모바일 융합 서비스 지원 공통 프레임워크 설계

탁미경*, 김행곤**
* **대구가톨릭대학교 컴퓨터정보통신공학부
e-mail: {lebbenl, hangkon}@cu.ac.kr

Design of Mobile Convergence Service-Oriented Common Framework

Mi-Kyung Tak*, Heang-Kon Kim**
* **Dept of Computer information & Communication Engineering
Cathoinc Univ. of Daegu, Korea

요 약

모바일 융합 서비스를 개발하기 위해 단순 컴포넌트의 조립(integration)이나 합병(merge)과 융합 서비스를 식별하는 기존의 개념만으로 부족하다. 비즈니스 서비스 모델에 적합한 비즈니스 기능을 가진 융합 서비스를 지원하는 응용 시스템 개발을 위해 비즈니스 서비스 모델에서부터 서비스 모델로의 매핑과 다양한 자산(assets)정보가 필요하며 이들을 설계 단계에서 구현 단계를 지원하는 적절한 프레임워크가 절대적으로 필요하다. 따라서 본 논문에서는 소프트웨어 집약적인 신기술을 도입하여 시장 전략과 융합 모바일 도메인에 적합한 소프트웨어 개발 도구 및 환경 지원을 위한 모바일 융합 서비스 기반 공통 프레임워크(SOCF:Service Oriented Common Frameworks)를 연구하며 실생활의 모바일 서비스가 제공자와 고객 간에 재사용 단위의 추상화 수준을 단순 컴포넌트를 넘어 서비스 제공과 사용 사이(provide/consume)의 서비스 수준까지 확대하여 다양한 서비스 자원들을 통합하고 재사용하여 서비스 융합 시스템 개발을 지원하는 프레임워크 설계 및 구현에 관한 내용을 서술한다.

1. 서론

최근 서비스 지향 아키텍처(Service-Oriented Architecture)가 기업 인프라의 복잡성 및 유지비용을 최소화하고, 기업의 생산성과 유연성을 극대화할 1)것으로 기대되어 차세대 소프트웨어 아키텍처로써 주목받고 있다. 서비스 지향 아키텍처의 서비스 식별 방법에 대한 연구는 시버스 지향 아키텍처를 지원하기 위한 개발방법론인 SODA(Service Oriented Development of Applications), SOUP(Service Oriented Unified Process), SOMA(Service Oriented Modeling and Architecture), SOAD(Service Oriented Analysis and Design) 등에 언급되어 있으나, 실제로 모바일 융합 개발 환경을 적절하게 지원하는 융합형 프레임워크에 대한 구체적인 연구는 진행되지 않고 있다.[1][2][3]

따라서 많은 개발자들이 융합 컴포넌트 모델과 융합 서비스 모델간의 차이를 모호하다고 생각한다. 융합 컴포넌트 기반의 시스템을 융합 서비스 지향 프레임워크로 매핑 또는 전향하는 경우, 기존의 컴포넌트를 그대로 ESB(Enterprise Service Bus)나 모바일 융합 서비스에 얹어 서비스화 하는 것이 기업에서 시스템을 서비스 지향 아키텍처화 하는 가장 쉬운 길이라고 오해하고 있다. 이러한 접근은 비즈니스 모델로부터 출발한 것이 아니라 기존의 엔터프라이즈 어플리케이션으로부터 서비스를 도출해 나가는 방식으로서 융합 서비스 지향분석이나 설계를 반영할 수 있는 프레임워크를 적용하는 절차나 방법이 고려도고 있지 않다. 이는 개발 프로세스에서 응용 도메인 비즈니스 모델로부터 출발하지 않고 단순 엔터프라이즈 어플리케이션 시스템의 조립(Composition)을 통해 규모가 좀 더 큰 컴포넌트를 생성하는 데에 초점을 두는 모바일 융합 소프트웨어 개발접근을 시도하고 있기 때문이다. 따라서 컴포넌트 시스템의 구현 환경만 모바일 융합 서비스 지향 아키텍처로 바꾸는 작업은 실제 서비스 지향 아키텍처 도입의 초보적인 단계로 언급되고 있으며, 궁극적인 서비스 지향 아키텍처의 설계 목표인 민첩성(agility), 융통성(flexibility), 약결합(loosely coupled) 등을 달성하기 어렵게 만든다.

