AHP를 통한 SOA와 WOA의 서비스 구현 복잡도 평가*

윤광열○, 박소현, 김성철, 최종무, 유해영
단국대학교 컴퓨터학부
soulmix@dankook.ac.kr, sohyun.psh@gmail.com, sckim@kookje.ac.kr,
choijm@dankook.ac.kr, yoohy@dankook.ac.kr

An Evaluation of the Service Implementation Complexity of SOA and WOA through AHP

Gwangyeul Yun○, Sohyun Park, Seongcheol Kim, Jongmoo Choi, Haeyoung Yoo
Department of Computer Science and Engineering, College of Engineering, Dankook University

요 약

IT 자원의 상호운용성 및 재활용성 등의 장점 통하여 새로운 비즈니스 환경변화에 가장 빠르게 대응할 수 있는 최적의 대안으로 서비스 지향 아키텍처(SOA : Service Oriented Architecture)가 최근 각광받고 있다. 그러나 구현의 복잡성 및 그에 따른 낮은 ROI(Return on Investment) 평가와 같은 SOA의 문제점들을 보완하기 위해 웹 지향 아키텍처(WOA: Web Oriented Architecture)가 제안되었다. 하지만 WOA 또한 보안 및 안정적인 메시지 전달 등의 문제점들을 안고 있다. 본 논문에서는 SOA와 WOA 구현의 핵심 개념을 연구하고, SOA 또는 WOA를 이용하여 서비스를 구현할 경우 중요한 핵심 개념의 복잡도를 AHP(Analytic Hierarchy Process) 기법을 통하여 평가하였다. 이를 통하여 SOA 또는 WOA 구현 시 요구되는 핵심 요구사항의 중요도를 평가하고 개발자에게 미치는 구현 복잡도를 측정할 수 있다.

1. 서 론

새로운 비즈니스 환경변화에 가장 빠르게 대응할 수 있는 최적의 대안으로 각광받고 있는 SOA(Service Oriented Architecture)는 IT 자원의 상호 운용성 및 재활용성 증대 등의 장점을 지니고 있다[1]. 하지만 많은 기업이 SOA의 도입에 신중을 기하고 있는 가장 큰 이유는 부족한 SOA 전문 인력과 ROI에 대한 의구심 때문이다[2]. 실제로 SOA를 구현하기 위해서는 기업의 비즈니스 프로세스와 로직을 완벽하게 이해하고 SOA의 구현 방법을 능숙히 다룰 수 있는 IT 전문가가 필요하지만 SOA의 구현 방법 자체가 복잡하기 때문에 전문가가 그리 많지 않다.

이러한 SOA의 단점을 보완하기 위하여 최근 대두된 것이 WOA(Web Oriented Architecture)이다[3]. WOA는 SOA에서 사용되는 SOAP(Simple Object Access Protocol)에 비하여 간단한 REST(Representational State Transfer)를 사용하여 보다 적은 전문가를 통해 쉽고 편리하게 웹 서비스를 구현할 수 있다는 점에서 각광받고 있다.

본 논문에서는 SOA 또는 WOA를 이용하여 서비스를 구축할 경우, 실제 현장에서 개발자들이 느끼는 구현 복잡도를 AHP(Analytic Hierarchy Process)를 통하여 검증하여 지속적인 서비스의 추가 확장 시 개발자들이 구현의 복잡도에 따라 SOA 또는 WOA 중 알맞은 방법을 선택할 수 있는 지표를 마련하였다.

본 논문의 구성은 다음과 같다. 2장에서는 SOA와 WOA의 핵심요소 및 장단점을 분석하였으며, 검증 방법인 AHP에 대하여 분석하였다. 3장에서는 AHP를 통하여 SOA와 WOA의 구현 복잡도를 검증하였으며, 끝으로 4장에서는 검증을 통한 효과와 향후 연구과제에 대하여 서술하였다.

2. 관련 연구

2.1 SOA

2.1.1 SOA 핵심 요소 분석

과거의 경영 환경에서의 비즈니스 변화는 현재의 변화에 비하여 평이하였으므로 기존의 IT 시스템을 통하여 경영 환경의 변화를 예측하는 것이 가능했다[4]. 하지만 현대의 경영 환경에서의 비즈니스 변화는 예측이 불가능할 정도로 빠르고 급격하기 때문에 기존 IT 시스템을 통하여 현재의 경영 환경을 예측하는 것은 불가능하며, 현재 경영 환경에 맞춘 새로운 IT 시스템 패러다임이 필요하게 되었다. 이와 같은 필요에 따라 등장한 개념이 SOA이다[1].

