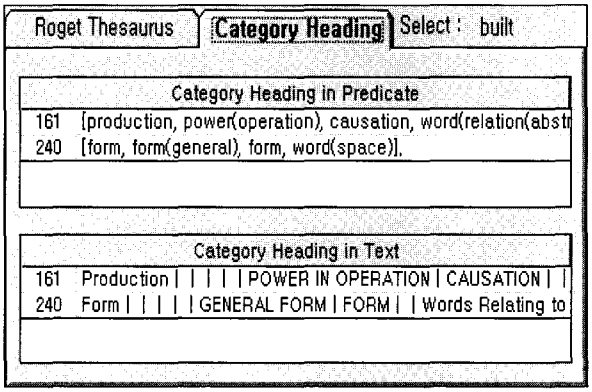
(그림 9)에서 ①은 입력문장을 나타내며, 'Dear Abby' 상담문 중에서 한 문장을 선택할 수 있다. ②는 ①의 문장에 대한 한글 번역문이다. ③은 입력문장 ①을 LGPI로 분석한 결과로써 문장의 통사구조를 보여준다. 여기에서는 문장구성성분들 간의 관계를 표식고리로써 나타낸다. 또한, 입력문장 ①에 대한 MiniPar+와 APPLE PIE의 구문분석결과도 볼 수 있다. ④는 문장에서 선택된 단어의 Roget 시소러스 범주 참조정보이며 ⑤는 ④에서 나열된 범주들을 참조정보를 이용하여 재분류한 OfN 정보로써 Closer 우선순위를 적용하였다. 아래 (그림 10)은 텍스트 기반의 범주 표제정보로 저장되어 있는 것을 Prolog 술어형태로 변환하여 표현한 것이다.



(그림 8) 문장분석 처리과정 : ISAAC



(그림 9) ISAAC



(그림 10) 범주 표제정보

(그림 11)은 예문에 대한 LGPI+와 MINIPAR+, APPLE PIE의 구문분석 결과이다. ①은 LGPI+, ②는 MINIPAR+, ③은 APPLE PIE의 구문분석 결과이다.

