상황인식 컴퓨팅 환경에서 동적 서비스 컴포지션 기술 동향 분석*

신동천** · 장효선***

Analysis of Trend in Dynamic Service Composition Technology for Context-Aware Computing Environments*

Dong-Cheon Shin** · Hyo-Sun Jang***

■ Abstract ■

A service composition has received much interest to provide users with the ad-hoc, seamless, and context-sensitive services dynamically composed from various networked services in context-aware computing environments. In this paper, with the purpose of guiding trends in the researches for dynamic service composition technologies, we investigate and analyze the major related works. For this, we separate semantic based composition from syntax based composition and then establish the comparison criteria by dividing the service composition processes into 3 steps : definition, construction, and execution.

Keyword : Service Composition, Context-aware Computing, Ubiquitous Computing, Service Provision

논문투고일 : 2010년 07월 07일 논문수정완료일 : 2010년 09월 01일 논문게재확정일 : 2010년 09월 11일
* 본 논문은 2009학년도 중앙대학교 연구장학기금 지원에 의한 것임.
** 중앙대학교 정보시스템학과 교수, 교신저자
*** 중앙대학교 일반대학원 정보시스템학과 석사과정

1. 서 론

유비쿼터스 컴퓨팅 환경 하에서 사용자들은 자신들을 둘러싼 수많은 컴퓨터들의 존재를 전혀 의식하지 않고 자신에게 제공되는 실제적인 서비스에만 관심을 가지게 될 것이다. 이를 위해서는 사람들이 현재 전기나 수도를 쓰는 것과 마찬가지로 자유롭고 쉽게 컴퓨터를 통해 필요한 서비스를 제공받을 수 있는 환경이 필요하게 될 것이고, 이러한 IT 서비스의 유틸리티화를 통해서 진정한 의미의 유비쿼터스 컴퓨팅 환경이 실현될 수 있을 것이다.

이러한 유비쿼터스 컴퓨팅 환경을 구축하기 위한 기반 기술로 제일 주목 받고 있는 것이 상황 인식(context-aware) 컴퓨팅 기술이다. 상황 인식 컴퓨팅 기술은 사용자의 현재 위치, 시간, 주변에 있는 다른 사람이나 정보가전 기기들, 사용자의 행동 및 작업 이력 등과 같은 사용자의 현재 상황 정보를 파악하고 분석하여 사용자가 현 상황에서 필요로 하는 서비스를 검색하여 구동시켜 주는 기술이다. 이러한 상황 정보는 센서 네트워크 내의 수많은 센서로부터 수집된 자료들을 분석하여 파악될 수 있다. 실제로 여러 분야에서 현재 이용되고 있는 위치기반 서비스도 상황 인식 컴퓨팅 기술 중의 한 분야이다. 이에 따라 다양한 센서로부터 얻은 정보들을 사용자의 목적에 맞게 동적으로 서비스를 구성하는 서비스 컴포지션 기술에 대한 관심도 높아지고 있다.

서비스 컴포지션은 다양한 단일서비스들의 조합을 통해 사용자의 요구사항을 만족시키는 것을 목표로 한다. 주로 웹 서비스 기반의 서비스 컴포지션에 대한 연구가 많이 진행되었으며 웹 서비스 컴포지션에 대한 대표적인 연구로는 SHOP2[23] 등이 있다. SHOP2는 OWL-S(Ontology Web Language-for Services) 기반 웹 서비스 컴포지션으로 HTN(Hierarchical Task Network)[29] 기반의 인공지능 플래닝 기술을 적용하여 OWL-S로 의미가 기술된 웹 서비스 컴포지션을 수행한다. 또 다른 연구로는 SWORD 프로젝트[30]가 있다. SWORD에서는 룰 기반 전문가 시스템이 자동적으로 프로세스 컴포지션을 결정하는데 사용되고, 이러한 컴포지션을 실현하기 위한 하나의 프로세스 계획을 반환한다.

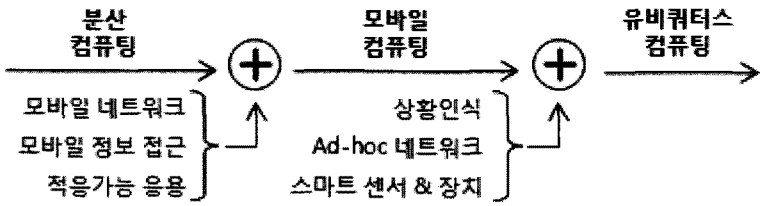
지금까지 연구된 대부분의 웹 서비스 컴포지션 시스템은 정적인 환경에서 이루어진다. 그러나 유비쿼터스 컴퓨팅 환경 하에서는 동적인 장치 및 환경 센서 등으로부터 얻을 수 있는 상황 데이터로부터 추출한 상황 지식과 사용자의 선호도 등을 이용하여 능동적으로 서비스를 구성하는 동적 서비스 컴포지션 기술에 대한 연구가 필요하다. 설계 시에 서비스 컴포지션이 결정되는 정적 서비스 컴포지션과 달리 동적 서비스 컴포지션은 서비스 컴포지션이 실행 시에 결정되는 것이 특징이다.

본 논문에서는 지금까지 연구된 동적 서비스 컴포지션을 비교 분석하고 앞으로의 연구 방향에 대해 고찰한다. 우선, 동적 서비스 컴포지션을 의미 기반(semantic based) 서비스 컴포지션과 구문 기반(syntactic based) 서비스 컴포지션으로 구분하였고 비교 분석을 위해 서비스 컴포지션 단계를 세 단계인 서비스 정의 단계, 구성 단계, 실행 단계로 나누어 비교 기준을 설정하였다.

본 논문의 구성은 다음과 같다. 제 2장에서는 상황인식 컴퓨팅의 주요 연구들을 간략히 소개하고, 제 3장에서는 동적 서비스 컴포지션을 의미 기반 서비스 컴포지션과 구문 기반 서비스 컴포지션으로 구분하여 소개한다. 제 4장에서는 비교 기준을 세우고, 동적 서비스 컴포지션의 연구들을 비교 분석한다. 마지막으로 제 5장에서 결론을 내린다.

2. 상황인식 컴퓨팅

유비쿼터스 컴퓨팅은 [그림 1]에서 볼 수 있듯이 분산 컴퓨팅과 모바일 컴퓨팅에서 진화된 개념으로 상황 의존성(context dependency)은 유비쿼터스 컴퓨팅 시스템의 최근 연구 영역에서 주요한 이슈이다[31]. 따라서 상황 인식에 대한 개념은 1990년대 이래 분산 시스템에 의해 중요성이 증가 되

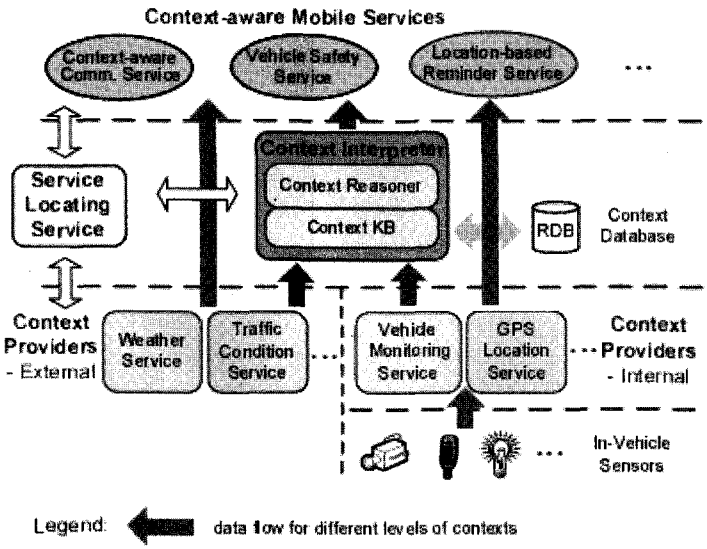


[그림 1] 유비쿼터스 컴퓨팅의 진화 단계[31]

었으며 끊임없이 변화하는 환경에서 모바일 단말 사용에 수반되었던 수 많은 문제점들에 대한 하나의 해결책으로 인식될 수 있다.