SOA는 서비스 소비자, 서비스 제공자, 그리고 서비스 디렉토리 간의 상호작용을 정의하며, 서비스의 흐름은 (그림 1)과 같이 묘사할 수 있다. 서비스 제공자는 서비스 디렉토리에 자신이 제공하는 웹 서비스를 묘사한 WSDL(Web Service

* 본 과제는 정보통신산업진흥원의 SW공학 요소기술 연구개발사업의 결과물임을 밝힙니다.

Description Language)를 제출하며, 서비스 소비자는 UDDI(Universal Description, Discovery, Integration)를 사용하여 서비스 디렉토리로부터 웹 서비스를 찾아 서비스 제공자가 제공한 웹 서비스의 WSDL을 획득한다. 그리고 서비스 소비자는 SOAP(Simple Object Access Protocol)을 이용하여 서비스를 호출함으로써 서비스 제공자와 연결된다[5].

SOA의 핵심 개념은 서비스 지향이며, 이를 통하여 기존 방법론들과 차별화되고 현재 경영환경에 맞는 새로운 IT 시스템의 구축이 가능하다. 서비스란 비즈니스의 논리적 단위로 정의할 수 있다. 이러한 논리적 구성요소 단위들 간의 흐름을 통제하거나, 자료 형식을 변환시키려는 노력이 필요하기 때문에 서비스 컴포넌트들을 비즈니스 논리와 분리할 필요성이 있다[6]. 기술적 관점에서 서비스 수행 과정의 서비스들은 서비스 제공자와 요청자 간의 연속적인 네트워크 관계를 가진다. 이런 서비스가 이행되는 과정들의 네트워크 집합을 SOA라 볼 수 있다[7].

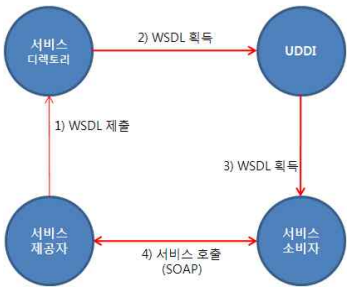

그림 1 SOA의 서비스 흐름

2.1.2 SOA 장·단점

Aloha Airlines의 CIO겸 수석 부사장 Soren Burkhart은 SOA의 실현을 통하여 얻을 수 있는 장점을 "비즈니스 프로세스를 웹 서비스를 통해 공개함으로써 전체 기업 환경의 데이터를 통합하고, 비즈니스 사용자, 외부 파트너, 고객에게 유용한 정보를 제공할 수 있다. 과거의 소프트웨어 컴포넌트는 서로 다른 시스템이 제공하는 경직된 인터페이스를 활용하는 방법밖에 없었지만, SOA를 통하여 느슨하게 결합된 형태의 인터페이스를 구현하고 BPEL(Business Process Execution Language)과 같은 표준을 활용하여, 비즈니스 프로세스에 더 집중할 수 있게 되었다."고 밝혔다[8].

하지만 SOA의 장점은 단순히 표준에 의거한 구축을 통하여 얻어지는 것은 아니다. IEEE Software Architecture의 멤버인 Mark M. Davydov는 "환경을 올바르게 이해하는 개발자에게 SOA는 매우 다양한 혜택을 제공하지만, 새로운 프로토콜과 툴들을 이용하기만 한다면 이러한 혜택이 마법과 같이 실현될 것이라고 믿는 사람들은 생각지도 못했던 문제로 어려움을 겪게 될 것입니다."라고 경고했다[9]. 즉, 적절한 개발 방법론과 툴의 조합에 의하여 SOA의 진정한 비즈니스 가치를 실현할 수 있다는 것이다. 환경을 올바르게 이해할 수 있는 개발자는 핵심 비즈니스 프로세스와 로직을 정확하게 파악할 수 있어야 하는데, 현실상 이러한 개발자의 영입은 어려울뿐더러 많은 비용을 요구하게 되는 단점을 야기한다.

