SOA 기반 소프트웨어의 구조적 복잡도 측정을 위한 메트릭스

김 은 미*

요 약

SOA 기반의 응용 프로그램의 개발이 급격히 진전되는 시점에서 이에 따른 SOA 기반 소프트웨어의 품질평가는 중요하며 특히, 이러한 품질에 영향을 주는 중요한 요인 중의 하나가 시스템의 복잡도임을 고려할 때 SOA 기반 소프트웨어의 복잡도 측정은 중요하다. 따라서 본 논문에서는 SOA기반 소프트웨어의 구조적 복잡도를 측정할 수 있는 메트릭을 제안한다. 제안한 메트릭은 시스템의 구조적인 관점에서 서비스의 크기, 서비스의 깊이 및 상호종속도로 구성된다. 마지막으로 제안한 메트릭을 예제에 적용하여 본다.

Metrics for Measuring a Structural Complexity of Softwares Based on SOA

Eun Mi Kim*

ABSTRACT

It is very important to evaluate the quality of softwares based on SOA, which develops rapidly. Particularly, one of the most important properties influencing on the quality of system is complexity. Therefore, we propose the metrics for measuring a structural complexity of softwares bsed on SOA. The proposed metrics is composed of the size of a service, the depth of a service and interdependency from the viewpoint of system structure, and finally we applied the proposed metric to an example.

Key words : Metric, SOA, Complexity

접수일 : 2010년 1월 29일; 채택일 : 2010년 2월 26일

* 호원대학교 컴퓨터&게임학부 교수

1. 서 론

최근 빠르게 변화하는 업무 환경에 쉽게 적응되는 SOA가 빠르게 성장하고 있다. SOA는 새로운 개념이라기보다는 객체지향이나 컴포넌트 기반의 소프트웨어가 가지고 있던 단점인 강결합(tight-coupling)구조를 보완하여 서비스의 구조 및 구현의 변화에 빠르게 대응할 수 있는 기민성과 표준화 된 인터페이스를 통해 느슨한 결합(Loosely Coupled)을 가지고 상호 연동할 수 있는 서비스들의 조합을 통해 어플리케이션의 개발을 가능하게 하는 정보 시스템이다[1, 2, 4].

SOA와 CBD의 관계를 비교함으로써 SOA에 대한 특징을 좀 더 명확히 알 수 있다. CBD의 컴포넌트는 몇 개의 제공 인터페이스를 보관 유지하는 소프트웨어의 단위이며, SOA의 서비스는 그 컴포넌트가 제공하는 인터페이스이다. 즉, CBD가 기업 내부 시스템에 대한 기능 구현이 중심이라면 SOA는 기업 내 외부 통합을 통한 비즈니스에 중점을 두고 있다. 또한, 연계와 활용 관점에서 보면 CBD는 해당 시스템을 구축한 기술에 그 영역이 제한되는데 비해 SOA는 웹 서비스라는 표준 프로토콜을 이용함으로써 실제 서비스의 구현 언어 등에 종속되지 않고 다양한 응용에서 동시에 사용하는 것을 허용한다. <표 1>은 CBD와 SOA의 차이점을 보여주고 있다. 이러한 SOA의 장점은 비즈니스 프로세스의 변경 시에 대응이 간단하다는 것과 서비스를 많은 시스템이 이용하므로 중복 개발이 적다는 것이다. 즉, SOA는 응용프로그램이나 다른 서비스를 개발할 때 소프트웨어의 기능들을 서비스로 제공받아 최종 프로그램이 구성되거나 전달되는 아키텍쳐 스타일로서 SOA의 중심은 서비스이며 이러한 서비스는 서비스를 제공하는 제공자와 이를 이용하는 이용자 간의 계약관계에 의해 이루어지며 이러한 관계가 유지되기 위해서는 일정한 수준의 서비스에 대한 품질을 보증해야 한다. SOA 기반의 응용 프로그램의 개발이 급격히 진전되는 시점에서 이에 따른 SOA 기반 소프트웨어의 품질 평가는 중요한 이슈이며 특히, 이러한 품질에 영향을 주는 중요한 요인 중의 하나가 시스템의 복잡도임을 고려할 때 SOA 기반 소프트웨어의 복잡도 측정은 중요한 연구과제이다. 따라서 본 논문에서는 SOA 기반 소프트웨어의 복잡도를 측정할 수 있는 메트릭을 제안한다. 이번 연구는 SOA 기반 소프트웨어의 구조적인 관점을 주로 고려하였다.

