소프트웨어 시스템과 서비스 시스템의 유사성에 기반한 서비스 시스템 개발을 위한 체계적 설계 기법

(A Systematic Design Method for Service System Development based on Similarity between Software System and Service System)

전 원 영 † 장 수 호 † 김 수 동 ††
(Won Young Jeon) (Soo Ho Chang) (Soo Dong Kim)

요 약 서비스 과학(Service Science)은 경영과 경제, 공학분야가 상호 연동하면서 서비스를 인식하는 새로운 응용분야이다. 서비스 시스템은 전통적인 소프트웨어 시스템과 같은 자동화된 기능을 제공하면서, 동적인 컨텍스트 인식 및 분석과 이를 기반으로 한 의사 결정이 적용되어 더욱 지능적인 기능을 제공한다. 전통적인 소프트웨어 개발 접근법은 서비스 요구사항을 모델링하고 서비스 시스템을 설계하는데 비효율적인 부분이 있다. 따라서, 서비스 시스템을 개발하기 위한 효과적이고 체계적인 설계 방법론이 요구된다. 본 논문에서는 전통적인 소프트웨어 시스템과 서비스 시스템의 특징을 비교함으로써 서비스 시스템의 특징을 도출한다. 그리고, 서비스 시스템을 설계하기 위한 프로세스를 아키텍처, 컴포넌트, 워크플로우의 측면에서 제안하고, 생활 보조 시스템 (Living Assistance System)의 한 분야인 응급 상황 처리 시스템의 설계과정에 적용한다. 제안된 프로세스로 전통적인 소프트웨어 시스템 설계에서 서비스 시스템 설계로의 이동이 효과적으로 진행될 수 있다.

키워드 : 서비스 과학, 서비스 지향 아키텍처, 개발 프로세스

Abstract Service science is a new application area that implements services in an interdisciplinary area of management, economics, and engineering. Service systems provide functionalities of traditional software systems, moreover the functionalities are more intellectual in that they require dynamic context awareness, analysis, and decision making based on the recognized and analyzed contexts. However, conventional software development approaches do not sufficiently provide methods to model the service requirements and to design service-intensive systems. Therefore, there is a great demand on effective methodologies for developing service systems. In this paper, we compare traditional software systems with service-intensive systems in order to identify characteristics of the service systems. And, we propose a step-wise process to model service systems, in terms of architecture, components, and workflows. Then, we show a case study on an emergency handling system which is a type of living assistant systems. We believe that the proposed approach can be used in developing high-quality service systems effectively.

Key words : Service Science, Service-oriented Architecture, Development Process

1. 서 론

서비스는 비즈니스와 마케팅 분야에서 경제적인 이윤을 목적으로 하는 활동으로 취급되어 왔다. 서비스의 종류가 다양하고 복잡해지면서 서비스를 효과적으로 제공하는데 전산학적 기술적인 도움은 중요하게 인식되고 있다. 이러한 시점에서, 서비스 과학(Service Science)은 경영과 경제, 공학분야가 상호 연동하면서 서비스를

· 본 논문은 숭실대학교 교내연구비 지원으로 이루어졌음
† 학생회원 : 숭실대학교 컴퓨터
wyjeon@otlab.ssu.ac.kr
shchang@otlab.ssu.ac.kr
†† 종신회원 : 숭실대학교 컴퓨터 교수
sdkim@ssu.ac.kr
논문접수 : 2006년 10월 24일
심사완료 : 2007년 3월 16일

인식하는 새로운 응용분야로 대두되고 있다[1]. 이러한 서비스를 기술적으로 구현한 서비스 시스템은 전통적인 소프트웨어 시스템과 같은 자동화 기능을 제공하면서, 동적인 컨텍스트 인식 및 분석과 이를 기반으로 한의사결정이 적용되어 더욱 지능적인 기능을 제공한다. 예를 들어 생활 보조 시스템(Living Assistance System, LAS)[2]은 장애자와 노약자를 위하여 삶의 질을 향상시키고 인간 노동 중심적 보조 요구를 최소화 서비스를 제공하는 시스템의 한 종류이다.

서비스 시스템은 요구되는 기능의 지능화 정도의 측면에서 전통적인 소프트웨어 시스템과는 다르다. 서비스 시스템은 서비스 대상의 컨텍스트를 동적으로 인식하고 분석하며, 이를 기반으로 적절한 서비스를 결정하고, 이를 제공하기 위해 다양한 다른 서비스 시스템 및 시스템 관련자들과의 상호작용한다. 따라서, 서비스 시스템을 구축하기 위해서는 소프트웨어의 모듈뿐만 아니라 센서나 엑츄에이터와 같은 하드웨어 장치, 네트워크 구성요소, 유비쿼터스 기술들의 통합되어야 한다. 그러나 객체지향형 분석 및 설계와 같은전통적인 개발 접근법은 이러한 서비스 요구사항을 모델하고 서비스 시스템을 설계하는데 부족한 부분이 있다. 즉, 컨텍스트 인식에 대한 모델링이나 이로부터 도출되는 설계에 대한 이슈들이 서비스 기능적인 분석 및 설계에 체계적으로 적용되어야 한다.

본 논문에서는 전통적인 소프트웨어 시스템과 서비스 시스템의 특징을 시스템의 개발적 측면에서와 시스템의 운영적 측면에서 비교하고, 이를 기반으로 서비스 시스템 고유의 특징을 도출한다. 그리고, 서비스 시스템을 설계하기 위한 프로세스를 요구사항 획득 및 분석, 시스템 아키텍처 설계, 컨텍스트 설계, 컴포넌트와 인터페이스 설계, 설계모델 검증의 다섯 활동의 단계로 정의한다. 또한, 제안된 프로세스를 기반으로 LAS의 한 형태인 응급 상황 처리 시스템을 도메인으로 한 사례연구를 수행한다. 제시된 기법은 전통적인 소프트웨어 개발과 서비스 시스템의 개발의 구분이 명확하지 않은 가운데 새롭게 대두되고 있는 서비스 시스템을 효과적으로 개발하는데 적용 될 수 있다. 전통적 소프트웨어 개발 기법을 기반으로 서비스 시스템의 특징을 고려한 서비스 시스템 개발 기법을 제안함으로써 서비스 시스템 개발의 효율성 향상이 기대된다.

2. 배 경

서비스는 경제학, 공학, 비즈니스 관리와 같은 다양한 학문분야에 관련이 있는 분야에서 자주 사용되어 왔으며, 여러 문헌에 서비스에 대한 다양한 정의[3-5]가 있다. 이러한 정의로부터 서비스에서 서비스 제공자, 구독자, 서비스 유저와 같은 다양한 서비스 시스템 관련자들이 존재한다는 것을 발견 할 수 있다. 그리고 서비스에 의해 창출된 이익들은 서비스 시스템 관련자들에게 다양한 형태로 나타난다. LAS에서는 인간의 생활을 보조함으로써 이익이 창출되고, 창출된 가치는 제공자들의 가치와 고객의 가치로 분류될 수 있다. 서비스 제공자에게 이익은 경제적 수입을 증가시킴으로 창출 될 수 있다. 고객에게 가치는 건강, 안전, 편안함과 같은 인간 생활의 질을 향상시킴으로 창출 될 수 있다.

시스템으로서의 서비스 구현은 아직 초기 단계이기 때문에 이러한 서비스 시스템을 개발하기 위한 효과적인 방법론들이 필요하다.