또한 SOA의 장점으로 부각되고 있는 표준을 통한 서비스의 제공은 표준의 범위가 계속해서 확대됨에 따라 점차 복잡해지게 되었으며, 이에 따라 너무 많은 표준에 의하여 제약사항이 증가하였고 개발의 복잡성도 증가하는 문제점을 초래하였다. 최초 SOA의 핵심 프로토콜인 SOAP의 장점으로 부각하였던 Simple의 의미는 2003년 6월 제출된 SOAP 1.2 버전에서 Simple이라는 의미를 삭제함으로 과도한 표준에 의한 문제점을 인정하였으며, 구체적으로 SOA는 타 기술과의 경쟁, 비준 등의 이유로 인하여 8년 이상 진화하는 동안 지원하는 표준들이 50개 이상으로 확대되었다[10][11].

SOA의 또 다른 문제점은 많은 기업에서 SOA를 구축함에 따라 점차 ROI(Return on Investment)에 대한 회의적인 시각이 확산되고 있다는 것이다. ROI에 대한 회의적인 시각은 SOA의 구현 시 비즈니스 부서가 아닌 IT 부서의 주도로 너무 기술적인 측면만 주목한 나머지 비즈니스 사용자의 요구를 적절히 수용하지 못하였으며, 디자인 원칙, 아키텍처를 의미하는 SOA의 개념을 솔루션과 동일시하여 프로젝트를 추진함에 따라 결과적으로 투자비용이 늘어나는 결과를 초래하고 있다는 것이다[12]. 2007년 9월 Information Week에서 조사한 바에 따르면 SOA를 도입한 기업 중 37%만이 ROI의 향상을 경험하였으며, ROI의 향상을 경험한 경우에도 조직 내부에서만 경험을 하였고, 조직외부, 엔터프라이즈 간 연계 등으로 확장되지는 않은 것으로 나타났다[13].

SOA의 개념을 적용하기 위하여 소요되는 인력만 살펴보아도 프로젝트 매니저, 분석가, 아키텍처, 모델러, 개발자, 테스터, 운영자 등 수 많은 인력들이 투입되며, 이들 모두가 핵심 비즈니스 프로세스 및 로직과 SOA의 표준들 및 SOAP, UDDI, WSDL의 정확한 이해가 수반되어야 한다는 제약사항이 발생하며, 이들의 비용 역시 ROI의 평가를 낮추는 중요한 요소이다. SOA의 장점과 단점에 대한 요약은 <표 1>과 같다.

<표 1> SOA의 장점과 단점

SOA의 장점	SOA의 단점
확장성	과도한 표준
분산 컴퓨팅 환경에 적합(웹을 이용)	낮은 ROI 평가
언어, 플랫폼, 통신 독립	개발의 복잡성
표준을 통한 웹서비스 이용	WOA에 비하여 어렵고 무거움

2.2 WOA

2.2.1 WOA 핵심 요소 분석

SOA의 ROI에 대한 회의적인 시각이 확산됨에 따라, SOA를 보완할 수 있는 WOA에 대한 관심이 증가하였다[14]. WOA는 데이터 중심적인 아키텍처이며, REST와 HTTP 프로토콜을 이용하여 웹 서비스를 생성한다. WOA의 최초 시작은 SOA의 복잡한 표준에 얽매이지 않고 쉬운 방식으로 웹 서비스를 구축하기위한 REST의 연구와 함께 시작되었으며, REST를 가장 적극적으로 이용한 것 역시 WOA이다.

최초 WOA의 명명은 2005년 가트너의 Nick Gall에 의하여 웹 기반의 SOA라는 의미로 불리워졌다. 하지만 이는 공식적인 정의나 의미에 대한 명확한 합의가 없었기 때문에 많은 이견이 대두되었다. 그리고 2008년 가트너는 WOA에 대한 공식적인 문서를 발행하였으며, “WOA는 웹의 아키텍처를 기반으로 전 세계적으로 연결된 하이퍼미디어를 통해 시스템과 사용자를 통합하는 SOA의 아키텍처 서비스타일이다. WOA는 다음의 다섯 가지 기본적인 인터페이스 제약을 통해 글로벌 네트워크 효과를 얻기 위한 인터페이스의 일반성을 강조한다.”라고 정의하였다[3].

WOA의 핵심 기술인 REST는 Representational State Transfer의 약자로, 웹과 같은 분산 하이퍼미디어 시스템을 위한 소프트웨어 아키텍처 스타일이라고 정의할 수 있다. 최초의 REST는 웹과 같은 대규모 네트워크 시스템을 위한 표준들의 집합이었으나, 최근에는 XML과 HTTP를 이용한 단순한 웹 기반 인터페이스를 지칭한다.