상황인식 컴퓨팅을 위해서는 몇 가지 주요 요소 기술이 있다. 상황정보를 얻기 위한 센싱 기술과 상황 정보를 가공하고 저장하며 공유하기 위한 상황인식 모델, 상위 상황정보를 유도하고 추론하는 상황인식 추론 기술이 필요하다. 마지막으로 사용자와 어플리케이션 사이에 중간 매개체 역할을 하는 상황정보 미들웨어 기술은 빠져서는 안 될 핵심 기술이다. [그림 2]는 상황인식 컴퓨팅을 위한 미들웨어 중 하나인 SOCAM[12]의 구조를 보여주고 있다. 최근에는 동적 센서로부터 얻은 정보들을 사용자의 목적에 맞게 동적으로 서비스를 구성하는 서비스 컴포지션 기술에 대한 관심도 높아지고 있다. 국내외 많은 기업과 학교에서는 다양한 상황인식 컴퓨팅 기술을 개발하고 적용하여 보다 지능적이고 효율적인 유비쿼터스 환경을 구축하려 노력하고 있다. 주요한 연구로는 CoBrA[5], Gaia[11], SOCAM[12], Scarlet-Context-aware infrastructure[8], CAMUS[14] 등이 있다.

CoBrA(Context Broker Architecture)[5]는 2004년까지 UMBC(University of Maryland, Baltimore Country)에서 개발한 지능형 공간상에 존재하는 모든 컴퓨팅 개체들을 위한 상황 정보 공유모델을 관리하는 상황 브로커(context broker) 에이전트 개발을 목표로 한 연구이다. 브로커는 서로 다른 정보 소스에서의 상황 정보 획득을 도와주고, 상황 정보 모델을 관리하고 유지하며, 에이전트들 간의 지식 공유를 도와준다. 이러한 브로커는 상황 및



[그림 2] SOCAM의 구조[12]

지식 저장 관리 모듈, 상황 추론 엔진, 상황 획득 모듈, 그리고 보안 관리 모듈로 구성된다.

Gaia[11]는 University Of Illinois에서 개발된 상황 인식 서비스 구조로 응용이 다양한 상황 정보를 얻고 추론할 수 있게 해준다. Gaia 시스템은 존재하는 자원들을 요청하고 이용할 목적과 동시에 상황 정보에 접근하고 사용할 목적으로 서비스들을 외부로 노출시킨다. 또한 Gaia 시스템은 사용자 중심적이며, 자원 인식과 여러 장치들의 사용이 가능하며, 상황 정보를 효과적으로 사용할 수 있는 모바일 응용들을 개발하기 위한 프레임워크도 제공하고 있다. Gaia 시스템의 아키텍처는 크게 Gaia Kernal과 Application Framework로 구성된다.

SOCAM(Service-Oriented Context-Aware Middleware)[12]은 Singapore 국립대학에서 주관하여 개발한 미들웨어로 상황 인식 서비스와 시스템 개발을 용이하기 위해 제안하였다. 또한 미들웨어 내에서 상황 정보 모델링을 위한 OWL(Web Ontology Language)를 사용하였다. SOCAM은 상황정보 제공자, 상황정보 번역기, 상황정보 데이터베이스, 상황인식 서비스, 서비스 위치 서비스(SLS) 컴포넌트로 구성된다.

Scarlet-Context-aware infrastructure[8]는 Illinois 기술 연구소에서 주관하여 이질적 플랫폼 간의 상황 정보 교환을 위해서 개발되었다. Scarlet는 상황인식 컴퓨팅 환경의 모든 측면을 다루고 있지 않고 단지 응용에 상황 정보를 어떻게 제공할 것인가에 대한 문제에만 역점을 두고 있다. 특히 이질적인 플랫폼 간에 상황 정보 전송을 위한 연구였다. Scarlet은 SOAP와 WSDL을 이용하여 플랫폼 간의 호환성을 유지하였다.

CAMUS(Context-Aware Middleware for URC Systems)[14]는 ETRI에서 개발하고 있는 URC (Ubiquitous Robotics Companion) 환경 내에서 획득된 상황 정보를 기반으로 환경 내에 있는 사용자에게 적절한 서비스를 제공할 수 있도록 상황 기반 응용의 개발과 실행을 지원하는 상황 인식 미들웨어이다. CAMUS에서는 온톨로지 기반의 계층적 상황 정보 모델을 사용하고 있으며, 이를 지원할 수 있는 상황 정보 관리 모듈을 제공하고 있다.

3. 동적 서비스 컴포지션

서비스 컴포지션을 분류하는 방법에는 여러 가지가 있다[9]. 그 중에서도 정적(static)과 동적(dynamic)으로 구분하는 게 가장 일반적이다. 설계 시에 서비스 컴포지션이 결정되는 것은 정적, 실행 시에 결정되는 것은 동적이다. 이전에는 정적 서비스 컴포지션이 대부분이었지만, 유비쿼터스 컴퓨팅 환경 하에서는 동적인 장치 및 환경 센서 등으로부터 얻을 수 있는 상황 데이터로부터 추출한 상황 지식과 사용자의 선호도 등을 이용하여 능동적으로 서비스를 구성하는 동적 서비스 컴포지션 기술에 대한 관심이 높아지면서 동적 서비스 컴포지션의 연구가 많이 이루어지고 있다.

서비스 선택 및 워크플로우 관리가 사전에 수동으로 이루어지는 정적 서비스 컴포지션에서는 주로 비즈니스 워크플로우 관리 이론을 기반으로 하여 BPEL(Business Process Execution Language)를 사용한다. BPEL은 WSDL에 정의된 서비스 모델에 기반을 두어 비즈니스 프로세스를 기술하는 웹 서비스 기반 표준 모델링 언어이다. 이 BPEL을 사용하는 기법들은 에러 처리나 트랜잭션 관리와 같은 비즈니스 환경에서 요구되는 실질적인 전체 기능을 지원한다[1]. BPEL의 대표적인 언어로는 그래프 중심으로 프로세스를 표현하는 WSFL (Web Services Flow Language)[16]과 구조적 중심으로 프로세스를 표현하는 XLang[33]를 통합한 BPEL4WS(Business Process Execution Language for Web Services)가 있다. BPEL4WS는 웹으로 비즈니스 프로세스를 정의하고 실행하기 위한 웹 서비스 표준이다. 일반적으로 BPEL4WS는 작업 흐름으로 표현되고 흐름은 액티비티(activity)로 나타낸다.

