# 헬스케어 서비스에서 동적인 데이터 전달을 위한 데이터 결합기 설계 및 구현

## (Design and Implementation of Data Binder for Dynamic Data Delivery in Healthcare Service)

강 규 창 † 이 전 우 †† 최 훈 †††
(Kyuchang Kang) (Jeunwoo Lee) (Hoon Choi)

**요 약** 본 논문은 서로 다른 벤더의 응용 프로그램과 생체 신호 측정 장치가 동적으로 데이터 전달을 할 수 있는 생산자(Producer)/소비자(Consumer) 패턴 기반의 데이터 결합기 구조를 제안한 것으로 휴대용 정보 단말에서 컴포넌트 기반의 프로그래밍이 가능하고 서비스 지향적인 동작 메커니즘을 제공하는 OSGi 플랫폼의 번들로서 구현된다. 데이터 결합기는 정적으로 데이터 생산자와 소비자를 연결하는 OSGi WireAdmin 서비스의 단점을 보완한 것으로, 데이터를 사용하는 주체인 응용 프로그램의 요구 사항을 응용 프로그램 설명자(Application Descriptor)로 형식화하고 데이터를 생산하는 주체인 생체 신호 측정 장치의 기능을 장치 설명자(Device Descriptor)로 형식화하여 런타임에 데이터 생산자-소비자 쌍을 만들어 동적으로 데이터가 연결되는 기능을 제공한다. 따라서 센서 기반 응용을 개발할 때 데이터 생산자와 데이터 소비자를 사이에서 빈번하게 일어나는 연결 관리를 동적으로 해주는 기능 구현에 활용 가능하다. 본 논문의 목적은 생체 신호 측정 장치와 같은 데이터 생산자와 헬스케어 응용 프로그램과 같은 데이터 소비자를 분리시켜 헬스케어 서비스 개발의 편이성을 제공하기 위한 것이다.

**키워드** : 헬스케어, 데이터 전달 방법, 생산자-소비자 패턴, 데이터 결합기

***Abstract*** This paper suggests producer/consumer-based Data Binder enabling applications and biomedical devices developed by mutually different vendors to transfer data dynamically. Data Binder is implemented as a bundle of OSGi platform providing component-based programming model and service-oriented operation architecture. Data Binder complements the disadvantage of OSGi WireAdmin service enabling static data delivery between a producer and a consumer of data. Data Binder normalizes an application requirement as an application descriptor and a device capability as a device descriptor so that it enables dynamic data delivery by making data producer/consumer pair in runtime. Therefore, Data Binder can be used as a connection management of a data link between a data producer and a data consumer in sensor-based application development. The object of this paper is to provide the facility of the healthcare service development by separating a data producer such as a biomedical device from a data consumer such as a healthcare application.

**Key words** : healthcare, data delivery mechanism, producer-consumer pattern, data binder

· 본 연구는 지식경제부 IT R&D 프로그램[2008-S-034-01(09MC2610)] 지원으로 수행하였습니다.

† 정 회 원 : 한국전자통신연구원 그린컴퓨팅연구부 선임연구원
k2kang@etri.re.kr
†† 비 회 원 : 한국전자통신연구원 그린컴퓨팅연구부 팀장
ljwoo@etri.re.kr
††† 종신회원 : 충남대학교 컴퓨터공학과 교수
hc@cnu.ac.kr

논문접수 : 2008년 10월 8일
심사완료 : 2009년 10월 7일



정보과학회논문지: 컴퓨팅의 실제 및 레터 제15권 제12호(2009.12)

## 1. 서 론

전세계적으로 유비쿼터스 기술[1]에 대한 관심이 증가하고 있고 유비쿼터스 세상을 실현하기 위한 많은 영역의 기술들이 융합되고 있다. 특히, 모바일 헬스케어나 센서 기반 네트워크를 이용하는 응용 분야에서는 사용자가 이용하는 장치의 수가 크게 증가할 것으로 예상된다. 또한 사용되는 장치들은 착용형의 소형 장치나 주변에 산재하는 형식으로 개발되고 서로 다른 벤더에서 제작될 것으로 예상되기 때문에, 장치 사용에 있어 많은 어려움이 있다. 이러한 장치들의 사용에 있어 표준화 정도가 미비하여 장치에서 측정된 데이터가 응용 프로그램으로 전달 되기 위한 다양한 방법이 혼재하게 될 것이다. 이러한 다양성은 프로그램 개발자의 응용 프로그램 개발을 어렵게 하고, 다양한 방식으로 장치가 사용되지 못하는 문제점이 있어 장치 개발자에게도 어려움을 준다.

따라서 본 논문은 장치 개발자와 응용 프로그램 개발자에게 상호 독립적인 개발 프로세스를 제공하기 위해 데이터 전달 인터페이스를 정의하고, 이를 기반으로 상호간의 데이터 전달을 효과적으로 수행할 수 있는 데이터 핫플러깅(Hot-Plugging) 메커니즘인 데이터 결합기(Data Binder)를 제안한다.