OWL 기반의 Web service ontology(OWL-S)는 서비스를 설명하기 위한 언어이다[6]. OWL-S는 서비스를 기술하기 위한 세가지 주요 분야로 구성되어 있다. OWL-S는 서비스들의 속성과 기능성을 명백하고 컴퓨터가 해석할 수 있게 설명하기 위한 핵심 구성체들의 집합으로 웹 서비스 제공자들을 위해 고안되었다. OWL-S로 설명된 서비스들은 웹 서비스 기반의 서비스들로 제한된다. 그러나 본 논문에서 명세하고자 하는 서비스는 다양한 하드웨어 장치와 소프트웨어 그리고 서비스 시스템 관련자들로 구성된 서비스이다.

컨텍스트 인지는 서비스를 구현하기 위한 핵심 요소이다. 컨텍스트에 대한 다양한 정의가 다음과 같이 존재한다[7-9]. 이러한 정의들은 사용자 식별, 위치, 시간과 같은 컨텍스트를 표현하는 데이타 형태를 제안한다. 그리고 이러한 정의들은 컨텍스트의 값들과 구성요소의 타입이 바뀌는 것과 같은 컨텍스트 정보의 동적인 측면을 강조한다.

Dey는 컨텍스트 인지 어플리케이션의 구축을 지원하는 아키텍처인 컨텍스트툴킷(ContextToolkit)을 제안하였다[10]. 컨텍스트툴킷(ContextToolkit)은 위젯 (Widgets), 인터프리터(Interpreters), 어그리게이터(Aggregators)와 같은 세가지 주요 요소로 구성된다. 위젯(Widget)은 특정 타입의 컨텍스트 정보를 획득하는 것과 획득된 정보를 컨텍스트 정보가 실제로 어떻게 획득되었는지에 관계없이 어플리케이션에 범용적으로 사용 가능하도록 만드는 역할을 한다. 인터프리터(Interpreter)는 하나 혹은 그 이상의 컨텍스트 타입을 받아들이고 단일 단위의 컨텍스트를 생성한다. 어그리게이터(Aggregator)는 컨텍스트를 수집하고 단일 요소를 위한 모든 컨텍스트에 대한 책임을 갖고 있다.

Gu, T는 Pervasive computing 환경에서 컨텍스트 인지 서비스를 구축하기 위한 효과적인 기반구조를 제공하기 위한 서비스 지향적 컨텍스트 인지 미들웨어 (Service-Oriented Context-Aware Middleware, SOCAM)

를 설계하였다[11]. SOCAM은 컨텍스트를 인지하는 서비스를 구축하기 위한 아키텍처이다. SOCAM은 센서로부터 컨텍스트 정보를 획득하는 컨텍스트 감지 계층, 인지된 컨텍스트 정보를 해석/추론하는 컨텍스트 미들웨어 계층, 분석된 컨텍스트 결과에 따라 동적으로 적절한 서비스를 제공하는 컨텍스트 어플리케이션 계층으로 구성되어 있다. SOCAM은 독립된 서비스 컴포넌트로 동작하기 위한 컨텍스트 제공기(Context provider), 컨텍스트 해석기(Context interpreter), 컨텍스트 데이타 베이스(Context database), 컨텍스트 인지 서비스(Context-Aware Service), 서비스 발견 서비스(Service Discovery Service) 등의 다수의 컴포넌트로 구성되어 있다. 본 논문은 온톨로지와 규칙을 이용하여 사용자에게 필요한 서비스를 제공하지만, 다양한 컨텍스트 정보에 따라 변해야하는 서비스 기능이 다양해질수록 이러한 서비스를 설계하기 위한 보다 체계적인 기법이 필요하다.

서비스 지향적 시스템 개발에 관한 대표적인 프로젝트로는 AMIGO[11], AMEC[12], BelAmi[13]가 있다. 이러한 프로젝트 들은 각각 홈네트워크 환경에서 다양한 신기술의 통합을 위한 미들웨어 구현, Ambient Intelligence(AmI) 분야의 시스템 개발 방법, 이종 하드웨어와 소프트웨어 컴포넌트를 통합하기 위한 구조적 프레임워크 개발을 목표로 수행되었다.

3. 서비스 시스템의 특징

2장에서의 배경연구를 기반으로 본장에서는 서비스를 아래와 같이 정의한다.

"서비스는 인간의 삶을 보조함으로써 제공자와 수혜자를 위하여 가치를 창조하는 사용자 수준의 워크플로우이다."

서비스 시스템은 이러한 서비스를 제공하는 개체로서 소프트웨어와 하드웨어 및 이들간의 상호연동을 포함한다.

어플리케이션 시스템의 특정화된 형태로서, 서비스 시스템을 개발하기 위해 전통적인 소프트웨어 시스템 개발 기법을 적용 할 수 있다. 이는 서비스 시스템과 전통적인 소프트웨어 시스템이 요구사항에 기초하여 기능성을 제공하고 기능성은 다수의 품질 속성과 제약사항을 요구한다는 측면에 있어 유사하기 때문이다.

그러나 이러한 유사점과 더불어 서비스 시스템에 한정된 특징이 존재한다. 본 절에서 서비스 시스템(Service System, SS)과 경영 정보 시스템과 같은 전통적인 소프트웨어 시스템(Conventional software system, CSS)을 비교한다. 표 1은 시스템 개발과 시스템 운영의 측면에서의 차이점을 보여준다.

시스템 개발의 관점에서, 전통적인 소프트웨어 시스템과 서비스 시스템은 적용된 기술, 구현 단위, 대표적 아키텍처 스타일, 외부 인터페이스 측면에서 비교할 수 있다. CSS에서는 객체 지향 접근법, CBD, MDA이 주로 사용된다. 그리고 DBMS와 같은 데이타 관리를 위한 다양한 방법들은 소프트웨어 개발에 있어 중요한 역할을 수행한다. SS에서는 유비쿼터스 컴퓨팅, AmI, 컨텍스트 인지, 서비스 지향적 아키텍처와 같은 접근법들이

표 1 전통적인 소프트웨어 시스템과 서비스 시스템간의 비교

	CSS	SS
시스템 개발		
대표적 적용 기술	· OOA/D, CBD · 데이타베이스 관리 · 프로그래밍 언어	· 유비쿼터스 컴퓨팅 · AmI · 컨텍스트 인지 · 서비스 지향적 아키텍처
구현 단위	· 대부분 소프트웨어 모듈에 초점	· 하드웨어와 소프트웨어의 통합된 시스템에 초점
대표적 아키텍처 스타일	· 일반적으로 중앙집중화	· 일반적으로 분산됨
외부 인터페이스	· 사용자 인식 가능한 인터페이스 · 다른 시스템과의 인터페이스	· 사용자에게 보이지 않는 인터페이스 · 다른 서비스와의 인터페이스 · 컨텍스트와 인터페이스
시스템 운영		
서비스 시스템 관련자	· 사용자	· 사용자, 구독자, 발행자(제공자)
시스템 구동	· 사용자에 의한 직접 호출	· 사용자에 의한 직접 호출 · 장치에 의하여 유발
정보 획득	· 사용자 입력 · 시스템 내에서 생성	· 사용자 입력 · 시스템 내에서 생성 · 컨텍스트 감지
입력 출력 순환	· 열린 loop 순환	· 닫힌 loop 순환
컴포넌트 상호작용과 기능성의 역동성	· 상호작용과 기능성은 미리 정의되어짐	· 상호작용은 동적으로 재구성될 수 있음
지능	· 요구되지 않음	· 요구됨

주로 적용된다. CSS의 구현 단위는 소프트웨어 모듈이다. 반면에 SS는 소프트웨어 모듈뿐 아니라 하드웨어 장치와 네트워크 단위로 구성된다. 그러한 단위의 아키텍처를 위하여 CSS의 요소들은 일반적으로 데이타 베이스에 집중된다. 반면에 SS의 그것들은 분산되고 유동성이 있다. CSS의 인터페이스는 사용자 인터페이스를 포함하고 외부 인터페이스로서의 다른 시스템과의 인터페이스를 포함한다. 반면에 SS의 인터페이스는 컨텍스트와의 인터페이스를 더불어 포함한다.