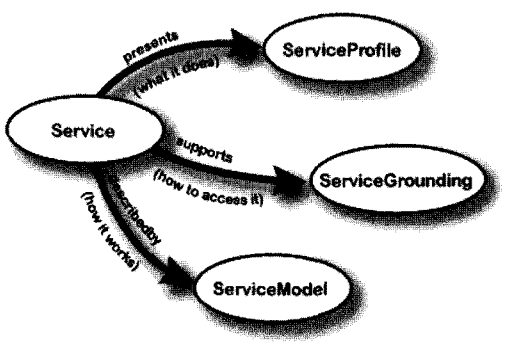
서비스 컴포지션을 분류하는 또 다른 방법으로는 의미 기반과 구문 기반으로 분류하는 방법이 있

다[34]. 의미 기반 서비스 컴포지션은 사용자의 요청에 대응하는 논리적 워크플로우를 만들기 위한 서비스들에 대해 의미 정보를 사용하는 것이다. 반면 구문 기반 서비스 컴포지션은 이미 존재하는 워크플로우를 추정하고 사용자가 미리 정의해놓은 제약조건들을 실행한다.

그 밖에 서비스 컴포지션의 분류는 서비스 컴포지션의 방법, 즉 기술로서 분류하는 방법이 있다. Rao는 웹 서비스 컴포지션을 워크플로우 기술을 사용하는 웹 서비스 컴포지션과 AI 플래닝 기술을 사용하는 웹 서비스 컴포지션으로 분류하여 조사하였다[28]. 또한, Dustdar는 모델 기반 서비스 컴포지션, 비즈니스 룰 기반 서비스 컴포지션, 설명문 컴포지션으로 구분하였다. 그 밖에도 자동화(automated) 서비스 컴포지션과 수동화(manual) 서비스 컴포지션으로 구분하였다[9].

3.1 의미 기반 서비스 컴포지션

의미 기반 서비스 컴포지션은 각 서비스의 의미들을 기술하는데 주로 OWL-S(Ontology Web Language-for Services)[21]와 같은 언어들을 사용한다. OWL-S는 OWL을 기반으로 웹 서비스를 기술하기 위해 작성된 웹 서비스의 의미를 정의하는 상위 온톨로지이다. OWL-S는 [그림 3]과 같이 웹 서비스를 정의하는 서비스 온톨로지, 서비스가 무엇을 하는지(what it does)를 정의하는 서비스



[그림 3] 서비스 온톨로지의 최상위 레벨[21]

프로파일 온톨로지, 서비스가 어떻게 동작하는지를 기술하는(how it works) 서비스 모델 온톨로지, 마지막으로 서비스의 접근 방법을 기술하는(how to access it) 서비스 그라운딩 온톨로지로 구성된다. 하나의 서비스는 서비스 클래스의 인스턴스로 정의되고, 속성으로 위의 세 개의 온톨로지의 인스턴스를 연결한다. 속성 presents는 서비스 프로파일, 속성 supports는 서비스 그라운딩, 속성 describedby는 서비스 모델과 연결된다. 서비스 모델은 서비스를 실행할 때 필요한 입력(Input), 서비스를 실행한 후에 얻게 되는 출력(Outputs), 서비스를 실행하기 위해 만족되어야 하는 전제조건(Preconditions), 그리고 서비스를 실행한 후에 발생하는 효과(Effects)의 IOPE를 사용하여 서비스를 정의한다.

의미 기반 서비스 컴포지션에 대한 연구가 많이 이루어지고 있다. [4]는 재수립 기술을 통해서만 컨텍스트의 변화에 적응하는 것을 해결하기 위해 확장된 AI 기반의 플래닝 기법을 제안하였다. 이 연구에서는 서비스를 명세하는 방법으로 FSM(Finite State Machines) 적응 행동을 제안하고, 서비스를 조합하기 위하여 FSM을 사용한다. 이러한 FSM들은 재수립 기술 없이 새로운 컨텍스트 변화에 적응하게 된다.

[26]은 서비스를 명세하기 위하여 OWL-S를 확장한 언어인 OWL-SC(Extending OWL-S with Context Elements)를 사용하는 온톨로지 기반의 프레임워크를 제안하였다. 또한, 글로벌 플래닝과 로컬 최적화를 위한 하이브리드(hybrid) 방법을 제안하였다.

[15]는 컨텍스트 정보들로 인한 동적의 복잡한 서비스들을 검색하고 조합할 수 있는 서비스 프로비저닝 미들웨어 시스템을 위한 프레임워크를 제안하였다. 또한 서비스 컴포지션을 위해 서비스 히스토리 정보와 데이터마이닝 기능을 가지고 있는 온톨로지 엔진을 이용한다. 온톨로지 엔진은 등록된 서비스 컴포지션들을 사용하고 사용자 쿼리들을 위한 동적 서비스 컴포지션을 제공하고, 데이터마이

닝 엔진은 서비스 히스토리 정보를 이용하여 새로운 서비스들을 조합할 수 있게 해준다.

[37]은 태스크를 수행하기 위하여 목표 지향의 서비스 컴포지션을 제안했다. 또한 사용자의 목표를 성취하기 위해 목표 지향 서비스 컴포지션 기반인 태스크 지향 의미 표현 모델을 구축하였다. 서비스 표현을 위해서는 태스크 기반 의미 표현 모델을 이용하고, 태스크 표현을 위해 TDL(Task Definition Language)를 사용한다. 또한 서비스 모델링 언어로는 OWL-S, 컴포지션 언어로는 BPEL 4WS를 사용한다.

[25]는 홈 지역 네트워크에 초점을 맞춘 서비스 컴포지션을 나타내고, 서비스 검색과 관련된 컴포지션 프로세스에서 제 3자인 서비스 제공자(service provider)를 포함한 서비스 컴포지션 모델을 제안한다. 제안된 모델이 태스크 컴퓨팅과 다른 점은 서비스 제공자를 포함한다는 것과 OSGI 기반의 서비스 그라운딩(grounding)을 사용한다는 점이다. 또한, 새롭게 만들어진 복합 서비스들은 UPnP[35]와 Jini[32]를 사용한다.

[13]은 PICO(Pervasive Information Communities Organization) 미들웨어 프레임워크를 사용하는 서비스 지향 환경을 구축할 수 있는 모델을 나타낸다. 서비스 컴포지션 메커니즘 모델은 서비스를 그래프로서 작성하고, 그래프 저장소들을 유지보수하며 동적으로 기본 서비스들이 결합할 수 있게 해준다.

[18]은 상황인식 서비스 컴포지션을 위해 상황인식 서비스 발견(discovery) 단계와 상황 인식 프로세스 통합(integration) 단계를 수행한다. 첫 번째 단계인 상황인식 서비스 검색에서는 컴포지션을 위한 후보 서비스들의 집합들을 제공하고, 두 번째 단계인 상황인식 프로세스 통합 단계에서 상황 요구사항들과 태스크들에 대응하는 컴포지션 계획들을 제공한다. 상황인식 프로세스 통합 단계에서는 미리 정의해놓은 Finite State Automata 모델을 사용한다.

[19]는 COCOA(Conversation-based service Composition in pervasivecomputing environments with QoS support)를 제안한다. COCOA는 QoS를 지원하기 위해 OWL-S 기반의 언어인 CO COA-L를 사용한다. 또한, 서비스 매칭을 통한 QoS 기반의 서비스 디스커버리인 COCOA-SD와 선택된 서비스들의 통합을 위한 COCOA-CI를 제안한다.