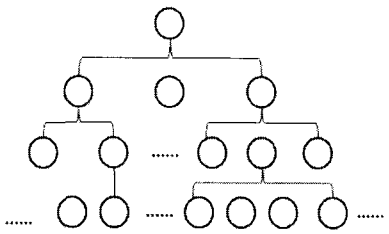
이 후 제 2장에서는 관련 연구에 대해 기술하고 제 3장에서는 SOA 기반의 소프트웨어의 구성 및 SOA 기반의 소프트웨어에 대한 구조적 복잡도를 측정할 수 있는 메트릭을 제안하고, 제안한 메트릭을 예제를 통해 적용하고 마지막으로 제 4장에서 결론 및 향 후 연구 방향에 대해 논의한다.

2. 관련 연구

Dmytro R.은 Component-based, Oriented 소프트웨어 및 웹어플리케이션과 SOA의 유사점과 차이점을 비교하고 SOA의 모든 특성을 반영하는 프로덕트 메트릭을 아래와 같이 복잡도, 신뢰성 및 성능 관점에서 정의하였다[1]. 이 중 복잡도 관점의 메트릭에 대해서만 기술하면 1) Network Cohesion in the System : 노드 간의 직/간접적인 호출의 수에 의해 측정한다. 2) Number of Service Involved in the Compound Service : 복합 서비스에 포함된 서비스의 수에 의해 측정한다. 3) Service Interdependence in Service : 시스템 내의 서로 종속된 서비스 쌍(pair)의 수로 측정한다. 그러나 복잡도 메트릭의 정의 시 복잡도와 직접적으로 연관된 노드의 크기, 순환호출관계 등으로 고려하지 않았다. 이외에도 Web Service Definition Language(WSDL)에 기반을 둔 SOA의 복잡도 측정에 대한 연구도 진행되었으며[2], 객체지향 소프트웨어에 적용한 메트릭을 기반으로 대규모의 SOA에 적용한 연구도 진행되었다[3].

〈표 1〉 CBD 와 SOA의 비교

관점	특성	CBD	SOA
구현	시스템	기업내부시스템 기능구현	기업 내 외부 통합
	모듈	기능중심	비즈니스 중심
	프로세스	개별컴포넌트에 집중	컴포넌트 간 프로세스에 집중
	모델링기법	UML	UML, EA, 컨설팅기법 등
	목표	시스템관점에서 컴포넌트 구축	BPM과 연계
연계	플랫폼	동일플랫폼기반	이기종 시스템간 연계
	방식	Tight Coupling(Serialization)	Loosely Coupling(SOAP)
인터페이스	활용방식	개별 정의된 인터페이스	공개적/표준화된 인터페이스
활용	응용연계	단일 응용에서만 사용	멀티 응용에서 동시 사용
	오류	오류 발생 시 전체에 영향	부분 오류 허용



(그림 1) SOA 시스템의 계층구조

3. 새로운 메트릭의 정의

3.1 SOA 시스템의 구조

SOA는 서비스 제공자에 의해 제공되고 있는 하나 이상의 서비스를 사용하는 서비스 사용자(service Consumer), 서비스 사용자가 호출 시 입력하는 값을 가공하여 그에 해당하는 결과를 제공하는 서비스 제공자(Service Provider), 서비스에 대한 설명 정보를 저장하고 있는 서비스 중개자(service Broker)로 구성되어 있다. 서비스 제공자는 경우에 따라 또 다른 서비스 제공자의 서비스를 사용하는 서비스 사용자가 될 수 있으며 서비스 제공자는 자신이 제공하고 있는 서비스를 서비스 중개자에 등록하고 서비스 사용자는 자신이 원하는 서비스를 서비스 중개자를 통해 찾아 사용한다. 이러한 SOA를 구조적 관점에서의 계층 구조로 표현하면 (그림 1)과 같다.

즉, Service-oriented 시스템은 provider의 노드들로 구성된다. 각 노드는 하나 또는 그 이상의 non-mobile 서비스를 제공하고 있으며, 각각의 서비스는 하나 또는 그 이상의 오퍼레이션들로 구성된다. 클라이언트들은 메시지의 상호 전달을 통해 서비스들과 통신을 한다[1]. 서비스들은 단독적으로 또는 복합적일 수 있다.

3.2 메트릭의 정의

본 장에서는 SOA의 구조적 특징을 고려한 새로운 메트릭을 정의한다. 새로운 메트릭은 시스템 내 서비스의 크기, 깊이 및 상호종속도의 관점에서 측정된 값을 통해 시스템 전체의 구조적 복잡도를

측정한다.