시스템 운영의 측면에서 서비스 시스템 관련자, 시스템 구동, 정보 획득, 컴포넌트 상호작용의 역동성, 기능성이 비교 될 수 있다. CSS의 서비스 시스템 관련자들은 서비스 사용자를 포함한다. 반면에 SS는 서비스 사용자뿐만이 아니라 구독자, 발행자로 상세화 된다. CSS는 사용자의 직접적 호출로 구동되는 반면 SS는 사용자 호출과 더불어 구독 조건이 충족 되었을 때 자동적으로 구동된다. CSS에서의 정보는 사용자나 시스템 구동에 의해서 생성되지만 SS의 정보는 사용자와 시스템 구동과 더불어 컨텍스트 감지를 통해서도 획득 될 수 있다.

시스템의 입출력 사이클은 비록 CSS에 프로세스에서의 상호작용 데이타가 데이타베이스에 저장되어있을 지라도, CSS에서 출력이 이루어지면 출력은 더 이상 이용가치가 없다는 점에서 CSS에서의 사이클은 개방되어 있다. 반면에 SS는 서비스를 제공하기 위하여 사용자의 입력과 컨텍스트의 스냅샷을 받아들인다. 그리고 서비스 결과는 추후에 피드백이나 컨텍스트의 히스토리로 사용될 수 있다. 이러한 측면에서 SS의 사이클은 닫혀있다.

그림 1은 서비스 모델과 컨텍스트의 관계를 보여주고 있다. 서비스 시스템은 시스템 인터페이스를 통하여서 사용자와 상호작용하게 되고 외부 센서를 통하여 컨텍스트를 획득하고 획득된 컨텍스트는 해석규칙에 기반하여 분석된다. 컨텍스트 분석을 통하여서 현재의 상황을 시스템이 인지하게 되고 서비스활동 규칙을 통하여 서비스활동을 결정하게 된다. 서비스활동의 실행은 엑츄에이터를 통하여 수행되고 이러한 활동을 통하여 변경된 컨텍스트는 다시 센서를 통하여 인식되어 시스템에 반영되게 된다.

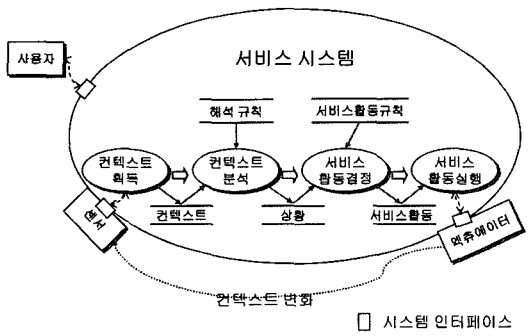


그림 1 서비스 모델과 컨텍스트

컴포넌트 간의 상호작용의 관점에서, CSS의 컴포넌트 상호작용은 설계 단계에서 결정된다. 반면에 SS에서의 컴포넌트 상호작용은 실행 시간의 컨텍스트에 따라서 동적으로 결정될 수 있다. 더불어 서비스 시스템은 획득된 컨텍스트 정보의 처리를 위하여 지능적 분석과 의사결정을 제공한다.

이러한 비교에 기초하여 서비스 시스템의 특징을 도출 할 수 있다.

- 서비스 시스템은 특정 분야 소프트웨어 컴포넌트, 컨텍스트 분석, 센서, 엑츄에이터와 같은 이종 컴포넌트들의 통합으로 전체 시스템이 구성된다.
- 서비스는 서비스 사용자에 따라 최적화 된다. 이것이 컴포넌트 상호작용이 동적으로 재구성 되는 이유이다. 그리고 더 중요한 점은, 서비스 시스템은 서비스 제공을 위해 컨텍스트 정보를 상당히 비중 있게 참조한다는 것이다.

4. 프로세스와 지침

이번 장에서는 서비스 시스템을 설계하기 위한 프로세스와 작업 지침을 제안한다. 서비스 시스템 개발의 핵심 단계는 그림 2에서와 같이 요구사항 획득 및 분석, 시스템 아키텍처 설계, 컴포넌트 설계, 컨텍스트 설계, 설계모델 검증의 다섯 가지 단계로 이루어진다. 본 논문에서는 시스템을 위한 요구사항이 존재한다고 가정한다.

서비스 시스템의 설계 단계에서, 단계 2에서 4까지는 각 단계에서의 산출물들이 다른 단계에 영향을 주면서 반복적으로 수행된다. 예를 들어 컨텍스트를 위한 시스

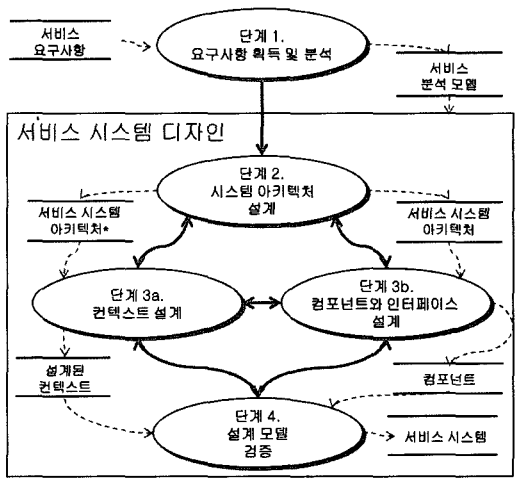


그림 2 서비스 모델링의 전체적인 프로세스

템 인터페이스는 시스템 아키텍처 설계 단계에서 개략적으로 설계된다. 그리고 그것은 컴포넌트 설계 단계에서 새로운 컴포넌트를 식별함으로써 정제된다. 새로운 컴포넌트는 시스템 아키텍처 설계에 적용되어야 한다.

4.1 단계 1. 요구사항 획득 및 분석

3장에서 살펴본 바와 같이 서비스시스템에서는 사용자, 구독자, 발행자와 같은 다양한 참가자들이 존재한다. 이러한 다양한 종류의 사용자들의 서비스 시스템에 대한 요구사항을 이들의 특성을 고려하여 요구사항을 수집하여야 한다.

요구사항 분석 활동은 사용자의 서비스 전체에 대한 사용자의 요구사항을 추출하여 서비스 요구사항을 명확하게 정의하며 요구사항들간에 충돌을 감지하고 해결한다. 이를 통해 서비스의 범위를 결정하고 서비스가 제공해야 할 기능성을 도출한다. 요구사항 분석 활동은 해당 서비스를 품질속성, 제약사항, 인터페이스, 기능적 요구사항, 텍스트 측면에서 요구되는 정보와 같은 품질 요구사항의 측면에서 분석하는 것이다.

첫째 활동은 서비스의 목표, 이익, 품질 속성, 기술적, 비 기술적 제약사항, 사회적 규율과 같은 서비스의 품질 요구사항을 식별하는 단계다. 서비스를 통해서 이루려는 목표, 서비스를 통해서 얻을수 있는 이익, 응답시간, 신뢰성, 성능, 정확도와 같은 품질 속성 등을 식별하는 단계이다.