[27]은 목표 명세서(goal specification)에 기반하고 STRIPS 기반 플래닝 기술을 이용하는 유비쿼터스 컴퓨팅 환경에서의 새로운 패러다임을 제시한다. 어플리케이션 개발자들 뿐만 아니라 사용자들도 목표들을 어떻게 성취하는지를 결정하는 플래닝과 태스크들을 기술할 수 있다. 이 시스템은 사용자가 최소한의 개입으로 태스크를 수행할 수 있게 해주고, 어플리케이션 개발자들이 오류 복구나 다양한 종류의 태스크들을 수행하는 것을 간소화해준다.

[10]은 의미 기반 동적 서비스 컴포지션을 위하여 의미 인지(semantics-aware) 컴포넌트 모델인 CoSMoS(Component Service Model With Semantics)와 미들웨어인 CoRE(Component Runtime Environment), 컴포지션 방법인 SeGSeC(Semantic Graph-Based Service Composition)을 제안한다.

3.2 구문 기반 서비스 컴포지션

구문 기반 서비스 컴포지션은 각 서비스의 의미들을 기술하는데 주로 WSDL(Web Services Description language)[6]과 같은 언어를 사용한다. WSDL은 웹 서비스가 제공하는 서비스에 대한 정보를 기술하기 위한 XML 기반의 언어이다. 즉, 원하는 서비스가 어디(Where)에 존재하며, 웹 서비스로 무엇(What)을 할 수 있고, 또 이를 실행하기 위해서는 어떻게(How) 해야 하는가를 제공한다. 이러한 것들을 제공하기 위해서 WSML은 데이터 타입, 메시지, 포트 타입, 바인딩 등이 XML 형식으로 기술되어 있다. 하지만 WSML은 해당 웹 서비스의 문법적인 규격에 대한 정보만 있을 뿐, 그 웹 서비스가 갖는 의미는 표현하지 않는다.

구문 기반 서비스 컴포지션에 대한 몇 가지 연구가 있다. [2]는 서비스 지향 컴포넌트 기반의 오픈 아키텍처를 제안한다. 제안하는 아키텍처는 상황인식 서비스 컴포지션을 따르고, 자율 매니저를 통해 홈 어플리케이션 문제를 해결한다.

[17]은 컨텍스트 정보를 추론하고 추론된 정보들로 서비스들을 조합하고 선택하는 템플릿 기반의 메커니즘을 제안하였다. 이 메커니즘의 주요한 세 가지는 태스크 기반의 컨텍스트 추론, 템플릿 기반의 태스크 실행과 패턴 기반의 서비스 컴포지션이다. 또한, 이러한 세 가지를 만족하기 위하여 태스크 기반의 아키텍처를 개발하였다.

[7]은 ECF(Executable Choreography Framework)를 제안한다. ECF는 XML 기반의 언어를 재정의한 ECL(Executable Choreography Language)를 사용하고, 세 개의 다른 기능들로 구현된다. 이 기능들은 message interception, transparent activity context propagation, dynamic rule deployment 이다.

[36]은 상황정보를 이용한 웹 서비스 컴포지션을 제안한다. 제안된 웹 서비스 컴포지션은 SHOP2 플래닝 시스템과 XML 기반의 컴포지션 언어인 BPEL4WS 을 사용한다.

[20]은 소프트웨어 에이전트의 개념과 컨텍스트를 사용하는 웹 서비스 컴포지션을 제안한다. 에이전트들은 서비스들을 조합하는 것과 관련된 composite-service-agents와 웹 서비스들과 관련 있는 master-service-agents와 웹 서비스 인스턴스들과 관련 있는 service-agents 로 이루어진다. 또한, 컨텍스트는 I-context, W-context, C-context의 타입으로 구성된다.

4. 비교 분석

서비스 컴포지션의 특징을 알아보고 비교한 몇 가지의 연구들이 있다. [9]에서는 웹 서비스 컴포지션의 특징을 알아보기 위해 웹 서비스 컴포지션의 특징을 7개의 기준에 맞추어 비교하였다. 컴포지션 전략, 실행 모니터(execution monitor), 동적 대화 선택(dynamic conversation selection), QoS 모델링, WSDL 언어 실행, 트랜잭션 지원(transaction support), 마지막으로 그래프 지원(graph support)이다.

[34]에서는 앰비언트 시스템들을 위한 동적 서비스 컴포지션의 특징을 비교하기 위해 비교 기준을 크게 4가지로 분류하였다. 언어 표현(Language expressiveness), 컴포지션 모델(composition model), 실행 모델(execution model), 그리고 실행 환경(execution environment)이다. 또한 각각의 분류들은 몇 개의 특징들로 이루어져 있다. 언어 표현은 Service modeling language, Semantic-aware, User task description method, Published services description method로 이루어져 있다. 컴포지션 모델은 Composition creation technique, Composition language, Context-aware, QoS-aware, User intervention으로 구성되고, 실행 모델은 Orchestration/Choreography, Replanning during the execution, Dynamic/Static service binding로 구성된다. 마지막으로 실행 환경은 Infrastructure, Service discovery method, Device that supports the execution으로 이루어져 있다.

[3]에서는 유비쿼터스 환경에서 서비스 컴포지션 메커니즘을 비교하기 위하여 비교 기준을 크게 컴포지션 사양(composition specification), 런타임(runtime) 그리고 전개(deployment)로 분류하였다. 또한 컴포지션 사양은 Specified by, Specified at, Specified in, Level, QoS로 이루어져 있다. 런타임은 Resource use, scalability, Infrastructure로 이루어져있고, 전개는 Topology, Evaluation으로 이루어져있다.

한편, 서비스 컴포지션 단계와 관련하여 [24]에서는 정의(definition), 스케줄링(scheduling), 구성(construction), 실행(execution), 진화(evolution)로 5단계로 나타냈고, [37]에서는 서비스 컴포지션의 단계를 정의, 구성, 실행 3단계로 나타내었다. 스케줄링 단계는 서비스가 언제 어떻게 실행될지 정의하

고 준비하는 단계로써 정의 단계와 통합될 수 있다. 또한 진화 단계는 서비스 컴포지션 이후 유지보수 단계로써 생략이 가능하다.

기존 연구에서 서비스 컴포지션을 위한 필수적인 단계는 정의, 구성, 실행 3단계이다. 정의 단계는 사용자의 업무나 목표를 정의하고, 서비스를 모델링 하는 단계로써 이 단계에서는 업무를 정의하는 방법과 서비스 모델링 언어가 필요하다. 구성 단계는 서비스 컴포지션을 정의하고 구축하는 단계로써 컴포지션 기술과 컴포지션 언어가 필요하다. 마지막 단계인 실행 단계는 서비스 컴포지션이 실행되는 단계를 나타낸다. 따라서 본 논문에서는 상황인식 컴퓨팅 환경에서 동적 서비스 컴포지션 연구들의 비교를 위하여 컴포지션 단계를 정의, 구성, 그리고 실행의 3단계로 한다. 그리고 각 단계의 수행과 관련된 주요 사항들을 세부기준으로 한다(<표 1> 참조).

