SLA 구현방법 비교: ASLM과 SLA@SOI

김상락*, 양재군*, 배재학*1), 장길상**
*울산대학교 전기공학부
**울산대학교 경영정보학과
e-mail:{shem0304, jgyang, jhjbae, gsjang}@ulsan.ac.kr

A Comparison of SLA Implementations: ASLM and SLA@SOI

Sang-Rak Kim*, Jae-Gun Yang*, Jae-Hak J. Bae*, Gil-Sang Jang**
*School of Electrical Engineering, University of Ulsan
**Dept. of Management Information System, University of Ulsan

요 약

최근 클라우드 컴퓨팅 기반의 IT 서비스가 증가하면서 기업들 사이에 SLA가 중요한 화두로 대두되고 있다. 상업용 SLA 관리 툴들은 어플리케이션 또는 데이터베이스 계층에 계약관련 로직을 내장하고 있다. SLA 규칙들이 구현 로직 내에 암시적으로 표현되어 있어서 유지보수가 어렵고 계약사항을 공유하는 데 많은 어려움이 따른다. 따라서 오늘날과 같은 서비스 지향 아키텍처, 온 디맨드, 유틸리티컴퓨팅, 클라우드 컴퓨팅 환경에서는 전체 서비스를 자동 관리할 수 있는 기술에 대한 연구가 반드시 필요하다. 본 논문에서는 능동문서기반의 ASLM(Active Service Level Management)과 유럽에서 진행 중인 SLA@SOI 프로젝트의 계약서 작성 및 실행에 대한 구현 방법을 비교하였다. 비교결과 유지보수성과 자동화 측면을 고려할 때 계약 규칙 로직을 어플리케이션 로직과 분리한 ASLM의 구현방법이 SLA@SOI 보다 SLA 계약처리 업무에 적합함을 알 수 있었다.

1. 서론

최근 클라우드 컴퓨팅 기반의 IT 서비스가 증가하면서 기업들 사이에 SLA(Service Level Agreement)가 중요한 화두로 대두되고 있다. SLA는 IT 서비스 수준의 정량적 측정을 통해 서비스 운영 성과를 평가하기 위하여 IT 서비스 제공자와 사용자가 요구되는 서비스 수준을 정의하고 이를 문서화한 계약 체계이다[1].

최근 상업적인 SLA 관리 툴들은 어플리케이션 또는 데이터베이스 계층에 계약 규칙을 내장하고 있다. 즉 다시 말하면 SLA 규칙들이 암시적으로 표현되어 있고 때로는 수정할 수 없는 경우도 있다. 이러한 구현 방식은 분산된 계약서를 상호 교환하기가 어려울 뿐만 아니라 수많은 SLA 계약업무들을 자동화하기가 어렵다. 기업들 사이에 SLA 계약을 체결할 때 대부분 일반 문서를 사용하고 있고, 거의 모든 계약들이 QoS(Quality of Service)에 초점을 두고 있다. 하지만 계약 이행에 대한 품질을 정량적으로 평가할 수 있는 정책과 평가도구는 클라우드 컴퓨팅 환경 확산에 필수적이다. 따라서 오늘날과 같은 서비스 지향 아키텍처, 온 디맨드, 유틸리티컴퓨팅, 클라우드 컴퓨팅 환경에서는 전체 서비스를 자동 관리할 수 있는 기술에 대한 연구가 반드시 필요하다. 본 논문에서는 능동문서[2] 기반의 SLA 구현 방법과 유럽에서 진행중인 SLA@SOI(Service Level Agreement@Service Oriented Infrastructure)[3]의 SLA의 여러 단계 중 계약 단계의 계약서 작성 및 실행에 대한 구현 방법을 비교한다.