둘째 활동은 해당서비스의 테스크들을 식별함으로써 해당 서비스의 기능성을 분석하는 활동이다. 3장에서 언급한 것과 같이 서비스 시스템은 시스템 인터페이스를 통하여서 사용자와 상호작용하게 되고 외부 센서를 통하여 컨텍스트를 획득하고 해석규칙에 기반하여 분석되며 서비스활동규칙에 의해 서비스활동이 결정되고 수행된다. 따라서, 서비스 시스템의 테스크는 컨텍스트 인식, 분석, 서비스활동결정, 서비스활동 수행으로 분류되어 분석 될 수 있으며 내부의 상세한 로직, 규칙은 서비스 도메인에 따라 다르게 분석된다. 테스크들의 일련의 순서들은 서비스가 수행되는 워크플로우로써 표현 된다. 서비스의 기능성을 분석하기 위하여 시나리오 기반 분석은 효과적으로 사용될 수 있고 UML의 유즈케이스 모델, 엑티비티 다이어그램들을 이용하여 모델을 표현할 수 있다. 예를 들어 응급상황 처리 서비스에서는 긴급상황 인지 분석, 긴급상황 레벨 판단, 통지, 엑츄에이터 작동, 긴급처치보조와 같은 서비스의 테스크의 구성을 식별하고 테스크 간의 워크플로우를 표현함으로써 서비스의 기능성을 분석 할 수 있다.

셋째 활동은 원시 속성과 분석된 상황을 식별함으로써 컨텍스트를 분석하는 활동이다. 원시 속성은사용되는 용도에 따라서 정의되어야 한다. 예를 들어 응급상황 처리 서비스에서 관찰 테스크는 반드시 환자의 혈압, 체온, 등의 컨텍스트를 참조해야 한다. 그리고 통지 테스크는 간호사, 의사, 가까운 병원의 위치를 사전에 인지하고 있어야 한다. 이러한 식별된 원시 속성들은 반드시 테스크들이 직접적으로 참조 할 수 있는 상황으로 해석되어야 한다. 해석은 해석 규칙에 의하여 수행 되어야 한다. 그림 3은 컨텍스트 분석의 일반화된 클래스 다이어그램이다.

컨텍스트 정보를 표현하기 위해 크게 원시 컨텍스트와 해석된 컨텍스트를 구분하고 이들 사이의 변환을 위해 해석 규칙을 정의한다. 원시 컨텍스트는 그 정보가 속성으로 표시되면 각 속성은 타입과 속성 명으로 표현된다. 해석된 컨텍스트는 해석된 상황에 대한 정보를 가지며, 이 해석된 상황은 원시 컨텍스트의 속성값을 기반으로 해석규칙을 수행하여 얻어진 이미 등록된 상황이다.

다이어그램에서 테스크A를 위해 필요한 원시 원시컨

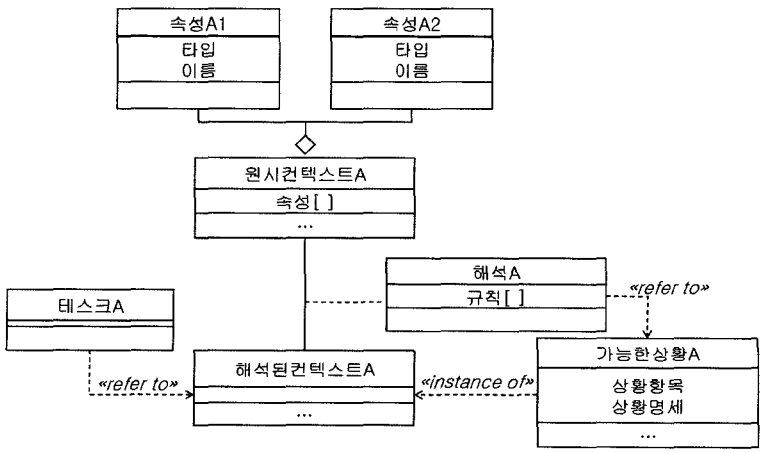


그림 3 컨텍스트 정보와 테스크

텍스트A, 해석규칙A, 가능한 상황A가 분석되었다. 그리고 해석된 컨텍스트A를 통해 표현된 실제 해석 결과는 사전에 정의된 가능한 상황A의 인스턴스로 표현되고 테스크A 클래스에 의해서 참조된다.

4.2 단계 2. 시스템 아키텍처 설계

전통적인 어플리케이션과 다르게 서비스 시스템은 이종의 장치들과 장치들간의 동적인 상호작용을 필요로 한다. 그러므로 이러한 컴포넌트들을 식별하고 통합하는 체계적인 기법이 필요하다.

첫째 활동은 아키텍처를 설계하는데 막대한 영향을 끼치는 요구사항인 아키텍처 드라이버를 선택하는 활동이다[15]. 서비스 시스템은 서비스 사용자 이동성, 서비스들 사이의 상호 작용성, 컨텍스트 정보의 적용성, 보안, 성과 등의 다수의 아키텍처 드라이버를 포함할 수 있다. 이러한 아키텍처 드라이버는 후에 아키텍처 스타일과 컴포넌트를 식별하는데 사용될 수 있다.

둘째 활동은 아키텍처 드라이버에 기반하여 적절한 아키텍처 스타일을 식별하는 활동이다. 스타일과 발생된 문제에 대하여 정해진 해결책을 제공함으로 효과적인 설계를 가능 하는 패턴을 이용하여 우리는 설계와 코드를 재 사용할 수 있다. 또한 시스템의 조직을 쉽게 이해할 수 있게 하고 해결 특징의 스타일 종속적인 분석에 대한 통찰성을 얻을 수 있다[16]. 이번 활동에서 드라이버의 기반이 되고 시스템 아키텍처를 구조화 할 수 있게 해주는 아키텍처 스타일을 이용한다.

예를 들어 그림 4에서는 하드웨어 장치 층, 인터페이스 층, 소프트웨어 논리 층, 데이타 베이스 층으로 구성된 MVC 모델과 결합된 계층화된 아키텍처 스타일을 보여주고 있다. 서비스시스템에서 컴포넌트들은 상호 이질적이고 그것들간의 상호작용은 복잡하다. 그러므로 그것들은 그들의 기능적인 측면에서 명백하게 구분되어야 할 필요가 있다. 이러한 계층구조를 적용함으로써 하드웨어 컴포넌트, 인터페이스 비즈니스 로직을 명백하게 정의 할 수 있다.

셋째 활동은 아키텍처 스타일을 구성하는 컴포넌트를 식별하는 활동이다. 서비스 시스템에 특징에 기인하여, 컨텍스트 인지, 분석, 의사 결정, 인터페이스, 도메인 종속적인 기능성을 갖고 있는 많은 이질적인 컴포넌트 타입이 있다. 컴포넌트는 그림 5에서 보여지는 바와 같이 구분될 수 있다.

