A Study on Building Web Services for Implementing Real Time Enterprise

Jungmin Lee[a]

[a] *Consultant, Web Services/SOA(Service Oriented Architecture) Team, Office of IT Consulting, SAMSUNG SDS*

16F Ilok Bldg. 707-19, Yoksam-2Dong, Gangnam-Gu, Seoul, 135-918, Korea
Tel:+ 82-010-4742- 8099, E-mail: jm80.lee@samsung.com, isaac-lee@hanmail.net

Abstract

To transform the enterprise system into the 'Real Time Enterprise' with respect to IT, I suggest the conceptual application model which is composed of pieces(components) extracted from different packaged applications working in a heterogeneous environment based on the 'business activity' in other words, business services provided by internal (inter-enterprise) and external (extended enterprise) application to support a business activity within in an enterprise and the design mechanism focusing service tier which acts as intermediate tier within application architecture.

Keywords:

Conceptual Application Model for a Real Time Enterprise; Composite System Design; Business Service Design; Web Services Component Design; Service Tier Design

제1장 도입

'기업의 IT는 비즈니스 활동 영역의 지리적인 확장, 다수의 비즈니스 파트너 및 시스템의 관여, M&A(Merge & Acquisition), 고객 요구사항의 다양화, 전략의 변화, business process의 변경에 대해 신속하고 민첩하게 대응하기 위해 어떻게 변화해야 하는가?'라는 질의에서 시작한 본 논문은 이와 같은 이슈사항에 대한 대안으로써 '지리적으로 산재한 여러 이종(異種)의 IT 시스템'에서 'business activity를 기준으로 특정 기능을 추출하여 component화'한 후 '기업의 IT Platform과는 무관하게 기업 내 외부와 서비스를 주고받도록 하는 Web Services component 화'한, 이른바 virtually composite system를 구현하기 위해 service tier를 설계하는 방법을 제안한다.

이 service tier는 지속적으로 변화하는 기업의 전략과 비즈니스 목표, 고객에게 제공하는 가치를 고려하여 legacy function(s)을 wrapping하고 공개 수준 등을 고려하여 기업 내 외부로 공개(expose)할 service별 business method를 정의한다. 해당 시스템의 사용자(타 시스템 포함)는 이 service tier를 통해 복잡한 내부 구현을 은닉하고 있는 interface를 통해 해당 시스템이 제공하는 서비스 즉 IT 시스템의 기능을 호출할 수 있게 됨에 따라 시스템이 사용하고 있는 기술 요소를 고려하지 않고 실시간 정보를 조회할 수 있게 된다. 시스템 설계자는 본 논문에서 구현하는 prototype system을 참고하여 service tier를 추가할 수 있으며, 본 논문에서 사용한 service tier를 design & implementation하는 방법을 참고하여 시스템을 재구성하는 과정에서 도움이 될 것이다.

제2장 이론적 배경과 적용

1. RTE 개념과 구축방안

일괄처리 또는 수작업을 최소화함으로써 기업 내 외부의 다양한 정보를 신속하고 정확하게 유통할 수 있고, 고객의 요구사항 또는 비즈니스 환경의 변화에도 재빠른 business process 변경과 효과적인 자원 활용을 통해 신속하고 민첩하게 대응할 수 있으되, 기존 자원과 자산을 재활용함으로써 시간과 비용 측면에서도 효율적인 기업 체계(enterprise system)를 소위 유기적인 민첩성(organizational agility), 민첩한 조직(agile organization), 실시간 기업(Real Time Enterprise; RTE)으로 정의하며[1], 이들을 통합하여 실시간 기업(RTE)이라 일컫는다.

실시간 기업(RTE)에서는 기업 내 외부의 연계와 통합, 자동화를 통해 정보 흐름의 실시간성을 보장하되[2], business process 단위의 기업 내 외부의 연계 및 통합, 자동화를 통해 재빠른 business process 변경을 가능하게 함으로써 기업의 유연성(flexibility)과 적응력(adaptability)을 제고하고[2], 정보뿐만 아니라 기존IT 자원과 자산의 활용 가치를 제고할 것을 강조한다[1, 3].

