碩士學位論文

ALE 기반 RFID미들웨어 설계 및 구현

濟州大學校 大學院

컴퓨터工學科

洪 蓮 美

2007年 12月

ALE 기반 RFID미들웨어 설계 및 구현

指導敎授 邊 暎 哲

洪 蓮 美

이 論文을 工學 碩士學位 論文으로 提出함

2007年 12月

洪蓮美의 工學 碩士學位 論文은 認准함

審査委員長 _____ 印

委 員 _____ 印

委 員 _____ 印

濟州大學校 大學院

2007年 12月

A Design and Implementation of ALE-compliant RFID Middleware System

Yeon-Mi Hong

(Supervised by professor Yung-Cheol Byun)

A thesis submitted in partial fulfillment of the requirement for the degree of Master of Computer Engineering

2007. 12.

This thesis has been examined and approved.

Thesis director, _____

Thesis director, _____

Thesis director, _____

December 2007

Department of Computer Engineering

Graduate School

Cheju National University

감사의글

어느덧 석사논문의 한쪽에 담을 감사의글을 쓰게 되었습니다. 돌아보면 만감이 교차합니다. 제가 제주도에 머문지도 횟수로 9년째가 되었습니다. “꽤 오래 살았구나~” 이런 생각이 들지만, 사실 추억을 들춰보면 학교를 6년동안 다녔더군요.

일찍 결혼을 하여 아이 둘 낳고 제 인생에 투자를 늦게 한 셈이죠. 그렇게 하고 싶던 공부를 시작한터라 정말 열심히 하고 싶었습니다.
“하늘은 스스로 돕는 자를 돕는다”라는 말이 맞는 것 같습니다. 열심히 할 때마다 도와주시는 분들이 계시더군요. IT라는 학문에 입문하면서 아이 둘 키우는 주부라고 격려와 배려를 아끼지 않으셨던 한라대학 컴퓨터정보과에 김형수교수님, 이상부교수님, 김영상교수님, 문석환교수님과 동기분들, 또 그와 반대로 전문인으로 거듭 설수 있게 채찍질과 쓴 경험담을 늘 해주셨던 현재 제주지식산업진흥원 총괄 실장님이신 김영철실장님 감사합니다.

전문대학 2년과 편입학 2년 그리고 대학원 2년 중간 중간 실무경험을 얻고자 회사에 입사했었고 다시 학교에 진학하는 일을 반복하였습니다. 지난 시간이 다 만족스럽지는 않지만 “그래도 열심히 달렸고, 결실을 맺을 수 있구나!” 하는 안도의 한숨을 잠시 쉬어봅니다.

아트피큐의 오태헌사장님, 강동섭이사님, 디자이너 변정윤씨와 그 때 제주도청 사이트 리뉴얼을 함께 진행했던 고수민씨, 현재는 공무원이 된 김영대씨 양재석씨 허물없는 동료였고, 친구이고 가족 같았던 사람들입니다. 함께할 수 있었음에 행복했고 감사합니다. 이글을 쓰고 있으니 머릿속에서 영화처럼 재생되고 감사할 사람들이 너무 많습니다.

우리 지도교수님은 열정과 강력한 카리스마를 지닌 분이십니다. 도전과 야망을 몸소 보여주셨습니다. 그것을 본받고 싶었고 닮고 싶었습니다. 변영철 교수님과 인연을 닿은지 학부때부터 대학원 생활까지 6년이 넘었습니다. 때론 스승으로 때론 경쟁자로 야속하기도 하고 미워하는 맘도 들었지만 인생에 단단한 인연이 되라고 이런 저런 감정을 갖게 했나봅니다. 감사합니다. 지금은 학교를 벗어나 사회에 공헌할 수 있는 전문인으로 걷기 시작했습니다. “항상 조금더 전문가로 인정받을 수 있는 사람이 되라 그러기 위해선 자기의 질을 높여라” 라는 말씀 가

슴에 새기고 살겠습니다.

또한 자판기 커피를 즐겨 드시던 김장형교수님, 학부생들에게 항상 인자하신 안기중교수님, 빌케이츠를 닮은 곽호영교수님, 동방문화진흥회 지회장을 역임하실 정도로 주역에 달인 변상용교수님, 온유함 속에 족집게 같은 날카로움을 지닌 이상준교수님, 얼굴에 항상 웃음과 도전 정신을 심어주시는 김도현교수님과 정은경, 이정하 조교선생님께도 감사합니다.

그리고 가장 최근까지 희노애락을 함께한 동료 영식, 지웅, 문석, 희준, 지영 또한 지난날 인터넷 컴퓨팅 연구실에서 함께했던 승건, 성남, 근애, 오상현박사님, 자바정보기술(주)에 재직중인 조윤상과장님 과 인터넷컴퓨팅연구실에 석사 1기 졸업생인 자바정보기술(주)의 박상열사장님께도 감사합니다. 감사함을 전할 모든 분들을 나열하려면 3박 4일 동안 써도 다 적지 못할 듯합니다. 힘들 때 서로 격려해주고 동고동락한 민성, 혜선, 부림, 성수 끝까지 잘 참아주었고 함께 졸업할 수 있어 너무 행복하고 컴퓨터공학과 대학원 선후배님들과 지금 현재 새로운 꿈을 함께 만들어가는 자바정보기술(주) 서울지사 식구들 강명호차장님, 오상현박사님, 조윤상과장님, 김광래과장님, 양윤석대리님, 혜영씨, 용철군과도 기쁨을 함께 나누고 싶습니다.

끝으로 넉넉한 마음으로 뒤에서 지켜봐준 가족들과 자기 일에 정신없이 바쁜 나쁜엄마의 걱정을 덜어주듯 스스로 자기 할 일을 척척 해내는 두 아들 오승훈, 오승민 사랑합니다.

목 차

그림 목차

표 목차

국문초록

ALE 기반 RFID미들웨어 설계 및 구현

컴퓨터공학과 홍연미
지 도 교 수 변영철

유비쿼터스 컴퓨팅을 실현하기 위하여 자동식별, 센서 네트워크, 홈 네트워크, 텔레매틱스 등 다양한 분야에서 연구가 진행되고 있다. 특히 자동식별 분야의 RFID 기술은 비 접촉식 무선인식 기술로서 상품에 부착된 바코드가 갖고 있는 인식 속도, 저장 공간 등을 더욱 발전시켜 유통·물류산업에 혁신적인 변화를 줄 수 있는 차세대 유비쿼터스 컴퓨팅의 핵심기술이다. 최근 RFID기술을 이용한 레거시 시스템과의 통합 및 데이터 수집, 장치 제어, 관리 등을 위하여 RFID 미들웨어의 중요성에 대한 인식이 확산되고 있다. 이에 따라 EPCglobal에서는 기존의 Savant를 대체하는 개념으로서 RFID 하드웨어와 응용 사이에 존재하는 ALE(Application Level Events) 스펙을 정의하였다. ALE 스펙은 RFID 리더 등 EPC 데이터 소스에서 애플리케이션의 이벤트로 이동하는 데이터의 양을 줄이는 프로세싱이 필요하며 물리적 계층에서 발생한 데이터를 애플리케이션 계층에서 분리하는 것이 시스템의 유연성 측면에서 효율적이라는 관점에서 국제 표준으로 제정된 것으로, 레이어의 내부 구조 및 구현 기술 등은 명시하지 않고 오직 외부 인터페이스만을 정의함으로써 향후 상호 운용성 및 표준 적합성 검증을 용이하게 한다. 본 논문에서는 이러한 ALE 스펙에 기반하여 표준적인 인터페이스를 설계함은 물론 표준 스펙에서 정의하고 있는 다양한 RFID 태그 데이터를 처리 기능을 갖는 RFID 미들웨어 시스템을 설계하고 구현한다.

ABSTRACT

A Design and Implementation of ALE-compliant RFID Middleware System

HONG, YEON-MI
Department of Computer Engineering
Graduate School
Cheju National University

Nowadays, to realize ubiquitous computing environment, many research activities have been going on within various kinds of research domains including automatic identification, sensor network, home network, telematics and so on. Especially, RFID middleware that supports the aggregation of RFID tag data, control and management, and the integration with legacy systems has recently gained a lot of attention. Meanwhile, EPCglobal defined an ALE (Application Level Events) standard specification, which exists between RFID readers and applications, and substitutes the previous systems called Savant. In the specification internal structures and implementation technologies of ALE are not mentioned and only external interfaces are defined. This approach easies the verification of standard compliance and inter-operability of the layer. In this thesis, we present the design of ALE-compliant RFID middleware systems that process RFID tag data efficiently.

약어표

RFID	Radio Frequency Identification
EPC	Electronic Product Code
GTIN	Global Trade Item Number
SSCC	Serial Shipping Container Code
GRAI	Global Returnable Asset Identifier
GIAI	Global Individual Asset Identifier
GID	General Identifier Auto-ID Center
GLN	Global Location Number
URI/URN	Uniform Resource Identifier/ Uniform Resource Name
URL	Uniform Resource Locator
ALE	Application Level Events
USN	Ubiquitous Sensor Network
TCP/IP	Transmission Control Protocol/Internet Protocol
DAO	Data Access Object
ISO	International Standardization Organization
UCC	Uniform Code Council
ONS	Object Naming Service
ARC	Architecture Review Committee
TBD	To Be Determine
Gen2	Class 1 Generation 2 UHF Air Interface Protocol
EPCIS	Electronic Product Code Information Service
API	Application Program Interface
WD	Working Draft
TDT	Tag Data Translation
UML	Unified Modelling Language
GS1	Global Standard 1
CSML	Context-driven Service Markup Language

I. 서 론

1. 연구 배경

유비쿼터스 컴퓨팅을 실현하기 위하여 자동식별, 홈 네트워크, 센서 네트워크, 텔레매틱스 등 다양한 분야에서 연구가 활발히 이뤄지고 있다. 특히 자동식별 분야에 있어서 사람 및 사물의 신원 정보를 제공할 수 있는 RFID가 주목받고 있다. 초기의 RFID 기술 연구 및 시장 형성은 주로 사물에 부착하기 위한 태그와 이를 무선을 통해 자동으로 인식하기 위한 칩, 리더 등의 하드웨어 중심으로 발전 되어왔다. 최근 RFID 기술을 이용하여 물류, 소매업, 의료, 공장・가정・사무실 자동화, 보안, 재난 방지, 재산 관리 등 편리한 서비스를 제공하는 응용들이 점차 늘어나고 있다[1, 2, 3]. 따라서 응용 개발자로 하여금 RFID 기술을 이용한 다양한 유비쿼터스 컴퓨팅 응용 서비스를 쉽게 구축할 수 있도록 하기 위해 RFID 태그가 부착된 객체와 응용 서비스의 교량 역할을 해주는 미들웨어의 중요성에 대한 인식이 확산되고 있다.[4, 5].

RFID 미들웨어는 다양한 센서를 관리하며, 센서로부터 다양한 프로토콜을 이용하여 데이터를 수집하고, 또한 수집된 가공되지 않은 데이터로부터 의미 있는 정보를 추출하여 응용 서비스인 비즈니스 시스템에 전달하는 기본적인 기능이 필요하다. 그러한 기능을 충족한 미들웨어 시스템 제품 및 솔루션들로는 Oracle의 Sensor Edge Server, CapTech의 TagsWare, SUN의Java System RFID Software와 Etri의 Ubicore가 출시되고 있다[7-9]. 하지만, 복잡한 비즈니스 로직의 처리나 다양한 형식을 가지는 센싱 된 데이터의 처리에 대한 고려가 미진한 상태이다[5]. 또한 응용과 미들웨어 간 독립성을 제공하기 위해 표준화된 방법을 따르지 않으므로 다양한 외부 비즈니스 로직과의 상호 운영성에 대한 서로 다른 제약의 문제점이 발생하여 비용 및 융통성 면에서 이점을 제공하지 못한다.

최근 EPCglobal에서 진행되고 있는 EPC 네트워크 구성 요소 중 RFID 미들웨어와 응용 사이에 존재하는 ALE가 기존 Savant의 개념을 대체하였는데, ALE 스펙에서는 미들웨어 내부에 대한 상세한 구현에 대해서는 다루지 않고 다만 사용자 인터페이스를 기술함으로써 미들웨어 와 응용서버 간 상호 운용성 및 표준 적합성 검증을 용이하고, ALE 인터페이스를 제공함으로써 어떠한 응용 시스템이

라도 동적으로 ECSpec(요구사항)을 제출하여 ECReports(처리 결과)를 전송 받을 수 있으므로 실시간으로 원활한 서비스를 제공 받을 수 있다[9-11].

본 논문에서는 다수의 응용 시스템이 서비스를 요청시 큐 핸들 동적 바인딩 처리 방법으로 데이터의 손실을 예방 할 수 있고, ALE 스펙에서 명시한 SOAP 통신 이외에 추가로 RMI 통신 프로토콜을 추가로 제공하여 JAVA 개발 환경에서는 더 높은 효율을 기대할 수 있다.

2. 연구 목적 및 방법

본 논문에서는 ALE 스펙을 적용한 RFID 미들웨어를 설계 및 구현한다. ALE 스펙을 기반으로 RFID 미들웨어를 설계하여 응용 개발자에게 ALE 인터페이스를 제공한다. 따라서 어떤 응용이라도 동적으로 간편하게 ALE 스펙에서 명시한 XML 인스턴스 형식으로 ECSpec(요구사항)을 기술하고 미들웨어에 독립적이고 보다 효율적으로 RFID 시스템 운영이 가능하게 하는데 목적이 있다. 제안된 RFID 미들웨어에서는 응용의 ECSpec을 처리하기 위해 개별 데이터처리를 위한 이벤트큐 생성과 ECSpec에 포함된 ECBoundarySepc에 기술된 조건동안 EPC데이터를 가져오는 일을 수행하는 이벤트 사이클 처리 블록과 이벤트 사이클 동안 쌓인 EPC 데이터를 필터링, 그룹핑 등 연산처리를 담당하는 연산 처리 블록, 연산 처리 블록의 결과로 얻어진 데이터를 응용이 원하는 코드 형식으로 생성 및 전송하는 역할을 담당하는 리포트 처리 블록과 이런 처리를 원활하게 하기 위한 전처리 단계인 데이터의 손실을 예방하는 브로드캐스트 처리 블록 등을 설계 및 구현한다.

3. 논문 구성

본 논문의 구성은 다음과 같다. Ⅱ장에서는 본 연구와 관련된 RFID 관련 국내외 표준과 EPCglobal의 ALE 스펙 및 기타 기술들을 기술한다. Ⅲ장에서는 ALE 스펙 기반으로 RFID 미들웨어를 설계한다. Ⅳ장에서는 RFID 미들웨어에 대한 구현 및 실험에 대해 기술한다. 마지막으로 Ⅴ장에서는 본 연구의 결론에 대해 서술한다.