컴포넌트는 의학 치료 장치와 같은 특정분야에 이용될 수 있는 컴포넌트와 RFID센서에 이용되는 컴포넌트와 같이 범용으로 이용될 수 있는 컴포넌트로 구분 될 수 있다. 다른 한편으로, 컴포넌트는 소프트웨어 모듈이나 하드웨어 장치들로 구현 될 수 있다. 소프트웨어 컴

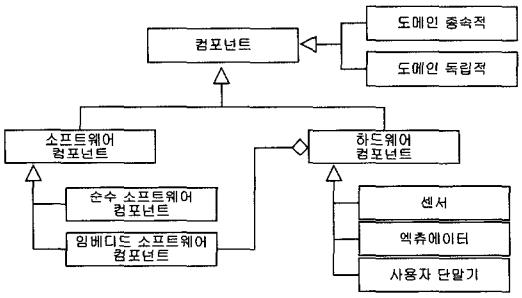


그림 5 서비스 시스템에서 컴포넌트 구분

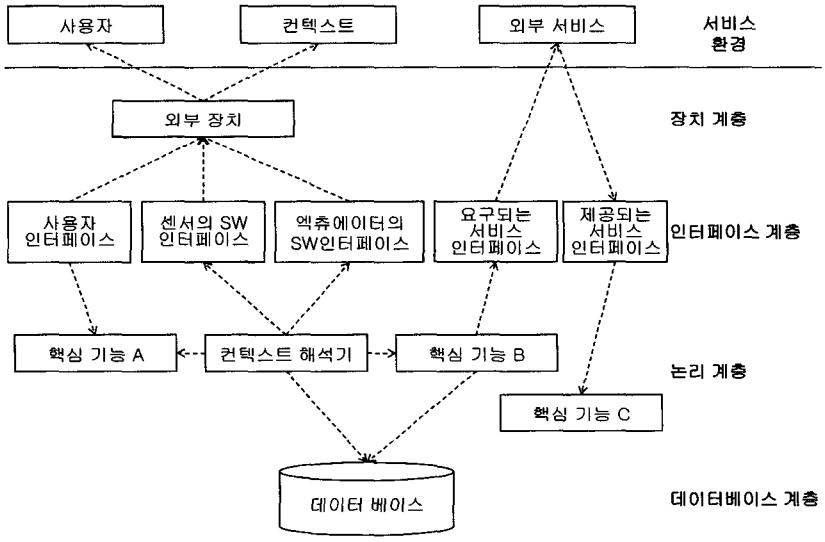


그림 4 서비스 시스템에서 컴포넌트 분류

포넌트는 서버나 데스크 탑에서 비즈니스 로직을 수행하기 위한 순수 소프트웨어 컴포넌트와 하드웨어 장치에 내장되어 있는 임베디드 소프트웨어 컴포넌트로 구분될 수 있다. 하드웨어 컴포넌트는 센서와 같은 컨텍스트 정보 획득 장치, 물리적 행동을 수행하는 액츄에이터와 같은 장치, 휴대폰이나 PDA같은 사용자 단말기로 구분 될 수 있다.

4.3 단계 3a. 컨텍스트 설계

인식된 컨텍스트 정보는 시스템은 개인화된 서비스를 제공한다는 측면에 있어서 서비스 시스템 개발에서의 가장 중요한 단계이다.

첫째 활동은 컨텍스트 정보를 요구하는 테스크를 구체화하는 활동이다.

둘째 활동은 원시 컨텍스트 클래스를 설계하는 활동이다. 이 활동에서, 타입, 이름, 값 그리고 속성의 근원 인터페이스들이 설계 수준에서 정의되고 결정된다.

셋째 활동은 감지된 컨텍스트 정보를 적절하게 해석하기 위한 해석 규칙, 가능한 상황, 해석된 컨텍스트 클래스들을 설계 하는 활동이다. 본 활동에서 설계된 컨텍스트는 단계1에서 분석된 컨텍스트 정보가 상세화된 것으로 그림 6과 같이 표현될 수 있다.

예를 들어 응급상황 처리 서비스의 경우 원시컨텍스트는 다양한 센서를 통하여 감지된 환자의 신원, 혈압, 체온, 맥박 등의 속성으로 구성된다. 원시컨텍스트는 체온, 맥박 등의 기준값으로 이루어진 해석규칙을 통하여 해석되고 시스템은 이를 통하여 가능한 응급상황을 해석하고 판단한다. 판단된 응급상황을 기준으로 컴포넌트는 응급상황에 적합한 활동(Action)을 수행한다.

규칙과 가능한 상황은 단계 1에서 분석된다. 그리고 분석 모델은 이번 단계의 설계 요소와 더불어 정제 될 수 있다. 예를 들어 해석된 컨텍스트A 클래스는 컨텍트 정보를 데이타 베이스에 저장하기 위한 오퍼레이션과 외부 컴포넌트에 컨텍스트 정보를 제공한다.

위와 같은 세 활동을 수행함으로써 표 2와 같은 명세가 제안된다.

표 2 Context 명세를 위한 템플릿

컨텍스트				
용도				
원시컨텍스트	이름	값	타입	인터페이스
해석 규칙	ID	규칙 명세		
해석된 컨텍스트	상황	명세		

4.4 단계 3b. 컴포넌트와 인터페이스 설계

시스템 아키텍처에서 컴포넌트의 기능성은 개략적으로 정의 될 수 있다. 컴포넌트 내부 설계 기능과 다른 컴포넌트와 상호 작용하기 위한 컴포넌트 인터페이스 설계는 이러한 활동을 통하여 정확하게 설계 될 수 있다.

첫째 활동은 컴포넌트에 입력데이타, 기능성, 출력데

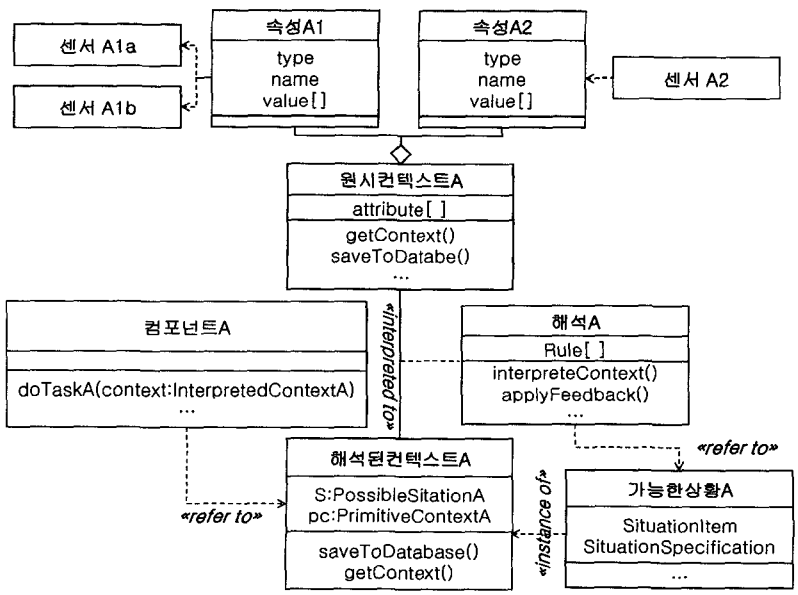
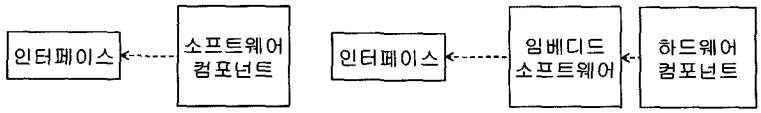


그림 6 테스크와 컨텍스트 사이의 관계 모델



1) 소프트웨어 컴포넌트를 위한 인터페이스

2) 하드웨어 컴포넌트내의 임베디드 소프트웨어를 위한 인터페이스

그림 7 소프트웨어 컴포넌트와 하드웨어 컴포넌트의 인터페이스

이타를 제공하는 컴포넌트 인터페이스를 정의 하는 활동이다. 그림 5에서와 같이 컴포넌트는 소프트웨어 컴포넌트와 하드웨어 컴포넌트로 분류 된다. 인터페이스의 경우 그림 7에서와 같이 컴포넌트의 인터페이스는 소프트웨어 컴포넌트를 위한 범용 소프트웨어 컴포넌트와 하드웨어 컴포넌트 내의 하드웨어 컴포넌트의 임베디드 소프트웨어 인터페이스로 세분화 된다. 임베디드 소프트웨어는 하드웨어 컴포넌트가 소프트웨어 컴포넌트와 메시지 전달을 할 수 있도록 하는 인터페이스이다. 예를 들어, 실내의 온도와 습도를 감지하는 센서(하드웨어컴포넌트)는 하드웨어적으로 온도와 습도를 감지하며 이를 리턴하는 임베디드 소프트웨어 API를 내장하고 있다. 그리고, 소프트웨어 컴포넌트와 임베디드 소프트웨어의 인터페이스는 메시지 전달을 하는 의존관계로 표현된다.