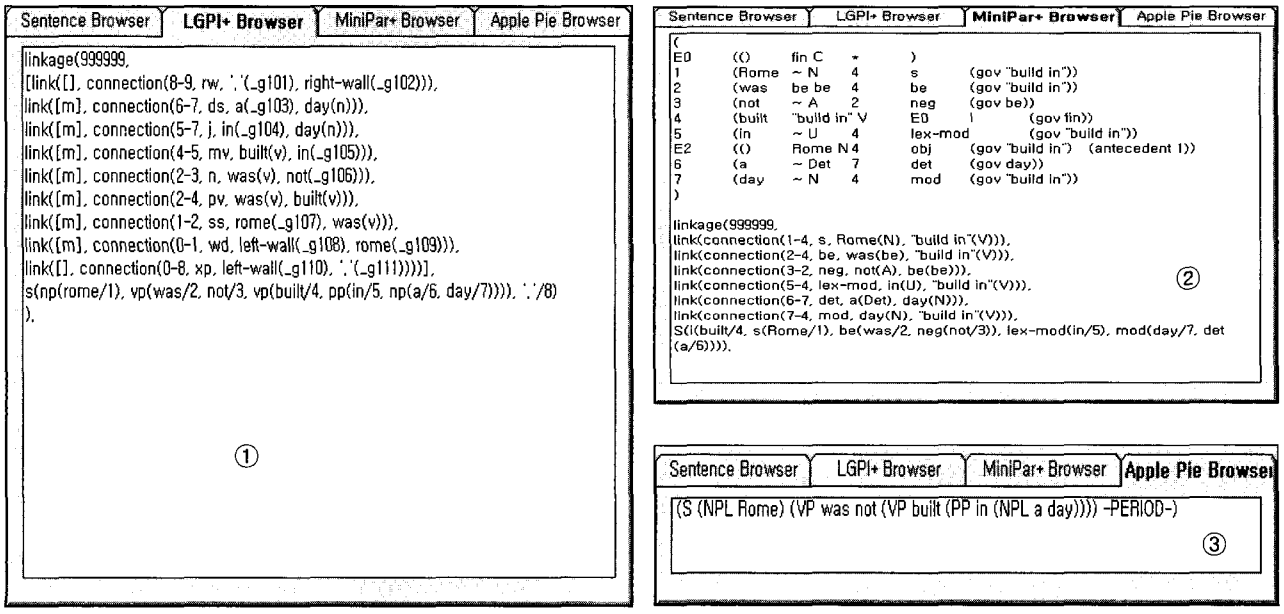
3.3 문장분석가의 작업효율 평가

문장분석 처리과정은 총 14단계로 나눌 수 있었다. I-SAAC에서는 이 단계들을 독립적인 기능을 가지는 4개의 모듈 - ① 문장의 통사구조 분석, ② 원형어휘 판별, ③ Roget 시소러스 범주정보 검색, ④ OfN 범주정보 검색 - 로 처리하게 되어 있다.

문장의 통사구조 분석단계는 문장을 의미적으로 분석하기 위한 선행단계로 문장을 구성하는 어휘들의 정보를 얻고 문장 구성성분들 간의 문법적인 관계를 파악하는 과정이다. 이를 위해, 구문분석기 LGP와 LGPI+를 이용한다. 본 논문에서는 이와 함께, 구문분석기 MINIPAR+와 APPLE PIE를 추가하여 활용하였다. 구문분석 결과는 기계가독형으로 전환되어 문장의 통사구조를 나타낸다. 이 과정에서 오류가 존재할 경우에는 LGP의 결과를 토대로 문장의 정규화 및 LPG에 수록된 어형사전을 참조하여 단어대용 과정을 거친다. 이러한 문장정규화나 단어대용 작업은 수작업으로 문장을 분석하는 경우에 가장 많은 시간이 소요되는 부분이다. 또한, 문장분석가의 판단오류가 존재할 가능성이 높은 단계이다. 이는 분석결과를 얻기까지의 효율성과 그 결과의 신뢰도에 영향을 미친다.

원형어휘 판별단계는 Roget 시소러스정보를 검색, 추출하기 위해 필요한 과정이다. 단어의 원형어휘를 찾기 위해 과거형, 복수형, 불규칙 등의 활용어휘 사전을 구축하고 그 단어의 원형을 우선 검색한다. 검색된 원형어휘는 Roget 시소러스 범주정보를 얻기위해 사용된다. 이렇게 추출된 Roget 시소러스 범주정보를 이용하여 OfN 범주정보를 결정한다.

ISAAC을 활용한 문장분석과 수작업으로 했을 때를 비교



(그림 11) 문장의 구문분석 결과 : LGPI+, MINIPAR+, APPLE PIE

해 봄으로써 문장분석가의 작업효율을 확인하였다. <표 1>은 문장분석의 단계별로 수작업과 ISAAC의 경우를 비교분석 한 것이다. 평가를 위한 말뭉치로는 DearAbby를 사용하였다. <표 1>에서 ●은 수작업으로 문장을 분석하는 과정에서는 존재하였으나, ISAAC에서는 통합되거나 내부적으로 처리되어 사라진 단계를 나타낸다.

〈표 1〉 ISAAC의 효율성 평가

분석과정	단계별 분석과정		수작업	ISAAC	비 고
문장의 통사구조 분석	1	LGPI+분석	○	○	1~3단계는 ISAAC에서 단일 과정으로 처리
	2	분석과정 오류	○		
	3	LGPI 분석	○		
	4	어휘 정보 기록	○	●	
원형어휘 판별	5	원형어휘 검색	○	●	ISAAC 활용어사전으로 각 어휘사전을 통합.
	6	과거형 어휘 검색	○		
	7	복수형 어휘 검색	○		
	8	불규칙 어휘 검색	○		
	9	(원형어휘 재검색)	○		
	10	원형어휘 검색 및 기록	○	●	
Roget 시소러스 범주정보	11	Roget 시소러스 검색	○	○	Roget 시소러스와 OfN의 검색, 추출과 기록은 ISAAC에서 실시간 처리
	12	범주정보 추출 및 기록	○	●	
OfN 범주정보	13	각 범주에 대한 OfN 검색	○	○	
	14	OfN범주정보 기록	○	●	