REST는 잘 정의된 URI를 통하여 웹 어플리케이션을 구동하며, 그 결과를 전달받아 처리하는 방식이다. 잘 정의된 URI로 리소스를 표현하는 REST는 웹 서비스에서 지향하는 4가지의 속성을 따르며, 이 속성들을 만족하였을 때 RESTful하다고 표현한다. RESTful 웹 서비스의 속성은 URI 표현 가능성(Addressability), 연결성(Connectedness), 요청의 일회성(Statelessness), 동일한 인터페이스(Homogeneous Interface)이다[15].

간략하게 말하자면 REST는 원하는 데이터의 위치를 URI를 이용하여 요청하고 그 결과를 XML로 전달 받는다. 이것을 확대하면 웹 의 모든 리소스들을 URI로 표현하고, 이를 구조적이고 유기적으로 연결하여 비 상태 지향적 방법으로 일관된 메소드를 통하여 리소스를 사용하는 웹 서비스 디자인이다.

2.2.2 WOA의 장·단점

SOA를 활용하여 쉽고 편리하게 서비스 지향의 비즈니스 어플리케이션 아키텍처를 구현하고자 했던 초창기의 취지가 사라지면서 SOA의 효용성에 의문이 제기되었다. SOA를 통하여 보다 편리하게 웹 서비스를 구축하고자 했으나 오히려 더 복잡해지는 결과를 초래한 것이다. 이에 따라 WOA의 필요성이 제기되었다.

WOA 역시 SOA와 마찬가지로 장점과 단점을 동시에 지니고 있으며, 그 장점과 단점은 <표 2>과 같다. WOA는 SOA와 다르게 기존 개발자를 활용하거나 개발자를 찾는 것이 쉽고, 매우 잘 알려져 있으며 거의 모든 사람이 사용하는 검증된 기술인 웹 기술(REST, HTTP)을 이용하여 웹 서비스를 쉽게 구축한다. 또한 웹을 적극적으로 이용하기 때문에 상호운영성이 뛰어나며 수행속도가 빠른 장점 역시 존재한다.

WOA의 장점을 요약해 보자면 별도의 솔루션 비용이 필요하지 않으며, 개발 인력도 SOA에 비하여 상대적으로 적으며 특히 고급 개발 인력의 소모가 적다. 따라서 ROI 측면에서 볼 때 SOA에 비하여 상대적으로 유리한 입장이며, ROI 측면의 유리함은 SOA의 불확실한 ROI에 의구심을 가지고 있던 기업들이 WOA에 주목하는 이유가 되었다. 또한 WOA는 SOA에 비해 쉬운 웹서비스 개발과 Mash-up을 통한 새로운 웹서비스 개발도 쉽게 해줌으로써 경쟁력을 갖추고 있음이 분명하며, 소비자 중심의 서비스 구현 역시 가능한 장점이 있다.

하지만 WOA는 분산 컴퓨팅 환경에서 메시지가 여러 중간 단계를 거쳐 보안이 취약하며, 정책 부분, 안정적 메시지 전달의 지원을 위한 표준이 부족하다는 단점이 있다. 특히 보안에 관련된 취약점은 기업의 경영을 위해 전사적인 프로세스를 WOA로 설계·관리하게 될 경우 큰 걸림돌로 작용할 수 있다. 따라서 WOA가 완벽하게 SOA를 대체하는 새로운 아키텍처가 아닌 SOA의 부족한 부분을 보완하는 보완재로써의 역할을 해야 한다는 것이다[14].

< 표 2 > WOA의 장점과 단점

WOA의 장점	WOA의 단점
언어와 플랫폼 독립적	보안 취약성
SOA에 비하여 간편한 개발 가능	지원을 위한 표준 부족
소비자 중심 서비스 구현 가능	
구현이 간편하여 고급 인력의 비용 감소	

2.3 AHP

이론의 단순성 및 명확성, 그리고 적용의 간편성 및 범용성이라는 장점을 통해 의사결정분야에서 널리 응용되어왔던 AHP는 의사결정의 목표 또는 평가기준이 다수이며 복합적인 경우 이를 계층화(Hierarchy)하고, 주요 요인과 그 주요 요인을 구성하는 세부 요인들로 분해한다. 그 후 이러한 요인들을 쌍대 비교(Pairwise Comparison)를 통해 중요도를 산출하는 분석방법이다. 즉, 의사결정요소들의 속성과 그 측정 척도가 다양한

다기준 의사결정문제에 효과적으로 적용되어 의사결정자가 선택할 수 있는 여러 가지 대안들을 체계적으로 순위화를 시키고, 그 가중치를 비율척도(ratio scale)로 도출하는 방법을 제시한다[19].