둘째 활동은 설계 클래스들과 컴포넌트가 구현된 기능성에 기초하여 그것들 사이의 관계를 식별하는 활동이다. 이 활동은 컴포넌트 기반의 개발에서의 디자인과 유사하기 때문에 상세한 기술은 생략한다.

4.5 단계 4. 설계모델 검증

컴포넌트와 컨텍스트의 상세하게 후, 설계모델은 검증과정을 거쳐야 한다. 설계모델 검증은 요구사항의 기능적 적합성을 검증함으로써 수행될 수 있다.

첫째로 설계모델의 검증은 사용자의 요구사항에서 명시된 기능적 요구사항이 설계모델에 어떠한 요소들로 나타나게 되는지를 검증해 봄으로써 수행될 수 있다. 4.1에서 언급된 것과 같이, 서비스 시스템에서 기능적 요구사항은 도출된 컨텍스트 인지기능, 컨텍스트 분석기능, 서비스 결정 기능, 서비스 실행 기능, 도메인 종속적 기능으로 구분할 수 있다. 도메인 종속적인 기능에 대한 검증은 서비스 종속적인 시스템 뿐만 아니라 전형적인 소프트웨어 개발과정에서 나타나므로 기존의 검증 기법을 참조 할 수 있다[17]. 따라서 본 절에서는 서비스 시스템에 종속적인 컨텍스트 인지, 컨텍스트 분석, 서비스 결정, 서비스 실행기능에 대한 검증기법을 제안한다. 그림 8은 각각의 기능적 요구사항이 설계모델의 어떠한 형태로 설계 되었는지를 보여준다.

컨텍스트 인지 기능은 Trace 1, Trace 2와 같이 서비스 시스템 아키텍처, 설계된 컨텍스트로 나타난다. 서비스 시스템의 아키텍처 구성요소들이 설계된 서비스 시스템 아키텍처에 누락되지 않고 적절하게 설계 되었는지 검증하고 인지된 컨텍스트가 설계된 컨텍스트에 표현되었는지 확인한다. 센서와 같은 컨텍스트 인지 장치는 서비스 시스템 아키텍처의 구성요소로 표현되고, 인지된 컨텍스트는 설계된 컨텍스트로 설계될 수 있다. 예를들어 응급상황 처리 시스템의 사례에서 컨텍스트 인지 기능의 기능적 요구사항은 그림 11에서와 같이 서비스 시스템 아키텍처에서 장치층의 *Sensor for Patient* 컴포넌트와 *Sensor for Individual Location* 컴포넌트로 설계된다. 인지된 응급환자의 컨텍스트는 컨텍스트 인지 기능의 기능적 요구사항의 또다른 형태의 설계 산

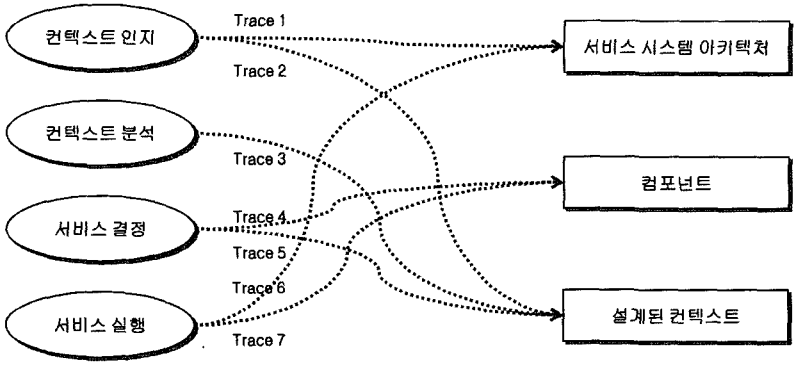


그림 8 기능적 요구사항과 설계산출물

출물로서 표 3의 컨텍스트 명세 형태로 설계 된다.

컨텍스트 분석 기능은 Trace 3과 같이 설계된 컨텍스트에 기반하여 현재 상황을 판단하는데 사용되고, 제공될 서비스에 필요한 컨텍스트는 시스템에 제공할 수 있는 기능성의 바탕이 된다. 요구사항에 기초한 컨텍스 분석을 통하여서 분석된 컨텍스트가 설계된 컨텍스트에 올바르게 설계되었는지 검증한다.

서비스 결정 기능은 Trace 4, Trace 5와 같이 컴포넌트와 설계된 컨텍스트의 설계 산출물로 나타난다. 서비스 시스템은 설계된 컨텍스트를 이용하여 어떠한 서비스를 제공해야 할지 결정하게 된다. 그러므로 제공 되어야 할 서비스를 결정하기 위한 적절한 컴포넌트가 도출되고 설계되어야 한다. 또한 제공되는 서비스는 설계된 컨텍스트 정보에 기반하여 서비스의 기능성을 제공하게 된다. 그러므로 요구사항에서 명시된 서비스의 기능성을 제공하기 위한 서비스 결정은 기능성단위의 컴포넌트로 설계되어야 하고 서비스 결정에 이용 되는 적절하게 설계된 컨텍스트가 산출 되었는지를 검증한다.

서비스 실행 기능은 Trace 6, Trace 7과 같이 서비스 시스템 아키텍처와 컴포넌트 설계 산출물로 나타난다. 서비스의 기능성을 제공하기 위한 엑츄에이터와 같은 장치는 시스템 아키택처의 구성요소로 도출되어야 하고 그것이 제공하는 기능성은 각각의 컴포넌트에 의하여 제공된다. 예를 들어 엑츄에이터와 같은 장치는 서비스 시스템 아키텍처의 구성요소로 표현되고 실행될 서비스는 컴포넌트의 구현을 통하여 서비스 시스템에서 제공해야 하는 기능을 구현한다.

기능적 요구사항을 만족시키기 위한 각각의 컴포넌트와 컨텍스트를 상세하게 설계 한 후에는 서비스 워크플로우의 유효성이 검증되어야한다. 워크플로우 검증은 워크플로우를 구성하는 테스크의 기능을 지원하는 컴포넌트와 인터페이스를 추적함으로써 수행될 수 있다. 예를 들어 다음과 같은 항목이 검증되어야 한다.

워크플로우 추적을 함으로서 아래의 예제와 같은 것들이 준수 될 수 있다.

- 적절한 장치들이 설계 되었는가?
- 메시지 전달 내에서 인터페이스들이 일치 하는가?
- 기능적 공백 없이 워크플로우가 자연스럽게 수행되는가?
- 예상치 못한 예외부분이 있는가?