따라서 본 논문에서는 모바일 융합 서비스를 개발하기 위해 단순 컴포넌트의 조립(integration)이나 합병(merge)만으로 융합 서비스를 식별하는 기존의 개념에서 탈피하여, 비즈니스 서비스 모델에 적합한 비즈니스 기능을 가진 융합 서비스를 지원하는 응용시스템 개발을 위해 비즈니스 서비스 모델에서부터 서비스 모델로의 매핑과 다양한 자산(assets)정보를 관리하며 이들을 설계 단계에서부터 구현 단계를 지원하는 적절한 프레임워크에 대해 논한다.

* 본 논문은 2010년도 한국연구재단 지역우수과학자 지원사업 지원을 받아 수행된 연구임.(No.R. 2010-0017089)

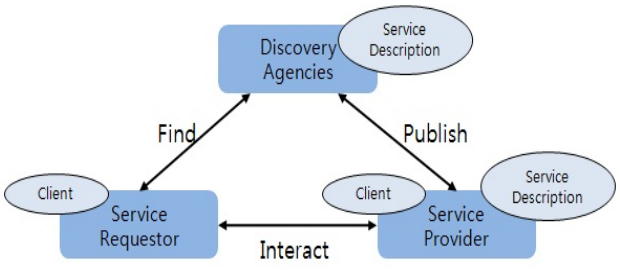
2. 관련연구

2.1 SOA(Service-Oriented Architecture)

기업 인프라의 복잡성 및 유지비용을 최소화, 기업의 생산성과 유연성을 극대화시킬 수 있는 것으로 경영환경이 빠르게 급변하는 최근에 떠오른 이슈이다.

기업 인프라의 복잡성과 유지비용을 최소화하고 기업의 생산성을 높일 것으로 전망되어 차세대 소프트웨어 아키텍처로 주목받으며 소프트웨어의 구성요소 및 이들의 가시적 속성과 이들의 관계로 구성된 시스템 구조를 의미한다. SOA기반의 차세대 소프트웨어 기술들의 등장은 기존 소프트웨어 시장의 침체된 상황을 정리하고 신규시장의 확대와 시장의 활성화 유도하고 최근의 비즈니스 환경은 과거 독립적인 조직 및 프로세스에 의해 주도되는 수직적 통합에서 고객, 공급자, 파트너 등 다수 기업과의 관계적 협업 관계가 중시되는 수평적 통합 환경으로 변화하고 있다. 이러한 관계적 협업이 중시되는 비즈니스 환경에서 경쟁력 있는 기업은 급변하는 시장요구에 민첩하게 대응할 수 있어야 한다. 이러한 비즈니스 요구에 효율적으로 대응하기 위한 기업 IT 아키텍처로 대표되는 것이 SOA이다. SOA는 전통적인 프로그램 중심의 설계/개발 방식에서 비즈니스 프로세스 관점에서 재활용 가능한 단위로 서비스를 설계/개발하게 함으로써 특정 프로세스나 서비스 변경 또는 내부/외부 시스템과의 비즈니스 통합 시 효율적이고 빠른 대응이 가능하다는 점에서 그 의미가 깊다

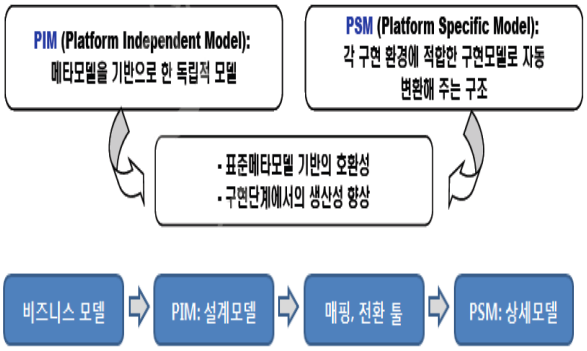
SOA는 특정 기술이나 플랫폼에 종속되지 않고 느슨한 결합(Loosely Coupled)을 가지고 상호 연동할 수 있는 서비스들의 조합으로 어플리케이션 개발을 가능하게 하는 정보 시스템 아키텍처이다. 즉, 한 덩어리의 방대한 코드로 이루어진 어플리케이션들을 각각 개발하는 대신 각각의 비즈니스 기능을 수행하는 서비스를 구성하고, 이 서비스를 조합하거나 분리함으로써 비즈니스 프로세스들을 구현할 수 있게 하는 정보시스템 구축을 목표로 한다.[4]