II. 관련 연구 및 고려사항

1. RFID 표준화 동향과 현황

1) RFID 표준화 동향

RFID는 비 접촉식 무선인식 기술로서 상품에 부착된 바코드가 갖고 있는 인식 속도, 저장 공간 등을 더욱 발전시켜 유통·물류산업에 혁신적인 변화를 줄 수 있는 차세대 유비쿼터스 컴퓨팅의 핵심기술이다. 이러한 RFID에 대하여 최근 국제 표준화 기구인 ISO에서는 관련 통신규약, 식별코드, 성능시험방법 등 대부분의 기술표준 제정을 완료하였으며, 한국은 ISO 규격 26종을 KS 규격으로 제정하였다. 또한, 컨테이너 보안용 전자봉인, 공급망관리(SCM)등의 응용 표준 등은 여러 인증시험을 거쳐 완성단계에 있다. EPC 보급에 앞장서고 있는 국제단체인 EPCglobal은 공급망관리상의 모든 주체들에 RFID를 적용하여 유용한 정보들을 서로 공유할 수 있도록 하는 EPCglobal 네트워크 구축을 완료하고 업종별 비즈니스 모델 발굴을 위해 유통. 물류분야와 의류, 약품 분야 등에 액션 그룹을 설치 및 운영하고 보급. 확산에 노력하고 있다.

2) 국제 표준화 현황

현재까지의 RFID 관련 ISO 표준화 현황에 대해 정리하면 표 1과 같다.

표 1. RFID 관련 ISO 국제 표준화 현황

그룹	그룹명	ISO/IEC	작업명	현단계	비고
SG1	Data 구문표준	15961	태그 Commands	IS	데이터프로토콜
		15962	Data Syntax	IS	
SG2	태그식별	15961	태그 식별자	IS	태그ID 식별
SG3	Air Interface	18000-1	Generic Parameters	IS	파라미터 규정
		18000-2	Below 135KHz	IS	FA, 동물인식
		18000-3	13.56GHz	IS	IC 카드, 신분증
		18000-4	2.45GHz	IS	Traceability 등
		18000-5	5.8GHz	CD부결	ITS
		18000-6	UHF 860-930MHz	IS	유통 물류
		18000-7	UHF 433MHz	IS	컨테이너(Active)
ARP	적용기술	TR18001	Application 요구사항	IS(TR)	적용조건조사

현재까지의 EPCglobal 네트워크 관련 표준화 현황에 대해 정리하면 표2와 같다.

표 2. EPCglobal Network 관련 표준화 현황

구분	표준	시기	상태
Object Exchange	UHF Class 0 Gen 1 RF Protocol	2003.02	Auto-ID Center 발표 비준(Ver. 1.09)
	UHF Class 1 Gen 1 RF Protocol	2002.11	
	HF Class 1 Gen 1 Tag Protocol UHF Class 1 Gen 2 Tag Protocol	2003.02 2005.01	
	EPC Tag Data Specification	2005.06	비준(Ver. 1.28)
Infrastructure	Reader Protocol	2006.06	비준(Ver. 1.1)
	Reader Management	-	WD
	Tag Data Translation	2006.01	비준(Ver. 1.09)
	Application Level Events(ALE)	2005.09	비준(Ver. 1.09)
	EPCIS Capture Interface	-	WD
Data Exchange	EPCIS Data Specification	-	WD
	EPCIS Query Interface	-	WD
	ONS	2005.10	비준(Ver. 1.09)
	EPCglobal Certificate Profile	2006.03	비준(Ver. 1.09)
	EPCIS Discovery Service	-	Business Action Groups, Architecture Review Committee에서 연구중

* 2006년 11월 기준 표준화 현황(EPCglobal)

3) 국내 표준화 현황

국내에서도 최근 국제표준으로 제정된 RFID용 프로토콜, 평가방법 등의 기술표준을 국가표준(KS)으로 도입하였다. 표 3은 2005년에 이어 추가적인 국가표준(KS) 14종을 2006년 말에 확정하였다. 또한 2008년까지 RFID관련 국가표준 60여종을 정비하고 국내 산업계에 보급할 계획에 있다.

표 3. 2006년 확정된 RFID 국가 표준(안) 14종

구분	KS 규격명	표준 분류	관련 국제규격
1	무선인식(RFID) 성능시험 방법	시험	ISO/IEC 18046
2	135KHz 이하 에어 인터페이스에 대한 적합성 시험방법	시험	ISO/IEC 18047-2
3	13.56MHz에어 인터페이스에 대한 적합성 시험방법	시험	ISO/IEC 18047-3
4	433MHz에어 인터페이스에 대한 적합성 시험방법	기반기술	ISO/IEC 18047-7
5	860-960MHz 에어 인터페이스에 대한 적합성 시험방법	기반기술	ISO/IEC 18047-6
6	2.45GHz 에어 인터페이스에 대한 적합성 시험방법	기반기술	ISO/IEC 18047-4
7	실시간 위치 추적 시스템(RTLS)에 관한 응용 인터페이스 프로그램	응용	ISO/IEC 24730-1
8	실시간 위치 추적 시스템(RTLS)에 관한 2.45Hz 에어 인터페이스 프로토콜	응용	ISO/IEC 24730-2
9	KS X 18000 에어 인터페이스에 대한 라이센스 플레이트용 기본 기능 정의	기반	ISO/IEC 24710
10	UHF 대역 18000-6 Type c의 태그 프로토콜	기반	ISO/IEC 18000-6
11	수송단위의 식별코드	데이터	ISO/IEC 15459-1
12	식별코드의 등록절차	데이터	ISO/IEC 15459-2
13	식별코드의 공통 규약	데이터	ISO/IEC 15459-3
14	공급망관리(SCM)를 위한 식별코드	데이터	ISO/IEC 15459-4

2. EPCglobal

1) EPCglobal의 개요

EPCglobal은 상품코드의 국제표준 개발 및 관리 기구인 EAN과 UCC가 통합되어 탄생한 GS1(Global Standard 1)이 2003년 11월에 설립한 자회사로서 EPC 코드와 EPCglobal Network의 전 세계 보급을 총괄하고 있는 국제 민간 기구이다. EAN과 UCC가 Auto-ID 센터를 흡수하여 설립한 비영리 기구로서 Auto-ID에서 개발한 EPC 네트워크에 대한 기술의 표준화, 상용화, 그리고 EPC 코드 보급과 관리 등을 목적으로 활동하고 있다[9, 11].

EPCglobal은 RFID 미들웨어의 표준화를 주도하고 있으며, 소프트웨어 관련 산업체에서는 이를 참조 모델로 솔루션을 개발하고 있다. EPCglobal은 RFID 태그로부터 외부 애플리케이션에 이르기까지의 구성 요소를 계층 구조로 표

현하고 각 계층 간의 인터페이스를 표준화 대상으로 삼고 있다. EPCglobal은 RFID 미들웨어와 관련하여 2002년에 구현 스펙 중심의 Savant Version 0.1과 2003년에 Savant Version 1.0를 제안하였고, 2005년 11월 다양한 센서로부터 EPC 코드 데이터를 받아 필터링하고 통합하여 응용에게 제공하는 미들웨어의 인터페이스에 관한 ALE 스펙 버전 1.0을 발표하였다[12, 15].

2) EPCglobal 네트워크 아키텍처

그림 1은 EPCglobal 네트워크 아키텍처를 나타낸 것으로 EPC 네트워크 아키텍처에서 녹색바는 EPCglobal 표준의 영향을 받는 인터페이스를 나타내며 반면에 청색 박스는 시스템의 하드웨어 또는 소프트웨어 및 컴포넌트의 역할을 나타낸다.

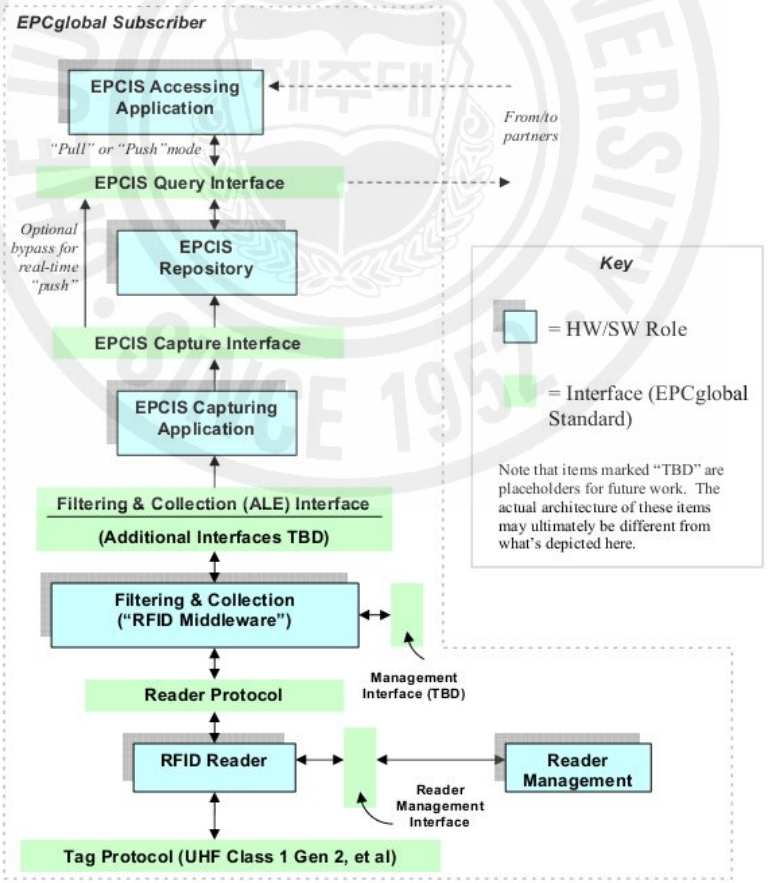

그림 1. EPCglobal 프레임워크 구조도

다음은 몇 가지 컴포넌트의 역할과 인터페이스에 관한 내용을 나타내면 아래와 같다.

(1) 리더(Reader)

태그가 판독 영역에 있을 때 RFID 태그를 계속 관찰한다.

(2) 리더 프로토콜 인터페이스(reader protocol interface)

리더에서 필터링과 콜렉션에 이르는 미처리 태그 판독의 제어 및 전송에 대해 정의한다. 이 부분의 인터페이스 이벤트를 “A 리더가 T 시간에 X의 EPC를 판독했다”라고 한다.

(3) 필터링과 콜렉션(filtering과 collection)

이 역할은 EPCIS 캡쳐 애플리케이션(예, 모션 감지기 작동)이 정의한 이벤트가 정해진 시간 간격마다 미처리 태그 자료를 필터링하고 수집한다.

(4) 필터링과 콜렉션 인터페이스(ALE)

필터링과 콜렉션에서 EPCIS 캡쳐 애플리케이션에 이르는 필터링되고 수집된 태그 판독 데이터의 제어 및 전송에 대하여 정의한다. 이 부분의 인터페이스 이벤트를 T1, T2 시간 사이에 위치 L에서 다음 EPC가 관찰됐다“라고 하며 EPC 목록은 사본이 없으며 EPCIS 캡쳐 애플리케이션에서 정의한 기준에 따라 필터링 된다.

즉 여기에서 ALE 계층은 데이터 수집하고 필터링하여 비즈니스 로직이 해석을 시작하도록 하는 의미 있는 이벤트를 생성하는데 관심이 있는 반면 비즈니스 계층은 비즈니스 처리 및 이벤트 저장에 관심이 있다. 저장된 이벤트는 이후 다양한 정보 처리를 위해 사용될 수 있다. 그러기 위해서는 원시 EPC 데이터를 획득하는 하부 구조 모듈과 그 데이터를 필터링하고 카운팅하는 구조적 모듈, 그리고 데이터를 사용하는 응용 간의 표준 인터페이스를 제공하여 독립성을 보장하여 미들웨어 기술 제공자와 수집에 의해 얻어진 이벤트 데이터를 사용하는 사용자에게 상당한 이점을 제공하는 ALE 계층이 강력히 필요하다[19].

3. ALE(Application Level Events)

ALE는 EPC 처리 시스템에서 RFID 태그를 인식한 리더가 애플리케이션 계층으로 데이터를 전달하는 역할을 하는 미들웨어이다. 분산 컴퓨터 시스템에서 미들웨어는 오퍼레이팅 시스템과 애플리케이션 시스템 사이에 서로간의 데이터를 중개해 주는 역할을 하는 소프트웨어 레이어로 정의된다. 미들웨어의 정의에 비춰 생각해 보면 태그를 읽어 들이는 리더에서 발생한 데이터를 EPCIS와 같은 EPC 애플리케이션으로 전달하는 역할을 하는 것이다. 즉 RFID 리더 등 EPC 데이터 소스에서 애플리케이션의 이벤트로 이동하는 데이터의 양을 줄이는 프로세싱이 필요하며 물리적 계층에서 발생한 데이터를 애플리케이션 계층에서 분리하는 것이 시스템의 유연성 측면에서 유리하다는 생각에서 개발된 것이다.

1) ALE의 동작 단위

ALE에서는 리드 사이클(read cycle), 이벤트 사이클(event cycle), 리포트(report)의 크게 3가지의 단위를 사용하여 실행된다.

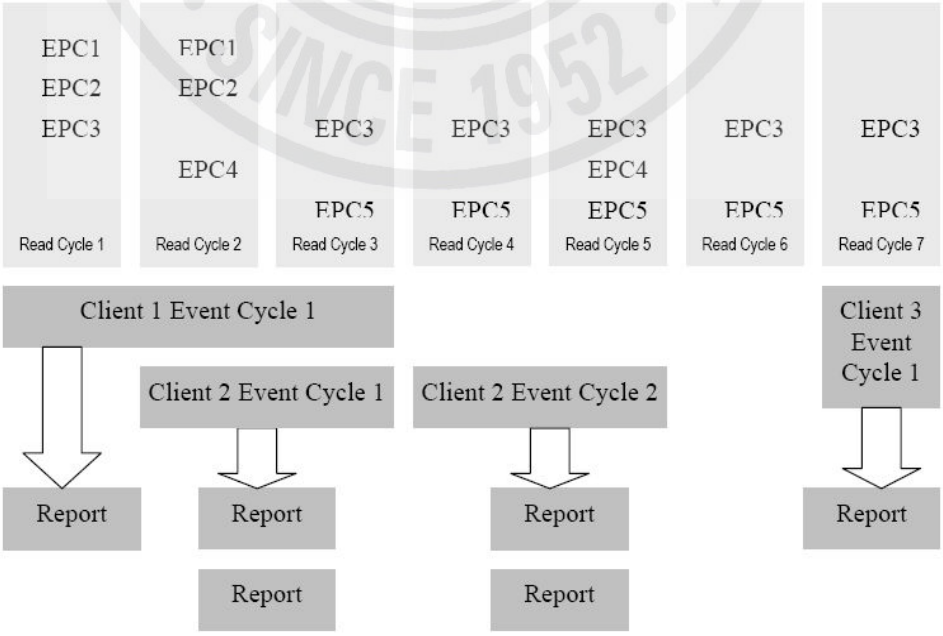

그림 2. ALE 동작 다이어그램

(1) 리드 사이클

리더와 상호작용하는 최소 단위(unit)이다. 가공되지 않은 태그 리드 레이어(raw tag read layer)와 ALE 레이어간의 인터페이스에서 리드 사이클의 출력은 ALE 레이어의 입력이 된다. ALE 관점에서는 리드 사이클은 하나의 EPC 세트만을 보유한 1회 이벤트이다.