세부기준은 기존 연구[34]의 세부기준을 따르도록 하는데, [34]에서는 상황인식 컴퓨팅 환경에서 동적 서비스를 분석하는데 적합하지 않은 비교기준들이 있다. 본 논문에서는 동적 서비스 컴포지션을 의미기반 서비스 컴포지션과 구문기반 서비스 컴포지션으로 분류하였기 때문에 세부기준인 Semantic-aware와 Dynamic/Static service binding은 불필요하며, 상황인식 기반이기 때문에 Context-aware도 불필요하다. 또한 Published services description method는 서비스들이 어떻게 묘사되는지를 나타낸 기준으로서, 동적 서비스 컴포지션의 경우 대부분 단일 서비스로 묘사되기 때문에 세부기준으로는 고려하지 않는다. User intervention의 경우 컴포지션이 이루어지는 동안 사용자가 참여 여부를 나타내는 지표로써 동적 서비스 컴포지션을 분석하는 데는 불필요하기 때문에 고려하지 않는다. 또한, 동적 서비스 컴포지션들은 대부분 중앙 프로세스에 의하여 실행되기 때문에 Orchestration/Choreography는 고려하지 않았고, 서비스 컴포지션에 초점을 맞추었기 때문에 서비스 검색 기술을 나타내는 기준인 Service discovery method도 고려하지 않는다. 마지막으로 동적 서비스 컴포지션 연구의 대부분이 동적 서비스 컴포지션이 실행되는 장치에 관해서는 언급하지 않았기 때문에 Device that supports the execution 또한 고려하지 않는다. 따라서 본 논문에서는 서비스 정의 단계에서는 사용자의 업무나 목표를 정의하는 방법을 나타내는 업무, 서비스 모델링 언어인 모델링 언어가 비교 기준이 된다. 구성 단계에서는 기술, 컴포지션 언어, QoS를 세부 기준으로 삼았다. 기술은 요구되는 목적(object)을 만족하기 위해 사용되는 컴포지션 기술을 나타내고, 컴포지션 언어는 서비스 컴포지션을 기술하고 실행하는데 사용되는 언어를 나타낸다. QoS는 QoS를 지원하는지를 나타낸다. 마지막 단계인 실행 단계에서는 세부 비교 기준으로 컴포지션이 실행되는 동안에 컨텍스트의 변화 등이 일어났을 경우 재수립 기술을 지원하느냐를 나타내는 재수립과 서비스 컴포지션이 일어나는 기반 구조를 나타내는 기반

〈표 1〉 동적 서비스 컴포지션 비교 기준

컴포지션 단계	세부 기준	설명
서비스 정의	업무	사용자의 업무나 목표를 정의하는 방법
	모델링 언어	서비스의 속성들을 기술하기 위한 언어
서비스 구성	기술	사용되는 컴포지션 기술
	컴포지션 언어	서비스 컴포지션을 기술하고 실행하는데 사용되는 언어
	QoS	QoS 지원 여부
실행	재수립	컨텍스트의 변화에 따르는 재수립 기술 지원 여부
	기반구조	서비스 컴포지션이 이루어지는 기반 구조

구조로 하였다.

<표 2>는 유비쿼터스 컴퓨팅 환경에서의 동적 서비스 컴포지션 연구들을 비교 기준에 따라 비교한 결과를 정리한 표이다. 의미 기반 서비스 컴포지션은 주로 OWL-S을 기본으로 서비스 모델링을 하는 것을 볼 수 있다. [29, 37]의 경우 OWL-S을 사용하여 사용자의 업무를 Goal로 정의하였다. [27]도 사용자의 업무를 Goal로 정의하였는데, [37]과는 다르게 의미 기반 웹의 표준 언어 중 하나인 DAML+OIL[22]을 사용하였다. OWL-S를 사용한 다른 연구인[4]는 서비스 컴포지션을 위해 사용자의 업무를 FSM의 조합으로 나타냈고, [10]은 의미 그래프로 나타내었다. 또한 [18, 19, 25]의 경우 사용자의 업무를 워크플로우로 나타내었고, [18, 25]는 OWL-S를 사용하며 [19]는 OWL-S 기반의 언어인 COCOA-L을 사용하였다. [26]은 OWL-S의 확

〈표 2〉 동적 서비스 컴포지션 비교

구 분	서비스 컴포지션	서비스 정의		서비스 구성			실행	
		업무	모델링언어	기술	컴포지션언어	QoS	재수립	기반구조
의미 기반	[37](2006)	Goal	OWL-S	goal-driven	BPEL4WS	Yes	No	SOAP
	[13](2007)	Simple graph	-	graph-theory -based	-	Yes	No	PICO middleware framework
	[27](2004)	Goal	DAML+OIL	AI Planning -STRIPS	Lisp	No	Yes	Gaia
	[26](2006)	Simple Service	OWL-SC	AI Planning -HTN	directed acyclic graph (DAG)	Yes	No	-
	[18](2006)	Workflow	OWL-S	AI Planning- Finite State Machines	OWL-S	Yes	No	-
	[25](2006)	Workflow	OWL-S	Workflow	OWL-S	No	No	OSGI
	[15](2006)	Simple Service	-	Data-Mining	-	No	No	Service Provisioning Middleware
	[19](2007)	Workflow	COCOA-L (OWL-S)	AI Planning- Finite State Machines	COCOA-L (OWL-S)	Yes	No	SOAP
	[10](2005)	Semantic graph	OWL-S, WSMO	Semantic Graph-based	OWL-S, WSMO	No	No	-
	[29](2004)	Goal	OWL-S	SHOP2 Planning	OWL-S	No	No	SOAP
	[4](2004)	Composite FSM	OWL-S	AI Planning - Finite State Machines	OWL-S	No	Yes	-
구문 기반	[7](2005)	Workflow	-	ECF	ECL	No	No	ECF
	[17](2007)	Template	-	Pattern-based	ACME	No	No	-
	[36](2004)	Goal	-	SHOP2 planning	BPEL4WS	No	No	-
	[2](2007)	-	-	Workflow	-	No	No	OSGI, iPOJO
	[20](2005)	Service Chart	WSDL	Agent Conversation based	State Chart Diagrams	Yes	No	SOAP

장 언어인 OWL-SC를 사용하여 사용자의 업무를 단일 서비스로 나타낸다. 반면 특정한 서비스 모델링 언어를 나타내지 않은 경우도 있는데, [13]은 그래프로, [15]는 단일 서비스로 사용자의 업무를 정의하였다. 구문 기반 서비스 컴포지션의 경우 서비스 모델링 언어로 특정한 언어를 언급하지는 않았지만, 주로 WSDL을 사용한다. [20]은 WSDL을 사용하면서 Service Chart로 사용자의 업무를 정의하였다. [17]의 경우 사용자의 업무를 Template 형태로 정의하고, [2, 7]은 워크플로우로 정의했다. 또한, [36]은 Goal로 정의한다.