그림 9에서와 같이 워크플로우는 UML의 시퀀스 다이어그램을 이용하여, 효과적으로 추적될 수 있다.

그림에서 두 워크플로우는 조건 제약 표현(Conditional Guards)을 사용하여 설계되었다. 각 메시지 전달에서 적절한 컴포넌트와 컴포넌트의 인터페이스 시그네이처들은 연결성을 고려하여야 한다. 추적은 활동1에서 분석된 시나리오에 기반하여 수행된다.

설계모델은 실제로 다양하게 존재 한다. 그러나 많은 설계 모델 중 최종적으로 완성된 설계 모델이 현재 주어진 요구사항이나 환경에 가장 적합하게 만들어 졌는지에 대한 검증이 중요하다.

5. 응급상황 처리 시스템 사례연구

4장에서 제안한 프로세스에 기반하여, LAS의 한 분야인 응급상황 처리 시스템(Emergency Handling System, EHS) 사례연구를 제안한다.

EHS는 특정위치에 거주하는 사람들의 건강에 문제가 있을 경우 등록된 보조자에게 사용자의 상태를 통보해 주거나 상황이 심각한 경우 인근 병원에 통보해 주고 응급 상황의 레벨에 따라 조치를 취한다. 혈압, 체온, 맥박 등을 통해 사용자의 건강상태를 항상 체크하고 사용자가 응급상황에 처했을 경우 응급상황을 인지하고 분

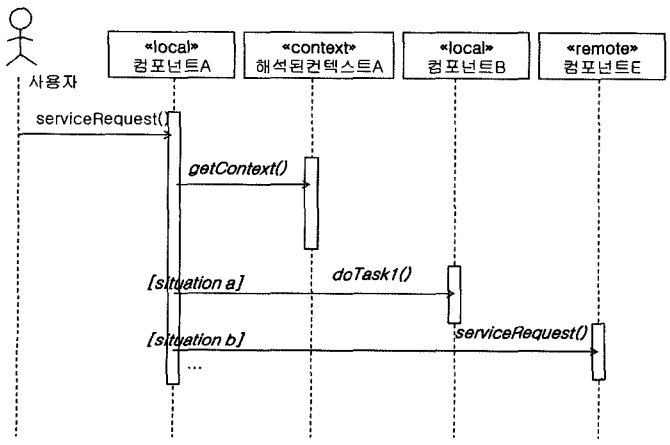


그림 9 컴포넌트간의 매크로 워크플로우

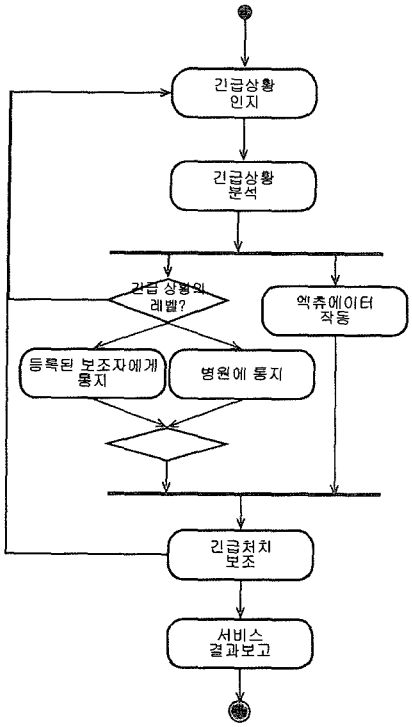


그림 10 EHS에서의 테스크와 매크로 워크플로우

석하여 응급상황의 레벨을 판단한다. 응급상황의 레벨에 따라 등록된 보조자에게 통지, 병원에 통지와 같은 조치를 취하고 상황이 심각하지 않은 경우 온도조절기, 조도조절기, 환풍기, 습도 조절기 등과 같은 엑츄에이터를 통해 사용자의 환경을 사용자의 현재 건강 상태에 따라 조절해 준다. 서비스 제공 후 상황에 따라 조치된 서비스의 결과를 보고한다.

시스템에는 일곱 개의 테스크들이 존재하고 테스크의 매크로 워크플로우는 그림 10에서와 같이 수행된다.

응급상황의 단계에 따라, 통지 활동은 병원 혹은 등록된 보조자에게 통지됨으로 이루어진다.

컨텍스트 명세를 위해 컨텍스트의 두 가지 용도가 분석된다. 표 3에서는 그것들 중 하나를 보여준다. CS-01은 응급상황 인지 테스크를 위해서 사용된다.

그림 11에서와 같이 시나리오는 4개의 층을 포함하고 13개의 컴포넌트를 갖는 시스템 아키텍처로 설계 된다.

아키텍처에 기반하여 컴포넌트, 인터페이스, 컨텍스트 명세, 매크로 워크플로우는 그림 12와 같이 나타난다. 메시지 3에서 환자의 현재 상태를 포함하고 있는 컨텍스트 CS-01을 참조하고, 메시지 5a에서는 의사나 간호사의 정보를 포함하고 있는 컨텍스트 CS-02가 참조된다.

6. 결 론

우리 사회의 새로운 중요한 어플리케이션의 형태로서, 서비스 시스템은 비즈니스모델, 경제효과, 생활 보조적 측면, 공학 개념의 완벽한 통합을 요구한다. 서비스 시스템은 단순히 기능성만을 제공하는데 그치지 않고 동적인 컨텍스트 인지, AmI, 의사 결정, 다양한 서비스 시스템 관련자들과의 상호작용을 제공한다. 그러나 전통적인 개발 접근법은 서비스 요구사항을 모델링하고 서비스 시스템을 설계하는데 충분한 구조를 제공하지 못한다.

본 논문에서 서비스 시스템의 명확한 특징을 표현하기 위해 서비스 시스템과 전통적인 소프트웨어 시스템을 비교하였다. 서비스 모델에서, 컨텍스트와 서비스 시스템은 원시 컨텍스트, 분석규칙, 해석된 컨텍스트로 구분되어 설계되었다. 그리고, 서비스 시스템을 설계하기 위해 아키텍처, 컴포넌트, 워크플로우의 측면에서 프로세스를 제안하였다. 제안된 프로세스에는 설계된 모델은 요구사항의 기능적 적합성 측면에서 검증하는 기법도

표 3 CS-01을 위한 컨텍스트 명세

<table>
<tr><td>Context ID</td><td colspan="4">CS-01</td><td></td></tr>
<tr><td>용도</td><td colspan="4">응급상황 인지 테스크가 호출되었을 경우</td><td></td></tr>
<tr><td rowspan="6">원시컨텍스트</td><td>이름</td><td>값</td><td>타입</td><td colspan="2">인터페이스</td></tr>
<tr><td>혈압</td><td rowspan="4">식별안됨</td><td>정수</td><td colspan="2" rowspan="4">환자 상태
감지 센서</td></tr>
<tr><td>체온</td><td>정수</td></tr>
<tr><td>맥박</td><td>정수</td></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td rowspan="4">해석규칙</td><td>ID</td><td colspan="3">명세</td><td></td></tr>
<tr><td>1</td><td colspan="3">If (BloodPresureA = 120 and BloodPresureB = 80) and
(36.6 <= 체온<= 37) and (60 <= 맥박<= 100)
Then normal;</td><td></td></tr>
<tr><td>2</td><td colspan="3">….</td><td></td></tr>
<tr><td>3</td><td colspan="3">….</td><td></td></tr>
<tr><td rowspan="2">해석된
컨텍스트</td><td>상황</td><td colspan="3">명세</td><td></td></tr>
<tr><td></td><td colspan="3">식별 안됨</td><td></td></tr>
</table>