(2) 이벤트 사이클

1개 이상의 리더로부터 수집된 1개 이상의 리드 사이클이며, 응용 관점에서 한 단위로 취급된다. ALE 인터페이스와 응용 간 상호작용의 최소 단위다.

(3) 리포트

ALE 레이어에서 응용으로 전달되는 이벤트 사이클에 관한 데이터이다. 리포트는 ALE 레이어의 출력으로 애플리케이션 비즈니스 로직 레이어로 전달된다.

그림 2는 리드 사이클, 이벤트 사이클, 리포트의 관계를 설명하고 있다. 한 개 리더에서 발생하는 리드 사이클을 나타내지만 실제로는 해당 이벤트 사이클은 한 개 이상 리더에서 발생하는 리드 사이클을 수집한다.

그림에서 표시된 것처럼, 어느 시점에서나 동작(active)하는 이벤트 사이클이 한 개 이상 있다. 다수의 이벤트 사이클은 다른 리드 사이클에 따라 시작 또는 종료되고 임의적으로 교차된다. 이벤트 사이클은 동일 리더, 다른 리더 또는 리더의 임의 교차 부분으로부터 데이터를 수집한다. 여러 동작 이벤트 사이클은 동시에 여러 요청을 하는 응용이나 독립된 응용에서 발생한다. 그러나 모든 경우에서 있어서, 동일 리드 사이클은 해당 리더로부터 데이터를 요청한 모든 동작 이벤트 사이클이 공유한다.

해당 리더기에서 수집된 해당 리드 사이클의 EPC 집합을 S로 표시하면 위 그림에서는 EPC 집합은 아래와 같다.

S_1= {EPC1, EPC2, EPC3}, S_2={EPC1, EPC2, EPC4}, S_3={EPC3, EPC5},...

응용은 1개 이벤트 사이클을 하나의 단위로 취급하며 응용은 이벤트 사이클 내부 구조를 알 수 없고 알 필요도 없다. 응용은 이벤트를 받아올 리더

를 설정하게 되는데 이벤트 사이클은 해당 리더의 리드 사이클을 수집한다. 수집된 리드 사이클들에서 중복된 것은 제거하여 이벤트 사이클을 생성한다. 즉, 수집된 리드 사이클의 합집합을 이벤트 사이클로 간주하면 된다. 그림3에서 응용 1 이벤트 사이클1은 아래와 같이 나타낼 수 있다.

$E1.1 = S_1 \cup S_2 \cup S_3 = \{EPC1, EPC2, EPC3, EPC4, EPC5\}$

응용은 리포트를 통해 해당 이벤트 사이클 정보를 얻는다. 리포트는 다음 세 가지 매개 변수의 조합으로 지정된다.

(4) 리포트 할 집합 R 설정

현재 리드 사이클들의 합집합으로 표현되는 이벤트 사이클의 집합

$\mathbf{R = E_{current}}$

이전 이벤트 사이클에 대한 현재 이벤트 사이클의 차이만 포함하고 있는 차집합

추가집합 : $\mathbf{R = E_{current} - E_{previous}}$

삭제집합 : $\mathbf{R = E_{previous} - E_{current}}$

(5) 필터 F(R) 옵션 설정

EPC Tag Data Standard 1.3의 필터 비트를 기준으로 한 팔레트, 케이스와 같은 제품객체와 창고, 트럭, 소매 진열대와 같은 위치객체를 필터 옵션으로 사용한다.

(6) 리포트 대상

F(R) 집합의 원소이며 표현 포맷은 아이덴티티 URI, 태그 URI, 2진수 등에서 선택 집합 F(R)의 원소의 수

즉, ALE의 동작은 합집합과 차집합으로 이루어진다. 이에 대한 구현은 다양한 방법이 존재할 수 있지만 응용은 제공된 ALE API를 바탕으로 적합한 형태의 리포트를 요구하기만하면 되고 내부의 구현은 알 필요가 없다. 이는 ALE를 제작하는 제작자의 몫이다.

2) 논리 리더(logical reader)

논리 리더는 응용의 사용 용도나 목적에 맞게 쉽게 이해할 수 있는 이름을

리더나 리더 그룹에 붙일 수 있다는 추상적인 개념이다. 예를 들어 그림3와 같이 42번문에는 물리 리더 3개를 묶어 DockDoor42라는 이름의 논리 리더로 관리할 수 있다. 이 경우 응용은 리더의 물리적 아이덴티티를 알 필요 없이 단지 DockDoor42에 대한 ALE의 이벤트 사이클 스펙을 요청하게 되고 ALE는 해당하는 3개의 물리적 리더에 대한 리포팅으로 처리한다. 물리적 리더에서 제공되는 EPC 이벤트 소스들은 논리 리더 네임으로 다양하게 맵핑되어 필요한 데이터를 응용에게 제공한다.

Logical Reader Name	Physical Reader Devices		
	Reader Name	Antenna	Protocol
DockDoor42	Acme42926	0	UHF
	Acme42926	1	UHF
	Acme43629	0	UHF
DockDoor43	Acme44926	0	UHF
	Acme44926	1	UHF
	Acme49256	0	UHF

그림 3. 논리 리더 네임에 따른 물리적 리더 디바이스

3) ALE API

ALE에서 중요하게 고려되는 인터페이스에 대하여 알아보자.

```
---
define(specName:string, spec:ECSpec) : void
undefine(specName:string) : void
getECSpec(specName:string) : ECSpec
getECSpecNames() : List   // returns a list of specNames as
strings
subscribe(specName:string, notificationURI:string) : void
unsubscribe(specName:string, notificationURI:string) : void
poll(specName:string) : ECReports
immediate(spec:ECSpec) : ECReports
getSubscribers(specName:String) : List // of notification
URIs
getStandardVersion() : string
getVendorVersion() : string
<<extension point>>
```

그림 4. ALE Main API

(1) ALE Main API

그림 4는 ALE Main API로 응용에게 제공되는 ALE Interface 메소드를 나타낸 것이다. 응용은 그림 5의 메소드를 이용하여 서비스 등록, 서비스 요청, 서비스 해지 등 적절한 요청을 할 수 있다.

(2) ALE API와 관련된 주 데이터 타입

그림 5은 ECSpec 데이터 타입을 uml 표기법으로 나타낸 것으로 이벤트 사이클에 포함되는 리드 사이클을 생성하는 논리 리더 목록과 이벤트 사이클의 경계가 어떻게 결정되는지에 대한 boundary에 대한 스펙과 이벤트 사이클에서 생성되는 리포트를 기술하는 각 스펙 목록이 있다.

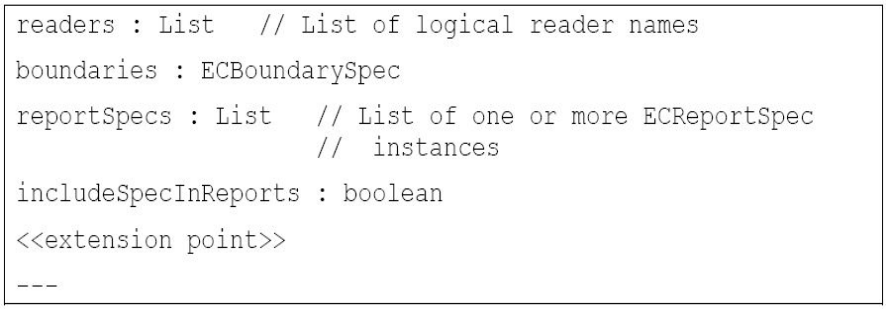

그림 5. ECSpec

그림 6은 ECReportSpec 데이터 타입을 uml 표기법으로 나타낸 것으로 이벤트 사이클 수행에서 얻어진 데이터를 반환할 리포트를 지정하는 데이터 타입으로 이벤트 사이클로 얻어진 데이터를 가공 처리할 조건을 기술하는 스펙이다.

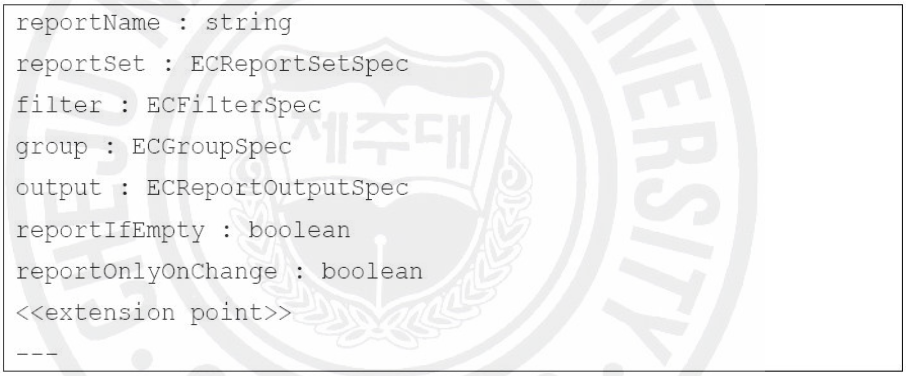

그림 6. ECReportSpec

그림 7은 이벤트 사이클의 시작과 종료를 지정하는 ECBoundarySpec으로 repeatPeriod는 이벤트 사이클의 반족 주기를 의미하고 startTrigger는 이벤트의 시작되는 조건을 의미한다. 이 둘은 상호 배타적이어서 둘 중에 하나만 사용해야한다. 이벤트 종료 조건으로 stableSetInterval, duration, stopTrigger의 3가지 경우와 ECSpec이 요청해지(UnSubsrcribe) 상태로 변경될 경우이다. 단 ECSpec의 요청해지 상태로 변경은 ECBoundarySpec에서는 명시항목은 아니다.

```
startTrigger : ECTrigger
repeatPeriod : ECTime
stopTrigger : ECTrigger
duration : ECTime
stableSetInterval : ECTime
<<extension point>>
---
```

그림 7. ECBoundarySpec

ECReports는 이벤트 사이클 동안 얻어진 가공된 결과 데이터 타입으로 그림 8과 같이 각 필드로 구성되어 있다.

```
specName : string
date : dateTime
ALEID : string
totalMilliseconds : long
terminationCondition : ECTerminationCondition
spec : ECSpec
reports : List   // List of ECReport
<<extension point>>
---
```

그림 8. ECReports

4. RFID 기반 미들웨어 제품 및 솔루션

Oracle의 Sensor Edge Server는 데이터 수집, 이벤트 처리, 데이터 분배(dispatching) 등의 기능을 지원하는 센서 기반 서비스 통합 플랫폼이다. 미리 정의된 필터들을 이용하여 응용이 원하는 데이터를 추출하며, 보다 복잡한 데이터를 처리하고자 할 경우 직접 로직 필터를 작성할 수 있다[7].

CapTech 사의 TagsWare는 RFID 태그 데이터를 응용에 전달하기 위한 링크, RFID 장비로의 표준 인터페이스를 지원하는 드라이버, 그리고 응용에서 링크와 드라이버의 사용을 지원하는 응용 기반 요소들로 구성된다. 리더 장치로부터 수집된

원시 데이터에서 태그 데이터를 추출하여 평활화(smoothing)한 후 응용에 전달하기 위한 링크 컴포넌트가 체인으로 연결된 구조를 제공한다[8].

SUN 사의 Java System RFID 소프트웨어는 다양한 센서로부터 오는 센서 데이터 스트림을 이벤트 관리기에서 처리하고, 리더 어댑터, 필터, 로거(logger), 엔터프라이즈 게이트웨이 등으로 구성된다. 델타(delta)와 평활화 질의를 할 수 있고, 필터들을 서로 연결하여 특정 마스크(mask) 조건을 만족하는 EPC 데이터 값을 얻을 수 있을 뿐만 아니라 사용자 정의 필터를 개발할 수 있는 도구를 지원한다[9].

ETRI의 UbiCore 시스템은 XML 기반 미들웨어 시스템으로 다양한 센서를 관리하며, XQueryStream이라는 XQuery에 기반한 연속 질의 언어를 제공한다. 필터링과 중간 결과 재사용을 통하여 스트림 데이터에 대한 질의 처리 속도를 개선하였고, 연속적으로 생성되어 들어오는 실시간 데이터뿐만 아니라 저장된 이력 데이터에 대한 질의를 지원하며 컨텍스트와 서비스의 연계 정보를 표현하기 위한 마크업 언어인 CSML(Context-driven Service Markup Language)를 제공한다[5].

이러한 RFID 미들웨어들은 다양한 기능 및 데이터 처리 방법을 제공함으로써 데이터 처리의 효율성 면에서 뛰어나다, 하지만 응용 서비스와 미들웨어 간 독립성을 제공하기 위한 사실상의 표준을 따르지 않음으로써 향후 다양한 외부 비즈니스 로직과의 상호 운영이 필요할 경우 서로 다른 제약으로 인한 문제점 발생으로 인하여 비용 및 융통성 면에서 개선의 여지가 있다. 또한 다수의 응용 시스템이 서비스를 요청시 서로 다른 이벤트 사이클 요청으로 큐의 핸들을 선점하기 때문에 발생하는 큐의 오버플로현상 또는 가장 오래된 데이터를 임으로 삭제하여 생기는 데이터 손실의 문제점들이 있다.

5. 고려사항

RFID 미들웨어를 설계할 때 일반적인 고려사항은 리더 인터페이스, EPC 코드 처리 및 여러 가지 응용 플랫폼에 대해서도 상호 운영성을 보장 할 수 있어야 한다. 상호 운영성을 보장하는 미들웨어를 개방형 미들웨어라 하며, 이를 위하여 표준화된 코드 , 정보표현, 교환규약을 준수하고 정보교환을 위한 메세징 기술을 적용해야하는데 이를 위해서는 EPC Specification, ISO 표준, 웹서비스 등과 같은 참조 표현을 준용해야 한다.

본 논문에서 제안하는 미들웨어는 스펙을 정확하게 분석하여 애플리케이션과 비용 측면을 충족시키면서 가급적 아키텍처 하위 레벨에서 데이터 필터링 및 카운팅을 처리해야한다. 응용 요청에 대한 응답으로 리포트를 생성하는 축적. 필터링. 카운팅 작업 표준에 유연한 인터페이스를 제공하여 위 목표 달성이 용이하도록 설계하여야 한다.

ALE 인터페이스는 만들어진 응용 요청과 이에 상응하는 리포트 사이를 순환하는데 요청은 두 종류가 있으며, immediate, 이 경우 정보는 요청 시에 한 번씩 리포트 되며 recurring, 이 경우 정보는 이벤트가 감지될 때마다 또는 지정된 시간 간격 마다 반복적으로 정보가 리포트 되어야 한다. 요청에 대한 응답으로 리포트된 결과는 요청한 응용에게 직접 전달되거나 요청자가 지정한 제3자에게 전달될 수 있어야 하며, 데이터를 정확하게 필터링하고 적절한 위치로 라우팅 할 수 있음을 확인해야 한다. 애플리케이션 통합 분야에서 RFID 데이터가 기업의 레거시 시스템과 안정적으로 연동되는데 필요한 메시징, 라우팅, 연결 편의성을 제공해야 한다. 또한 RFID 기반 미들웨어 제품 및 솔루션에서 문제점으로 대두되었던 다수의 응용 시스템이 서비스를 요청시 서로 다른 이벤트 사이클 요청으로 큐의 핸들을 선점하기 때문에 발생하는 큐의 오버플로현상과 가장 오래된 데이터를 임으로 삭제하여 생기는 데이터 손실의 문제점을 고려해야한다.