AHP는 정량적인 분석이 곤란한 의사결정 분야에 전문가들의 정성적인 지식을 이용하여 각 요소의 가중치 또는 중요도를 구하는데 유용하게 응용 될 수 있다는 점에서 수리적인 기법만을 활용한 기타 분석방법에 대해 강점을 가지며, 평가자들에 대한 일관성 여부를 추론할 수 있다는 장점도 가지고 있다.

인간이 의사결정시 두뇌가 단계적 또는 위계적 분석과정을 활용한다는 사실에 착안한 AHP의 3가지 원칙은 다음과 같다.

A. 계층적 구조의 설정 : 맨 윗부분에 Goal(목표)을 두며, 그 밑에 판단 기준이 되는 Evaluation Standard(기준)을 두고, 가장 아래 계층에 Alternatives(대안)을 두는 구조다. Evaluation Standard 아래 Sub-Evaluation Standard 두어 판단 기준 요소를 여러 단계를 나누는 과정을 계속적으로 추가할 수 있으며, 시스템이 난해하거나 심층적 분석을 요할수록 더 복잡한 계층 구조를 가지게 된다.

B. 상대적 중요도의 설정 : 여러 의사결정 요소들을 동시에 고려해서는 그들 사이의 중요도나 가중치를 산출하기가 사실상 불가능함으로 요소들의 쌍대비교(1:1비교)를 통하여 비교행렬을 구성, 중요도나 가중치를 산출한다.

C. 논리적 일관성의 유지 : 비교행렬의 고유벡터를 활용한 1:1 비교 결과의 통합과정에서 비일관성지수(Inconsistency Index)를 도출하여, 의사결정자의 논리적 일관성 유지 여부를 확인하고 의사결정의 합리성과 논리성을 높일 수 있게 된다.

3. AHP를 통한 SOA와 WOA의 구현 복잡도 평가

SOA와 WOA의 유용성을 평가하기 위해서는 시스템 개발을 위한 다중의 요소들에 대한 분석과 정성적인 요인을 고려해야 한다. 따라서 T.L.Satty가 제안한 다기능 문제 의사결정방법인 AHP(Analytic Hierarchy Process)의 적용을 통한 가중치(weight) 추정(estimation)을 통하여 SOA와 WOA의 구현 복잡도를 평가하였다[17][18].

3.1 AHP 및 가중치를 통한 구현 복잡도 평가

본 논문에서는 SOA와 WOA의 AHP 적용을 위해 계층 구조도를 그리고, 목적설정 및 평가항목을 선택하여 상대적 복잡도 및 가중치를 산출하여 구현 복잡도를 평가한다. 또한 산출된 가중치의 논리적 일관성 유지 평가를 위해 일관성지수를 도출하여 검증한다. (그림 2)는 SOA와 WOA에 대한 계층 구조도를 나타낸 것으로 이 계층 구조도에서의 목표는 SOA와 WOA의 구현 복잡도 평가이다. 평가 기준으로는 SOA표준, SOAP, REST, UDDI, 서비스개념을 두었으며, 대안은 SOA와 WOA로 개발된 서비스이다. 대안을 평가하기 위한 기준은 관련논문 및 연구보고서 등의 리서치를 통해 1차적으로 추출하고, IBM의 컨설턴트와 삼성SDS의 SOA전문가들, SERC(Software Engineering Research Center)의 SOA전문가들을 대상으로 설문조사 및 토의를 거쳐 5개의 평가 기준 항목을 선정하였다.

각 평가기준에 대해서 항목 간에 어느 쪽이 얼마만큼의 가중치를 갖는지를 AHP에서 사용하는 일대일 비교치 목록을 기준으로 쌍대 비교 하여 중요성의 정도에 따라 두 번째 원칙인 상대적 중요도의 설정을 위해 앞에서 정의한 평가항목들의 가중치를 도출한다. 즉, 평가항목들의 일대일 비교치 목록을 기준으로 쌍대 비교하여 수치로 부여한다. 개별 구성원들의 평가 자료를 종합하는 방법으로는 기하평균법(Geometric Average)을 이용하였다[20].