본 논문에서는 이와 같은 실시간 기업화를 위한 IT 측면에서의 그 구축방안을 다음과 같이 정리한다.

❑ 기업은 configuration을 통해 응용 시스템(들)을 customization할 수 있어야 한다. 즉 business process를 기준으로 여러 개의 다른 application software packages와 legacy applications 내 하위 기능들을 임의로 추출 및 수집(assemble; configure)할 수 있게 한다.

□ 연계와 통합, 자동화는 표준 방식으로 구현함으로써 상호연동보다 협업으로 나타날 수 있는 성과와 성능에 더욱 초점을 맞출 수 있어야 한다. 일반적으로 연계 대상이 증가할수록 구축비용은 크게 증가하나, 표준 방식을 통해 서로 다른 별개의 시스템 간의 통신을 원활하게 하게 함으로써 비용과 시간을 감축할 수 있도록 한다.

2. Web Services 이해와 적용방안

최근에는 인터넷 기반의 표준화 기술인 HTTP(Hyper Text Transfer Protocol)와 HTML(Hyper Text Markup Language), 정보의 의미와 구조를 표준화한 XML(eXtensible Markup Language)의 상용화, 이러한 표준화된 기술을 근간으로 확장된 Web Services, UDDI(Universal, Description, Discovery, and Integration), SOAP(Simple Object Access Protocol) 등의 국제 표준 방식의 연계 인터페이스를 지원하는 기술이 개발되어 이상적인 실시간 기업(Real Time Enterprise) 구현을 위한 컴퓨팅 기술로 주목 받고 있다[2, 4].

World Wide Web Consortium(W3C) 내 Web Services Architecture Working Group은 'Web Service란, software application이 이종(異種)의 운영 체제에서 실행되고 상이(相異)한 언어로 작성되었을 지라도 네트워크 상에서 상호 운용될 수 있도록 하는 software system이다. WSDL(Web Services Description Language)과 같이 기계에 의해 처리 가능한 포맷으로 인터페이스가 기술되므로 다른 시스템들은 해당 Web Services와 사전 규정된 형식의 SOAP(Simple Object Access Protocol) 메시지를 주고받으며 상호 작용한다. 이때 SOAP 메시지는 전형적으로 HTTP를 사용하여 전달되며, web과 관련한 다른 표준들과 연계되어 XML의 직렬화가 진행된다.'라고 정의하고 있다.

네트워크 상에서 이종(異種) 시스템 간의 대화(communication)는 공통 언어 역할을 하는 XML을 사용함으로써 가능하다. 이를 위해 기존의 자원(예컨대 기 구축되어 있는 EJB(Enterprise Java Beans) component)에 XML 문서를 생성하고 해독하는 기능만 더하도록 포장(wrapping)하여 서비스 형태로 전환하도록 한다.

Web Services는 그 개념으로부터 Web과 Service 측면으로 나누어서 특징 및 장점을 설명할 수 있다[5].

1) Web 측면에서의 Web Services 특징 및 장점

① **Web 기반의 protocol**: Web Services는 공용의 인터넷 상에서 HTTP를 이용하여SOAP 메시지를 전달한다. 전송을 위해 HTTP이라는 protocol을 사용하기 때문에 방화벽에 제한 받지 않으며 이기종 환경에서도 작동할 수 있다.
② **상호운용성(interoperability)**: SOAP은 이종(異種) 시스템 간 상호운용이 가능한 공용 표준으로 기술된다.
③ **XML 기반**: eXtensible Markup Language는 기계가 판독할 수 있으며, World Wide Web Consortium(www.w3.org)에 의해 지속적으로 관리되고 있는 개방형 web 표준이다.