(그림 1) Service Oriented Architecture

2.2 MDA(Model Driven Architecture)

MDA(Model Driven Architecture)는 OMG(Object Management Group)에서 공식 발표된 이후로 소프트업계에서는 매우 유망한 패러다임으로 받아들여지고 있다. 3년간의 짧은 시간에, 40 개가 넘는 회사에서 MDA를 구현하는 소프트웨어를 출시하였다. 아직 많다고는 할 수 없지만 규모가 있는 사례들을 통해 새로운 개념을 적용하고 또 성공하였다는 것은 의미하는 바가 크다. 이러한 결과를 바탕으로 MDA가 소프트웨어의 개발과 유지보수 업무 방식에 대변혁을 가져오는 잠재성을 가졌다고 말할 수 있으며 MDA가 빠른 속도로 대중화되어 가고 있다. MDA는 새로운 소프트웨어를 개발하는데 있어 더 큰 생산성을 얻기 위해서 필요하다. 여러 측면에서 봤을 때 소프트웨어 개발하는 속도가 하드웨어의 파워보다 뒤떨어지기 때문인데, 여기에는 이유가 있다. 첫째로, 소프트웨어 애플리케이션은 금융에서 국방에 이르기까지 광범위한 영역의 문제들을 대상으로 만들어진다. 게다가 이들 애플리케이션의 복잡성 또한 더욱 커지고 있다. 둘째로, 소프트웨어 기술 자체의 복잡성이 폭발적으로 증가하고 있으며, 개발자들은 HTML, XML, WML, 다계층 아키텍처, J2SE, J2EE, J2ME, .NET, 컴포넌트와 오브젝트 모델, 컨넥터, 메시지 브로커 등과 같은 여러 가지 기술을 익혀야만 한다. 소프트웨어 개발의 주요 과제는 생산성을 증대하면서 동시에 유연성과 표준화를 조화시키는 것이다. MDA가 소개되기 전까지, 기존 언어는 이런 요구사항 중 한 두 가지는 맞추었지만 모두를 만족시키지는 못하였다. MDA의 출현으로, 완벽한 유연성을 유지하면서, 표준 환경 내에서 높은 생산성이 가시적으로 현실화되었다. 또한, 수동적으로 작성되는 코드가 적어지고 최상의 사례를 적용할 수 있는 패턴을 사용함으로써 애플리케이션 품질도 개선된다.[5]



(그림 2) MDA 모델

3. SOCF 아키텍처 설계

3.1 SOCF 분석 및 설계

본 연구에서 수행하는 SOCF 서비스 융합 관리의 시작은 요구사항 관리 프로세스이며 SOCF 사용자와 함께 하는 활동을 포함하며, 프레임워크에서 명시하는 유저 스토리를 산출물로 작성해 이를 바탕으로 관리 명세서를 작성하는 활동들을 포함하여 연구한다. 표 1과 같이 1단계에서는 SOCF 요구사항 관리가 이뤄져야하며 그에 따른 활동으로는 유저스토리 작성과 개발자, 관리자, 고객이 한 자

리에 모여야 하는 활동을 명시하며, 모바일 서비스 소프트웨어 특성을 고려한 요구사항 분석 활동을 명시한다. 2단계에서는 1단계보다는 더 나아가 요구사항 변경에 대한 관리 활동을 명시한다. 이는 모바일 서비스 소프트웨어 특성상 자주 발생하는 요구사항 변경에 따른 대안 마련을 활동으로 명시하여 언제든지 요구사항 변경에 대비하고 이를 관리해야할 것이다. 3단계에서는 요구사항이 설계 혹은 개발 단계에서 이뤄져야 할 부분을 미리 검증을 통해 예측관리를 할 수 있도록 하는 것이다. 이에 따른 활동으로는 요구사항 변경 사항 자료를 항상 모니터링 하고 분석해야한다