III. ALE 기반 RFID 미들웨어 설계

1. 제안하는 시스템 개요

본 연구에서 제안하는 RFID 미들웨어는 다양한 RFID리더 장치로부터 들어오는 방대한 양의 노이즈(noise)데이터 또는 중복된 데이터이거나 불필요한 EPC 데이터를 응용이 사용하기 편리한 이벤트로 변환하여 비용성과 효율성을 극대화하고 상호 운용성을 높이기 위해 ALE Specification Version 1.0을 기반으로 설계하는데 RFID 미들웨어 시스템구조는 다양한 모바일리더기 및 EPC 호환 RFID 리더들과 연동할 수 있어야 하며 이것들로부터 수집된 데이터를 처리하기 위해 센서 관리와 연결 관리를 해야 한다.

이를 위해서는 다양한 데이터 포맷을 처리 할 수 있는 리더 인터페이스가 필요하고, 리더 인터페이스를 거쳐 생성된 논리리더로 들어오는 EPC데이터를 동적으로 생성되는 이벤트 큐에 연결하여 연속적으로 밀려오는 데이터스트림을 끊김 없이 개별 처리할 수 있게 하기 위한 분배기술을 담당하는 브로드 캐스팅 처리 블록이 필요하다. 또한 미들웨어 측면에서 응용에게 ALE 인터페이스를 제공하여 쉽게 서비스를 요청 할 수 있게 하기 위한 ALE 인터페이스 블록이 필요하고, 그런 서비스 요청을 받았을 때 응용에서 제출한 요구 스펙을 분석하고 그 스펙을 이용하여 이벤트 사이클을 수행하여 얻은 결과를 필터링, 그룹핑, 레포트셋 등 다양한 처리를 지원하는 블록이 필요하다.

본 논문에서 제안하는 RFID 미들웨어 시스템 구성은 크게 리더 인터페이스 처리 블록, 브로드캐스팅 처리 블록, ALE 인터페이스, ALE CORE 처리 블록으로 구성하였다.

그림 9는 제안하는 전체 시스템 구조도이다.

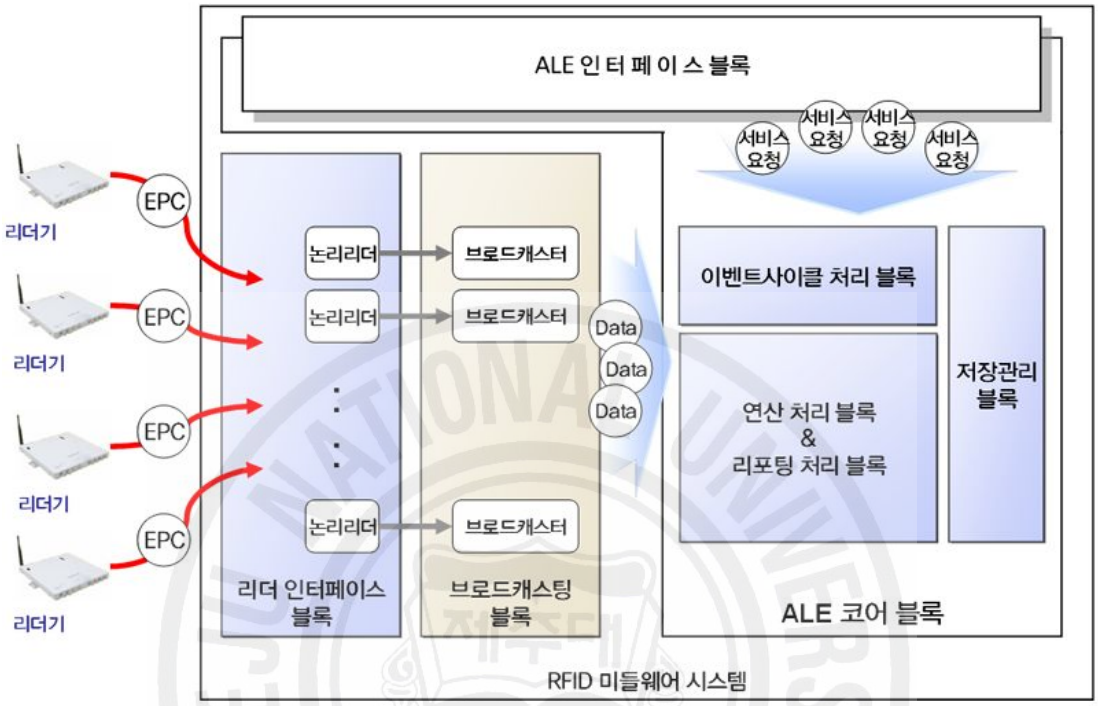

그림 9. ALE 기반 RFID 미들웨어 구조도

본 논문에서 제안한 미들웨어 특징은 다음과 같이 요약된다.

1) 다양한 리더 연동

리더 인터페이스는 Alien 사의 ALR-9640과 ALR-9780, ALR-9750 그리고 EPC 호환 RFID 센서들과 연동할 수 있으며 리더 인터페이스는 다양한 리더기로부터 센싱된 XML 스트림 데이터를 처리하기 위해 센서 관리와 연결 관리를 하고 연속적으로 미들웨어로 들어오는 XML로 된 EPC 데이터 스트림을 하나 혹은 여러 개의 논리 리더로 생성하고 관리하는 일을 담당한다[18].

2) 센서 데이터 스트림 실시간 처리

XML Schema를 이용한 센서 데이터 스트림 구조로 데이터의 실시간 스트림 처리를 위한 데이터 브로드캐스팅(분배) 기술을 지원한다.

3) ALE 인터페이스 제공

ALE 표준 인터페이스를 응용 개발자에게 제공하여 다양한 응용서비스가 RFID 미들웨어를 사용할 수 있도록 지원한다.

4) ALE Core 처리 기능

ECSpec 분석, 이벤트 사이클 처리, 필터링, 그룹핑, 리포트 전송 등 ALE 스펙에 명시된 항목들에 관한 처리를 할 수 있다.

5) 다양한 코드 형식 제공

응용 개발자가 원하는 다양한 코드 형식을 제공한다.

예) EPC, Tag, RawHex, RawDecimal

2. ALE Core의 동작 방식 및 데이터 처리 과정

본 논문에서 제안한 RFID 미들웨어의 데이터 처리는 크게 전처리 단계와 후처리 단계로 나눌 수 있는데 전처리 단계에서는 RFID 리더기로부터 들어오는 원시 데이터소스를 처리 하고, 후처리 단계에서는 ALE 인터페이스를 이용하여 응용이 서비스를 요청할 경우 그 요구사항에 맞게 데이터를 처리하는 과정을 의미한다. 본 절에서는 전처리 단계가 진행되고 있는 상황에 후처리에서 진행되는 작업을 중점으로 언급한다.

ALE 스펙을 따르는 미들웨어는 그림 10과 그림 11, 그림12와 같이 일련의 상황별로 동작되고 이것을 처리해야 한다.

1) ALE Core의 동작 방식

(1) 직접받기(Direct Subscription)

그림 10과 같이 직접받기의 상황일 경우 응용1은 ALE 인터페이스의 define 메소드를 호출하여 파라미터로 ECSpec과 해당 ECSpec을 호출할 수 있도록 스펙이름을 인자로 넘긴다. 응용1은 subscribe 메소드를 호출하는데 사용할

스펙이름과 결과를 리턴 받을 수 있는 URI를 인자로 넘긴다. 이때 ECSpec은 unrequest 상태에서 request 상태로 전이하고 시작조건이repeatPeriod 경우 ECSpec은 active 상태로 전이하여 이벤트 사이클이 시작된다.

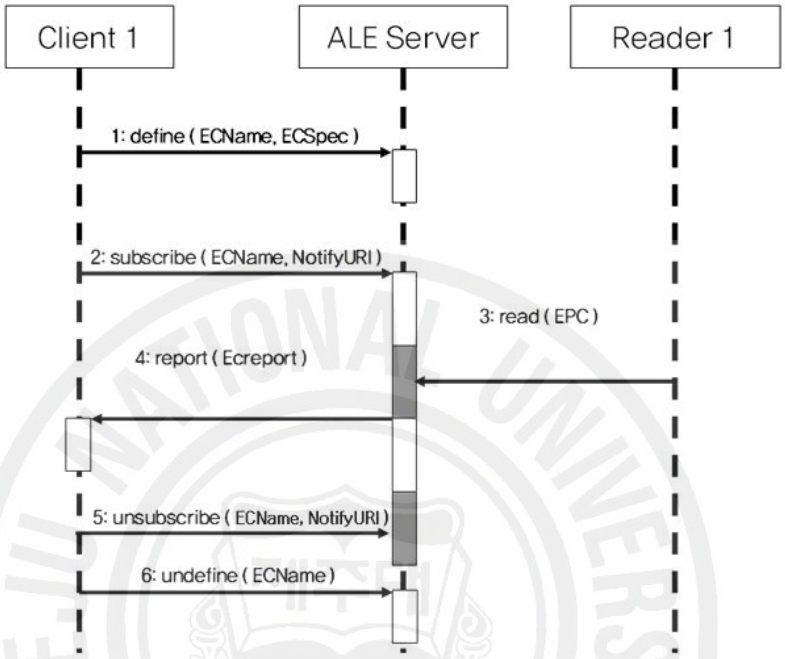

그림 10. 직접받기

즉, 응용이 필요한 EPC 데이터를 수집, 가공처리를 한다. unsubscribe 메소드가 호출되지 않으면 주기적으로 repeatPeriod는 계속 실행되고 그 주기 동안 ECSpec에 지정된 필터 조건을 충족시키지 않으면 리포트 되지 않고 (ECReports가 생성되지 않음) 충족시키면 repeatPeriod 종료시 요청한 ECReports가 생성되어 응용으로 전달된다.

(2) 간접받기(Indirect Subscription)

그림 11은 응용이 그림 10과 같이 define 메소드를 호출하는데 ECSpec에 시작조건과 종료조건이 trigger로 작성된 경우로subscribe 메소도를 호출했을 경우 ECSpec은 request 상태로 되고 startTrigger를 발생했을 때 active 상태가 된다. stopTrigger를 수신할 경우 ECSpec은 active 상태에서 request 상태로 전이되고 ECReports가 생성되어 응용에게 비동기식으로 전달된다.

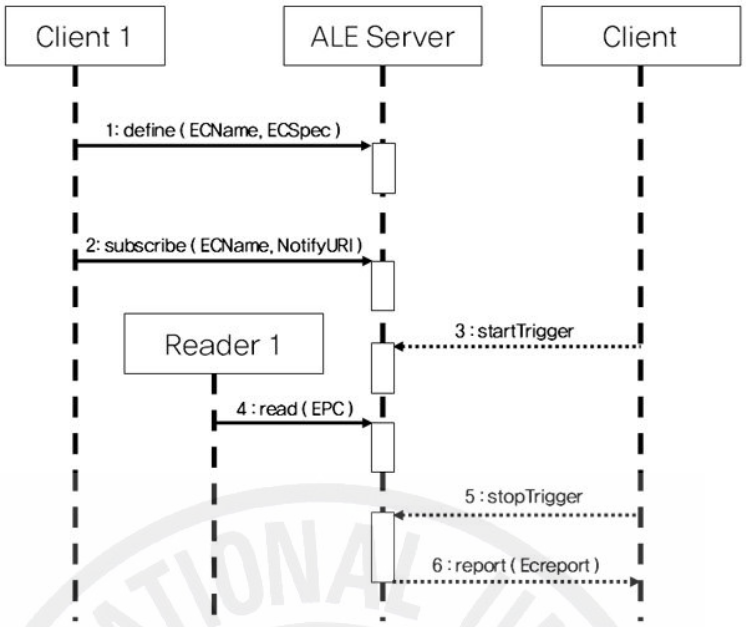

그림 11. 간접받기

(3) Poll과 Immediate

그림 12는 poll과 immediate를 호출하는 경우로 poll은 define 메소드를 호출하여 poll을 호출한다. 그림 12에서는 ECSpec의 종료조건은 duration이 지정 되어 있을 경우 poll 메소드가 호출되면 ECSpec은 request 상태에서 active 상태로 전이되고 duration 시간동안 이벤트 사이클은 수행된다.

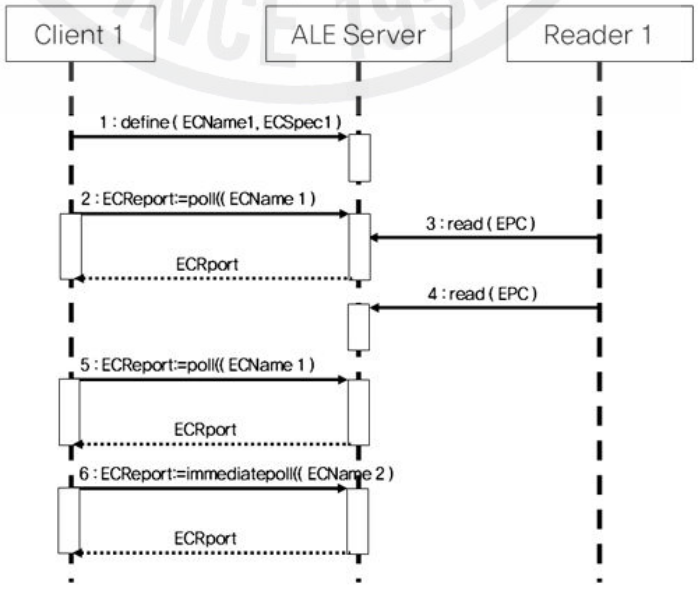

그림 12. poll. immediate

그림 10과 다른 점은 그림 12의 경우 한 번의 이벤트 사이클을 수행한다는 점이다. 다시 이벤트 사이클을 수행하려면 poll 또는 immediate 메소드를 다시 호출해야한다. 또한 메시지 전달 방신은 동기식 방식으로 처리한다.

immediate 메소드는 define 된 스펙을 사용하지 않고 메소드 호출시 ECSpec을 직접 전달한다는 점이 다르다. 동작 방식은 poll과 같다.

2) ALE Core 내부 데이터 처리 흐름도

ALE Core 내부 데이터 처리 흐름은 그림 10, 그림 11 그리고 그림 12와 같은 동작 방식을 이용하여 응용이 ECSpec을 작성한 후 define 인터페이스 메소드의 파라미터로 미들웨어 시스템에 등록한다. 이때 ECSpec은 스펙을 관리하는 스펙 리스트 자료구조에 등록된 후 예상치 못한 시스템의 다운 및 이상이 발생할 경우에 대비하기 위하여 일정한 시간 간격으로 백업 모듈에 의하여 데이터베이스에 저장된다.