2) Service 측면에서의 Web Services 특징 및 장점

① **Modular**: Web Services component는 그 자체로도 유용하며, 재사용가능하고, 더 큰 component로 조립될 수 있다.
② **Self-described**: Web Services는 기계가 판독할 수 있는 언어로써 어떠한 서비스를 제공하고 있는지, 어디에 위치하고 있으며, 자신을 호출하기 위해서는 어떻게 해야 하는 지에 대한 정보를 담고 있는 interface를 서술하고 있다(WSDL; Web Services Description Language).
③ **Implementation-independent**: Web Services의 interface와 실제 구현은 분리되어 있다. Web Services 내 business logic을 구현하고 있는 언어와 플랫폼과는 무관하게 SOAP 메시지를 주고받을 수 있다.
④ **Published**: Service description(WSDL)은 repository(예컨대UDDI(Universal, Description, Discovery, and Integration))로 publish될 수 있으며, 사용자가 repository를 검색하여 서비스에 접근할 수 있다.

이와 같은 특징 및 장점을 가진 Web Services를 상위에서 도출한 실시간 기업(RTE)의 구축 방안에 접목한다. 즉, legacy functionality를 encapsulated Web Services component(s)로 추출(extract)하여 재조립한 composite application(s)를 구성함으로써 소위 application assembly environment를 구현할 수 있도록 한다.

3. 소프트웨어 아키텍처 스타일과 Service Tier

일반적으로 기업의 Architecture는 크게 비즈니스 아키텍처(Business Architecture), 소프트웨어 아키텍처(Software Architecture), 기술 아키텍처(Technical Architecture)의 3개의 주요 아키텍처로 구분할 수 있다.

시스템 설계 시에는 비즈니스 아키텍처(BA)에서 도출된 단위 프로세스를 시스템의 기능적, 비기능적인 요구사항에 맞추어 component들을 도출하고, component들 간의 관계와 component 내부구조에 대한 다양한 표준모델을 나타내기 위해 소프트웨어 아키텍처(SA)를 사용한다.

전체 구성도에서 소프트웨어의 작동원리를 중심으로 티어(tier) 혹은 레이어(layer)를 정의하기 위해서는 소프트웨어 아키텍처의 표현 중 소프트웨어

아키텍처 스타일을 이용하여 소프트웨어를 구성하는 클래스 요소와 이들 간의 상호작용을 표현하게 되는데, 본 논문에서는 이 소프트웨어 아키텍처 스타일에서 service tier를 추가함으로, business activity를 수행하는 IT Services를 구현하는 여러 구성 컴포넌트들 간의 관계와 흐름, 컴포넌트 각각의 정의는 은닉하고 해당 IT Services의 도식화한 행동을 표현해 보고자 한다.

표 1 - service tier를 추가한 소프트웨어 아키텍처 스타일

비즈니스 아키텍처							
소프트웨어 아키텍처	어플리케이션 아키텍처						
	소프트웨어 아키텍처 스타일	Presentation Tier	Workspace Tier	Workflow Tier	Service Tier	Business Tier	Integration Tier
		①	②	③		④	⑤
	설계 메커니즘						
기술 아키텍처							

제 3장 RTE 구현을 위한 Web Services

1. [Step 1: Workflow-based Component Identification]

Web Services는 기계와 사람 모두가 이해할 수 있는 가독성 있는 인터페이스 즉 WSDL(Web Services Description Language)로 스스로를 설명(self-describable)할 수 있기 때문에 외부의 서비스 사용자가 개별 Web Services의 business logic을 일일이 살펴보지 않고 인터페이스만으로 용이하게 검색하고 구현할 수 있다는 점에서 객체나 컴포넌트보다 재활용성이 뛰어나다. 또한 각각의 Web Services는 독립적이고 의미 있는 소프트웨어 단위이기 때문에 하나 이상의 Web Services를 모은 후 조립(composable)함으로써 또 다른 의미 있는 소프트웨어를 구축할 수 있다.

이와 같은 특성을 가진 Web Services를 business process에 기반하여 구현한다면 재활용에 따른 시간과 비용 절감 외에도 Web Services 각각이 business process를 의미한다는 점에서 business process가 변경된다면 이에 따라 시스템도 민첩하고 신속하게 변경될 수 있다.