서비스 구성 단계에서도 컴포지션 언어로 OWL-S를 쓰는 몇몇의 연구가 있다[4, 10, 18, 25, 29]. 이 연구들의 경우 서비스 모델링 언어로도 OWL-S를 사용하였고, 모두 의미 기반 서비스 컴포지션이다. 또한, [4, 18]의 경우는 서비스 컴포지션을 위해 AI 기반의 플래닝 기법인 FSM을 사용했으며, [18]의 경우 QoS 지원을 한다. [10]의 경우 의미 그래프 기반의 서비스 컴포지션 기술을 사용한다. OWL-S를 사용하는 또 다른 연구인 [25]는 서비스 모델링 기법과 같이 워크플로우 기법을 컴포지션 단계에서도 사용하며, [29]는 SHOP2 플래닝 기법을 컴포지션 기술로 사용한다. OWL-S의 확장된 형태인 COCOA-L을 쓰는 [19]의 경우 FSM 기법을 사용하며 QoS를 지원한다. 의미 기반 컴포지션 중 컴포지션 언어로 OWL-S 다음으로 많이 쓰이는 것은 XML 기반의 컴포지션 언어인 BPEL4WS이다. BPEL4WS를 사용하는 연구의 경우 [37]이 있는데, 이는 서비스 컴포지션 기술로 goal-driven 기술을 제안하고, QoS를 지원한다. BPEL4WS를 사용하는 다른 연구로는 [36]이 있는데, 이 연구는 구문 기반 서비스 컴포지션으로써 SHOP2 플래닝 기법을 서비스 컴포지션 기술로 사용하였다. 또 다른 언어로 Lisp를 사용하는 의미 기반 컴포지션인 [27]의 경우 서비스 컴포지션 기술로 STRIPS 기반의 플래닝 기법을 사용한다. [26]의 경우 DAG (Directed Acyclic Graph)로 서비스 컴포지션을 나타내고, 플래닝 기법과 하이브리드 방법을 제안했으며 QoS를 지원한다. 의미 기반 서비스 컴포지션에서 서비스 컴포지션 언어를 특별히 지정하지 않는 연구가 있는데, [13]의 연구에서는 특정한 서비스 컴포지션 언어를 지정하지 않되 서비스 컴포지션을 위해 graph-theory 기반인 기술을 제안하고 QoS를 지원한다. 또한, [15]의 연구에서도 특정한 서비스 컴포지션 언어를 지정하지 않았고, 서비스 컴포지션을 위하여 데이터마이닝 기법을 사용했다. 구문 기반 컴포지션인 [17]은 ACME로써 서비스 컴포지션을 기술하며 패턴기반의 서비스 컴포지션 기술을 제안했다. [7]은 XML 기반의 언어를 재정의한 ECL(Executable Choreography Language)를 사용하며 서비스 컴포지션 기술로 ECF(Executable Choreography Framework)를 제안한다. [20]은 서비스 컴포지션을 State Chart Diagrams을 사용하여 정의하며 서비스 컴포지션 기술로 Agent Converation 기술을 제안했고, QoS를 지원한다. 특별한 컴포지션 언어를 정의하지 않은 구문 기반 서비스 컴포지션으로는 [2]가 있는데, 이 경우 서비스 컴포지션 기술로 워크플로우 기법을 사용했다.

실행 단계에서의 분류 기준인 재수립을 지원하는 서비스 컴포지션은[27]과 [4] 두 가지로 나타났다. 기반구조의 경우 SOAP과 OSGI 기반의 기반구조를 가장 많이 사용했는데, [19, 20, 29, 37]의 연구에서는 SOAP을 사용했다. 또한, OSGI 을 사용한 연구로는 [2, 25]가 있고, [2]의 연구에서는 OSGI와 iPOJO를 함께 사용했다. 반면 [13]의 연구에서는 PICO 미들웨어 프레임워크를 사용하고, [27]에서는 Gaia를 사용한다. [15]의 연구에서는 Service Provisioning Middleware를 제안했고, [7]은 ECF를 제안했다. 그 밖의 연구에서는 특정한 기반구조를 지정하지 않았다.

서비스 컴포지션에는 여러 가지가 있다. Goal-driven은 사용자 업무나 목표를 goal로써 정의하고, 동적으로 사용자의 목표를 성취하는데 목적을 두고 있다. 이 기법을 이용하면 사용자와 유비쿼터스 컴퓨팅 환경간의 상호작용을 간단히 해주고 서비스 협동과 컴포지션의 유연성을 향상시켜 준

다. 하지만, 트랜잭션이 고려되지 않았다는 점과 데이터 변환의 문제점이 있다. 인공지능 플래닝 기법의 경우 STRIPS, HTN, FSM 등으로 나뉘게 된다. STRIPS는 비계층적인 계획 기법으로서 STRIPS 플래너가 목표인 goal을 만족하기 위하여 현 상태와 목표 상태의 차이를 줄이면서 진행한다. STRIPS 플래닝은 최소한의 조정으로 사용자의 업무를 수행하고, 어플리케이션 개발자들의 업무를 간단화 시키는데 도움을 준다. 하지만 비계층적이기 때문에 세부적인 계획을 많이 가지는 경우 많은 경비가 소요되고 goal들 간의 충돌을 관리하기가 어렵다는 단점이 있다. 계층적 계획인 HTN 플래닝은 태스크를 단위 태스크까지 나누어 가는 태스크 분해 방법이다. HTN은 태스크 기반의 플래닝으로 플래닝 과정의 모니터링이 쉽고 목표 태스크까지의 도달성이 높다는 장점을 가지고 있다. FSM 은 특정한 상태를 정의하기 위한 개념적 모델로서 여러 개의 제한된 상태(State)가 존재하고, 그 존재들이 특정 조건에 물려 서로 연결되어 있는 형태로 Finite State Automata라고도 부른다. FSM은 특정한 조건에 따라 그 조건으로 상태가 전환이 되면서 해당되는 상태를 처리하게 되는데, 표로써 나타내기 때문에 개념 적립이 쉽고, 새로운 상태의 추가 삭제가 용이하다는 장점이 있다. 또한, 인공지능 플래닝 기법인 HTN 플래닝 기술을 적용한 서비스 컴포지션의 기술로 SHOP2 플래닝 기법이 있다. SHOP2 플래닝 기법은 HTN 기반의 인공지능 플래닝 기술을 적용하여 태스크를 구현하기 위해 하나의 독립된 플래너를 사용하며, HTN 플래닝의 장점을 가지고 좀 더 수월하게 HTN 플래닝 기법을 이용한 플래닝을 좀 더 수월하게 해준다. Workflow 는 작업 절차를 통한 정보 또는 업무의 이동을 의미하며, 작업 흐름이라고 할 수 있다. Workflow 기법은 비즈니스 분야에서 성공적으로 활용되고 있으며, 서비스 컴포지션에서도 BPEL4WS나 OWL-S 등의 언어를 사용하여 다양한 형태로 구현될 수 있다.

의미 기반 서비스 컴포지션에서 가장 많이 쓰이는 언어인 OWL-S은 온톨로지를 이용하여 자동화된 서비스 검색이나 컴포지션을 가능하게 해준다. 즉, OWL-S는 온톨로지로 웹 서비스를 기술하여 자동으로 서비스를 발견하고, 컴포지션하고, 실행하는 것을 지원하도록 디자인 되어 있다. 따라서 OWL-S는 의미 기반 서비스 컴포지션에 유용하고 많이 쓰이지만, 서비스 검색이나 컴포지션을 통해 가져온 서비스의 검증하는 수단을 제공하지 않는다는 점 등의 단점이 있다. 이런 OWL-S를 기반으로 한 언어로는 COCOA-L이 있다. COCOA-L은 대화로써 업무를 모델링하고, QoS 속성을 가지고 있다는 특징이 있다. 주로 구문 기반 서비스 컴포지션에서 쓰이는 BPEL4W는 XML 기반 명세언어로 비즈니스 플로우를 정적으로 정의하고 요구되어지는 서비스 컴포지션을 수행한다. BPEL4WS는 그래프 중심으로 프로세스를 표현하는 WSFL과 구조적 중심으로 프로세스를 표현하는 XLang의 장점을 통합했지만, 의미 개념은 지원하지 않는다는 단점이 있다. 또 다른 컴포지션 언어인 DAG는 방향성 비순환 그래프로서 프로세스를 모델링하는데 사용한다. DAG는 액션들의 입력, 출력, 실행의 집합을 표현하고 후보 액션들을 찾는다. 또한, 노드들 간의 다른 경로들을 만들어 서비스를 컴포지션 한다. DAG는 공간을 효율적으로 사용한다는 장점이 있다. 하지만, 노드수가 증가할수록 복잡해지고 수행시간이 길어진다는 단점이 있다. 또 다른 언어인 State Chart Diagram은 상태에 따른 흐름을 나타내주며, 서비스 컴포지션을 명세하기 위하여 사용된다. 또한, State Chart Diagram은 조합된 서비스들의 Service Chart Diagram의 집합으로 이루어져 있다.