<표 1> SOCF 요구사항 프로세스 활동

SOCF 요구사항 관리 프로세스		
산출물		유저스토리, 요구사항명세서
1 단계	목표	요구사항을 관리한다.
	활동	· 모바일 서비스 융합소프트웨어 요구사항을 유저스토리 혹은 그에 준하는 문서를 작성한다. · 기능적/비기능적 요소를 식별한다. · 개발자, 관리자, 고객이 한 자리에서 모임을 갖는다. · 모바일 서비스 융합소프트웨어 특성을 고려하여 요구사항을 분석한다.
2 단계	목표	요구사항 변경을 관리한다.
	활동	· 고객의 요구사항 변경사항을 수집, 관리한다. · 문제 발생에 대한 대안을 마련한다.
3 단계	목표	요구사항 추적성을 확보하고 요구사항을 검증한다.
	활동	· 요구사항 추적성을 확보하기 위한 변경 사항 자료를 분석한다. · 모바일 서비스 융합소프트웨어가 탑재될 하드웨어 제약사항을 검토하고, 요구사항명세서를 검증한다.

3.2 개발 SOCF 아키텍처

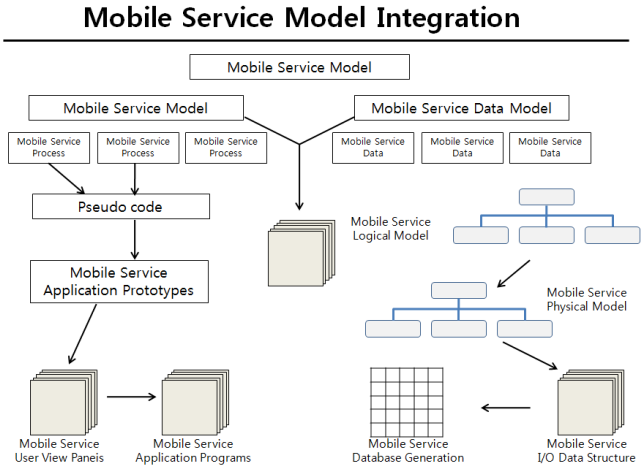
SOCF 아키텍처 설계는 정규화된 프로세스에 기반하며 요구사항 관리 프로세스 단계에서 모바일 서비스 사용자 중심으로 작성한 요구사항 문서를 바탕으로 쉽게 알아 볼 수 있는 문서로 작성하는 것이 주요 목표다. 표 2와 같이 1단계에서 목표는 앞서 설명된 은유적 설명이 포함된 설계에 관한 활동을 명시하며 2단계에서는 작성된 모바일 서비스 소프트웨어 아키텍처를 관리해야한다. 3단계에서는 구현단계에서 예측할 수 있도록 아키텍처를 모델링하여 최소 템플릿 소스까지 나올 수 있도록 활동을 명시한다.

<표 2> 아키텍처 설계 프로세스 활동

SOCF아키텍처 설계 프로세스		
산출물		프로토타입, 아키텍처 설계서
1 단계	목표	은유적 설명이 포함된 간단한 설계를 한다.
	활동	· 모바일 서비스 융합소프트웨어 아키텍처를 고객이 이해할 정도로 심플하게 설계한다. · 프로토타입을 작성한다. · 기능 구현 관련하여 비용을 따져 구현/재사용/구매를 결정한다.
2 단계	목표	모바일 서비스 융합소프트웨어 아키텍처를 관리한다.
	활동	· 모바일 서비스 융합소프트웨어 아키텍처와 하드웨어 제약사항을 참고하여 평가, 검증한다.
3 단계	목표	모바일 서비스 융합소프트웨어 아키텍처를 모델링한다.
	활동	· 템플릿 소스작성이 가능한 모델링을 한다. · 모바일 서비스 융합시스템 아키텍처로 확장한다.