업무처리흐름인 business process를 기준으로 재정렬된 service tier를 설계하기 위해서는, 먼저 business activity와 연관된 IT Services를 파악할 수 있어야 한다([그림 1] 참조). 산재한 여러 classes와 business logic 등을 business activity를 기준으로 식별 가능하고 목적 지향적인 집합으로 묶는(grouping)다. 한 묶음(grouping)은 하나의 workflow-based component이자, business component을 의미하게 된다.

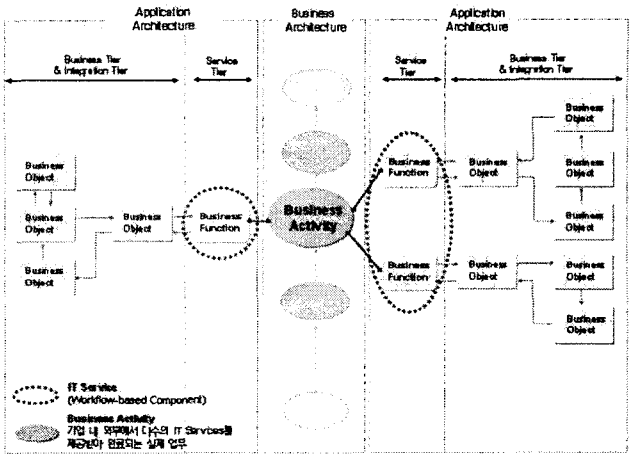

그림 1 - 기업 내 외부로부터 다수의 IT Services를 제공받아 완료되는 Business Activity

2. [Step 2: Business Interaction Identification]

일련의 business process를 완료하기 위해 하나 이상의 기업 내 외부에서 제공되는 business components가 필요하다. Business component가 기업 내 외부로 공개(expose)되는 경우는, 하나의 business activity를 완료하기 위해 기업 내·외부에 위치하는 비즈니스 관계자들이 소유하고 있는 응용 시스템과 business interactions가 발생하는 경우이다. 이것은 business service가 되며, [Step3~ 4] 과정을 통해 추가적인 분석과 정제작업을 통해 Web Services component로 개발된다.

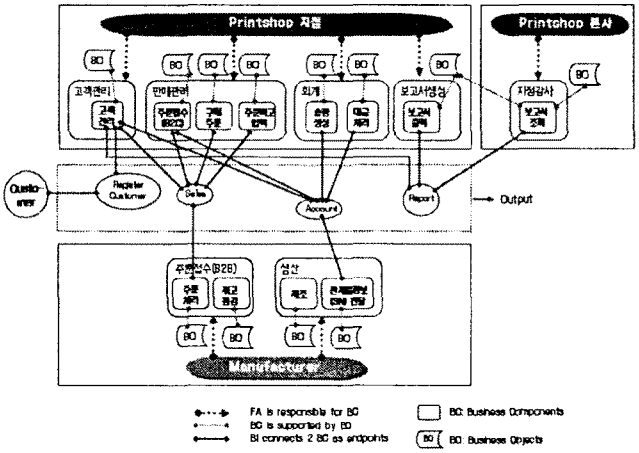

그림 2 - IT서비스의 제공 또는 이용과 관계된 business interactions의 식별

3. [Step 3: Candidate Service Identification]

[Step1]에서 business activity에 기반하여 workflow-based component(business component)를 도출하고, [Step2]에서 business activity를 완료하기 위해 추가적으로 외부로부터 제공받아야 하거나 외부로 공개해야 할 business component를

파악하였다면, [Step3]에서는 business component를 기업 내 외부와 서비스를 주고받도록 하는 Web Services component로 전환할 수 있도록 명세하는 단계이다.