ECSpec의 XML 인스턴스의 예는 아래와 같다.

```
<?xml version="1.0" encoding="UTF-8"?>
<ale:ECSpec xmlns:ale="urn:epcglobal:ale:xsd:1"
      xmlns:epcglobal="urn:epcglobal:xsd:1"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:epcglobal:ale:xsd:1 Ale.xsd"
      shcemaVersion="1.0"
      creationDate="2003-08-06T10:54:06.444-05:00"
      includeSpecInReports = "true">
  <logicalReaders>
    <logicalReader>FRONT_DOOR</logicalReader>
    <logicalReader>BACK_DOOR</logicalReader>
  </logicalReaders>
  <boundarySpec>
    <startTrigger>http://218.49.163.218/eventcycle?specname=Spec_A1C_TSU!triggertype=
                start!methodtype=su!methoduri=218.49.163.218:5500</startTrigger>
    <stopTrigger>http://218.49.163.218/eventcycle?specname=Spec_A1C_TSU!triggertype=
                stop!methodtype=su!methoduri=218.49.163.218:5500</stopTrigger>
  </boundarySpec>
```

```
  <reportSpecs>
    <reportSpec reportName="Spec_A1C_T"
                              reportIfEmpty="true" reportOnlyOnChange="true">
      <reportSet set="CURRENT"/>
      <filterSpec>
        <includePatterns>
          <includePattern>urn:epc:tag:sgtin-64:1.1.13.*</includePattern>
          <includePattern>urn:epc:tag:sgtin-64:1.1.9.*</includePattern>
          <includePattern>urn:epc:tag:sgtin-64:1.1.10.*</includePattern>
          <includePattern>urn:epc:tag:sgtin-64:1.1.20.30</includePattern>
        </includePatterns>
        <excludePatterns>
          <excludePattern>urn:epc:tag:sgtin-64:1.1.13.21</excludePattern>
        </excludePatterns>
      </filterSpec>
      <groupSpec>
        <pattern>urn:epc:tag:sgtin-64:1.1.9.*</pattern>
        <pattern>urn:epc:tag:sgtin-64:1.1.10.*</pattern>
        <pattern>urn:epc:tag:sgtin-64:1.1.18.*</pattern>
      </groupSpec>
      <output includeCount="true" includeEPC="true" includeTag="false"
                              includeRawHex="false" includeRawDecimal="false"/>
    </reportSpec>
  </reportSpecs>
</ale:ECSpec>
```

응용은 subscribe와 poll 메소드를 이용하여 시스템에 등록되어 있는 ECSpec을 지정함으로써 원하는 데이터를 요청할 수 있다. subscribe 메소드를 호출하여 데이터를 요청하면 해당 ECSpec의 상태가 SpecStatesList 라는 자료구조에 등록된다. 그런 다음 논리리더로부터 들어오는 데이터를 저장하기 위한 이벤트 큐를 생성하고 이벤트 사이클의 시작 조건과 종료 조건에 의해 얻어진 중복 제거된 EPC 데이터를 수집하여 필터링, 그룹핑, 데이터 셋, 전송할 데이터 유무 체크 등의 처리 과정을 수행한 후 지정된 데이터 형식으로 변환하여 최종 결과물인 리포트 객체를 만들어 응용으로 전송한다.

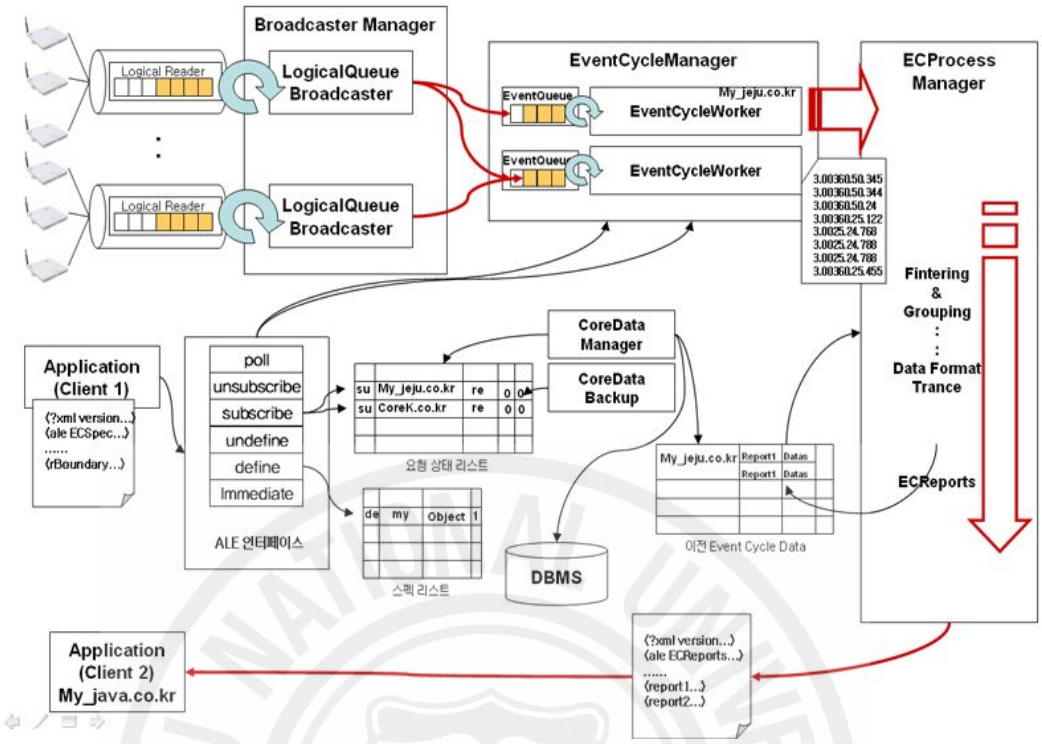

그림 13. ALE Core 내부 데이터 처리 흐름도

3) ALE Core에서 처리되는 데이터의 예

RFID 리더기에서 보내는 데이터는 다음과 같은 형식으로 리더기 제조사별로 포맷은 상이하다.

<STX>NaruSmart001,192.168.1.200,4000,20070918103045,02,000001,EF045942B296AB00<TAG>302C000000000001DDFA9972</TAG><END>

논리리더에 적재되어있는 데이터는 다음과 같이 중복된 데이터를 포함하고 있다.

urn:epc:tag:sgtin-96:1.3.0.0.8030478453
urn:epc:tag:sgtin-96:1.3.0.0.8031648354
urn:epc:tag:sgtin-96:1.3.0.0.8030464293 <--중복된 데이터
urn:epc:tag:sgtin-96:1.3.0.0.8030464293 <--중복된 데이터
urn:epc:tag:sgtin-64:1.1.13.24
urn:epc:tag:sgtin-64:1.1.9.1
urn:epc:tag:sgtin-96:1.3.0.0.8030464293 <--중복된 데이터

논리리더에 적재되어 있는 중복을 포함한 데이터들은 브로드캐스팅 블록을 통

해 응용이 요청한 데이터를 처리하기 위한 이벤트큐로 분배되는데 이벤트큐에 적재되는 데이터는 아래와 같다.

urn:epc:tag:sgtin-96:1.3.0.0.8030478453
urn:epc:tag:sgtin-96:1.3.0.0.8031648354
urn:epc:tag:sgtin-96:1.3.0.0.8030464293 **<--중복 제거된 데이터**
urn:epc:tag:sgtin-64:1.1.13.24
urn:epc:tag:sgtin-64:1.1.9.1

이벤트큐에 적재된 데이터들은 응용이 작성한 ECSpec을 참조하여 필터링, 그룹핑 등 각종 연산처리를 거쳐 알맞은 이벤트데이터를 생성한다.

다음은 레포트로 보내질 데이터의 예이다.

<tag>**urn:epc:tag:sgtin-96:1.3.0.0.8030464293**</tag>
또는 <epc>urn:epc:id:sgtin:1.13.26</epc>

3. 데이터 처리를 위한 주요 모듈 설계

그림 13과 같이 ALE Core에서 데이터 처리를 위해 설계할 모듈들에 대하여 기술한다.

1) 브로드 캐스팅 블록

브로드캐스트 관리자(broadcast manager) 와 브로드캐스터(broadcaster)는 ALE Core 전처리 단계로 그림 9의 시스템 구성도에서 브로드캐스팅 처리 블록에 해당하는 모듈중의 하나로 응용에서 사용할 논리 리더(원하는 데이터가 있는 곳)를 동적으로 생성되는 이벤트 큐에 연결하여 처리할 수 있게 하기 위한 것으로 브로드캐스트 관리자는 큐맵에서 해당하는 논리 큐를 하나씩 꺼내고 큐에 대응하는 브로드캐스터 객체를 생성하고 브로드캐스터 객체를 관리하기 위해 맵을 생성하고 관리한다. 브로드캐스터는 연결된 논리 리더에서 사용자가 필요로 하는 이벤트 큐로 메시지를 보내는 목적이 있다.

브로드캐스트 처리 블록의 동작 방식은 그림 14와 같이 논리 큐에서 데이터 가져온 후 브로드캐스터에 이벤트 큐의 인큐 핸들이 바인딩 되면 해당 이벤트 큐로 데이터 전송한다.

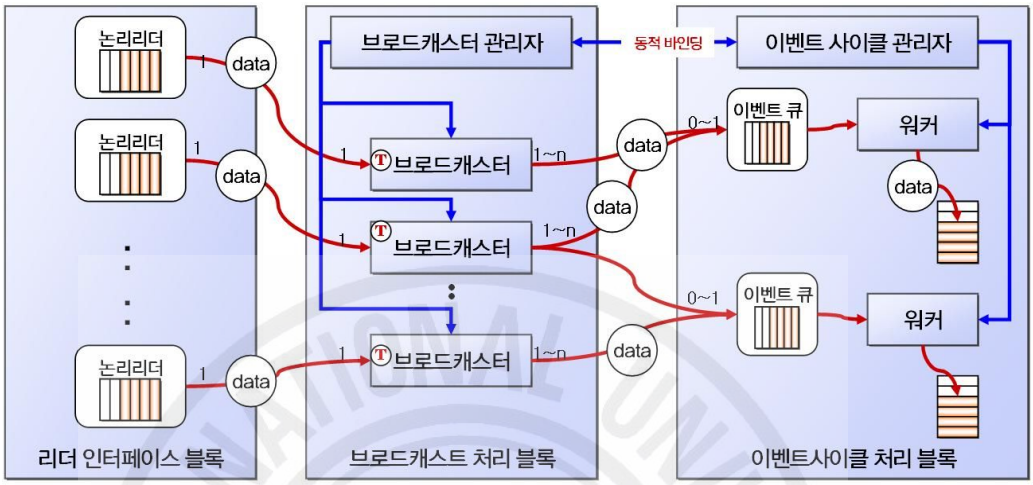

그림 14. 브로드캐스트 처리 블록 데이터 흐름도

2) 이벤트 사이클 처리블록

(1) 이벤트 사이클 관리자(eventcycle manager)

이벤트 사이클 관리자는 그림 9의 시스템 구성도에서 ALE Core 블록중 이벤트 사이클 처리 블록에 해당한다.

그림 13과 같이 응용이 subscribe, poll, immediate, unsubscribe등 RFID 미들웨어에게 서비스를 요청하면 이벤트 사이클 워커를 할당하고, 이벤트 데이터를 처리하기 위해 이벤트 큐를 생성한다.

생성한 큐와 이벤트 사이클 워커를 관리하기 위해 각각 큐맵과 해쉬 테이블로 저장 관리한다. 또한 이벤트 큐 생성시 인큐 핸들은 ALE Core 블록의 전처리 블록인 브로드캐스팅 처리 블록으로 전달하고, 디큐 핸들은 이벤트 사이클 워커에게 전달하여 데이터를 처리하게 한다.

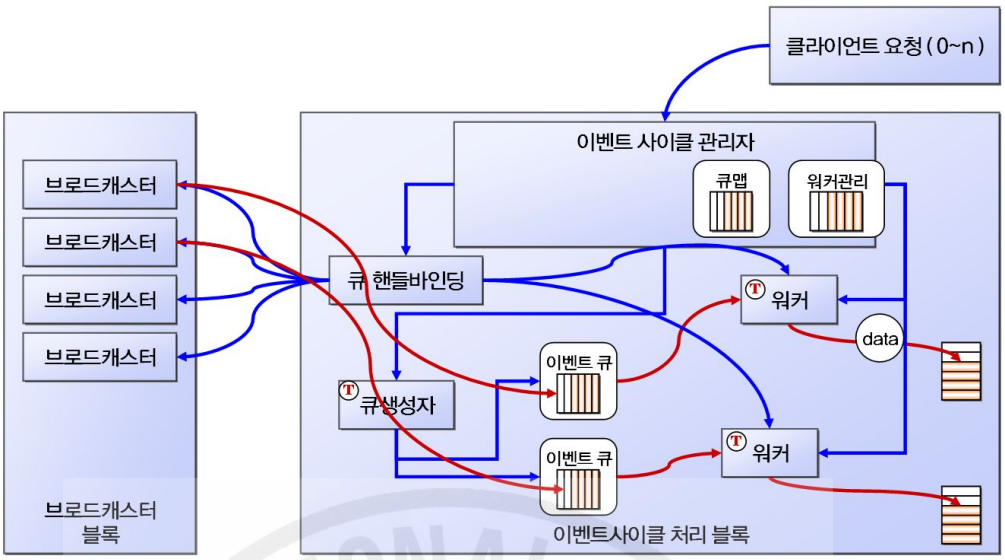

그림 15. 이벤트 사이클 처리블록의 데이터 흐름도

(2) 이벤트 사이클 워커

이벤트 사이클 워커는 ECSpec에 명시된 Boundary 조건을 이용하여 실제로 데이터를 가져오는 모듈로 논리리더 리스트를 얻어와 이벤트 큐에서 데이터를 가져온다. 가져온 데이터는 해쉬셋에 저장되고, EC프로세스 관리자(ECProcessManager)에게 해쉬셋 객체를 보내는 역할을 한다.

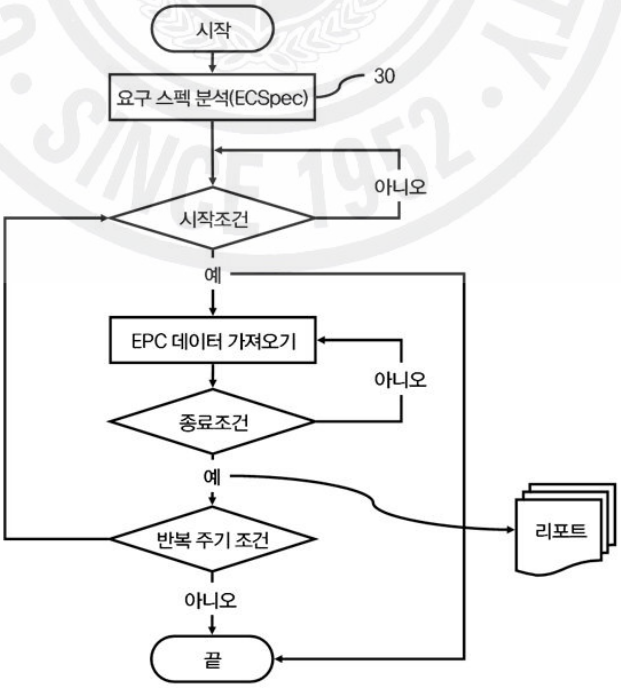

그림 16. 이벤트 사이클 처리 순서도

그림 16은 이벤트 사이클 처리 순서도를 나타낸 그림이다. 응용으로부터 제출된 ECSpec을 분석하여 이벤트 사이클의 시작과 끝의 조건을 추출하여 시작조건과 종료조건에 의해 한번 또는 주기적으로 이벤트 사이클을 수행한다.