데이터 결합기는 응용 프로그램과 하부 장치 사이의 결합 관점에서 볼 때, 응용 설명자와 장치 설명자로 표현되는 메타데이터의 상호 정합 관계를 사용한다. 데이터 결합기는 상호 정합 관계가 확인되고 자동적인 데이터 전달을 위한 방법론으로 OSGi 아키텍처의 Wire-Admin 서비스[2]를 기반으로 하는 동적인 와이어 생성, 연결 및 관리 메커니즘을 제공한다.

구현된 데이터 결합기를 사용할 경우, 데이터 소비자는 사용하기를 원하는 데이터의 타입에 대해서만 알면 되고 이를 응용 설명자에 명기하면 된다. 데이터 생산자는 어떤 타입의 데이터를 생성하는지 장치 설명자에 명기하면 되고 상호간 서로에 대해 알 필요가 없는 구조이다.

본 논문의 구성은 다음과 같다. 제1장의 서론에 이어 제2장에서는 본 논문에서 제안하는 데이터 결합기 구현을 위한 관련 기술 분석 및 접근법에 대하여 설명한다. 제3장에서는 본 논문에서 제안한 데이터 결합기 설계 및 구현에 대하여 설명하고 제4장에서는 제안한 방식을 적용한 사례 연구 및 검토 사항에 대하여 설명한다. 마지막으로 제5장에서는 논문의 요약 및 결론에 대해 설명하고 향후 연구방향에 대하여 기술한다.

## 2. 관련 기술 분석 및 접근방법

본 논문의 목적은 BAN(Body Area Network)[3] 내에서 상호 연결된 생체 신호 측정용 착용형 장치들로부터 데이터를 수집하여 휴대용 사용자 단말로 데이터를 전달하는 방법에 관한 것으로 이를 위한 데이터 전달 방법론과 관련된 기술에 대해 먼저 알아본다.

데이터 중심의 설계는 데이터 중심의 분산된 내장형 시스템을 구축하기 위한 주요 방법론으로 등장하고 있다. DDS(Data Distribution Service)[4]와 JMS(Java Message Service)[5]는 잘 알려진 미들웨어 표준 API (Application Programming Interface)로서, 사용하기 편리한 발간/구독(publish/subscribe) 통신 모델의 장점을 제공하여 약한 결합의 확장성 있는 분산 애플리케이션을 개발할 수 있는 방법론을 제공한다. 또한 데이터 중심의 시스템 구축을 위한 다른 패턴으로 생산자/소비자(Producer/Consumer) 기반의 설계 패턴도 사용되고 있다.

DDS는 OMG(Object Management Group)에 의해 표준화된 것으로 복잡한 네트워크 프로그래밍을 단순화한 네트워킹 미들웨어로서 노드간에 데이터, 이벤트, 명령어 등을 주고 받기 위한 발간/구독 모델을 구현한 것이다. 그러나 BAN 기반으로 연결되고 소형 경량의 휴대용 사용자 단말을 사용하는 모바일 헬스케어 환경에서 DDS를 설치 운용하는 것은 오버헤드가 큰 문제가 있다.

JMS는 J2EE 기반의 응용 컴포넌트가 메시지를 생성, 송수신, 및 읽기를 할 수 있도록 지원하기 위해 JCP (Java Community Process)에서 JSR 914[6]로서 표준화된 메시징 방식으로서 약한 결합의 신뢰성 있는 비동기 방식의 분산 통신 모델이다. 그러나 모바일 헬스케어 서비스를 위한 소형 경량의 휴대용 단말에 JMS를 설치 운용하는 것 또한 오버헤드가 큰 문제가 있다.

생산자/소비자 (Producer/Consumer) 설계 패턴[7]은 마스터/슬래이버(Master/Slave) 패턴을 기본으로 하고 있으며 서로 다른 속도로 실행 되고 있는 다수의 루프에서 데이터를 공유하기 위해 사용된다. 전통적인 마스터/슬래이브 설계 패턴에 비해 생산자/소비자 설계 패턴은 서로 다른 속도로 데이터를 생산하고 소비하는 프로세스를 상호 분리하여 사용하는데 유용하다. 생산자/소비자 설계 패턴의 병렬 루프는 데이터를 생산하는 부분과 생산된 데이터를 사용하는 부분으로 나눌 수 있다.

생산자/소비자 패턴은 동시에 서로 다른 속도로 동작하는 복수의 프로세스를 조작하는데 편리하여 프로세스 사이의 버퍼링(Buffering) 된 통신 방식에서 효율적이며, 발간/구독 패턴에 비해 푸시(Push)와 풀(Pull) 방식의 통신이 가능하다는 장점을 가지고 있다.

따라서 이러한 기술들 중에서 본 논문에서 목적으로 하고 있는 착용형의 소형 측정 장치를 사용하고 BAN

영역 내에서 데이터 중심의 시스템 구축을 위해서는 생산자/소비자 패턴의 데이터 전달 방식이 적절하고 특히 센서 기반의 응용 영역에서는 효과적이다[8].