<표 1>에서 확인되는 바, ISAAC을 활용한 문장분석은 각 단계별 작업에서 얻어지는 정보의 기록 여부에 따라 반복적이거나 수동적인 작업을 줄일 수 있다. 문장분석 과정은 선행단계의 결과를 다음단계에서 필요로 한다. 수작업으로 하는 문장분석의 경우에는, 이러한 정보들을 수기나 파일의 형태로 기록·참조하게 된다. 이 과정에서 인적오류로 인하여 불확실한 정보의 기록, 정보의 손실, 판단의 오류 등이 생길 수 있다. 이는 문장분석의 효율성을 떨어뜨리거나, 분석결과의 신뢰도를 낮게 할 수 있다. 이러한 점은 문장분석의 모든 단계에 영향을 미치게 된다.

그러나, ISAAC을 통한 문장분석은 전체 14단계의 처리과정이 4개의 단계로 줄어든다. 수작업에서 발생할 수 있는 오류를 줄이고 반복적인 검색이나, 유의어사전 및 어휘사전을 이용한 Roget 시소러스 범주정보와 OfN 범주정보의 선택 등의 작업에서 관련정보를 내부적으로 처리한다. 이것은 문장분석가의 작업효율을 3.5배 이상 향상시킬 수 있음을 의미한다. 문장분석 소요시간의 경우에는 정량적으로 확인하기는 어려우나 분석단계의 감소비율보다 더 높은 효율을 가져올 것을 예상할 수 있다. 뿐만 아니라, 각 단계별 처리에 필요한 지루한 정보기록 이전작업을 ISAAC이 담당하게 함으로써 문장분석정보의 정확성도 높일 것으로 예상할 수 있다.

4. 결 론

문장분석은 다양한 언어학적 도구와 자원을 필요로 한다. 활용되는 도구와 자원들은 대부분 개별적으로 개발되고 축적된 것들이다. 이러한 도구와 자원을 이용하여 문장분석 정보들을 단계적으로 관리하고 처리하기에는 어려움이 있다. 또한 문장분석가의 반복적인 작업은 문장분석의 효율성을 떨어뜨린다.

본 논문에서는 이러한 점에 착안하여 가용한 도구와 자원을 통합하여 편리한 사용자 인터페이스를 제공하도록 하였다. ISAAC은 문장의 통사구조를 밝히는 구문분석기로 LGPI+, MINIPAR+, APPLE PIE를 활용한다. 이와 더불어, 온톨러지 OfN을 사용한다. OfN은 Roget 시소러스를 어휘사전으로 삼아 범주를 재편성하고 고유명사자원과 수사구조의 연구결과를 활용하여 얻은 것이다. ISAAC은 이러한 개별적으로 존재하는 언어학적 자원과 도구들의 모든 기능들을 유지하면서 문장분석 정보들을 시각화하고 체계적으로 관리할 수 있게 한다. 구문분석기 MINIPAR+와 APPLE PIE는 구문분석의 성공률과 문장 분석정보의 상호보완성을 높이기 위하여 ISAAC에 추가한 것이다. 또한 각 기능간 문장분석정보의 공유와 상호작용을 위해 ISAAC을 구성하는 도구와 자원간 입·출력 변환 알고리즘을 적용하였다.