3) 연산 처리 블록과 리포트 처리 블록

RFID 미들웨어의 연산 처리 블록은 이벤트 사이클 주기 동안 수집된 데이터를 바탕으로 사용자가 원하는 데이터 형태로 필터링과 그룹핑 등의 처리를 바탕으로 데이터를 정제하여 보다 양질의 RFID 태그 데이터를 응용으로 제공한다.

그림 17과 같이 RFID 미들웨어의 연산 처리 블록은 데이터 필터링(Filtering), 데이터 변경유무 검사(OnChange), 데이터 전송집합 구하기(ReportSet), 데이터 유무 검사(ReportIfEmpty), 이전데이터 보존(ChangePreData), 데이터 그룹핑(Grouping)의 기능을 수행하는 관리기로 구성된다.

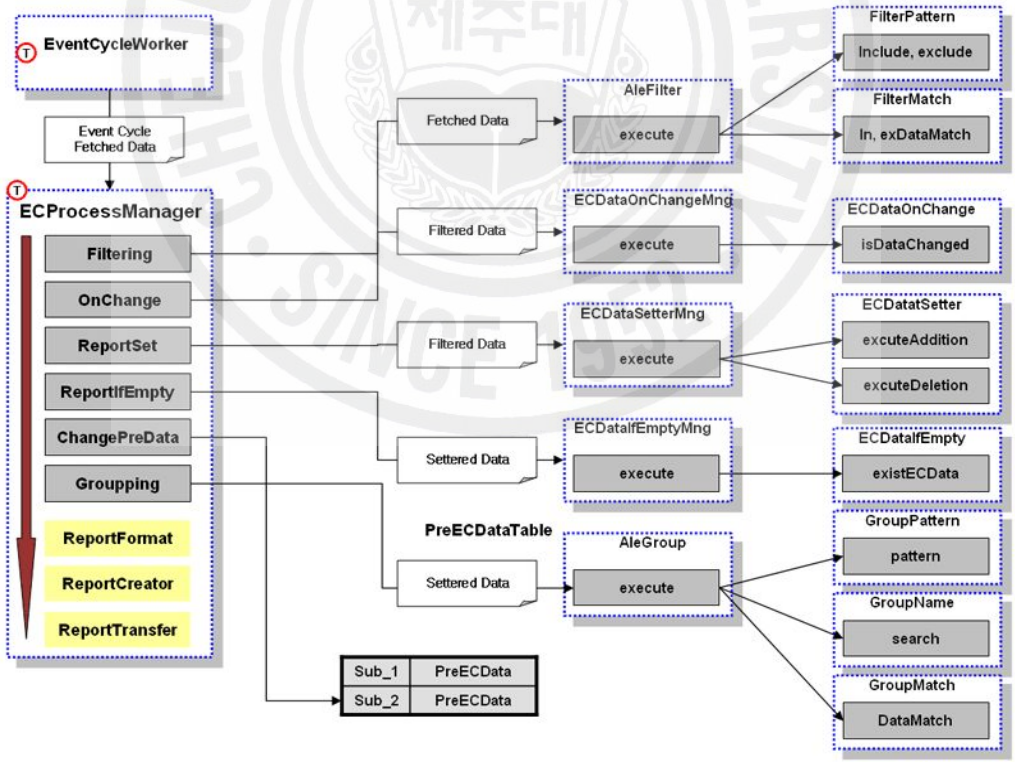

그림 17. 데이터 연산처리 블록

(1) 필터링

ECSpec에서 필터 조건을 추출하여 필터 패턴 관리기를 실행한다. 필터 패턴 관리기는 특정 조건에 포함되는 집합으로 필터링 할 것인지, 혹은 특정 조건에 의해 제외되는 집합으로 필터링 할 것인지를 분석하며 이는 패턴 리스트로 저장된다.

필터 처리기는 패턴 리스트 정보와 이벤트 사이클로 얻은 원시 데이터를 입력으로 연산 처리를 하는데 includePattern과 excludePattern이 동시에 있을 경우 excludePattern을 먼저 실행하고 includePattern을 나중에 실행하여 필터링을 효율적으로 처리한다.

(2) 데이터 변경유무 검사

데이터 변경유무 검사에서 하는 처리는 현재의 이벤트 사이클 데이터와 바로 이전의 이벤트 사이클 데이터를 비교하여 만약 두 데이터가 동일할 경우 응용이 두 사이클 동안 수집된 동일한 데이터를 중복적으로 받지 않아도 될 경우 유용하게 이를 사용자가 정한 true/false의 조건으로 변경된 새로운 데이터 집합만을 받게끔 할 수 있거나 동일한 자료라도 상관없이 계속 받을 수 있다. 데이터 변경유무 처리기는 위와 같은 처리를 통하여 변경유무 표시문자와 이전 데이터와 동일한 혹은 변경된 필터링 데이터를 출력한다.

(3) 데이터 전송집합 구하기

데이터 전송집합 관리기는 ECSpec, 이벤트 사이클 동안 얻은 데이터 처리 건수 정보가 있는 리포트 카운트, 변경유무 검사 관리기의 결과 데이터 그리고 이전 데이터 보관소에 저장된 이전데이터를 입력으로 데이터 전송 집합 구하기 처리기가 실행된다.

(4) 데이터 유무 검사

데이터유무 관리기는 ECSpec, 이벤트 사이클 동안 얻은 데이터 처리건수 정보가 있는 리포트 카운트, 데이터 전송집합 관리기를 통해 얻은 데이터를 입력으로 실행한다. 사용자가 정한 true/false 조건으로 이벤트 사이클 동안 얻은 데이터가 존재할 경우만 리포트를 할 수 있다. 즉, RFID 미들웨어가 응용으로 전송할 데이터가 없으면 미들웨어가 이하 부수적인 처리를 하지 않고

새로운 수집처리를 하게 되어 보다 효율적인 운영을 할 수 있다. 응용 역시 수집된 RFID 태그 데이터가 있을 경우에만 자료를 받아볼 수 있어서 보다 유용한 자료를 받을 수 있다. 그리고 전송할 데이터가 있을 경우와 데이터가 없을 경우 모두 처리가 가능하기 때문에 보다 유연하게 처리가 가능하다. 데이터유무 처리기는 위와 같은 처리를 통하여 전송유무 표시문자와 응용으로 전송할 데이터를 출력한다.

(5) 이전데이터 보존

이전 데이터 보존은 현재 이벤트 사이클에서 얻은 데이터를 저장하는 것으로 다음 이벤트 사이클 데이터와 비교할 경우 사용될 목적으로 저장 관리하는 데이터이다. 표 6은 이전 데이터 저장테이블 스키마로 스펙이름과 URI정보와 이벤트 사이클 데이터를 저장할 수 있다. 여기에 저장된 데이터는 데이터 변경유무 검사와 데이터 전송집합 구하기 관리기에서 활용한다.

표 6. 이전 데이터 저장테이블

KIND	SPEC_NAME	URL	ECDATA
C/A/D	test_spec	#203.253.….…	urn:epc:tag:……

(6) 데이터 그룹핑

그룹핑 관리기는 ECSpec, 이벤트 사이클 동안 얻은 데이터 처리건수 정보가 있는 리포트 카운트, 상위 데이터 전송집합 관리기에서 생성된 가공된 데이터 입력으로 실행된다. 그룹핑 패턴 관리기에서는 ECSpec에서 그룹핑 조건 추출하여 패턴 리스트로 저장하고 필터링과 유사한 처리를 보이나, 필터링과는 다르게 데이터 매칭 전에 해당 그룹 조건에 의해 전송할 데이터에서 그룹 이름을 추출한다. 그룹핑 처리기는 생성된 그룹 이름 리스트와 전송할 데이터를 입력으로 사용자가 원하는 데이터 그룹으로 매칭 처리를 수행하여 데이터를 생성한다. 또한 응용으로 리포트를 전송하기 위한 최종 형태인 ECReports 리스트를 생성한다.

(7) 데이터 포맷 변환

데이터 포맷 변환 관리기는 실제 모든 연산처리가 끝난 뒤 얻어진 데이터를 사용자가 받고자 하는 코드 형식으로 변환 처리 하는 과정으로 하나 또는 미들웨어에서 지원하는 수만큼 동시에 처리할 수 있다. 현재 제공하는 형식은 데이터 카운트, RawHex, RawDecimal, 태그, EPC 등이 있다.

(8) 리포트 생성과 전송

리포트 생성기는 필터링에서 RFID 포맷 변환까지 처리된 최종 자료인 최종 전송 리스트를 바탕으로 응용이 수신 받게 되는 ECReports를 생성하며, 이는 XML를 기반으로 처리된다. 아홉 번째로 리포트 전송기는 리포트 생성기에서 생성한 ECReports를 자료가 필요한 응용으로 전송하며, Socket, RMI, HTTP, SOAP 통신인터페이스를 통하여 전송된다.

4) 저장관리블록과 XML 관련 모듈

그림 9의 미들웨어 전체 구조도에서 저장관리블록과 그 외 XML 관련 모듈에 대해 기술한다.

EPC 데이터 처리를 위하여 필요한 자료구조들이 있는데, 표 7는 ECSpec을 관리하기 위한 스펙풀(spec pool) 테이블로서 define 메소드를 호출할 경우 ECSpec 관련 정보를 저장한다. 이 테이블은 ECSpec 이름에 따라 ECSpec이 유일하게 존재하도록 하기 위하여 해쉬맵 자료구조를 이용하여 설계하였다.

표 7. 스펙 관리 테이블

method	specName	spec	Dirty bit
define	mySpec<String>	ECSpec<Doc>	1

define, undefine, subscribe 등의 메소드를 호출함에 따라 ECSpec의 상태가 달라진다. ECSpec 이름을 키로 하여 스펙 객체가 저장되며, 메소드 이름 필드에는 ECSpec의 상태를 변화시킨 메소드 이름을 저장한다. ECSpec 상태 정보가 변경되면 백업 비트가 1이 되며, 이에 따라 백업 모듈이 ECSpec 상태 정보를 데이터베이스에 백업한다.

(1) 코어데이터관리자(CoreDataManager)

코어데이터관리자는 ECSpec 정보 리스트, 요청상태 메모리 정보, 이전 이벤트 사이클 데이터 리스트 등 데이터 백업 플래그와 메소드 플래그를 체크하여 해당 항목을 리스트에서 삭제 하는 모듈이다.

ECSpec 정보 리스트의 경우 메소드 플래그가 UNDFINE이고 백업플래그가 OFF일 경우 리스트에서 삭제한다.

요구 상태 리스트의 경우 메소드 플래그가 UNSUBSCRIBE이고 백업플래그가 OFF이고 상태비트가 OFF일 경우 리스트에서 삭제한다.

이전 이벤트 사이클 데이터리스트의 경우 도착비트가 ON이고 백업플래그가 OFF일 경우 리스트에서 삭제한다.

(2) 백업모듈(BackupModule)

각 정보리스트를 가져와 데이터 백업 플래그를 체크하여 해당 플래그가 1로 되어 있는 항목을 데이터베이스에 저장 및 업데이트를 하고 해당 플래그를 1에서 0으로 초기화하는 모듈이다.

(3) XML파서모듈(XMLParserModule)

XML파서모듈은 XML 문서를 ECSpec 객체와 ECReports 객체로 만드는 모듈로 XML파서모듈에 포함된 setXML2ECSpec(String xml) 메소드는 인자값으로 문자열 xml을 넘기고, 그 결과를 ECSpec 객체로 만든다. 또한 setXML2ECRepors(String xml) 메소드는 인자값으로 문자열 xml 문서를 받아들여서 그것을 ECRepors 객체로 만든다.

(4) EC스펙XML(ECSpecXML)

ECSpec 객체를 XML 문서로 만들어 주는 모듈이다. EC스펙XML에 포함된 setECSpecToXml(ECSpec obj) 메소드는 인자값으로 ECSpec 객체를 받아들여서 그것을 XML 문서로 만든다.

4. 데이터베이스 설계

표 8은 RFID 미들웨어의 시스템 운영에 필요한 메타데이터와 물리적 센서, 논리적 센서 등 각종 정보를 저장 및 관리 하는 테이블 정보이며 표 9는 각종 정보들의 관리를 체계적이고 효율적으로 사용할 수 있도록 테이블 레이아웃을 구축하였다.

표 8. 테이블 정보

테이블 ID	테이블명
IC_user	RFID 미들웨어 접근 사용자 테이블
IC_readermodel	RFID 리더 모델 관리 테이블
IC_readergroup	RFID 리더 그룹 관리 테이블
IC_reader	RFID 물리리더 관리 테이블
IC_logicalreader	RFID 논리리더 관리 테이블
IC_logicalreaderinfo	RFID 논리리더 정보 관리 테이블
IC_schemainfo	스키마 정보 관리 테이블
IC_privilege	권한 관리 테이블
IC_grant	접근 허가 관리 테이블
tb_requestspec	서비스 요청 관리 테이블
tb_pre_ecdata	이전 EPC 데이터 관리 테이블
tb_ecspec	EC스펙 관리 테이블

표 9. 테이블 레이아웃

테이블명	필드명	속성명	Type	키형태	Nulls
IC_user	UserName	사용자명	varchar(64)	PK	no
	Password	비밀번호	varchar(64)		no
IC_readermodel	readermodelname	리더모델명	varchar(64)	PK	no
	mfrdescription	리더설명	varchar(100)		yes
	readerconfiguration	리더설정	varchar(100)		yes
IC_readergroup	groupname	그룹명	varchar(64)	PK	no
	parentgroupname	상위 그룹명	varchar(64)		no
	description	설명	varchar(100)		yes
IC_reader	readername	물리 리더명	varchar(64)	PK	no
	readermodelname	리더모델명	varchar(64)		yes
	readeripaddress	리더 주소	char(20)		yes
	readerport	리더 포트	char(10)		yes
	readergateway	리더게이트웨이	char(20)		yes
	readernetmask	리더 넷마스크	char(20)		yes
	readerdns	리더 DNS	char(20)		yes
	hostipaddress	호스트 주소	char(20)		yes
	hostport	호스트 포트	char(10)		yes
IC_logicalreader	sourcename	논리 리더명	varchar(64)	PK	no
	readername	물리 리더명	varchar(64)		no
IC_logicalreaderinfo	sourcename	논리 리더명	varchar(64)	PK	no
	sourcetype	논리리더 타입	int(11)		yes
IC_schemainfo	schemaname	스키마명	varchar(64)	PK	no
	XMLschema	XML스키마	blob		yes
	QueueMessageType	큐타입	int(11)		yes
	QueueCapacity	큐용량	int(11)		yes
	SchemaType	스키마타입	int(11)		yes
tb_requestspec	method_name	메소드명	char(2)		yes
	spec_name	스펙명	varchar(100)	PK	no
	url	전송 주소	varchar(255)	PK	no
	state	상태 정보	char(1)		yes
tb_pre_ecdata	KIND	데이터 종류	char(1)		yes
	spec_name	스펙명	varchar(100)	PK	no
	url	전송 주소	varchar(255)	PK	no
	ecdata	EPC 데이터	varchar(255)		yes
tb_ecspec	method_name	메소드명	varchar(2)		yes
	spec_name	스펙명	varchar(100)	PK	no
	spec_doc	스펙문서	text		yes

IV. 시스템 구현 및 테스트

이 장에서는 Ⅲ장에서 ALE 스펙을 기반으로 설계한 RFID 미들웨어를 구현하여 미들웨어의 효율성 및 미들웨어의 자체 테스트를 위해 수행한 결과를 분석한다.