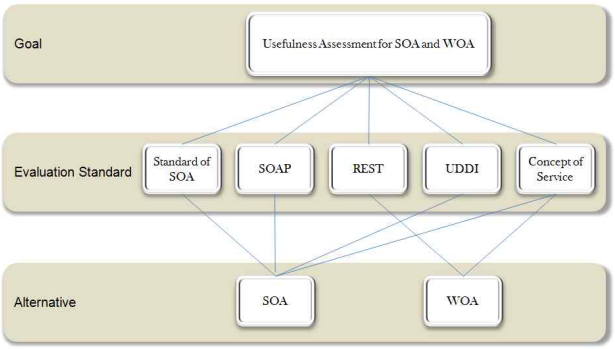

그림 2 Hierarchical layer for AHP analysis

각 평가 항목의 지속가능부문 이슈를 A1....An로 두고 지속가능 정도를 V1....Vn이라 할 때, AHP에서 사용하는 일대일 비교치 목록인 <표 3>를 참고하여 쌍대 비교 값을 정방행렬 [A]로 배열하는 방법은 <표 4>과 같다. V1/V1는 A1자신에 비교한 것으로 그 값은 1이다. V1/V2는 A2에 비교한 A1의 복잡도의 정도를 나타낸 값이고, V1/Vn은 An에 비교한 쌍대비교 값을 의미한다.

<표 3> AHP Pairwise comparison matrix

value of a_{ij}	interpretation
1	Objectives *i* and *j* are of equal importance
3	Objectives *i* is weakly more important than objective *j*
5	Experience and judgement indicate that objective *i* is strongly more important than objective *j*
7	Objective *i* is very strongly or demonstrably more important than objective *j*
9	Objective *i* is absolutely more important than objective *j*
2, 4, 6, 8	Intermediate values
reciprocal	Objectives *j* is more important objective *i*

각 항목의 모든 쌍을 쌍대 비교 하여 <표 5>과 같이 행렬로 나타내고, (식 1)을 이용하여 각 값들의 기하평균을 산출한 다음 (식 2)과 같이 산출한 기하평균 전체의 합에 각 항목의 기하평균 값을 나누어 각각의 가중치를 계산한다. 그 결과는 <표 6>과 같다.

<표 4> 쌍대비교 값의 정방행렬 배열

	A_1	A_2	…	A_n
A_1	V_1/V_1	V_1/V_2	…	V_1/V_n
A_2	V_2/V_1	V_2/V_2	…	V_2/V_n
…	…	…	…	…
A_n	V_n/V_1	V_2/V_n	…	V_n/V_n

$$A \left\{ \prod_{k=1}^{n} a_k \right\}^{\frac{1}{n}} = a_1 a_2 \cdots a_n \quad (\text{식 } 1)$$

$$W_i = \frac{GA_i}{\sum_{k=1}^{n} GA_k} \quad (\text{식 } 2)$$

쌍대비교행렬은 의사결정자의 주관에 따라 결정되므로, 전문가 집단이 주관적으로 판단한 요소간의 중요성을 얼마나 일관성 있게 응답했는가를 논리적으로 판단하기 위해 일관성 검사(consistency test)를 수행한다. 이것은 쌍비교행렬로 계산한 가중치를 사용하여 실제로 얻어진 행렬과의 차이를 수량적으로 검토하는 것이다. 일관성 검사에는 일관성 지수(consistency index : CI)와 일관성 비율(consistency ratio : CR) 값이 필요하다. 먼저 CI는 다음과 같이 계산한다.

① 쌍대비교행렬과 가중치 벡터를 곱하여 가중치 합 벡터를 구한다.

② 가중치 합 벡터의 값을 상응하는 가중치 값으로 나눈다.

③ 위에서 구한 값의 평균인 최대고유치(principal Eigenvalue) λ max를 구한다.