3.2.1 서비스의 크기(Quantity of Service : QoS)

서비스 요청에 의해 전달되는 데이터의 크기를 의미한다. 이는 각 서비스에 포함된 오퍼레이션의 수에 의해 측정된다. 서비스되는 데이터의 크기가 클수록 서비스 제공 시간이 길어지며 시스템의 성능을 저하시키는 요인이 될 수 있다. 서비스의 크기는 한 노드에 포함된 각각의 서비스에 대한 크기와 전체 노드에 대한 크기로 구분 할 수 있다.

가) 한 노드내의 개별 서비스의 크기 (Individual Quantity of Service in a node)
: 각 서비스에 포함된 오퍼레이션의 수에 의해 측정한다.
$IQoS(i)$: 각서비스 i에포함된 오퍼레이션의수

나) 한 노드내의 전체 서비스의 크기(Total Quantity of Service in a node)
: 각 서비스에 포함된 오퍼레이션의 수를 합산하여 측정한다.

$$TQoS(i)=\sum_{j=1}^{n} QoS(j)$$

n: 한노드 i에포함된 서비스의 총수

다) 평균 서비스의 크기(Average Quantity of Service in a node)

$$AQoS(i)= TQoS(i)/n$$

n: 한노드 i에 포함된 서비스의 총수

3.2.2 서비스의 깊이(Depth of Service : DoS)

한 노드에서 제공되는 서비스의 깊이는 서비스의 호출에 대한 단계를 의미한다. 하나의 서비스를 호출 시 그 서비스의 수행을 위해 또 다른 연관관계를 가진 서비스가 존재한다면 복잡도는 증가할 것이다. 이는 객체지향이나 컴포넌트 기반 소프트웨어의 상속과 유사한 의미를 지닌다. 서비스의 깊이가 깊을수록 내부 coupling이 증가되고 서비스 제공 시간이 길어진다. 또한, 서비스의 이해용이성 및 사용성이 저하된다. 서비스의 깊이는 한 노드의 평균 서비스의 깊이와 노드에 포함된 서비스의 최대 깊이를 측정한다.

가) 한 노드의 평균 서비스의 깊이(Average Depth of Service : ADoS)

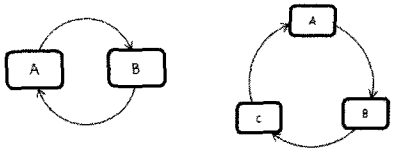
$$ADoS(i)=\frac{\sum_{j=1}^{n} d_j}{n}$$

d_j : 노드 i내에 정의된 서비스 j의 깊이
n : 노드 i내에 정의된 서비스 j의 총수

나) 한 노드의 최대 서비스의 깊이(Maximum Depth of Service : MDoS)
$MDoS(i)$ = 노드 i내 서비스 깊이 중 최대값

3.2.3 서비스의 상호종속도(InterDependency of Service : IDoS)

서비스가 서로 호출하여 하나의 쌍을 이거나 상호 순환 관계를 이루는 호출관계를 의미한다(그림 2 참조). 이는 상호 종속되거나 상호 순환관계에 의한 호출은 서비스 간 또는 노드간의 종속도를 증가하므로써 두 객체간의 coupling을 강하게 한다. 따라서 서비스 제공 시간의 지연 및 복잡도를 증가시키는 요인이 된다. 노드 i에 상호종속성은 노드 내의 각 서비스에 대한 상호종속도의 합에 의해 측정한다.