지금까지 각 비교 기준에 맞추어 동적 서비스 컴포지션을 비교해보았다. 비교결과를 정리하자면 먼저, 의미 기반 서비스 컴포지션에서 서비스 모델링 언어로 가장 많이 쓰이는 것은 OWL-S인 것을 볼 수 있고, OWL-S를 기반으로 하는 OWL-SC, COCOA-L이 쓰이기도 한다. 반면 구문 기반 서비스 컴포지션은 특정한 서비스 모델링 언어를

언급하지는 않은 연구가 대부분이다. 또한, 의미 기반 서비스 컴포지션에서 서비스 컴포지션 언어로 가장 많이 쓰이는 언어도 OWL-S이며 다음으로는 많이 쓰이는 서비스 컴포지션 언어는 BPEL 4WS이다. 서비스 컴포지션 기술은 AI 플래닝 기법이 가장 많이 쓰이는 것을 볼 수 있다. 이 AI 플래닝 기법은 STRIPS나 HTN과 여러 가지 형태가 쓰이는 것을 볼 수 있다. 다음으로 QoS를 지원하는 서비스 컴포지션을 보면, 대부분 의미 기반 서비스 컴포지션인 것을 알 수 있다. 마지막으로 최근 서비스 컴포지션 연구에서는 re-planning 기능을 지원하지 않는 경우가 많다.

의미 기반 서비스 모델링 언어로 웹 서비스 모델링 언어인 OWL-S를 많이 사용하는 것은 OWL-S가 온톨로지를 이용하여 서비스의 의미를 표현하는데 용이하기 때문으로 보인다. 하지만 서비스 검색이나 컴포지션을 통해 가져온 서비스의 검증하는 수단을 제공하지 않는다는 점 등의 문제점이 있다. 따라서 다양한 개발 환경에 맞추어 서비스 모델링 언어를 선택하거나 다양한 특성을 반영하고, 유연한 서비스 모델링 언어의 연구가 필요하다.

서비스 컴포지션 기술의 경우 매우 다양한 것을 볼 수 있다. 서비스 컴포지션 기술은 위에서 설명하였듯이 각각의 장단점이 있으므로 특정한 서비스 컴포지션 기술이 좋다고는 할 수 없다. 개발하려는 환경에 따라 서비스 컴포지션 기술을 선택하고, 특성에 맞는 기술을 개발하여야 한다.

QoS는 유비쿼터스 컴퓨팅 환경에서 사용자의 만족도를 높이는데 중요한 역할을 한다. 따라서 서비스 컴포지션이 수행되는 동안 기능적 요구사항뿐만 아니라 비기능적 요구사항들인 latency, confidentiality, integrity 등도 고려해야 한다[34]. 동적 서비스 컴포지션 비교 결과 QoS를 지원하는 서비스 컴포지션은 의미 기반 서비스 컴포지션이 대부분인 것을 볼 수 있다. 이는 의미 기반 서비스 컴포지션이 구문 기반 서비스 컴포지션보다 QoS를 지원하는데 더 적합하다고 평가할 수 있다.

기반구조의 경우 특정한 프레임워크를 지정하지 않거나 특히 많이 쓰이는 프레임워크는 없는 것으로 나타났다. 이는 유비쿼터스 컴퓨팅 환경에서 쓰는 장치들이 다양하기 때문으로 보인다. 따라서 동적 서비스 컴포지션의 기반구조는 개발하려는 환경에 맞는 프레임워크를 선택할 필요가 있다.

유비쿼터스 컴퓨팅 환경에서는 사용자의 환경에 맞게 컨텍스트 정보가 변하기 때문에 사용자에게 끊임없는 서비스를 지원할 수 있는 기능이 필요하다. 이를 위해서는 서비스 컴포지션이 실행되는 동안에도 컨텍스트 정보가 바뀌었을 경우 재수립을 통해 새로운 서비스를 컴포지션 하여 사용자에게 지원할 수 있어야 한다. 하지만, 비교결과에서 볼 수 있듯이 최근 연구에서는 이러한 재수립 기능이 빠져 있다. 앞으로는 이러한 재수립 기능을 지원할 수 있는 연구가 필요하다.

5. 결 론

본 논문에서는 유비쿼터스 환경에서 동적 서비스 컴포지션에 대한 비교하고 분석하였다. 분석 결과 의미 기반 서비스 컴포지션에서 가장 많이 쓰이는 모델링 언어는 OWL-S로 나타났고, 서비스 컴포지션 언어로 가장 많이 쓰이는 언어도 OWL-S로 나타났다. 서비스 컴포지션 기술로는 AI 플래닝 기법이 가장 많이 쓰이고, 다음으로는 Finite State Automata가 많이 쓰이는 것을 볼 수 있었다. 또한 QoS는 의미 기반 서비스 컴포지션에서 주로 지원하고, 마지막으로 최근 서비스 컴포지션 연구에서는 재수립 기능을 지원하지 않는 경우가 많은 것으로 분석되었다.

본 논문에서 보인 결과 분석으로 다음과 같은 결론을 얻을 수 있다. 첫째, 유비쿼터스 컴퓨팅 환경은 매우 다양하기 때문에 개발하려는 환경에 맞추어 서비스 모델링 언어를 선택하거나 다양한 특성을 반영하고, 유연한 서비스 모델링 언어의 연구가 필요하다. 둘째, 서비스 컴포지션 기술은 매우 다양하고 각각의 장단점이 있으므로 특정한 서

비스 컴포지션 기술이 좋다고는 할 수 없기 때문에 개발하려는 환경에 따라 서비스 컴포지션 기술을 선택하고, 특성에 맞는 기술을 개발하여야 한다. 셋째, 구문 기반 서비스 컴포지션보다 의미 기반 서비스 컴포지션이 QoS를 지원하는 데 더 적합하다고 보인다. 넷째, 동적 서비스 컴포지션의 기반구조는 개발하려는 환경에 맞는 프레임워크를 선택할 필요가 있다. 마지막으로 최근 연구에서는 재수립 기능이 빠져 있지만, 앞으로는 이러한 재수립 기능을 지원할 수 있는 연구가 필요하다.

참 고 문 헌

[1] 이용주, "시맨틱과 워크플로우 혼합기법에 의한 자동화된 웹 서비스 조합시스템", 「정보처리학회논문지」, 제14-D권, 제2호, 통권 제112호(2007), pp.265-272.

[2] Bottaro, A., J. Bourcier, C. Esco_er, and P. Lalanda, "Autonomic context-aware service composition", Proceedings of IEEE International Conference on Pervasive Services, 2007.

[3] Bronsted, J., K. M. Hansen, and M. Ingstrup, "A Survey of Service Composition Mechanisms in Ubiquitous Computing", Second Workshop on Requirements and Solutions for Pervasive Software Infrastructures at Ubicomp, 2007.