다음으로 헬스케어 환경에서 데이터 전달 메커니즘을 적용하기 위한 기술적 현황과 생산자/소비자 패턴 기반의 데이터 전달 메커니즘을 적용할 수 있는 소프트웨어 플랫폼 및 기반 환경에 대해서 살펴본다.

모바일 헬스케어 시스템의 구성 환경은 사용자가 착용하는 소형의 측정 장치와 PDA 또는 스마트 폰과 같은 사용자가 휴대할 수 있는 소형 컴퓨팅 장치로 구성된다.

일반적으로 모바일 헬스케어에서 생체 신호를 측정하기 위한 측정 장치는 여러 개의 센서를 내장하고 무선 통신 기능을 갖추고 있다. 그러나 측정 장치는 소형 경량으로 제작되기 때문에 최소한의 컴퓨팅 기능만을 가지며 데이터의 측정 및 전달 기능을 수행한다.

측정 장치에서 측정된 데이터는 휴대용 단말에서 처리되며 데이터 전송이 필요할 경우 무선랜이나 이동 통신망을 통해 헬스 포털과 같은 서버로 전송된다. 그러나 휴대용 단말 또한 자원 제한적인 소형의 컴퓨팅 장치이기 때문에 휴대용 단말의 소프트웨어 플랫폼은 필요한 서비스 및 구동 드라이버가 동적으로 구성될 수 있는 환경을 요구 한다.

모바일 헬스케어 분야에서 사용되는 휴대용 단말은 제한된 자원과 소형의 프로세스를 구비하고 있기 때문에 복잡한 사용자 개입 없이 서비스 배포 및 관리가 가능한 소프트웨어 플랫폼을 필요로 한다. 이러한 환경 제공을 위한 일환으로 휴대용 장치를 위한 차세대 모바일 자바 플랫폼에 대한 표준화가 진행되고 있고 JSR 248 (Mobile Service Architecture, or MSA)[9] 및 JSR 249(Mobile Service Architecture Advanced)[10]와 같은 규격이 있다. JSR 248은 정적인 실행환경을 정의한 것으로서 개발자가 사용할 수 있는 고정된 API를 이용하여 한번에 하나의 응용 프로그램을 다운로드하고 실행할 수 있다. 이러한 정적인 환경에 대한 보완으로서 JSR 249는 동적으로 새로운 API를 다운로드 할 수 있는 기능을 제공한다. JSR 248 및 JSR 249는 MIDP [11] 형식의 패키지로 응용 프로그램을 만들어 휴대용 단말로 다운로드 할 수 있는 플랫폼으로, 주로 게임이나 멀티미디어 관련 API를 제공하고 있지만 원격 관리 관련 API를 충실히 제공하지 못한다. J2ME 환경에서 더 많은 API를 제공하고 동적인 실행환경을 제공하는 것으로서 OSGi 환경이 있다. OSGi 실행 환경은 응용 프로그램의 다운로드, 클라이언트 미들웨어 뿐만 아니라 플랫폼의 모니터링까지 지원하고, 휴대용 단말에 적재된 다양한 API에 대한 접근 권한을 정책기반으로 관리할 수 있는 매우 안전한 CDC(Connected Device Configuration)[12] 기반의 프레임워크이다. OSGi 실행환경의 특징은 JSR 291(Dynamic Component Support for Java SE)[13] 및 JSR 232(Mobile Operational Management)[14]로서 JCP(Java Community Process)의 표준화 과정에 포함되어 있다.

본 연구의 목표 시스템은 관리자에 의해 동적으로 서비스를 배포하고 관리할 수 있는 플랫폼이 요구되고, PDA나 스마트 폰 수준의 휴대용 단말을 사용하기 때문에 소프트웨어 플랫폼으로 OSGi 실행 환경이 적합하다. 따라서 본 논문에서의 데이터 결합기는 OSGi 실행환경에서 실행되는 소프트웨어 모듈로 구현된다. 그러나 OSGi 규격에서 생산자/소비자 패턴으로 유연한 데이터 전달 메커니즘을 제공하는 WireAdmin 서비스는 데이터의 생산자와 소비자를 연결하기 위한 구조만 정의하고 있고 동적으로 진출입 되는 데이터의 생산자와 소비자와의 연결을 동적으로 해주는 방법을 제공하지 못한다. 따라서 이러한 환경에서 데이터 생산자와 소비자는 상호간에 서로에 대해 알 필요 없이 장치가 플러그인 되거나 응용 프로그램이 설치될 때 시스템에 의해 데이터 생산자/소비자 쌍으로 동적인 데이터 전달을 가능하게 하는 결합기가 필요한데, 본 연구의 데이터 결합기가 이러한 요구 사항을 해결하는 하나의 방법론을 제공한다.

## 3. 데이터 결합기(Data Binder) 설계 및 구현

본 장에서는 응용 프로그램과 하부 측정 장치와의 자동화된 데이터 전달을 가능하게 하는 데이터 결합기의 설계 및 구현에 관해 설명한다.