(가) 종속관계 (나) 순환관계

(그림 2) 관계의 정의

$$IDoS(i) = \sum_{j=1}^{n} (id_s + id_n + id_{cs} + id_{cn})$$

i : i번째 노드

j : 노드 i에 포함된 서비스

id_s : 서비스 j와 상호종속성을 가진 서비스의 수

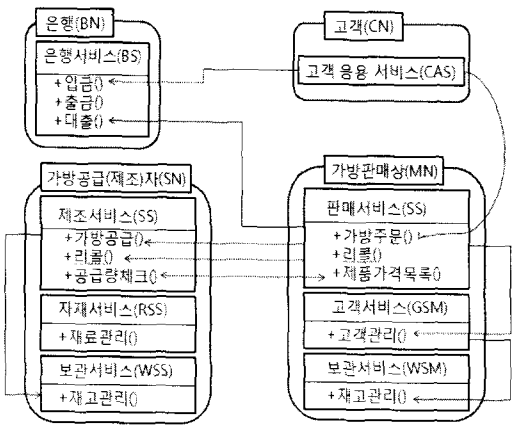
id_n : 서비스 j와 상호종속성을 가진 노드의 수

id_{cs} : 서비스 j와 상호 순환 관계를 가진 서비스의 수

id_{cn} : 서비스 j와 상호 순환 관계를 가진 노드의 수

4. 적 용

본 장에서는 제안한 메트릭을 (그림 3)의 Service-oriented 시스템의 간단한 예제를 통해 적용해 본다. 본 예제 시스템은 4개의 노드(BN, CN, MN, SN)로 구성되어 있으며, 각 노드는 각각의 서비스를 1개 또는 3개씩 포함하고 있다. 그림에서 모서리가 둥근 사각형은 노드를 나타내며, 사각형은 각 노드에 포함된 서비스를 나타낸다.



(그림 5) Service-oriented 시스템의 예

<표 2>는 예제 시스템에 대한 기본 데이터 값을 산출한 결과이다. 이 결과를 이용하여 제안한 메트릭의 값을 산출할 수 있다.

〈표 2〉 (그림 1)의 기본 데이터 값

노드	서비스	오퍼레이션의 수(a)	서비스 깊이(b)	상호/순환 호출(c)
BN	BS	3	1	0
CN	-	-	1	0
MN	SS	3	1	1
	GSM	1	2	0
	WSM	1	3	0
SN	SS	3	1	1
	RSS	1	1	0
	WSS	1	2	0

4.1 서비스의 크기(Quantity of Service : QoS)

가) 한 노드내의 개별 서비스의 크기(IQoS(i))는 <표 2>의 (b)열에서 메트릭의 값을 알 수 있다. 따라서 예제에 대한 전체 서비스의 크기(TQoS(i))는 12이며, 평균 서비스의 크기(AQoS(i))는 12/4 = 3이다.

4.2 서비스의 깊이(Depth of Service : DoS)

서비스의 깊이는 자신을 1로 놓았을 때 이 노드에 의해 호출되는 서비스는 그 깊이가 1씩 증가

〈표 3〉 서비스의 깊이

노드	서비스	Dos	ADoS	MDoS
BN	BS	1	1	0
CN	-	1	1	0
MN	SS	1	3	3
	GSM	2		
	WSM	3		
SN	SS	1	1.3	2
	RSS	1		
	WSS	2		

한다. 예를 들어, SN 노드의 서비스 SS는 실행 시 서비스 WSS를 호출하여 사용한다. 이 때 SS의 깊이는 1이며 SS에 의해 호출된 WSS의 깊이는 2로 산출한다. <표 3>은 각 노드에 대한 서비스의 깊이를 나타낸다.

4.3 서비스의 상호종속도(InterDependency of Service : IDoS)

본 예제에서는 상호호출에 의한 종속도만 포함되어 있다. 따라서 서비스의 상호종속도는 (표 6)과 같다.

(표 4) (그림 1)의 데이터 값

노드	서비스	상호/순환 호출(서비스당)	MDoS
BN	BS	0	0
CN	-	0	0
MN	SS	1	1
	GSM	0	
	WSM	0	
SN	SS	1	1
	RSS	0	
	WSS	2	

5. 결론 및 향 후 연구

본 논문에서는 SOA 기반 소프트웨어의 구조적 복잡도를 측정할 수 있는 메트릭을 제안하고 예제에 적용하여 값을 산출하였다. 이번 연구는 메트릭에 대한 첫 단계로서 SOA 시스템의 구조적인 관점만을 반영하였다. 향 후 이러한 메트릭 값의 분석 및 활용에 대한 연구가 진행되어야 하며, SOA의 서비스 간의 동적인 특징을 분석하여 복잡도 메트릭을 지속적으로 보완해 나갈 예정이다.

참 고 문 헌

[1] WDmytro R. 외 2인, "Product metrics for service-oriented infrastructures", Proc. of IWSM/MetriKon, 2006, 2006.

[2] Quynh P. T. 외 2인, "A complexity measure for web service", Proc. of 2009 International conference of Knowledge and System Engineering, pp. 226-231, 2009.

[3] Helge H. 외 1인, "Supporting service-oriented design with metrics", Proc. of 12th International IEEE Enterprise Distributed Object Computing Conference, pp. 191-200, 2008.

[4] http://yagukorea.com/xe/workspace/866.

김 은 미

1991년 전북대학교 전산통계학과 (공학사)

1993년 전북대학교 전산통계학과 (공학석사)

1997년 오사카대학교 정보공학부 (공학박사)

1997년~현재 호원대학교 컴퓨터&게임학부 교수