[4] Carey, K., D. Lewis, S. Higel, and V. Wade, "Adaptive composite service plans for ubiquitous computing", 2nd International Workshop on Management Ubiquitous Communications and Services, 2004.

[5] Chen, H., "An Intelligent Broker Architecture for Context-Aware Systems", A PhD. Dissertation Proposal in Computer Science, 2003.

[6] Christensen, E., F. Curbera, G. Meredith, and S. Weerawarana, "Web Services Description Language(WSDL) 1.1", W3C Note, 2001.

[7] Cottenier, T. and T. Elrad, "Adaptive embedded services for pervasive computing", Workshop on Building Software for Pervasive Computing-ACM SIGPLAN conf. on Object-Oriented Programming, Systems, Languages, and Applications, 2005.

[8] Daftari, A., N. Mehta, S. Bakre, and X. H. Sun, "On Design Framework of Context Aware Embedded Systems", Monterey Workshop on Software Engineering for Embedded Systems, 2003.

[9] Dustdar, S. and W. Schreiner, "A survey on web services composition", *International Journal of Web and Grid Services*, Vol.1, No.1 (2005), pp.1-30.

[10] Fujii, K. and T. Suda, "Semantics-Based Dynamic Service Composition", *IEEE Journal on selected areas in communications*, Vol.23, No.12(2005), pp.2361-2372.

[11] Gaia project, http://gaia.cs.uiuc.edu/.

[12] Gu, T., X. H. Wang, H. K. Pung. and D. Q. Zhang, "An Ontology-based Context Model in Intelligent Environments", Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference, 2004.

[13] Kalasapur, S., M. Kumar, and B. A. Shirazi, "Dynamic Service Composition in Pervasive Computing", *IEEE Transactions on Parallel and Distributed Systems*, Vol.18, No.7 (2007), pp.907-918.

[14] Kim, H., Y. J. Cho, and S. R. Oh, "CAMUS -A Middleware Supporting Context-aware Services for Network-based Robots", IEEE

Workshop on Advanced Robotics and Its Social Impacts, 2005.

[15] Lee, S. Y., J. Y. Lee, and B. I. Lee, "Service composition techniques using data mining for ubiquitous computing environments", *IJ CSNS International Journal of Computer Science and Network Security*, Vol.6, No.9 B(2006), pp.110-117.

[16] Leymann, P. and D. Engineer, "Web services flow language (WSFL 1.0)", http://www-4.ibm.com/software/solutions/webservices/pdf/WSFL.pdf, 2001.

[17] Molina, A. J., H. M. Koo, and I. Y. Ko, "A template-based mechanism for dynamic service composition based on context prediction in ubicomp applications", Proceedings of the First International Workshop on Intelligent Web Based Tools, 2007.

[18] Mokhtar, S., B., D. Fournier, N. Georgantas, and V. Issarny, "Context-aware service composition in pervasive computing environments", *Lecture notes in computer science*, Vol.3943(2006), pp.129-144.

[19] Mokhtar, S. B., N. Georgantas, and V. Issarny. "Cocoa : Conversation-based service composition in pervasive computing environments", Proceedings of the IEEE International Conference on Pervasive Services, 2006.

[20] Maamar, Z., S. K. Mostefaoui, and H. Yahyaoui, "Toward an agent-based and context-oriented approach for web services composition", *IEEE Transactions on Knowledge and Data Engineering*, Vol.17, No.5(2005), pp.686-697.

[21] Martin, D., M. Burstein, J. Hobbs, O.Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. R. Payne, E. Sirin, N. Srinivasan, and K. Sycara, "OWL-S : Semantic markup for web services", W3C Member Submission, 2004.

[22] Mcguinness, D. L., R. Fikes, J. Hendler and L. A. Stein, "DAML+OIL : An Ontology Language for the Semantic Web", *IEEE intelligent systems*, Vol.17, No.5(2002), pp.72-80.

[23] Nau, D., T. Au, O. Ilghami, U. Kuter, J. Murdock, D. Wu and F. Yaman, "SHOP2 : An HTN planning system", *Journal of Artificial Intelligence Research*, Vol.20, No.1 (2003), pp.379-404.

[24] Orriens, B., J. Yang, and M. P. Papazoglou, "A Framework for Business Rule Driven Service Composition", Proceedings of the Fourth International Workshop on Conceptual, 2003.

[25] Pourreza, H. and P. Graham. "On the fly service composition for local interaction environments", Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, 2006.

[26] Qiu, L., Z. Shi, and F. Lin, "Context optimization of ai planning for services composition", Proceedings of the IEEE International Conference on e-Business Engineering, 2006.

[27] Ranganathan, A. and R. H. Campbell, "Autonomic pervasive computing based on planning", International Conference on Autonomic Computing, 2004.

[28] Rao, J. and X. Su, "A Survey of Automated Web Service Composition Methods", Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition, (2004), pp.43-54.

[29] Sirin, E., B. Parsia, D. Wu, J. Hendler, and D. Nau, "HTN planning for Web Service composition using SHOP2", *Web Semantics : Science, Services and Agents on the World Wide Web*, Vol.1, No.4(2004), pp. 377-407.

[30] Shankar, P. and A. Fox, "SWORD : A developer toolkit for web service composition", Proceedings of the Eleventh International World Wide Web Conference, 2002.

[31] Strang, T. and C. Linnhoff-Popien, "A Context Modeling Survey", Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004-The Sixth International Conference on Ubiquitous Computing, 2004.

[32] Sun Microsystems, "Jini Technology core Platform Specification", http://java.sun.com/developer/products/jini/index.jsp, 2003.

[33] Thatte, S., "XLang : Web services for business process design", Microsoft Corporation, 2001.

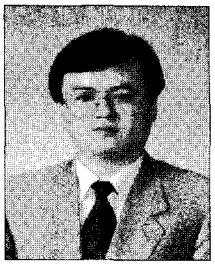
[34] Urbieta, A., G. Barrutieta, J. Parra, and A. Uribarren, "A survey of dynamic service composition approaches for ambient systems", Proceedings of the 2008 Ambi-Sys workshop on Software Organisation and MonIToring of Ambient Systems, 2008.

[35] UPnP Forum, "UPnP Device Architecture 1.0", http://www.upnp.org, 2003.

[36] Vukovic, M. and P. Robinson, "Adaptive, planning based, web service composition for context awareness", *2nd International Conference on Pervasive Computing*, Vol.178 (2004), pp.247-252.

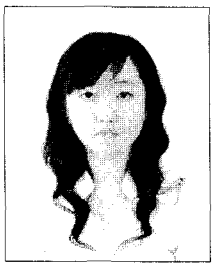
[37] Zhang, K., Q. Li, and Q. Sui, "A goal-driven approach of service composition for pervasive computing", 1st International Symposium on Pervasive Computing and Applications, 2006.

◈ 저 자 소 개 ◈



신 동 천 (dcshin@cau.ac.kr)

서울대학교 컴퓨터공학과를 졸업한 후 한국과학기술원 전산학과에서 석사와 박사학위를 각각 취득하였다. 현재 중앙대학교 정보시스템학과에서 교수로 재직 중이다. 최근의 주요 관심분야는 모바일 데이터베이스, IT 서비스 모델링, 기업정보시스템 등이다.



장 효 선 (yuz@hanmail.net)

중앙대학교 정보시스템학과를 졸업하고, 현재 중앙대학교 정보시스템학과 석사과정에 재학 중이다. 주요 관심분야는 데이터베이스, 서비스, 온톨로지 등이다.