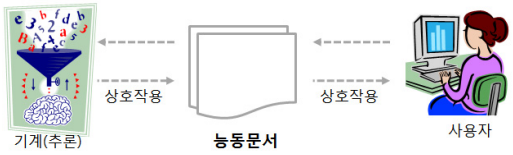
1) 교신저자

2. 관련 연구

본 논문과 관련 있는 능동문서, RBSLA[4], SLA@SOI 내용을 세부적으로 살펴보면 다음과 같다.

2.1 능동문서

(그림 1)은 능동문서 처리 흐름이다. 능동문서 시스템은 문서를 중심으로 사용자와의 상호작용을 지원하는 시스템이다. 이는 문서양식과 절차를 내포하는 행동 가능한 문서를 통해 지능적인 웹 어플리케이션을 구현하고자 하는 접근방법이다.

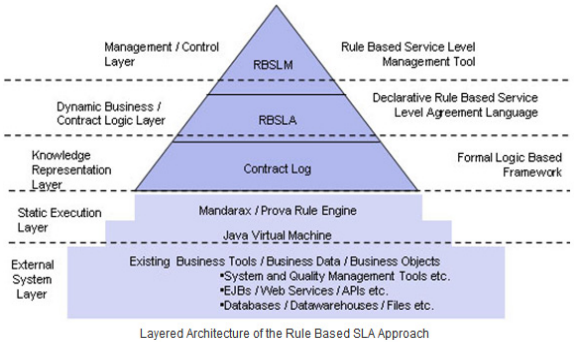


(그림 1) 능동문서 처리 흐름

2.2 RBSLA

RBSLA(Rule-based Service Level Agreements)는 전자계약, SLA 등의 계약관리 서비스를 효율적으로 지원하기 위한 프레임워크이다. 본 프레임워크 구성은 (그림 2)와 같다. 서비스의 이행상황을 자동적으로 감지하여 계약 내용 위반 시 자동으로 관련 담당자들에게 통지하여 신속한

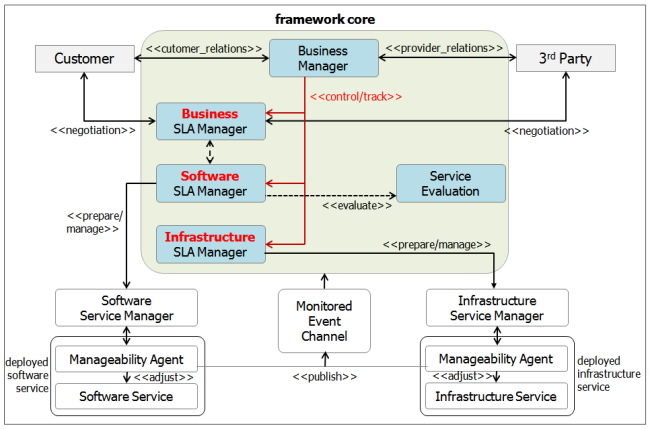
복구를 취할 수 있도록 하고 있으며, 계약 수준을 실시간으로 모니터링 할 수 있는 도구도 내장하고 있다.