1. 구현 환경

본 논문에서 제안한 ALE 기반 RFID 미들웨어는 표 10의 환경에서 구현하였다.

표 10. 시스템 구현 환경

구분	하드웨어	소프트웨어	비고
Middleware	AMD Athlon(tm) 64 X2 Dual Core 3800+ 2.01GHz RAM 1024MB	Microsoft Windows 2003 Sun Java SDK 1.5.0_07 Eclipse SDK 3.1 MySQL 5.0	java 이용
Reader System	ALR 9750-A Frequency 902.6 MHz - 927.4 MHz Hopping Channels 63 Communications Interface RS-232, LAN TCPI/IP 모바일 리더	ale_Reader_데모 Alien 에뮬레이터	java 이용
Tag	ALN-9350-R "I" - EPC Class 1, Gen 1 - 148.5 (w) x 10 (h) mm		
Application	AMD Athlon(tm) 64 3800+ 2.01GHz RAM 1024MB	Microsoft Windows XP Pro Sun Java SDK 1.5.0_07	

제안하는 시스템은 이클립스를 이용하여 자바로 구현하였다. 미들웨어의 메타정보 저장 및 EPC 데이터 연산처리를 위한 일시적 데이터 저장을 위해 DB는 MySQL 5.0을사용 하였다. RFID 리더로는 Alien사의 900Mhz 9640모델과 에뮬레이터를 이용하였다.

2. 미들웨어 패키지 구현

ALE 기반 RFID 미들웨어 패키지 구현은 그림 18과 같이 구현하였다.

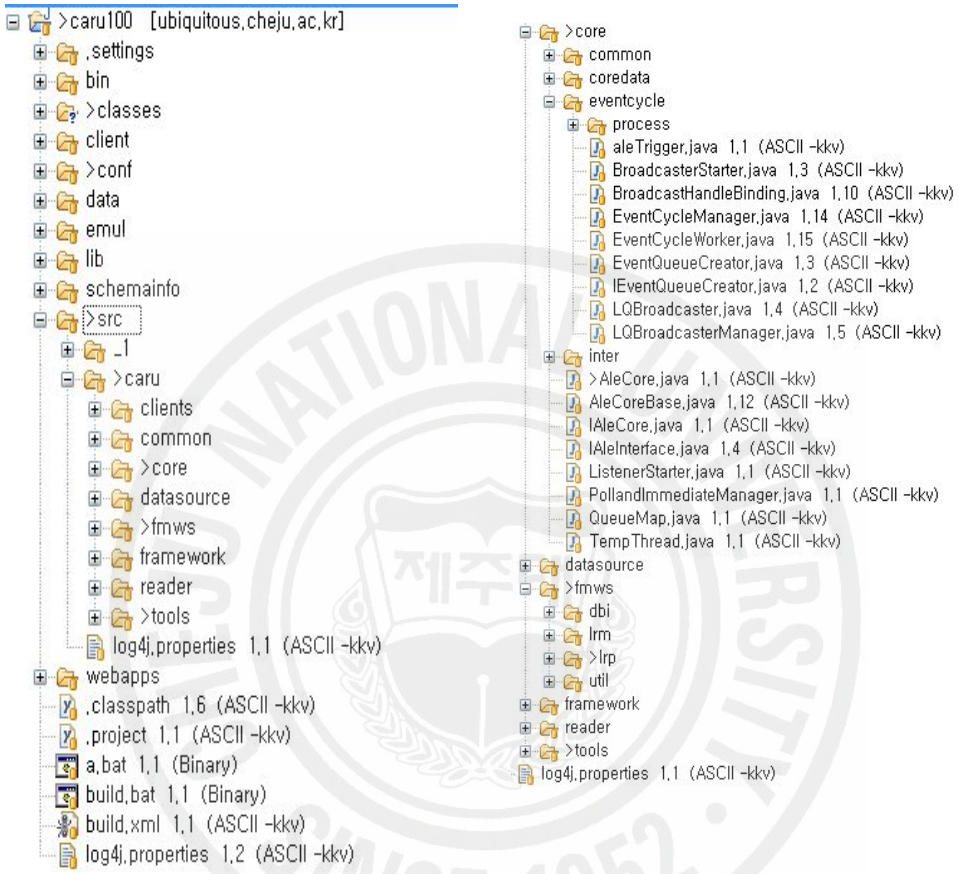

그림 18 미들웨어 패키지 설계

1) 프레임워크(Framework) 패키지

그림 19는 프레임워크 패키지에 포함된 클래스 다이어그램이고 표 11은 각 클래스에 대한 설명이다. 프레임워크 패키지에 포함된 클래스들은 미들웨어의 구동 및 RFID 리더기에서 보낸 데이터를 받을 수 있도록 리스너 등이 포함되어 있다.

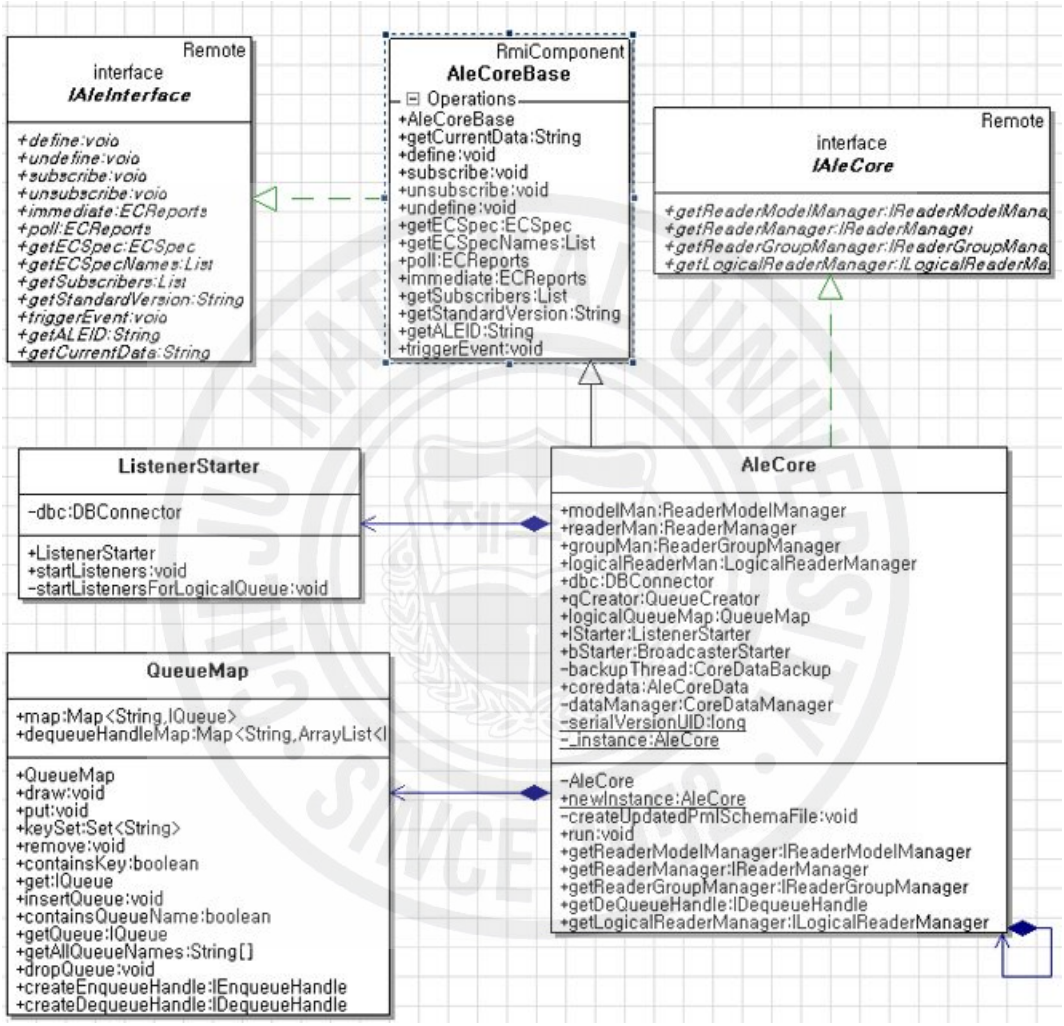

그림 19. Framework 패키지 포함 클래스 다이어그램

표 11. Framework 패키지에 포함 클래스

Class Name	설 명
AleCore	ALE Core를 구동시키는 클래스. ALE 미들웨어를 실행하면 이 클래스가 먼저 실행된다.
AleCoreBase	IAleInterface를 구현한 클래스
IAleCore	논리리더와 물리리더에 관련된 인터페이스
IAleInterface	RMI로 연결하는 ALE 인터페이스
ListenerStarter	물리적 리더기에서 데이터를 받을 수 있게 리스너를 구동시킴.

2) ALE 인터페이스 패키지

그림 20과 같이 ALE 인터페이스 패키지에 포함된 클래스에는 응용이 미들웨어에게 쉽게 서비스를 호출 할 수 있도록 제공하는 ALE 인터페이스가 포함된 클래스이다. 표 12는 각 클래스에 대한 설명이다.

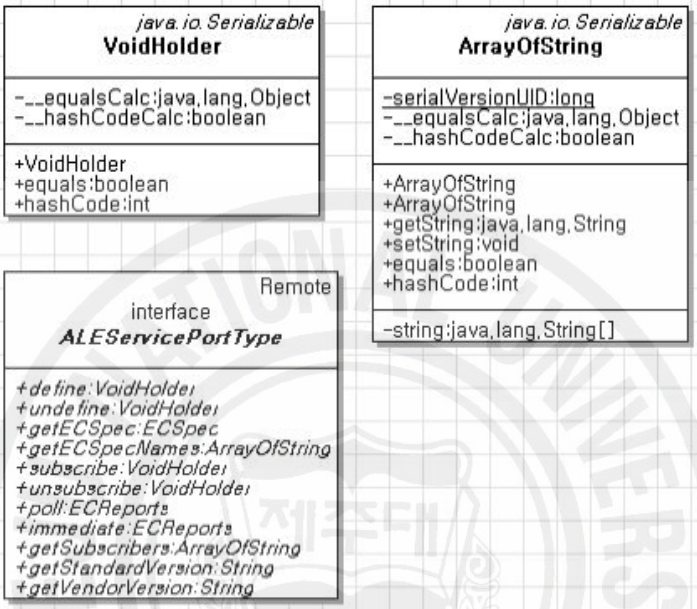

그림 20. ALE 인터페이스 패키지에 포함된 클래스 다이어그램

표 12. ALE 인터페이스 패키지에 포함 클래스

Class Name	설 명
ALEServicePortType	SOAP를 위한 ALE 인터페이스
VoidHolder	SOAP 인터페이스에서 void를 쓸수 없으므로 그것을 대체하며 아무것도 하지 않고 단지 Serializable된 클래스
ArrayofString	SOAP 인터페이스에서는 String[]을 반환할 수 없으므로, String []을 갖고 있는 클래스.

3) 데이터 자료형

그림 21은 ECSpec, ECReports, ECTime 등 데이터 자료형의 클래스 다이어그램을 나타낸 것이다. 표 13은 각 클래스에 대한 설명이다.