Business process element(business activity)별 ①필요한 IT Services, ②해당 IT Service의 위치, ③해당 IT Service의 공개여부, ④해당 IT Service에 대한 비기능적 요구사항(nonfunctional requirements)을 작성한다([표2~ 3] 예시 참조). 이 단계에서 business process에서 요구하는 IT Services 목록을 작성함으로써 중복되는 IT Services를 발견할 수 있으며, 해당 IT Service를 제공하는 organization 정보를 통해 기업 내 외부로 공개(expose)되어야 하고 제공받아야 하는 서비스를 발견할 수 있다. 또한, 해당 IT Service에 대한 비기능적 요구사항(nonfunctional requirements)을 참고하여 서비스 별 차별적인 운영과 관리가 가능하다.

표 2 - Retailer-side에서 식별된 candidate service

Retailer-Side							
Business Functions	Business Activity	Associated IT Services		IT Services Location	공개	Nonfunctional Requirements	Associated Organization
고객관리	고객관리	고객관리 서비스	C	지점 1 호	o	N/A	지점 1 호, 지점 2 호
판매관리	주문접수 (B2C)	고객관리 서비스	R	지점 1 호	-	-	지점 1 호, 지점 2 호
		주문접수(B2C) 서비스	C	지점 1 호	o	Availability >= 0.90	지점 1 호, 지점 2 호
	구매주문 (B2B)	구매주문(B2B) 서비스	C	지점 1 호	o	Execution Duration < 5.0sec	지점 1 호, 지점 2 호
			C	manufacturer	-	-	지점 1 호, 지점 2 호, manufacturer
	주문비고입력	주문접수(B2C) 서비스	R	지점 1 호	-	-	지점 1 호, 지점 2 호
		주문비고입력 서비스	C	지점 1 호	o	N/A	지점 1 호, 지점 2 호
외계		고객관리 서비스	R	지점 1 호	-	-	지점 1 호, 지점 2 호, manufacturer
		주문접수(B2C) 서비스	U	지점 1 호	-	-	지점 1 호, 지점 2 호
			C	지점 1 호	o	Fidelity >= 0.80	지점 1 호, 지점 2 호, manufacturer
	대금처리	고객관리 서비스	R	지점 1 호	-	-	지점 1 호, 지점 2 호
			U	지점 1 호	-	-	지점 1 호, 지점 2 호, manufacturer

		대금처리 서비스	C	지점 1 호	o	Fidelity >= 0.9989	지점 1 호, 지점 2 호

표 3 - Manufacturer-side에서 식별된 candidate service

Manufacturer-Side							
Business Functions	Business Activity	Associated IT Services		IT Services Location	공개	Nonfunctional Requirements	Associated Organization
주문 접수 (B2B)	주문처리	주문처리 서비스	C	manufacturer	o	Reliability >= 0.90	지점 1 호, 지점 2 호, manufacturer
	재고점검	재고점검 서비스	C	manufacturer	x	N/A	manufacturer
생산	제조	제조 서비스	C	manufacturer	x	N/A	manufacturer
	완제품정보(SN) 전달	완제품정보(SN) 전달 서비스	C	manufacturer	x	Fidelity >= 0.9989	manufacturer, 지점 1 호
	완제품정보 (SN) 전달		R	지점 1 호	-	-	지점 1 호, 지점 2 호, manufacturer

위 [표 2~ 3]의 표기 중 굵게 표시된 cell은 [Step 1]단계의 결과로, business process를 참고하여 도출된 workflow-based component를 의미하며, 각 component는 기업 내 외부로 공개되어 IT 서비스를 제공할 수 있는 독립성과 재사용성을 가지고 있다. Associated IT Services는 [Step 2]단계의 결과로, 하나의 business activity를 외부로부터 제공받아야 하는 associated IT services를 의미한다.

4. [Step 4: Web Services component Identification]

Candidate Service 중 기업 내 외부로 공개(expose)할 services를 도출하고, Web Services 기술을 적용하여 Web Services component를 개발하는 단계이다.

여기서 주목할 점은 특정 business activity를 완료하기 위해 필요한 Web Services component가 반드시 제공해야 할 business method와 입출력정보를 정의하기 위해 sequence diagram을 작성하고 있다는 점이다. Sequence diagram의 actor는 최종 사용자이며 interactions의 주체는 IT Services이다.