개선한 ISAAC을 평가하기 위하여 수작업으로 하는 문장분석과 비교해 보았다. 그 결과, ISAAC을 이용한 문장분석의 경우, 수작업에서 반복적으로 행해지던 분석단계들을 독립적인 기능을 가지는 모듈로 처리함으로써 문장분석에 소요되는 시간을 줄일 수 있었다. 구체적으로는 총 14개의 문장분석 단계들이 독립적인 기능을 가지는 4개의 모듈 - ① 문장의 통사구조 분석, ② 원형어휘 판별, ③ Roget 시소러스 범주정보 검색, ④ OfN 범주정보 검색 - 로 처리된다. 이는 문장분석가의 작업효율을 3.5배 이상 높일 수 있음을 알 수 있다. 뿐만 아니라, 각 단계별 처리에 필요한 지루한 정보기록 이전작업을 ISAAC이 담당하게 함으로써 문장분석정보의 정확성도 높일 것으로 예상할 수 있다.

본 논문을 통하여 구현한 문장분석용 통합 사용자 인터페이스 ISAAC은 문장분석에 필요한 도구와 자원을 통합하고 편리한 사용자 인터페이스를 제공함으로써 문장분석가의 작업 효율을 높일 수 있음을 알 수 있었다. 향후 연구로는 ① 구문분석기의 추가 및 ② 다수의 구문분석기로부터 얻는 분석정보의 정확도를 결정하는 문제, ③ 온톨러지 OfN을 활용한 문장의 의미분석 정보로부터 문장추상화 결과 도출, ④ 이들을 활용한 문단의 주제문 선별[3] 등을 생각할 수 있다. 이러한 ISAAC의 활용은 궁극적으로 이해기반의 문서요약[4]에도 그 효용가치가 있을 것으로 기대된다.

참 고 문 헌

[1] 김곤, 김민찬, 배재학, 유해영, 이종혁, "문장분석용 통합사용자 인터페이스 ISAAC의 개선", 한국정보처리학회 춘계학술발표대회 논문집, 제10권 제1호, pp.325-328. 2003.

[2] 김명수, 김민찬, 배재학, "문장분석에 활용할 종합적인 사용자 인터페이스", 한국정보처리학회 춘계학술발표대회, 제9권 제1호, pp.535-538, 2002.

[3] 양재군, 배재학, "온톨로지 정보를 이용한 범주 재편성 : Roget 시소러스의 경우", 한국정보처리학회 춘계학술발표대회, 제9권 제1호, pp.515-518, 2002.

[4] Bae, J.-H. J. and Lee, J.-H. "Topic Sentence Selection with Mid-Depth Understanding," Proc. of ICCPOL, pp. 199-204, 2001.

[5] Bae, J.-H. J. and Lee, J.-H. "Mid-Depth Text Understanding by Abductive Chains for Topic Sentence Selection," IJCPOL, Vol.15, No.3, pp.341-357, 2002.

[6] Roget's Thesaurus. http://promo.net/cgi-promo/pg/t9.cgi?entry=22&full=yes&ftpsite=ftp://ibiblio.org/pub/docs/books/gutenberg/.

[7] Link Grammar. http://www.link.cs.cmu.edu/link/.

[8] SWI-Prolog. http://www.swi-prolog.org/.

[9] DearAbby. http://www.dearabby.com/.

[10] D. Marcu, "From Discourse Structures to Text Summaries," in Proc. ACL'97 and EACL'97 Workshop on Intelligent Scalable Text Summarization, Madrid, Spain, pp. 82-88, 1997.

[11] Proper Names Wordlist. http://clr.nmsu.edu/cgi-bin/Tools/CLR/clrcat#I4.

[12] H. Cunningham, Y. Wilks and R. Gaizauskas. "GATE - a General Architecture for Text Engineering," *In proceedings of the 16th Conference on Computational Linguistics (COLING-96)*, Copenhagen, August, 1996.

[13] K. Bontcheva, H. Brugman, A. Russel, P. Wittenburg and H. Cunningham. "An Experiment in Unifying Audio-*Visual and Textual Infrastructures for Language* Processing R&D," *In Workshop on Using Toolsets and Architectures To Build NLP Systems at COLING-2000*, Luxembourg, 2000.

[14] B. A. Myers, "Why are Human-Computer Interfaces Difficult to Design and Implement?," Technical report CS-93-183, Carnegie Mellon University, School of Computer Science, July, 1993.