데이터 결합기는 응용 프로그램과 하부 장치 사이의 결합 관점에서 볼 때, 응용 설명자(Application Descriptor) 및 장치 설명자(Device Descriptor)로 표현되는 메타데이터의 상호 정합 관계를 사용한다. 데이터 결합기는 상호 정합 관계가 확인되고 자동적인 데이터 전달을 위한 방법론으로 WireAdmin 서비스를 기반으로 하는 동적인 와이어 생성, 연결 및 관리 메커니즘을 제공한다.

자동화된 데이터 전달을 위해 데이터를 생성하는 주체, 예를 들어 각 측정 장치의 기능 드라이버는 OSGi WireAdmin 패키지(org.osgi.service.wireadmin)에서 정의하고 있는 Producer 인터페이스를 구현해야 하고 데이터를 소비하는 주체, 예를 들어 응용 프로그램은 WireAdmin 패키지의 Consumer 인터페이스를 구현해야 한다.

### 3.1 WireAdmin 서비스 및 제약사항

WireAdmin는 OSGi 서비스 플랫폼에서 와이어링 토폴로지를 제어하는 관리적 서비스로서 그림 1과 같은 연결 메커니즘을 제공한다.

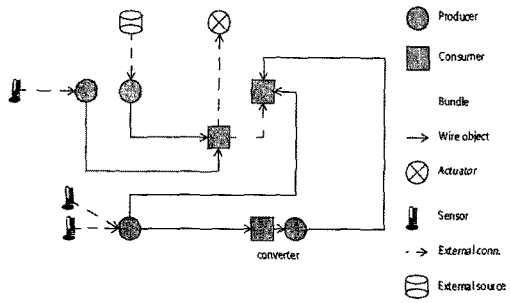


그림 1 OSGi 환경에서의 데이터 와이어링

WireAdmin 서비스는 데이터를 생산하는 Producer서비스 객체와 데이터를 소비하는 Consumer서비스 객체를 Wire 객체를 통해 연결하는 아래와 같은 관리 메소드를 제공한다.

```
public interface WireAdmin{
  public Wire createWire(String producerPID, String
    consumerPID, Dictionary properties);
  public void deleteWire(Wire wire);
  public Wire[] getWires(String filter) throws
    InvalidSyntaxException;
  public void updateWire(Wire wire, Dictionary
    properties)
}
```

그러나 Producer 서비스와 Consumer 서비스가 프레임워크에 등록이 되면 WireAdmin 서비스를 사용하는 관리자에 의해 정적으로 연결될 수 있으나 Producer 서비스와 Consumer 서비스의 등록 및 해지를 추적하여 동적으로 와이어(wire)를 연결하는 메커니즘을 제공하지 못하는 단점을 가지고 있다.

### 3.2 데이터 결합기(Data Binder)

WireAdmin 서비스를 사용하여 구성되는 모바일 헬스케어 응용에서 생체 신호 측정 장치와 응용 프로그램 사이의 구성은 그림 2와 같은 구조를 가진다.

데이터 결합기는 WireAdmin 서비스의 표준 규격을 구현할 때 정적인 연결에 의해 데이터 생산자와 소비자가 연결되는 구조를 개선한 것이다. 데이터 결합기는 개발자에게 직접적으로 제공되는 인터페이스는 가지지 않고 백그라운드에서 Producer 및 Consumer 서비스의 등록 및 해지 이벤트를 감시한다. 데이터 결합기는 Producer나 Consumer 서비스 이벤트가 발생하면 장치 설명자와 응용 설명자로부터 추출한 정보를 저장하고 있는 관리 테이블에서 상호 정합되는 Consumer와 Producer를 찾고, 이들을 연결하는 Wire 객체를 생성하여

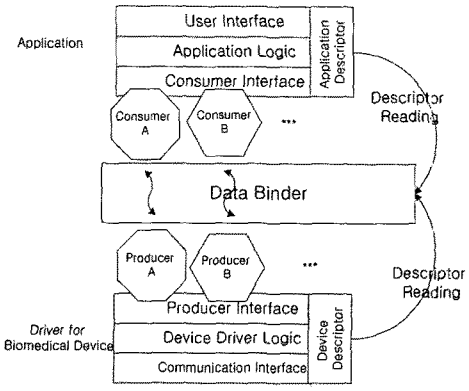


그림 2 WireAdmin 서비스가 적용된 헬스케어 서비스를 위한 응용프로그램과 측정 장치와의 동작 구성도

Consumer와 Producer에게 전달하는 역할을 수행한다. 이렇게 Wire 객체에 의해 Consumer와 Producer가 연결되면 데이터는 상호간의 풀(pull) 또는 푸시(push) 방식으로 전달된다.