일관성 지수 CI는 λ, 비교 요소의 수를 n이라고 했을 때, (식 3)과 같이 계산한다.

$$CI = \frac{\lambda - n}{n - 1} \quad (\text{식 } 3)$$

일관성 비율 CR은 확률 지수(Random Index)와 CI 값을 이용하여 (식 4)와 같이 계산하며, 일관성이 작을수록 CR 값이 크다. 일반적으로 CR이 0.1 이하일 경우 논리적 일관성을 유지한 것으로 판단한다. 확률 지수 RI는 쌍대비교의 대상 개수 n에 따라 <표 6>과 같다.

$$CR = \frac{CI}{RI} \quad (\text{식 } 4)$$

< 표 6 > Value of the random index

n	2	3	4	5	6	7	8	9
RI	0.0	0.58	0.90	1.12	1.24	1.32	1.41	1.45

< 표 5 > 의 쌍대비교 행렬 값과 도출한 가중치의 일관성 검사는 다음과 같다.

$$\begin{pmatrix} 1 & 3 & 6 & 2 & \frac{1}{2} \\ \frac{1}{3} & 1 & 3 & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{6} & \frac{1}{3} & 1 & \frac{1}{4} & \frac{1}{7} \\ \frac{1}{2} & 3 & 4 & 1 & \frac{1}{3} \\ 2 & 4 & 7 & 3 & 1 \end{pmatrix} \begin{pmatrix} 1.783 \\ 0.092 \\ 0.044 \\ 0.174 \\ 0.042 \end{pmatrix} = \begin{pmatrix} 1.365 \\ 0.476 \\ 0.223 \\ 0.899 \\ 2.155 \end{pmatrix} \quad (\text{식 } 5)$$

$$\lambda = \frac{1}{5}\left(\frac{1.365}{1.783} + \frac{0.476}{0.092} + \frac{0.223}{0.044} + \frac{0.899}{0.174} + \frac{2.155}{0.042} \right) = 5.128 \quad (\text{식 } 6)$$

$$CI = \frac{\lambda - n}{n - 1} = \frac{5.128 - 5}{5 - 1} = 0.032 \quad (\text{식 } 7)$$

$$CR = \frac{CI}{RI} = \frac{0.032}{1.12} = 0.029 \quad (\text{식 } 8)$$

평가항목 간 쌍대비교행렬의 일관성 비율 CR이 0.029로 0.1보다 작으므로 일관성에 문제가 없음을 검증하였다.

AHP를 응용한 구현 복잡도 평가를 위하여 SOA와 WOA 서비스 구현시 개발자들이 반드시 알아야하는 5가지 핵심 요소를 설문을 통하여 도출하였다. AHP를 통해 도출된 가중치는 하나

< 표 5 > 각 평가기준에 대한 쌍대비교행렬과 기하평균 및 가중치

쌍대비교	SOA표준	SOAP	REST	UDDI	서비스개념	기하평균	가중치
SOA표준	1	3	6	2	1/2	1.783	**0.269**
SOAP	1/3	1	3	1/3	1/4	0.608	**0.092**
REST	1/6	1/3	1	1/4	1/7	0.289	**0.044**
UDDI	1/2	3	4	1	1/3	1.149	**0.174**
서비스개념	2	4	7	3	1	2.787	**0.421**
					합계	6.616	**1.000**

의 SOA 또는 WOA 서비스를 개발하기위해 투입되는 인력들이 5가지 핵심 요소에 대하여 느끼는 구현의 난이도를 측정하였으며, 가중치의 결과는 서비스의 개념, SOA 표준, UDDI, SOAP 그리고 REST 순으로 도출되었다.

5가지의 핵심 요소 중 SOA 서비스의 개발을 위하여 필요한 항목은 서비스의 개념, SOA 표준, UDDI 그리고 SOAP이며, WOA 서비스의 개발을 위하여 필요한 항목은 서비스의 개념과 REST이다. 따라서 SOA와 WOA 서비스의 개발을 위하여 투입된 개발자들이 느끼는 구현 복잡도를 비율로 확인하자면 SOA : WOA = 0.956 : 0.465과 같은 결과를 도출할 수 있다.

위의 결과를 통해 SOA 서비스는 WOA 서비스의 개발에 비하여 2.056배의 높은 복잡도가 예상된다. 총 5개의 서비스를 SOA로 신규 생성하여 추가 시 4.78의 구현 복잡도가 측정된다고 가정한다면, 5개의 서비스 중 2개의 서비스를 WOA를 통하여 개발하였을 경우 3.78의 구현 복잡도를 측정할 수 있다. 따라서 WOA를 통한 개발의 경우 단순히 SOA를 통한 구현에 비하여 개발자들이 느끼는 구현 복잡도가 상대적으로 낮으며, WOA를 통한 외부 서비스의 추가가 확대될수록 개발자의 구현 복잡도는 SOA를 통한 구현에 비하여 상대적으로 낮아질 것이다.