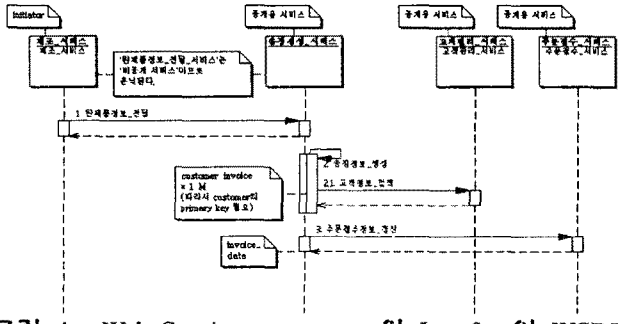

그림 4 - Web Services component의 Interface인 WSDL 도출을 위해 작성하는 Sequence Diagram

이상의 Web Services components 간 interactions를 sequence diagram을 통해 명세함으로써 Web Services

component의 Interface인 WSDL(Web Services Description Language) 상에서 공개용 operations를 직관적으로 파악할 수 있다. Sequence diagram을 통해 파악한 공개용 operations를 기반으로 아래 기업 내 외부로 공개(expose)할 business services를 설계한다.

예컨대 [그림 5]의 제조_서비스, 송장생성_서비스, 고객관리_서비스, 주문접수_서비스는 [그림 4]의 sequence diagram으로부터 파악한 공개용 operations를 구현하게 된다.

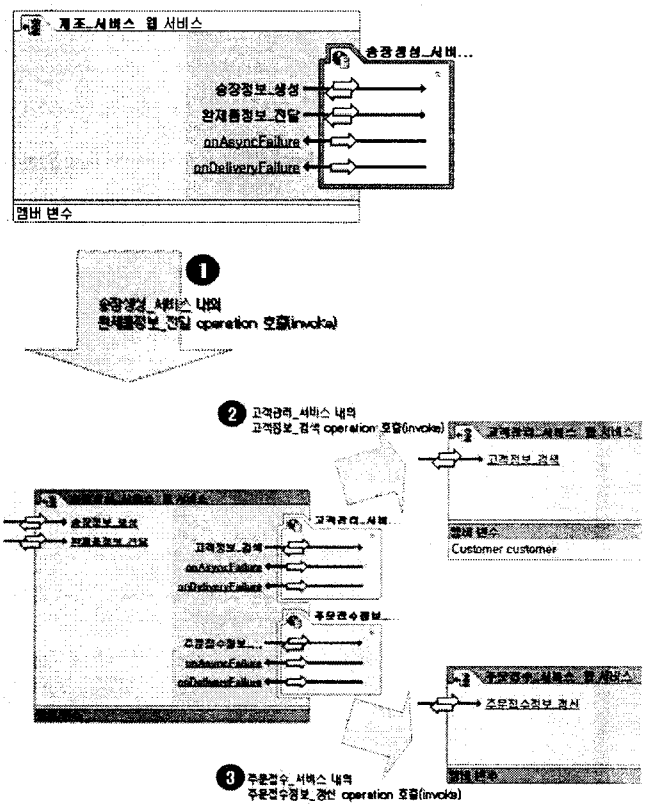

그림 5 - '제조'라는 Manufacturer System 내 기능(use case)을 완료하기 위해 제공받는 서비스 간 흐름도

일반적으로 지점이나 자회사에서 client module만 가지고 중앙의 server module을 invoke하는 경우나 기업 내(local integration)에서 중 하위 수준의 EAI(Enterprise Application Integration)를 이용하여 business functions와 business objects로의 접근을 위한 internal business interactions를 구현한 경우에는 fine-grained service를 도출하지만, B2C와 B2B applications에서 사용할 통합 인터페이스가 필요한 external business integrations를 구현할 경우에는 일련의 연결된 business functions의 coarse-grained services를 도출하도록 한다[6].