데이터 결합기는 두 가지 방법으로 구현될 수 있다. 첫 번째는, WireAdmin 서비스와 별도의 패키지로 제작하여 데이터 결합기가 3rd 파티의 어떤 OSGi 프레임워크에도 실행될 수 있도록 하는 방법이다. 두 번째는, WireAdmin 서비스 표준 규격을 구현할 때 추가 기능으로 바인딩 기능을 통합하여 WireAdmin 패키지를 제작하는 것이다. 본 논문에서는 첫 번째 방법으로 데이터 결합기를 구현한 것으로, 다양한 벤더가 제공하는 프레임워크에서 표준 패키지의 변경 없이 동작되도록 하기 위함이다.

데이터 결합기는 그림 3과 같이 구성된다. 데이터 결합기는 WireAdmin 서비스의 기능을 사용하고 Producer/Consumer 서비스를 추적하는 두 개의 ServiceTracker, 응용 설명자와 장치 설명자의 메타데이터 정보를 저장 및 관리하는 MetaDataManager, 연결된 와이어 정보를 저장 및 관리하는 WireManager, Producer 서비스와

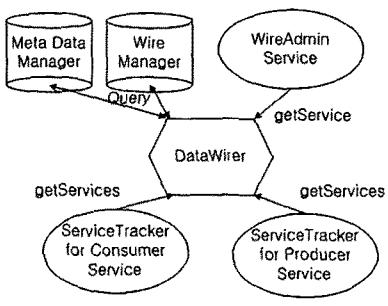


그림 3 데이터 결합기의 구성도

Consumer 서비스를 모니터링하고 서비스 등록이나 등록 해지 이벤트가 있을 때 데이터 생산자와 소비자를 연결하는 DataWirer로 구성된다.

아래의 예는 DataWirer에서 Producer 서비스와 Consumer 서비스를 추적하기 위한 코드의 예이다. DataWirer는 org.osgi.util.tracker의 ServiceTracker 클래스를 사용하여 Producer와 Consumer 이름으로 등록되는 서비스 객체를 추적한다.

```
***
ServiceReferene wireAdminRef =
        bc.getServiceReference(WireAdmin.class.getName());
WireAdminService wireAdminService =
        (WireAdmin)bc.getService(wireAdminRef);
ServiceTracker consumerTracker =
        new ServiceTracker(bc, Consumer.class.getName(), null);
consumerTracker.open();
ServiceTracker producerTracker =
        new ServiceTracker(bc, Producer.class.getName(), null);
producerTracker.open();
***
```

MetaDataManager는 응용 프로그램이 설치될 때 응용 프로그램 번들의 메타데이터로부터 응용 설명자의 내용을 저장하고, 하부 측정 장치가 연결될 때 장치나 장치 벤더로부터 장치 설명자의 내용을 저장한다. WireManager는 DataWirer에 의해 생성된 Wire 객체를 유지 관리하고, Wire의 연결점인 Producer와 Consumer 서비스 객체가 등록 해지될 때 연결된 Wire를 제거한다. WireManager는 ManagedWire라는 객체로 와이어 연결 정보를 저장하고, 데이터 소비자의 pid(Persistent IDentification), 데이터 생산자의 pid 및 Wire 객체를 담고 있다.

데이터 결합기는 사용자 인터페이스를 가지지 않는다. 데이터 결합기는 응용 프로그램과 같은 데이터 소비자의 요구사항과 측정 장치와 같은 데이터 생산자의 제공 사항을 분석하여, 동적인 연결 관리를 할 수 있는 백그라운드에서 실행되는 모듈 프로그램으로 아래와 같은 절차로 수행된다.

```
public class DataWirer implements ServiceListener{
  /*Consumer, Producer 서비스가 등록 및 해지되는 것을
    모니터링*/
  public void serviceChanged(ServiceEvent event){
    //그림 4 의 동작도 참고
    If Consumer 서비스 등록:
      1.MetaDataManager 로부터 Consumer 서비스가 사용하는
        Data 이름 검색
      2.검색된 Data 이름을 제공하는 Producer pid 검색
      3.Producer ServiceTracker 로 이 pid 로 등록된 Producer
        검색
      4.이 pid 의 Producer 발견되면 createWire 호출로 wire 생성
      5.WireManger 로 ManagedWire 객체로 등록(consumer pid,
        producer pid, Wire object 내용)
    Else Consumer 서비스 등록 해지:
      1.WireManager 에 등록된 ManagedWire 를 탐색
      2.Consumer pid 를 포함하는 ManagedWire 획득
      3.deleteWire(Wire) 호출로 wire 제거
      4.WireManager 에서 ManagedWire 제거
    //Producer 서비스의 등록 및 등록 해지인 경우에도 같은
순서로 동작
  }
}
```

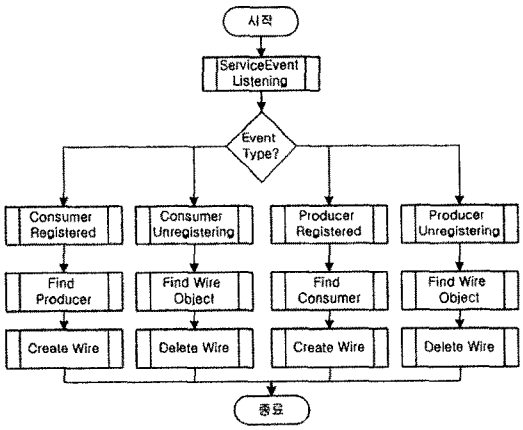


그림 4 Consumer/Producer 서비스 이벤트를 수신하고 처리하는 데이터 결합기의 동작도

### 3.3 메타데이터 정의

데이터 결합기에 의해 측정 장치로부터 응용 프로그램으로 동적으로 연결되어 데이터가 전달되기 위해서는 응용 설명자와 장치 설명자가 정의되어야 한다.