4. 결론 및 향후 연구

급변하는 현대의 비즈니스 환경에서 많은 기업들은 SOA에 대한 적극적인 도입의 움직임을 보이고 있으며, 또한 SOA에 대하여 부정적인 의견들 역시 비례하여 증가하고 있다. 이러한 부정적인 의견의 증가는 WOA에 대한 관심의 증가를 가지고 왔으며, WOA를 SOA의 대체재로 바라보는 시각이 존재하기 시작하였다.

본 논문에서는 최근 IT 업계의 화두인 SOA와 WOA의 구현 복잡도를 AHP 기법을 통하여 검증하였다. AHP의 기법을 통한 검증은 1차적으로 기술에 대한 연구를 진행하였으며, 다음 단계로 전문가들을 통한 설문 및 토의를 통하여 SOA와 WOA 서비스 구현을 위하여 필요한 5가지 필수 요소들을 도출하였다. 이를 통하여 AHP 방식으로 가중치를 산정하여 구현 복잡도를 평가하였다. 구현의 복잡도는 개발에 투입되는 인력의 비용과 직결된다. 구현의 복잡도가 높을수록 개발 기간을 맞추기 위하여 인력을 확대하거나, 고급인력을 필요로 하기 때문이다. 따라서 상대적으로 낮은 복잡도가 측정된 WOA를 통하여 SOA의 일부분을 대체하는 방식으로 기존 SOA에서 문제가 되었던 비용부분을 향상시킬 수 있을 것으로 기대된다.

현재 기존 논문에서 연구된 SOA와 WOA의 통합 아키텍처 설계를 바탕으로 서비스 추가 시 기존의 단점을 보완하는 아키텍처를 실제 구현하고 있으며, 이를 통하여 SOA와 WOA의 기술적인 단점을 모두 보완할 수 있는 통합 아키텍처의 유용성을 실험적으로 검증할 계획이다.

5. 참고 문헌

[1] 전병선, "SOA, What & How," 와우 북스, 2008.

[2] KIPA, “SOA 시장 활성화를 위한 벤더의 역할,” SW산업동향, 2008.

[3] Nicholas Gall, Daniel Sholler, Anthony Bradley, "Tutorial: Web-Oriented Architecture: Putting the Web Back in Web Services," Gartner Research Paper, 2008.

[4] Gartner, "Business Intelligence Software market Grows 12 Percent in 2006," 2007.

[5] Peter Brittenham, Francisco Cubera, Dave Ehnebuske, Steve Graham, "Understanding WSDL in a UDDI registry," IBM DeveloperWorks, 2001.

[6] Norbert Bieberstein, Sanjay Nose, Marc Fiammante, Keith Jones and Rawn Shah, "Service-Oriented Architecture(SOA) Compass: Business Value, Planning, and Enterprise Roadmap," IBM Press, 2005.

[7] David Sprott, "The Business Case for Service Oriented Architecture," CBDI, 2004.

[8] Alan Joch, "Modern Design," Oracle Magazine, 2005.

[9] Mark M. Davydov, “Beyond SOA: Principles of Service Engineering,” JAVAPro, 2004.

[10] Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen, Anish Karmarkar and Yves Lafon, "SOAP Version 1.2 Part 1: Messaging Framework," W3C Recommendation, 2007.

[11] Nilo Mitra, Yves Lafon, "SOAP Version 1.2 Part 0: Primer," W3C Recommendation, 2007.

[12] Anne T. Manes, "SOA is Dead; Long Live Services," Burton group blogs, 2009.

[13] John Soat, "The ROI of SOA: Get It Right!," Information Week, 2007.

[14] KIPA, "SOA의 구원투수 WOA," SW산업동향, 2008.

[17] T.L.Satty. “The Analytic Hierarchy Process,” McGraw-Hill, 1980.

[18] 이상완, “e-비지니스 패턴을 이용한 효율적 SOA 구축 방안”, 동아대학교 대학원, 박사학위논문, 2006

[19] Zahedi, F., "The Analytic Hierarchy Process-A Survey of the Method and its Applications," Interfaces, Vol 16, No.4 pp.96-108, 1986.

[20] Hannan, E.L., "An Eigenvalue Method for Evaluating Contestants," Computer and Operations Research, 1983.