3.3. SOCF에서 모바일 서비스 모델 통합

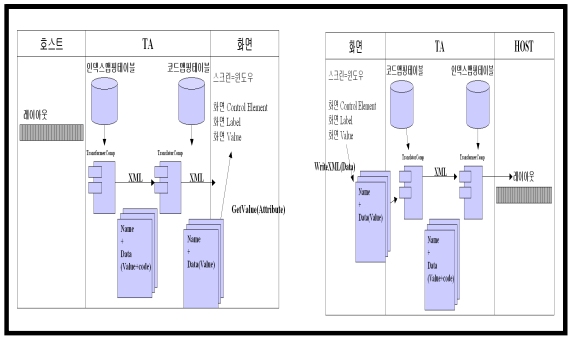
SOCF에서 초기 단계에서 생성되는 모델은 기존의 모바일 서비스 응용 모델과 모바일 서비스 데이터 모델로 분류하며(그림 2) 이들 모델을 플랫폼 독립적인 환경에서 검색하고 physical model을 통해 실행 하도록 하는 어댑터를 설계 구현한다. 다양한 개념적, 명세적 그리고 상세적 모델 타입으로서 모델 정의와 실행, 모니터링 기술이 이 어댑터를 통해 제공된다.



(그림 3) SOCF 모바일 서비스 모델 통합

3.4 SOCF 서비스 융합 어댑터 설계

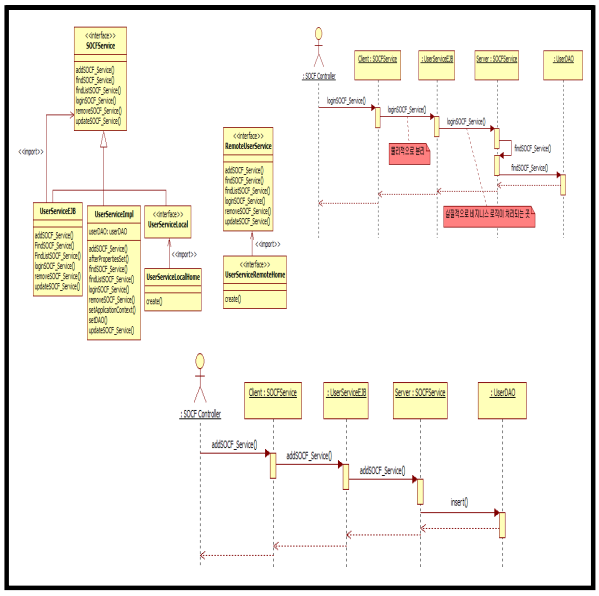
기존 레거시 시스템을 새로운 모바일 비즈니스 서비스 융합에 적합한 적용 가능한 자산으로 재구성 하여 효율적 재사용을 지원하는 어댑터를 개발 한다. 서비스간의 인터페이스의 차이를 보이는 두 서비스 간에 개조기 패턴을 활용하여 두서비스 사이의 인터페이스 불일치를 해결한다. 레거시 시스템의 안정적 서비스는 새로운 모바일 서비스 기반의 아키텍처와 통합되어야 한다. 레거시 시스템은 다른 장비, 다른 외부 애플리케이션으로부터 서비스 요청을 처리할 수 있는 기반을 갖추고 있다. 다시 말하면 레거시 시스템을 리소스 레이어로 보면 새로운 애플리케이션 레이어로부터의 요청된 서비스를 제공할 수 있다는 것이다. 복잡한 시스템을 모바일 서비스 아키텍처 정의 및 명세(Mobile Services Architecture Definition & Specification Definitione)를 통하여 SOCF 서비스를 생성, 획득, 조립을 위한 표준 참조 모델을 정의하고 이에 따른 메타 데이터와 서비스 명세 방법을 결정함으로써 서비스 수요, 공급, 활용의 핵심 지침을 마련할 예정이다. SOCF 융합 어댑터를 이용하여 클라이언트쪽에서 호스트쪽으로 서비스를 처리하는 방법을 연구하며 기본 계획은 그림과 같이 인덱스 맵핑 테이블을 사용하는 융합 애답터를 통해 해당 채널 클라이언트가 요구하는 형태의 데이터로 변환하여 전달한다(그림 3).



(그림 4) SOCF 서비스 재사용 어댑터

3.5 서비스 지원 관점 SOCF 상세 명세화

SOCF 아키텍처 설계 및 구현 단계에서 상세 명세화는 기존의 UML 방법을 도입하여 모델링한다. 그림 1과 같이 필요한 stake-holder를 Use case로 정의하고 필요한 class들을 추출하여 모델링한다. 마지막으로 SOCF 상세 명세는 sequence diagram을 통해 동적 행위를 서술하게 된다.