(그림 2) RBSLA 아키텍처 계층구조

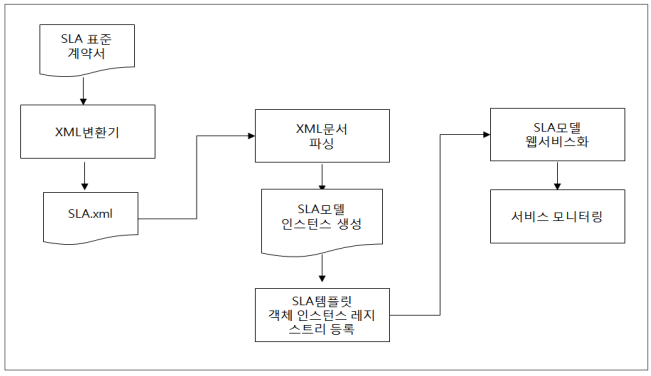
2.3 SLA@SOI

(그림 3)은 SLA@SOI 시스템 아키텍처이다. 서비스공급자와 서비스 수요자는 SLA 서비스 내용을 실시간으로 모니터링 할 수 있다. 서비스 내용이 서비스 수요자와 합의한 수준에 도달하지 못할 경우에 이벤트를 발생시켜 관련 담당자들에게 통지하여 정해진 절차에 따라 조치를 할 수 있도록 한다. Business Manager는 서비스종류, 서비스 정책, 가격, 평가, 보고서, 서비스 이해관계자 등을 관리한다. Software SLA Manager는 소프트웨어에 관한 서비스 관리 및 예외발생 처리를 담당한다. Infrastructure SLA Manager는 인프라 자원에 대한 서비스 관리 및 예외발생 처리를 담당한다[5].



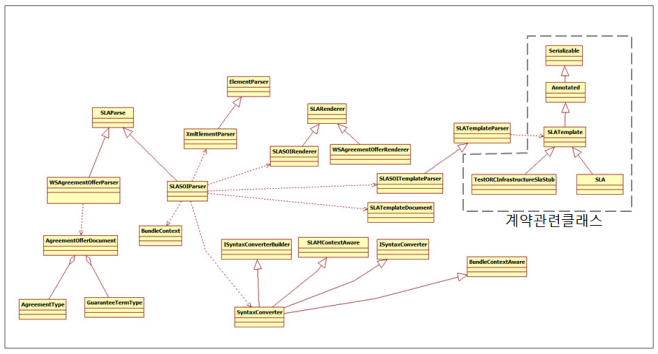
(그림 3) SLA@SOI 시스템 아키텍처

(그림 4)는 자연언어로 작성된 SLA계약서를 실행 가능한 언어로 변환하는 흐름을 표현한 다이어그램이다. 자연언어로 작성된 계약서를 XML 변환기를 사용하여 SLA.xml 문서를 만든다. XML문서를 파싱하여 SLA 모델 객체를 생성한다. SLA 모델을 객체 레지스터리에 등록하고, 그 모델을 웹서비스화한다. 웹서비스 모듈은 SLA@SOI 시스템에서 서비스 이행 상황을 판단하는데 사용한다.



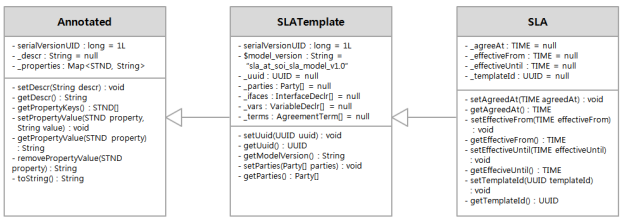
(그림 4) SLA XML 파일을 자바코드로 변환 다이어그램

(그림 5)는 SLA 계약문서 XML 파일을 자바 클래스로 맵핑하는 클래스와 SLA 문서를 파싱하여 SLA 클래스 인스턴스로 만들어 주는 클래스 다이어그램이다.



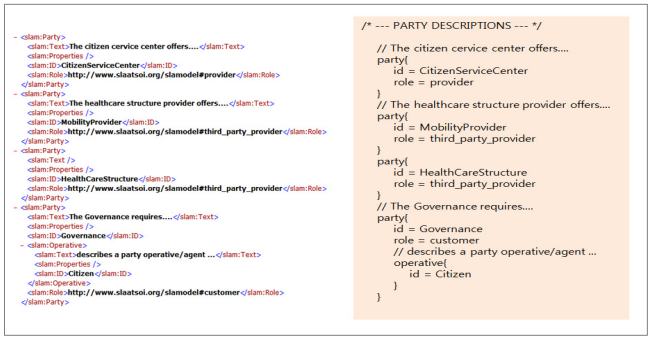
(그림 5) SLA@SOI 계약 관련 클래스 다이어그램

SLA 계약문서 관련 클래스는 멤버변수와 멤버함수는 (그림 6)과 같다. SLA 클래스는 SLATemplate 클래스와 Annotated 클래스를 상속받아 SLA 계약문서 XML 파일 정보를 저장한다.