응용 설명자는 응용 프로그램이 실행되기 위해서 필요한 정보를 명기해야 한다. 응용 프로그램이 데이터 결합기에 의해 동적으로 연결되어 장치로부터 데이터를 수신하기 위해 응용 설명자는 다음 예와 같이 필요한 데이터의 이름, 형식, 식별자 및 업데이트 간격과 같은 정보를 명기해야 한다.

```
<consuming>
  <data>
```

```
    <name>HR</name>
    <type>Integer</type>
    <pid>kr.re.etri.bio.util.Consumer.HR</pid>
    <update_interval>1000</update_interval>
  </data>
  <data>
    <name>PPG</name>
    <type>Float</type>
    <pid>kr.re.etri.bio.util.Consumer.PPG</pid>
    <update_interval>2000</update_interval>
  </data>
</consuming>
```

장치 설명자는 장치가 제공할 수 있는 데이터 정보를 명기하고 있다. 측정 장치가 데이터 결합기에 의해 동적으로 연결되어 응용 프로그램으로 데이터를 전달하기 위해서, 장치 설명자는 다음 예와 같이 제공 가능한 데이터의 이름, 형식, 식별자 및 업데이트 간격과 같은 정보를 명기해야 한다.

```
<producing>
  <data>
    <name>HR</name>
    <type>Integer</type>
    <pid>kr.re.etri.bio.util.Producer.HR</pid>
    <update_interval>1000</update_interval>
  </data>
  <data>
    <name>PPG</name>
    <type>Float</type>
    <pid>kr.re.etri.bio.util.Producer.PPG</pid>
    <update_interval>2000</update_interval>
  </data>
</producing>
```

### 3.4 응용 프로그램 및 장치 드라이버의 요구사항

데이터 결합기에 의한 동적인 데이터 전달 메커니즘이 적용되기 위해 응용 프로그램 모듈에서 필수적으로 해야 하는 것은 두 가지이다. 첫 번째는 응용 설명자를 작성하여 번들을 패키징할 때 번들 매니페스트 파일에 응용 설명자의 위치를 명기하는 것이다. 두 번째는 org.osgi.service.wireadmin 패키지의 Consumer 인터페이스를 구현하는 것이다.

```
public interface Consumer{
    public void producersConnected(Wire[] wires);
    public Object updated(Wire wire, Object value);
}
```

데이터 결합기에 의해 동적으로 데이터 전달 메커니즘이 적용되기 위해 장치 드라이버 모듈에서 필수적으로 해야 하는 것은 두 가지이다. 첫 번째는 장치 설명자를 작성하여 측정 장치의 플러그인 과정에서 플랫폼으로 전달 해야 하는 것이다. 두 번째는 org.osgi.service.wireadmin 패키지의 Producer 인터페이스를 구현하는 것이다.

```
public interface Producer {
  public void consumersConnected( Wire[] wires );
  public Object polled( Wire wire );
}
```

## 4. 사례 연구 및 토의

본 논문에서 제안한 생산자/소비자 중재기반의 데이터 전달방법의 가용성을 확인하기 위하여 모바일 헬스케어 시스템에서 측정 장치와 헬스케어 응용 프로그램에 적용해 보았으며 적용 환경은 그림 5와 같이 구성하였다.

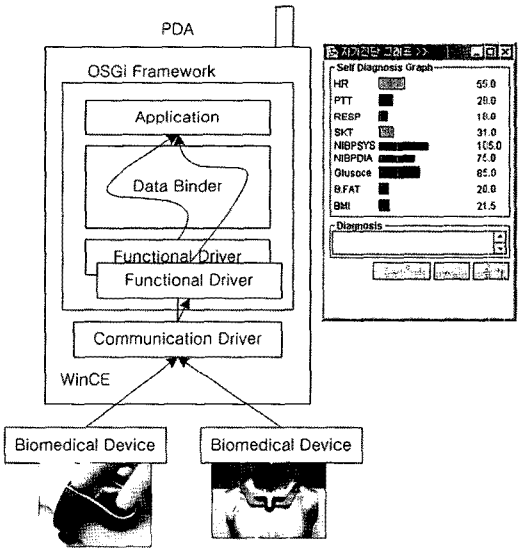


그림 5 적용 환경

- 착용형 생체 신호 측정 장치: ZigBee 통신 기반 시계형, 목걸이형 측정장치
- 측정 장치의 기능 드라이버: OSGi 번들로 구현되며 시계형은 EKG(Electrokadiogram), PPG(Photoplethysmography), SKT(Skin Temparature), ACC(Acceleration), Tilt 정보를 생산하는 5개의 Producer로 구성되고 목걸이형은 EKG(Electrokadiogram), RES(Respiration), ACC(Acceleration)를 생산하는 3개의