[15] MINIPAR. http://www.cs.ualberta.ca/~lindek/.

[16] D. Lin, "Dependency-based Evaluation of MINIPAR," *In Workshop on the Evaluation of Parsing Systems*, Granada, Spain, May, 1998.

[17] SUSANNE Corpus. http://www.grsampson.net/.

[18] Boitet C. and M. Seligman. "The Whiteboard Architecture : A Way to Integrate Heterogeneous Components of NLP Systems," Proceedings of COLING, 1994.

[19] Melody Y. Ivory and Marti A. Hearst, "The state of the art in automating usability evaluation of user interfaces," ACM Computing Surveys, Vol.33, No.4, pp.470-516, 2001.

[20] W. I. Wittel, Jr. and T. G. Lewis, "Integrating the MVC paradigm into an object-oriented framework to accelerate GUI Computer Science," Tech. Rep. 91-60-06, Department of Computer Science, Oregon State University, 1991.

[21] H. Cunningham, K. Humphreys, R. Gaizauskas and Y. Wilks. "Software Infrastructure for Natural Language Processing," *In Proceedings of the Fifth Conference on Applied Natural Language Processing*, (ANLP-97), Mar, 1997.

[22] Atif M. Memon, Martha E. Pollack and Mary Lou Soffa, "Hierarchical GUI Test Case Generation Using Automated Planning," IEEE Transactions on Software Engineering (TSE), Vol.27, No.2, pp.144-155, 2001.

[23] INUI Takashi and INUI Kentaro, "Committee-based Decision Making in Probabilistic Partial Parsing," The 18th International Conference on Computational Linguistics, 2000.

[24] Jensen, K., G. E. Heidorn, and S. D. Richardson., Natural Language P processing : The PLNLP Approach., Kluwer Academic Publishers, 1993.

[25] Apple Pie Parser(version 5.7). http://nlp.cs.nyu.edu/app/.

[26] Satoshi Sekine, '*Corpus-based Parsing and Sublanguage Studies*,' Ph.D. thesis, New York University, 1998.

[27] Penn TreeBank. http://www.cis.upenn.edu/~treebank/home.html.

김 곤

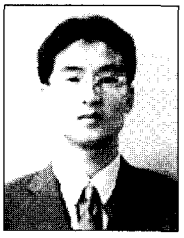
e-mail : gonkim@ulsan.ac.kr

1997년 울산대학교 전자계산학과(공학사)

2000년 울산대학교 대학원 컴퓨터·정보통신공학부(공학석사)

2002년~현재 울산대학교 대학원 컴퓨터·정보통신공학부 박사과정 중

관심분야 : 자동프로그래밍, 인공지능, 문서요약, 전자상거래



김 민 찬

e-mail : tomatuli@ulsan.ac.kr

2002년 울산대학교 컴퓨터·정보통신공학부 (공학사)

2002년~현재 울산대학교 대학원 컴퓨터·정보통신공학부 석사과정 중

관심분야 : 자동프로그래밍, 인공지능, 문서요약, WFMS

배 재 학

e-mail : jhjbae@ulsan.ac.kr

1981년 중앙대학교 전자계산학과(이학사)

1983년 한국과학기술원 전산학과 (공학석사)

2003년 포항공과대학교 컴퓨터공학과 (공학박사)

1985년~현재 울산대학교 컴퓨터·정보통신공학부 교수

관심분야 : 자동문서요약, (자동, 논리)프로그래밍, 지식경영 및 기술, 전략경영정보시스템, 교육인적자원정보시스템

이 종 혁

e-mail : jhlee@postech.ac.kr

1980년 서울대학교 수학교육과(이학사).

1982년 한국과학기술원 전산학과(공학석사)

1988년 한국과학기술원 컴퓨터공학과 (공학박사).

1991년~현재 포항공과대학교 컴퓨터공학과 교수

관심분야 : Natural Language Processing, Computer Processing of the Korean Language, Machine Translation between Japanese, Chinese, English and Korean, Cross-Language IR, Text Summarization, Text Classification