(그림 6) SLA 클래스 계층도

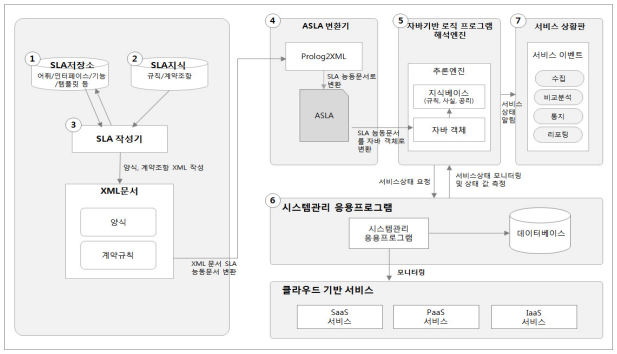
SLA@SOI에서 사용하는 SLA 계약문서는 (그림 5)와 같이 SLATemplate 클래스를 상속받아 XML로 작성한 SLA 계약문서를 자바코드로 변환한다. (그림 7)은 XML로 작성한 SLA 계약문서를 자바코드로 변환한 내용 일부이다. SLA@SOI의 XML로 만든 SLA 계약문서 내용 중에서 서비스와 관련 있는 당사자들에 대한 정보를 SLA 모델 자바클래스에 맵핑한 정보를 보여주고 있다.



(그림 7) SLA@SOI의 SLA 계약문서 자바코드 변환

3. ASLM

능동형 SLM(ASLM, Active Service Level Management)은 능동문서를 기반으로하는 SLA 관리 시스템이다. ASLM의 아키텍처는 (그림 8)과 같다.



(그림 8) ASLM 시스템 아키텍처

SLA 저장소(1)는 어휘, 인터페이스, 기능등의 정보를 가지고 있고 SLA 지식(2)은 규칙과 계약조항을 가진다. SLA 작성기(3)는 SLA 서식과 SLA 규칙을 포함하는 XML을 작성한다. 능동형 SLA(ASLA, Active Service Level Agreement) 변환기(4)는 프롤로그를 표현한 ERML(Excutable Rule Markup Language)[6]에 서식과 지식을 추가하여 ASLA로 변환한다. 자바 기반 로직 프로그램 해석엔진(5)은 형식, 논리 계약 규칙을 실행하는 컴포넌트와 추론엔진 역할을 한다. 시스템관리 응용프로그램은(6) SLA 서비스를 실행, 이행, 모니터링을 가능하게 한다. 서비스 상황판(7)은 모니터링 상황을 한 눈에 파악할 수 있도록 가시화 한다. 또한 이행사항 위반, 미터링, 과금, 통지서비스 등 SLM 프로세스를 지원한다.

ASLM 시스템에서 핵심영역(1-4)인 ERML을 이용한 실행 가능한 규칙마크업 언어는 다음과 같은 5단계를 거쳐 처리된다: (단계1) 규칙의 자연어 표현, (단계2) 자연어 표현을 Prolog 규칙으로 표현, (단계3) Prolog 규칙을 ASLA로 변환, (단계4) 추론엔진에서 사용 가능한 규칙생성, (단계5) 추론을 위한 질의를 실행한다.

(그림 9)는 SLA 규칙의 자연어 표현 문장이다. 서비스 이행 미준수시 조치해야 하는 내용이다. (그림 10)의 프롤로그 표현내용을 ERML을 이용하여 실행 가능한 언어로 변환한 후 서식과 지식을 추가하여 ASLA로 변환한다[7].

- 오전 8시부터 오후 6시까지 서비스 이용이 정상적으로 수행되고 있는지 매분마다 확인한다.
- 서비스가 제대로 이용되지 않을 경우 프로세스 사무원에게 통보한다.