예컨대 [표 2]에서 보는 바와 같이 business activity별 associated IT services 중 공개 대상이 되는 것은 fine-grained service가 되어 지점2호의 client module이 지점1호의 service module를 invoke할 수 있게 된다. 반면, [표 3]의 생산 중 완제품정보(SN) 전달이라는 business activity와 같이 Manufacturer-Side의 '완제품정보(SN) 전달 서비스'가 호출(invoke)되면 이어서 [표 2]의 Retailer-Side의 '송장생성 서비스'가 호출(invoke)되고 Retailer-Side로부터 '송장생성 서비스'의 완료 통보를 받음으로써 B2B에 의한 '완제품정보(SN) 전달'이라는 business activity가 완료될 경우, coarse-grained service로 구현된다. [그림 5]는 하나의 기능(use case)을 완료하기 위한 서비스 간 흐름(service flow)으로 표현된 경우, 개별 fine-grained services가 서비스 흐름에 따라 더 큰 서비스로 묶음(grouping)되어 coarse-grained service로 제작된 예이다.

제4장 연구의 의의

1) 본 논문은 비즈니스 영역(business domain)에서 기업 내 외부로 공개(expose)되는 Web Services component의 적절한 granularity를 산출하기 위해 다음을 고려한다.

① 하나의 business activity에서 여러 비즈니스 관계자들과 business interactions가 발생할 경우 해당 business activity는 fine-grained service로써 기업 내 외부로 공개(expose)하나, 기업 내부에서 사용하는 (즉, 기업 내 외부로 공개되지 않는) 경우에는 은닉(encapsulate)하거나 다른 candidate service (즉, Web Services 기술을 적용하기 전의 business component)들과 조립되어 business logic을 수행할 수 있도록 한다.

② 지점이나 자회사에서 client module만 가지고 중앙의 server module을 invoke하는 경우나 기업 내(local integration)에서 중 하위 수준의 EAI(Enterprise Application Integration)를 이용하여 Business Functions(BF)와 business objects로의 접근을 위한 Internal business interactions를 구현한 경우에는 fine-grained service를 도출하지만, B2C와 B2B applications에서 사용할 통합 인터페이스가 필요한 external business integrations를 구현할 경우에는 일련의 연결된 Business Functions의 coarse-grained services를 도출하도록 한다.

③ Business process의 변경이 발생하는 영역(boundary)을 고려한다면, 변경이 발생할 부분에 대해서는 은닉(encapsulate)하고 그렇지 않은 부분에 대해서는 외부로 공개(expose)함으로써 외부의 여러 비즈니스 관계자들 간 interface의 변경에 효과적으로 대응할 수 있도록 한다.