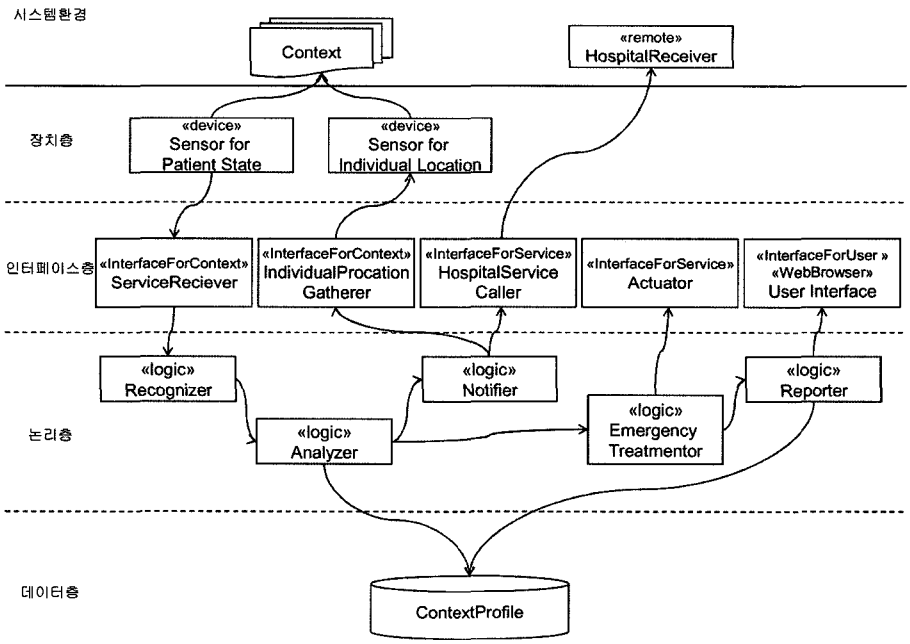


그림 11 EHS의 시스템 아키텍처

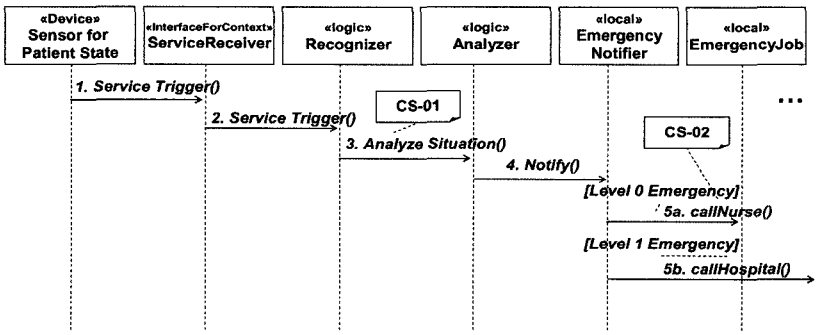


그림 12 EHS에서 컴포넌트와 컨텍스트의 부분 매크로 워크플로우

포함한다. 또한, EHS의 사례연구를 통해 구체적인 수행 예제를 제시하였다. 제시된 기법은 전통적인 소프트웨어 개발과 서비스 시스템의 개발의 구분이 명확하지 않은 가운데 새롭게 대두되고 있는 서비스시스템을 효과적으로 개발하는데 적용 될 수 있다. 전통적 소프트웨어 개발 기법을기반으로 서비스 시스템의 특징을 고려한 서비스 시스템 개발 기법을 제안함으로써 서비스 시스템 개발의 효율성 향상이 기대된다.

참 고 문 헌

[1] Chesbrough, H., and Spohrer, J., "A Research Manifesto for Services Science," *Communications on the ACM*, Vol. 49, No 7, pp. 35-40, ACM Press, 2006.

[2] Jürgen, N., et al., "Living Assistance Systems: An Ambient Intelligence Approach," *On the proceedings of the 28th international conference on Software engineering (ICSE '06)*, pp. 43-50, 2006.

[3] Hill, T., "On goods and services," *The Review of Income and Wealth*, Series 23, Blackwell Publishing, 1977.

[4] Services Sciences, Management and Engineering; www.research.ibm.com/ssme.

[5] Sheth, A., Verma, K., and Gomadam, K., "Semantics to Energize the Full Services Spectrum," *Communications on the ACM*, Vol. 49, No 7, pp. 55-61, ACM Press, 2006.

[6] The OWL Services Coalition, "OWL-S: Semantic Markup for Web Services,"http://www.daml.org/services.

[7] Dey, A. "Providing Architectural Support for

Building Context-Aware Applications," Ph.D. Thesis Dissertation, College of Computing, Georgia Tech, Dec. 2000.
[8] Brezillon, P., "Context Dynamic and Explanation in Contextual Graphs," *On the proceedings of Modeling· and Using Context (CONTEXT '03), LNAI 2680,* pp. 94-106, 2003.
[9] Edmonds, B., "Learning and exploiting context in agents," *On the proceedings of International Conference on Autonomous Agents (AAMAS '02),* pp. 1231-1238, 2002.
[10] Dey, A. and Abowd, G., "The Context Toolkit: Aiding the Development of Context-Aware Applications," *On the proceedings of Human Factors in Computing Systems (CHI'99),* pp. 434-441, 1999.
[11] Kalaoja, J., "The Vocabulary Ontology Engineering for the Semantic Modeling of Home Services," *On the proceedings of 8th International Conference on Enterprise Information Systems (ICEIS'06),* May, 2006.
[12] IETA, "Design direction workshop integrating technology, people and design results," June, 2005.
[13] Anastasopoulos, M., Bartelt, C., Koch, J., Niebuhr, D., and Rausch, A., "Towards a Reference Middleware Architecture for Ambient Intelligence Systems," *On the proceedings of the Workshop for Building Software for Pervasive Computing 20th Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA '05),* October, 2005.
[14] Gu, T., Pung, H. and Zhang, D., "A service-oriented middleware for building context-aware services," Journal of Network and Computer Application 2005.
[15] Thiel, S., and Hein, A., "Systematic Integration of Variability into Product Line Architecture Design," *On the proceeding of SPLC2, LNCS 2379,* Springer-Verlag Berlin Heidelberg, 2002.
[16] Garlan, D., et al., "Exploiting Style in Architectural Design Environments," *On the proceedings of 2nd Acm-Sigsoft Symposium on the Foundations of Software Engineering (SIGSOFT'94),* Foundations of Software Engineering, 1994.
[17] Roger, S., *Software Engineering A Practitioner's Approach,* McGraw Hill, 2005.

전 원 영
2005년 숭실대학교 컴퓨터학부 공학사
2006년~현재 숭실대학교 컴퓨터학과 석사과정. 관심분야는 객체지향 S/W 공학, 서비스지향 아키텍처(SOA)

장 수 호
2003년 숭실대학교 컴퓨터학부 공학사
2005년 숭실대학교 컴퓨터학과 공학석사
2005년~현재 숭실대학교 컴퓨터학과 박사과정. 관심분야는 제품계열 공학(PLE), 소프트웨어 아키텍처, 서비스지향 아키텍처(SOA)

김 수 동
정보과학회논문지 : 소프트웨어 및 응용 제 34 권 제 4 호 참조