(그림 9) 서비스 계약 규칙

```
escalate(S) :-
    sysTime(T),                 % notify
    not(holdsAt(unavailable(S),T)),
                                % escalate only once
    add("outages","happens(outage(_0),_1).",
        [S,T]),                 % add event to KB
    notify("process_manager", unavailable(S)).
```

(그림 10) 서비스 계약 내용 Prolog 표현

(그림 10)은 서비스 계약 내용 Prolog 표현이고 그림(11)은 RuleML 기술을 응용한 ERML로 변환한 것이다.

```
<?xml version="1.0" encoding="euc-kr"?>
  <RuleSet>
    <hn>
      <relationship>
        <relator> :- </relator>
        <relationship>
          <relator> escalate </relator>
          <var> S </var>
        </relationship>
        <relationship>
          <relator> , </relator>
          <relationship>
            <relator> sysTime </relator>
            <var> T </var>
            .....
```

(그림 11) Prolog 표현을 ERML로 변환

4. 구현방법 비교

4.1 서비스 구조

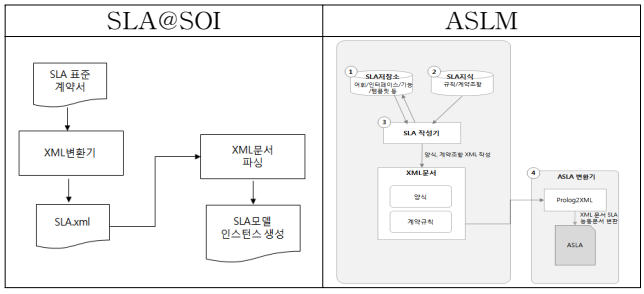
(그림 12)는 SLA@SOI와 ASLM의 SLA 계약문서 변환 아키텍처를 비교한 그림이다. SLA@SOI는 자연어로 작성된 계약문서를 XML변환기를 이용하여 규칙에 맞게 변환한다. 계약문서 XML을 실행상태로 만들기 위해 자바로 작성된 xml2java 변환 모듈을 사용하여 변환 작업을 수행한다. ASLM은 SLA저장소의 정보와 SLA지식을 이용하여 SLA 작성기를 통해 서식과 규칙을 포함한 실행 가능한 XML 문서인 ASLA를 만든다. ASLA을 이용하여 서비스 계약상황 확인, 모니터링, 과금 등의 업무를 자동화할 수 있다.

4.2 서비스 기술

<표 1>은 SLA@SOI와 ASLM 개발 기술을 비교하여 요약한 표이다. SLA@SOI는 SLA 상용화 제품을 만들 목적으로 유럽에서 수행한 프로젝트의 결과물인데 반해 ASLM은 컴퓨터가 웹 정보자원의 의미를 이해하고, 정보

<표 1> 서비스 기술 비교

구분		SLA@SOI	ASLM
배경 및 현황		• 실물경제 중심 사회에서 서비스경제 중심 사회로의 변화 • 서비스경제 중심 사회에서 반드시 필요한 도구	• 컴퓨터가 웹 정보자원의 의미를 이해하고, 정보의 검색, 추출, 해석, 가공등 제반 처리를 사용자 대신할 수 있는 웹 개발에서 출발 • 서식, 지식, 규칙을 내장한 ASLA을 플랫폼 독립적인 환경에서 실행할 수 있는 연구
특징		• 계약 템플릿 변경시 자바코드 수정 과다 발생	• 유지보수성, 확장성, 자동화 측면에서 효율적임.
적용기술	프로토콜	SOAP/HTTP	HTTP
	서비스명세	WSDL	ASLA
	서비스 레지스트리	UDDI	제한 없음
	지식표현	BDD(Binary Decision Diagrams)	RDF, OWL, RuleML, ERML
	사용언어	Java, XML, BNF(Backus - Naur Form)	Java, XML, Prolog
주요 적용 분야		IaaS, PaaS, SaaS 등 클라우드 서비스 분야	서비스 계약 및 과금 자동화 분야
현재의 문제점		계약의 규칙 로직이 어플리케이션 내에 하드코딩 되어 유지보수성과 자동화 측면에서 불편	계약의 규칙 로직을 어플리케이션 로직과 분리시켜 계약변경에 따른 시스템 유지관리가 용이