Producer로 구성

- 모바일 헬스케어 소프트웨어 플랫폼: WinCE 5.0 기반의 PDA(520MHz CPU, 64MB RAM, CDMA, and WLAN communication)에 J2ME CDC personal profile 기반의 CrE-ME[15] 가상 머신이 설치되고 가상 머신상에 OSGi R3 호환 프레임워크 설치
- 헬스케어 응용 프로그램: 자가진단 서비스는 자신의 생체 신호 또는 생체 지수를 PDA를 통해서 모니터링하고 간단한 건강 관련 정보를 제공하는 서비스

본 구성에서 측정 장치의 기능 드라이버는 데이터 생산자 역할을 하고 자가진단 응용 프로그램은 데이터 소비자 역할을 수행한다.

Producer와 Consumer를 이용하여 데이터 결합기에 의한 동적인 와이어 연결 및 해지에 대한 지연 시간을 측정하였다. 시험을 위해 100개의 테스트 Consumer와 100개의 테스트 Producer를 사용하였고, 연결하는 Wire 수를 10개에서 100개까지 증가 시키면서 동적으로 와이어가 생성 및 제거되는 시간을 PDA에서 측정하였으며 그림 6과 같은 결과를 얻었다.

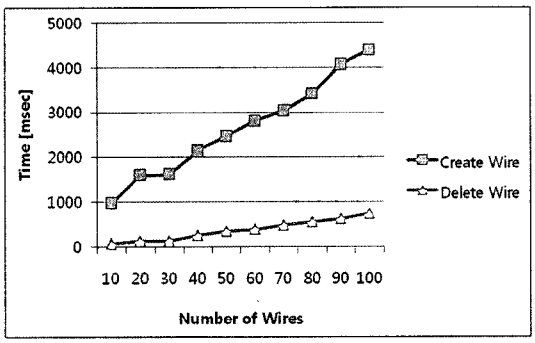


그림 6 Consumer/Producer동적 연결 및 해지에 따른 지연 시간

와이어 하나 당 연결되는 평균 지연 시간은 약 60 [msec]이고 와이어가 삭제되는 평균 지연 시간은 약 7[msec] 소요된 것으로 측정되었다. 데이터 결합기가 실제 응용에서 적용될 때 메타데이터가 정의되어 있으면 데이터를 주고 받는 쌍이 만들어지는데 약 0.06초 정도 소요되므로 거의 실시간으로 측정 장치 및 응용 프로그램의 플러그인시 동작된다고 판단할 수 있다.

데이터 결합기를 모바일 헬스케어 분야에서 실제 적용할 경우, 사용자가 사용하는 생체 신호 측정 장치는 보통 두 개 또는 세 개로 한정된다. 그리고 데이터를 생산하는 측정 장치당 데이터 Producer는 BAN 통신의 한계 등으로 인하여 보통 10개 이하로 운용되기 때문에 데이터 결합기에 의한 동적인 데이터 전달은 실시간으로 이뤄진다고 판단된다.

유사한 연구와의 비교 측면에서 살펴보면, 서비스 바인더[16]는 OSGi 서비스 플랫폼에서 서비스간 의존관계를 관리하는 메커니즘을 제안한 것이다. 서비스 바인더는 의존 관계에 있는 서비스 객체들의 자동 발견, 서비스 등록 및 해지를 모니터링하여 적응적인 번들 실행을 가능하게 하지만 Consumer 서비스, Producer 서비스로 등록되는 서비스 객체간 자동화된 동적인 와이어링에 대한 방법론을 제시하지 못한다.

산업계에서 많이 사용되는 데이터 전달 방법론과의 비교 측면에서 살펴보면, OMG의 DDS[4]는 분산 시스템에서 토픽(topic)을 기반으로 하는 QoS(quality of service) 지원 발간/구독(publish/subscribe) 방식의 데이터 분배 서비스이고, JCP의 JMS[5]는 분산 메시징을 지원하는 자바 프로그래밍 인터페이스로서 메시지 큐를 이용하는 일대일 대응의 방식과 토픽(topic)을 사용하여 다수의 수신인에게 메시지 전달이 가능한 발간/구독 방식을 지원한다. 그러나 DDS 및 JMS는 데이터의 생산자와 소비자의 플러그인 또는 플러그 아웃 때 동적으로 양단을 연결하는 기능 및 폴링(polling) 방식으로 데이터를 읽을 수 있는 기능을 지원하지 못한다.