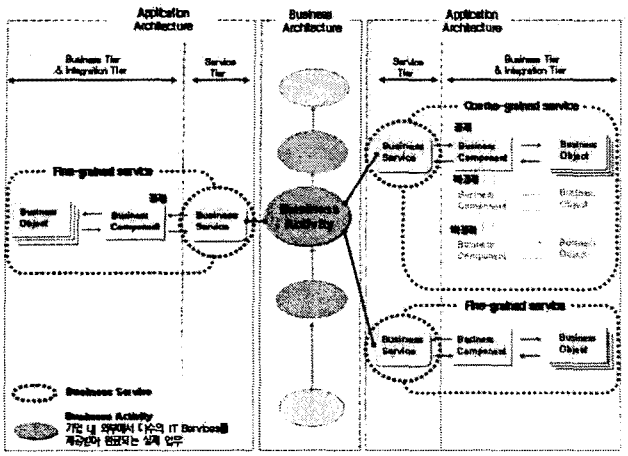

그림 6 - service tier로 배포된 fine-grained service와 coarse-grained service

2) 본 논문은 Web Services component가 business events 또는 customer demands, technical innovation의 변화에도 신속하고 유연하게 대응할 수 있으되, 기존의 IT 자원과 자산을 효과적으로 사용할 수 있도록 하기 위해, [그림 6]에서 보는 바와 같이 소프트웨어 아키텍처 스타일 중에서 service tier로 Web Services component를 배포(deploy)하도록 한다.
통상적으로 기존의 object 또는 component보다 더 큰 단위(coarse-grained)의 business function service를 구현하기 위해서는 하나 이상의 intermediate tiers가 필요하다[7]. Intermediate tiers는 최종 고객군의 요구사항, 재활용 정책, 공개수준 등을 고려하여 업무에 필요한 어플리케이션 실현 방식이나 데이터 처리 흐름 등을 적절히 은닉(encapsulation)함으로써 기업 내 외부에는 필요한 정보와 로직만을 공개할 수 있게 한다. 따라서 intermediate tiers를 적용하면 이전보다 재활용성(reusability), 추상화 수준(abstract level), 유연성(flexibility), 민첩성(agility)은 제고될 수 있다[7].
여기서는 intermediate tier의 장점을 확대한 service tier를 적용할 것을 제안하고 있다. 즉, 본 논문에서 제안하는 service tier는 business activity를 완료하기 위해 기업 내 외부로부터 제공받는 business functions 즉 IT Service가 되는 Web Services components로써 구성된다. 따라서 본 논문에서 제안하는 service tier는 intermediate tier와 Web Services, business process-based component가 제공하는 장점들을 통합한 것이므로 비즈니스 활동 영역의 지리적인 확장, 다수의 비즈니스 파트너 및 시스템의 관여, M&A(Merge & Acquisition), 고객 니즈의 다양화, 전략의 변화, business process의 변경에 대해 더욱 신속하고 민첩하게 대응할 수 있는 역량을 제공한다는 점에서 그 의의를 찾아볼 수 있다.

3) 본 논문은 Web Services component의 interface 작성을 위한 design mechanism을 고려한다.
특정 기능(use case)을 실현(realization)하는데 참가하는 services 간 상호 교환 메시지를 참고하여 외부에서 호출(invoke)할 수 있는 business method를 정의할 수 있는 design mechanism을 고려할 수 있어야 한다.
본 논문에서는 이를 위해 sequence diagram을 활용하였다. 이 경우 actor는 최종 사용자이며 interactions의 주체는 IT services (즉, Web Services components)이다. IT services 간 전달 메시지를 통해 WSDL(Web Services Description Language) 상에서 정의해야 할 business method (즉, 공개용 operations)를 직관적으로 파악할 수 있다는 점에서 그 의의를 찾아볼 수 있다.

Acknowledgments

Thanks to Father, Mother, Joseph(Brother), Mina(Sister) and the Lord my Jesus. I love you Really Really!

References

[1] Boden T. (2004). "The grid enterprise- structuring the agile business of the future". *BT Technology Journal*, Vol. 22, no. 1, pp. 107-117.
[2] Khosla Vinod, and Murugan Pal. (2002). "Real Time Enterprise: A Continuous Migration Approach". *Information Knowledge System Management*, Vol. 3, pp. 53-79.
[3] Jin Kai, Tongsen Wang, and Annamalai Palaniappan. 2005. "Improving the Agility of Automobile Industry Supply Chain". *ACM International Conference Proceeding Series*, Vol. 113, pp. 370-374.
[4] Tweney, D. (2001). *What's going on down at the plant.* Business 2.0.com
[5] Fremantle Paul, Sanjiva Weerawarana, and Rania Khalaf. (2002). "Enterprise services: examining the emerging field of web services and how it is integrated into existing enterprise infrastructures". *Communications of the ACM*, Vol. 45, no. 10, October, pp. 77-82.
[6] Baghdadi Youcef. (2005). "A web services-based business interactions manager to support electronic commerce applications". *ACM International Conference Proceeding Series*, Vol. 113, pp. 435-445
[7] Frankel David, and John Parodi. (2002). "White Paper: Using Model-Driven Architecture™ to Develop Web Services". IONA Technologies PLC, Second Edition.