(그림 12) 계약문서 변환 서비스 구조 비교

의 검색, 추출, 해석, 가공 등 제반 처리를 사용자 대신할 수 있는 웹 개발에서 출발하였다. SLA@SOI는 지식에 대한 표현을 위해 절차적 언어를 사용하였고, ASLM은 로직 프로그래밍 언어인 프롤로그를 사용하였다. 본 연구의 범위로 제한을 두고 두 개의 기술을 비교한다면 규칙 변경에 따른 작업의 용이성 및 자동화 측면에서 ASLM이 뛰어나다.

5. 결론

본 논문에서는 클라우드 서비스 확산에 중요한 도구인 SLA 시스템에 관한 연구를 하였다. SLA 시스템인 ASLM과 SLA@SOI의 가장 중요한 차이점은 SLA 업무 중에서 프로비저닝의 자동화 여부이다. ASLM SLA 시스템은 로직 프로그램 언어인 Prolog를 사용하여 SLA 프로비저닝을 자동화시켰다. 하지만 SLA@SOI는 SLA 표준 템플릿을 자바 객체화하여 템플릿이 변경되면 모든 관련 자바 클래스를 수정해야 하는 문제점을 가지고 있다. 또한 SLA@SOI는 어플리케이션 내에 업무규칙이 하드코딩되어 있어서 규칙 변경 발생 시 어플리케이션 소스를 일일이 변경해야 한다. 본 연구는 SLA 시스템 자동화 도구 개발의 초기단계이다. 추가적인 연구로는 계약 규칙을 모니터링 하는 체계에 대한 연구와 초기계약과 이행실적에 따른 과금체계 자동화에 대한 연구, 소프트웨어와 하드웨어 중요 측정지표들을 자동으로 측정하는 소프트웨어 에이전트에 관한 연구 등이다. 클라우드 컴퓨팅은 IT산업의 새로운 패러다임으로 IT환경의 혁신적인 변화의 주역이다. 하지만 서비스 관리 도구의 부재는 이러한 기술의 지속적인 발전에 걸림돌이 될 것이다. 따라서 본 논문에서 연구하고 있는 SLA 시스템은 클라우드 컴퓨팅의 중요한 서비스 도구가 될 것이다.

감사의 글

본 연구는 정보통신산업진흥원의 IT/SW 창의연구과정의 연구결과로 지식경제부와 에스케이씨앤씨 주식회사에 의해 지원된 과제로 수행되었음 (NIPA-2010-1405))

참고문헌

[1] 서한준, 여명구, “국내외 SLA/SLM 추진 사례”, 한국정보산업연합회, 2004.

[2] 남철기, 배재학, 장길상, “능동문서에 대한 새로운 접근법과 그 응용”, 한국정보과학회논문지, 제30권 제3·4호, pp.347-353, 2003.

[3] SLA@SOI, http://sla-at-soi.eu.

[4] RBSLA, http://ibis.in.tum.de/projects/rbsla.

[5] SLA@SOI-Overview, http://sla-at-soi.eu/wp-content/uploads/2010/10/SLA@SOI-Overview.pdf.

[6] 남철기, 배재학, “업무규칙을 포함한 능동문서”, 한국정보과학회 2002년도 봄 학술대회논문집 제29권 제1호(A), pp.352-354, 2002.

[7] 김상락, 장선아, 양재군, 배재학 “지능형 서비스 계약문서 처리 프레임워크”, 2010년도 한국산업경영시스템학회 추계학술대회 논문집 p. 39, 2010.