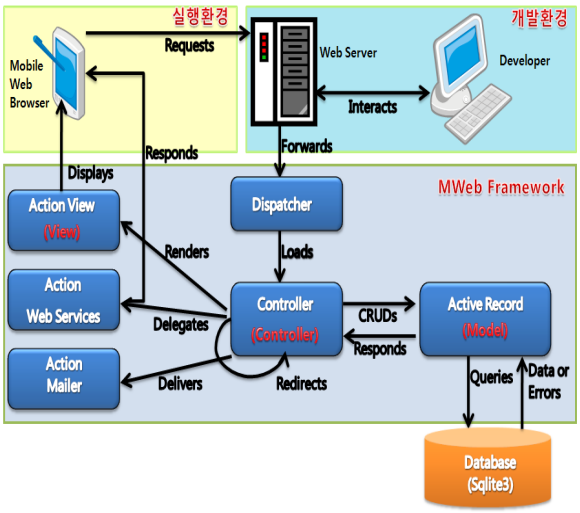


(그림 5) SOCF 상세 명세화 예 (UML 표현)

4. 실행·통합 뷰 및 모니터링 환경 설계

SOCF를 통해 개발한 애플리케이션을 다양한 환경에서 다양한 서비스를 융합하여 구동하면서 동작 상태를 확인(실행)하며 결과를 통해 애플리케이션의 전체 구동상태(통합) 및 개발한 모바일 서비스 애플리케이션의 간단한 내용을(뷰) 검토한다(그림 4).

모바일 웹 브라우저를 통해 SOCF와 적절한 interaction을 통해 모바일 융합 서비스의 Action들을 뷰 및 dispatch 그리고 제어를 하게 되며 이들 정보는 개발 단계에서 피드백 하여 서비스 애플리케이션을 수정·보완하게 된다.



(그림 6) SOCF 실행·통합 뷰 및 모니터링 환경

7. 결론

비즈니스 서비스 모델에 적합한 비즈니스 기능을 가진 융합 서비스를 지원하는 응용 시스템 개발을 위해서는 비즈니스 서비스 모델에서부터 서비스 모델로의 매핑과 다양한 자산(assets)정보가 필요하며 이들을 설계 단계에서부터 구현 단계로 지원하는 적절한 프레임워크가 절대적으로 필요하다. 그리하여 본 논문에서는 소프트웨어 집약적인 신기술을 도입하여 시장 전략과 융합 모바일 도메인에 적합한 소프트웨어 개발 도구 및 환경 지원을 위한 모바일 융합 서비스 기반 공통 프레임워크(SOCF:Service Oriented Common Frameworks)를 연구하여 실생활의 모바일 서비스가 제공자와 고객 간에 재사용 단위의 추상화 수준을 단순 컴포넌트를 넘어 서비스 제공과 사용사이의 서비스 수준까지 확대하여 다양한 서비스 자원들을 통합하고 재사용하여 서비스 융합 시스템 개발을 지원하는 프레임워크 설계 및 구현에 관한 내용을 서술하였다.

참고문헌

[1] Gregg Kreizman, "How to Build a Business Case for Service-Oriented Development of Applications in Government," Gartner. Industry Research, Sept. 2005

[2] 이현주,최병주,이정원, "서비스지향 아키텍처를 위한 컴포넌트기반 시스템의 서비스 식별, 정보과학회논문지 소프트웨어 및 응용, 제 35권 제2호 pp.70-80, 2008

[3] Ash Parikh, Rajesh Pradhan and Nirav Shah, "Modeling of Web Services : A Standards-Based Approach," Software Magazine, May, 2004

[4] 최은만, "Service Oriented Architecture와 재사용." 정보과학회지, 정보과학회지 제 24권 제11호, pp32-37, 2006

[5] Gabor Karsai, Sandeep Neema, David Sharp, "Model-driven architecture for embedded software: A synopsis and an example" Science of Computer Programming, Volume 73, No.1, pp.26-38, 2008