## 5. 결 론

모바일 헬스케어는 다양한 형태의 착용형 생체 신호 측정 장치와 사용자가 휴대하는 정보 단말로 구성된다. 이러한 환경에서 유연한 상호 동작을 위해서는 응용 프로그램과 같은 데이터를 사용하는 구성 요소와 측정 장치와 같은 데이터를 생산하는 구성 요소가 가능한 느슨한 결합으로 동작되는 것이 효과적이다.

본 논문의 데이터 결합기는 응용 프로그램이 요구하는 데이터 요구사항과 장치가 제공하는 데이터 제공사항을 상호 정합시켜 필요한 데이터가 자동으로 연결되는 구조를 제안한 것이다. 따라서, 센서 기반 응용을 개발할 때 데이터 생산자와 데이터 소비자 사이에서 빈번하게 일어나는 연결 관리를 동적으로 해주는 기능 구현에 활용 가능하다. 구현된 데이터 결합기를 사용할 경우 데이터 소비자는 사용하기를 원하는 데이터의 타입에 대해서만 알면 되고 이를 응용 설명자에 명기하면 된다. 데이터 생산자는 어떤 타입의 데이터를 생성하는지 장치 설명자에 명기하면 되고 상호간 서로에 대해 알 필요가 없는 장점을 제공한다.

향후 연구 과제로는 측정 장치와 응용 프로그램 사이에서 주고 받는 생체 신호 데이터를 표현하는 방법에 있어 일관성을 기하기 위해 HL7[17]과 같은 국제 표준화 단체에서 표준화가 진행되고 있는 표현법으로 생체 데이터를 정의하고 매핑하는 일이 필요할 것이다.

## 참 고 문 헌


[1] M. Weiser, "The computer for the 21st century," *Scientific American*, pp.94-104, 1991.

[2] OSGi Alliance, "OSGi Service Platform: The OSGi Alliance," IOS Press, 2003.

[3] Body Area Network (BAN), http://en.wikipedia.org/wiki/Body_Area_Network

[4] OMG, "Data Distribution Service (DDS) Specification," http://www.omg.org/technology/documents/dds_spec_catalog.htm

[5] Sun Microsystems, "Java Message Service (JMS)," http://dlc.sun.com/pdf/816-5904-10/816-5904-10.pdf

[6] Java Community Process, Java Message Service, http://jcp.org/en/jsr/detail?id=914

[7] National Instruments, "Application Design Patterns: Producer/Consumer," http://zone.ni.com/devzone/cda/tut/p/id/3023

[8] N. Nillson, "Connecting Producers and Consumers," position paper at OOPSA Workshop on References Architectures and Patterns for Pervasive Computing, 27 October 2003.

[9] Java Community Process, "Mobile Service Architecture," http://jcp.org/en/jsr/detail?id=248

[10] Java Community Process, "Mobile Service Architecture," http://jcp.org/en/jsr/detail?id=249

[11] Sun, "Mobile Information Device Profile," http://java.sun.com/products/midp/

[12] Sun, "Connected Device Configuration," http://java.sun.com/javame/reference/apis.jsp

[13] Java Community Process, "Dynamic Component Support for Java SE, http://java.sun.com/javame/reference/apis.jsp

[14] Sun, "Mobile Operational Management," http://jcp.org/en/jsr/detail?id=232

[15] NSIcom, CrE-ME Java virtual machine, http://www.nsicom.com/

[16] H. Cervantes, R.S. Hall, "Automating Service Dependency Management in a Service-Oriented Component Model," 6th International Symposium on Component-Based Software Engineering (CBSE), 2003.

[17] HL7, "Health Level Seven," http://www.hl7.org/



강 규 창

1994년 경북대학교 전자공학과 졸업(학사). 1997년 경북대학교 전자공학과 졸업(석사). 2009년 충남대학교 컴퓨터공학과 졸업(박사). 1997년~2001년 국방과학연구소 종합시험단 근무. 2001년~현재 한국전자통신연구원 IT 융합기술 연구부문 선임연구원. 관심분야는 모바일 컴퓨팅 미들웨어, 분산 미들웨어, 컴포넌트 기반 아키텍처 및 가상화 기술



이 전 우

1983년 경북대학교 전자공학과 졸업(학사). 1985년 경북대학교 전자공학과 졸업(석사). 1998년 경북대학교 전자공학과 졸업(박사). 1985년~현재 한국전자통신연구원 IT 융합기술연구부문 팀장. 관심분야는 홈서버 구조, Post-PC 플랫폼, 가상화 기술, 클라우드 컴퓨팅



최 훈

1983년 서울대학교 컴퓨터공학과 졸업(학사). 1990년 Duke University 전산학과 졸업(석사). 1993년 Duke University 전산학과 졸업(박사). 1983년~1996년 한국전자통신연구원 광대역통신망연구부 근무. 1996년~현재 충남대학교 컴퓨터공학과 교수. 2000년 미국 NIST(National Institute of Standards and Technologies) 객원연구원. 관심분야는 모바일 컴퓨팅/분산 시스템 미들웨어, 운영체제