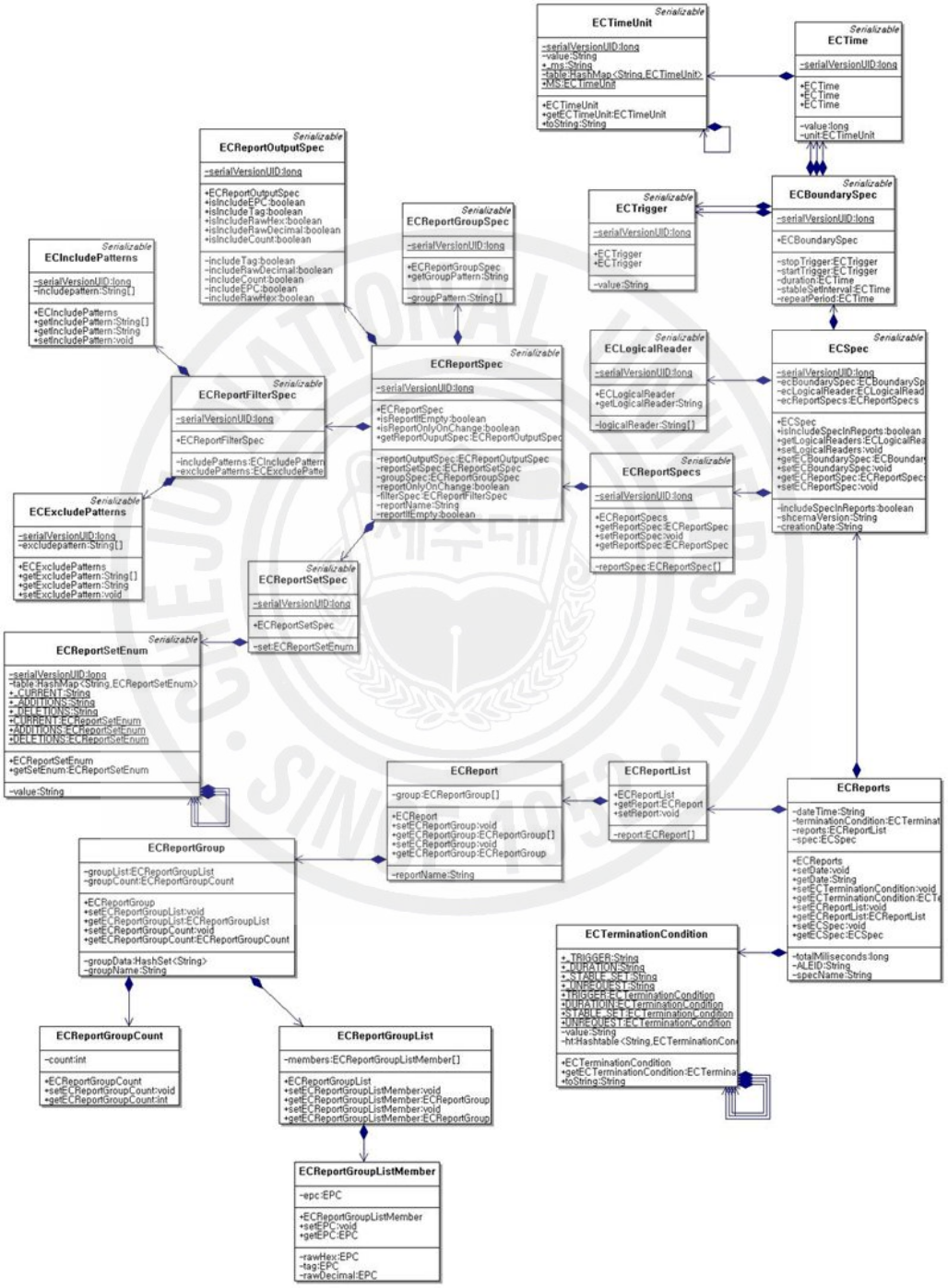

그림 21. ALE 자료형 패키지에 포함된 클래스 다이어그램

표 13. ALE 데이터 자료형 클래스 목록

Class Name	설 명
ECSpec	응용이 정의하는 ECSpec 객체를 담는 데이터형 클래스
ECLogicalReader	논리 리더를 넣는 데이터형 클래스
ECBoudanrySpec	Boundary를 담는 데이터형 클래스
ECTrigger	StartTrigger, StopTrigger 조건을 담는 데이터형 클래스
ECTime	Duration, RepeatPeriod, SetStableInterval 조건을 담는 데이터형 클래스
ECTimeUnit	Time Unit (MiliSeconds)를 담은 데이터형 클래스
ECReportSpecs	Report []을 담는 데이터형 클래스
ECReportSpec	하나의 Report 조건을 담는 데이터형 클래스
ECRportSetSpec	Addition,Current, Deletion의 조건 중 하나를 담는 데이터형 클래스
ECReportSetEnum	ADDITION, CURRENT, DELETION 조건을 설정한 클래스
ECReportGroupSpec	GroupPattern이 들어 있는 클래스
ECReportOutputSpec	Output 조건 - includeTag, includeEPC, includeCount, includeRawHex, includeRawDecimal 조건 중 여러가지가 들어 있을 수 있는 클래스
ECIncludePatterns	includePattern 즉, 필터링 시 데이터가 포함될 수 있는 조건을 있는 클래스
ECReportFilterSpec	IncludePattern과 ExcludePattern이 들어 있는 클래스를 담는 클래스
ECExcludePatterns	excludePattern 즉, 필터링 시 데이터가 제외되는 조건이 있는 클래스
ECReports	응용에 보내는 ECReports 객체를 담는 데이터형 클래스
ECReportList	ECReprot [] 배열을 담고 있는 클래스
ECReport	하나의 ECReport를 담고 있는 클래스
ECReportGroup	Grouping이 된 데이터를 갖고 있는 클래스
ECReportGroupCount	Grouping이 된 데이터를 개수로 갖고 있는 클래스
ECReportGroupList	Grouping이 된 데이터의 목록을 갖고 있는 클래스
ECReportGroupListMember	Grouping이 된 데이터을 갖고 있는 클래스
ECTerminationCondition	EventCycle이 끝난 종료 조건이 들어 있는 클래스, DURATION, UNREQUEST, SETSTABLEINTEVAL, STOPTRIGGER이 있다.

4) 익셉션(Exception) 패키지

그림 22는 익셉션 패키지에 포함된 클래스 다이어그램이다. 표 14는 각 클래스에 대한 설명으로 응용이 인터페이스를 통해 메소드를 호출시 발생되는 익셉션 등 다양한 익셉션 클래스가 있다.

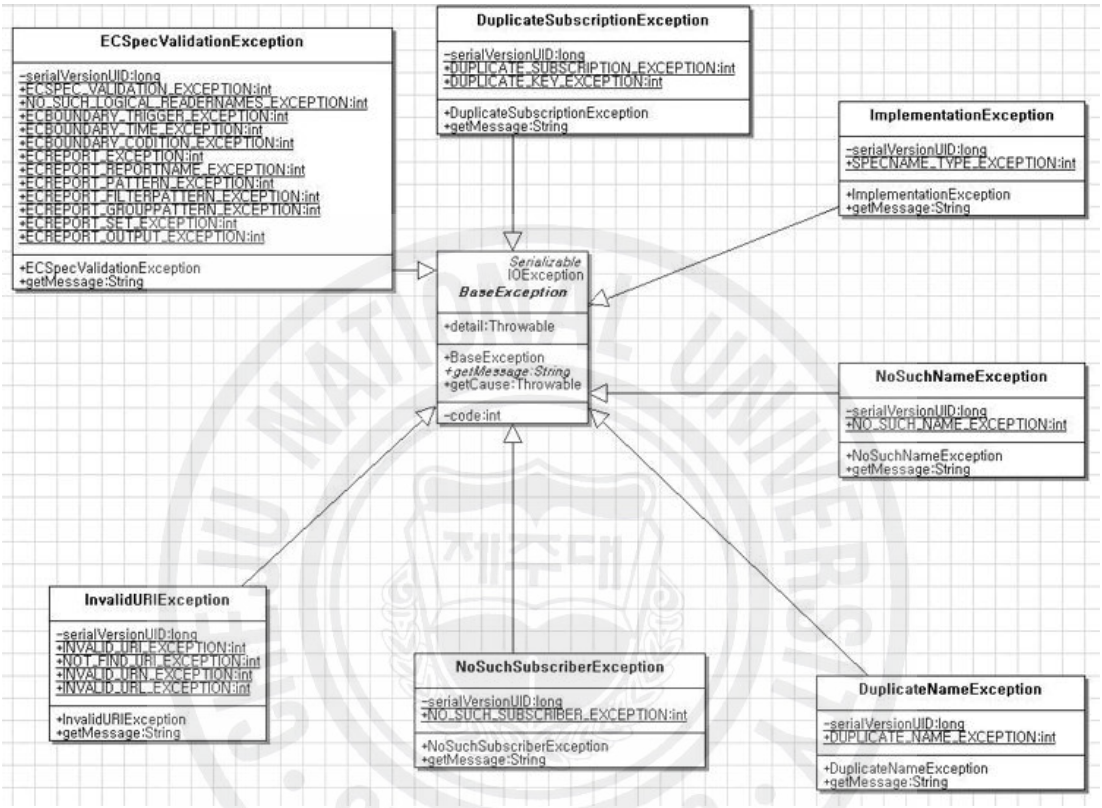

그림 22. exception 패키지에 포함된 클래스 다이어그램

표 14. Exception 패키지에 포함 클래스

Class Name	설 명
BaseException	IOException을 상속받아서, ALE Exception을 정의하는 추상 클래스 응용에서 이 Exception만 처리해줘도 된다.
ImplementationException	ALE 구현 상에 발생되는 Exception
DuplicateNameException	ECSpec 이름이 중복이 일어날 경우 발생되는 Exception
NoSuchNameExcetpion	ECSpec이름이 존재하지 않을 경우 발생되는 Exception
NoSuchSubscribeException	Subscriptor가 존재하지 않을 경우 발생되는 Exception
InvalidURIException	URI가 유효하지 않을 경우 발생되는Exception
ECSpecValidationExceptin	ECSpec이 유효하지 않을 경우 발생되는 Exception
DuplicateSubscriptionException	Subscription의 중복이 일어날 경우 발생되는 Exception

5) Validation 패키지

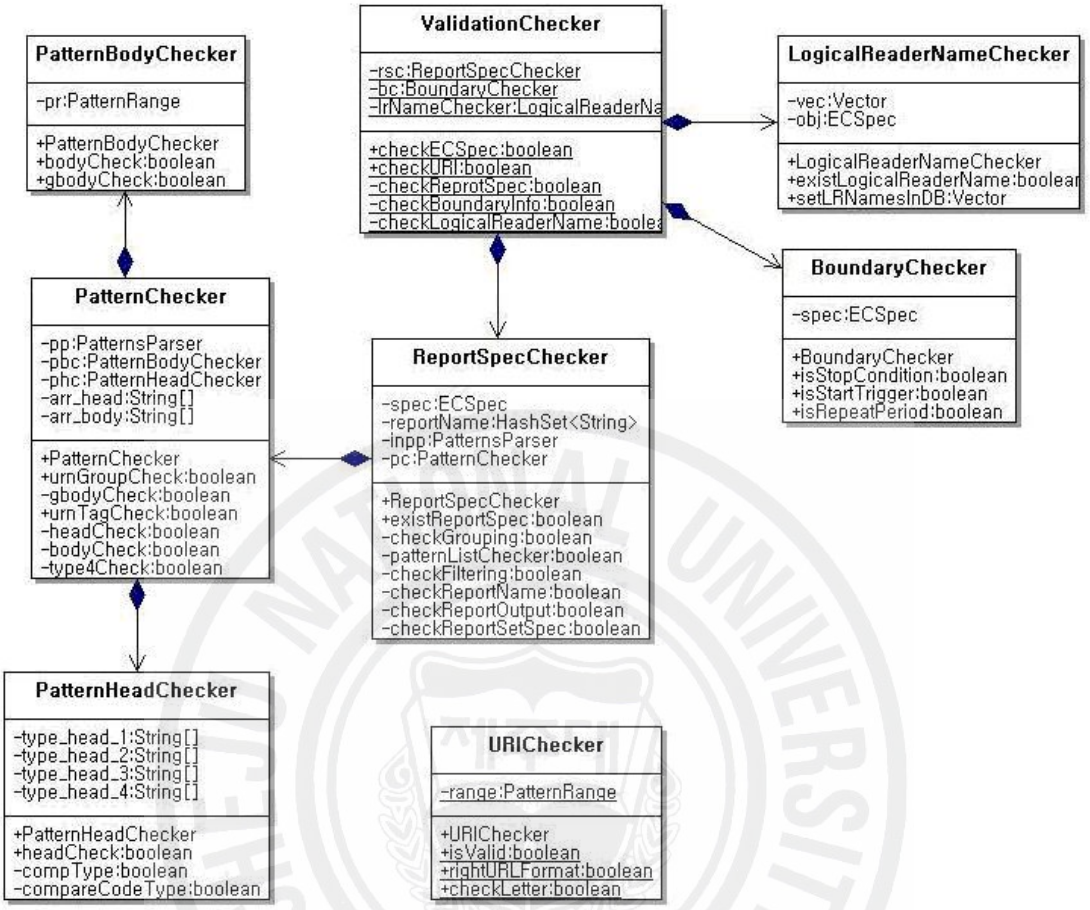

그림 23. ValidationChecker 패키지에 포함된 클래스 다이어그램

표 15. Checker 패키지 포함 클래스

Class Name	설 명
ValidationChecker	ECSpec의 유효 여부를 검사하는 클래스
LogicalReaderNameChecker	ECSpec의 논리 리더의 이름의 유효 여부를 검사하는 클래스
BoundaryChecker	ECSpec의 BoundarySpec의 유효 여부를 검사하는 클래스
ReportSpecChecker	ECSpec의 ReportSpec의 유효 여부를 검사하는 클래스
PatternChecker	ReportSpec 중 필터링과 그룹핑이 disjoint 여부를 검색하는 클래스
PatternBodyChecker	URN 패턴의 태그 바디 부분을 disjoint 여부를 검사하는 클래스
PatternHeadChecker	URN 패턴의 태그 헤드 부분을 disjoint 여부를 검사하는 클래스
URIChecker	URI 형식이 올바르지 여부를 검사하는 클래스

6) Util 패키지

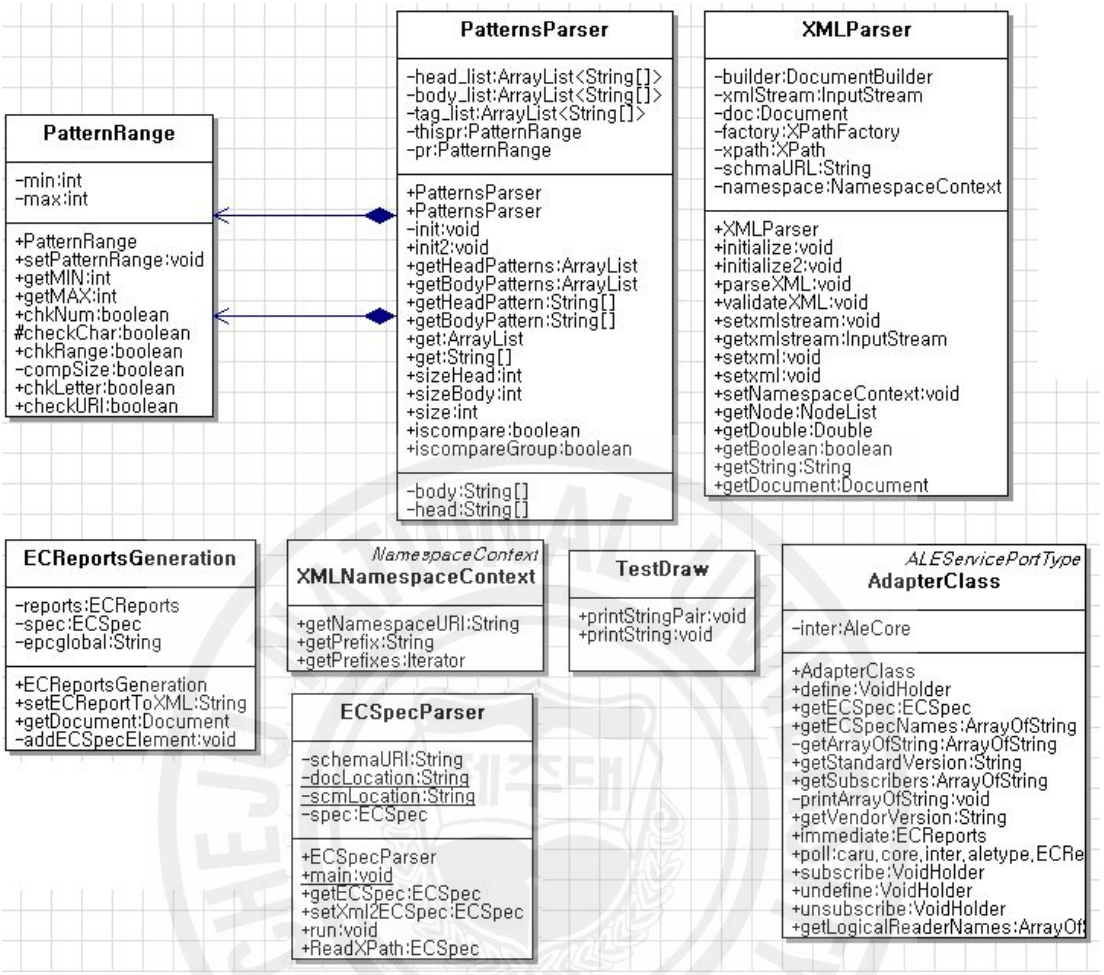

그림 24.Util 패키지에 포함된 클래스 다이어그램

표 16. Util 패키지 포함 클래스

Class Name	설 명
PatternsParser	ECSpec의 패턴에 관련된 EPC 코드를 Head부분과 Body부분으로 나누어서 서로 비교한다.
PatternRange	PatternsParser가 비교할 때 쓰이는 각종 쓰이는 함수를 정의한 클래스
XMLParser	XML 문서를 객체로 반환한다. 현재는 쓰이지 않는다.
XMLNamespaceCountext	XML 네임스페이스 설정을 하는 클래스이다.
AdapterClass	SOAP Interface인 ALEServicePortType 인터페이스를 구현한 클래스. 이 클래스를 weapps/api/web-inf/classes/web-inf/xfire/service.xml의 구현 클래스 태그에 패키지명과 클래스 이름을 써주어야 한다. 이 클래스의 메소드는 ALEServicePortType 인터페이스에 설명과 같으므로 생략을 한다.
ECSpecParser	ECSpec 객체를 Serializable XML 문서로 만들어주는 클래스. 현재는 쓰이지 않는다.

7) CoreData 패키지

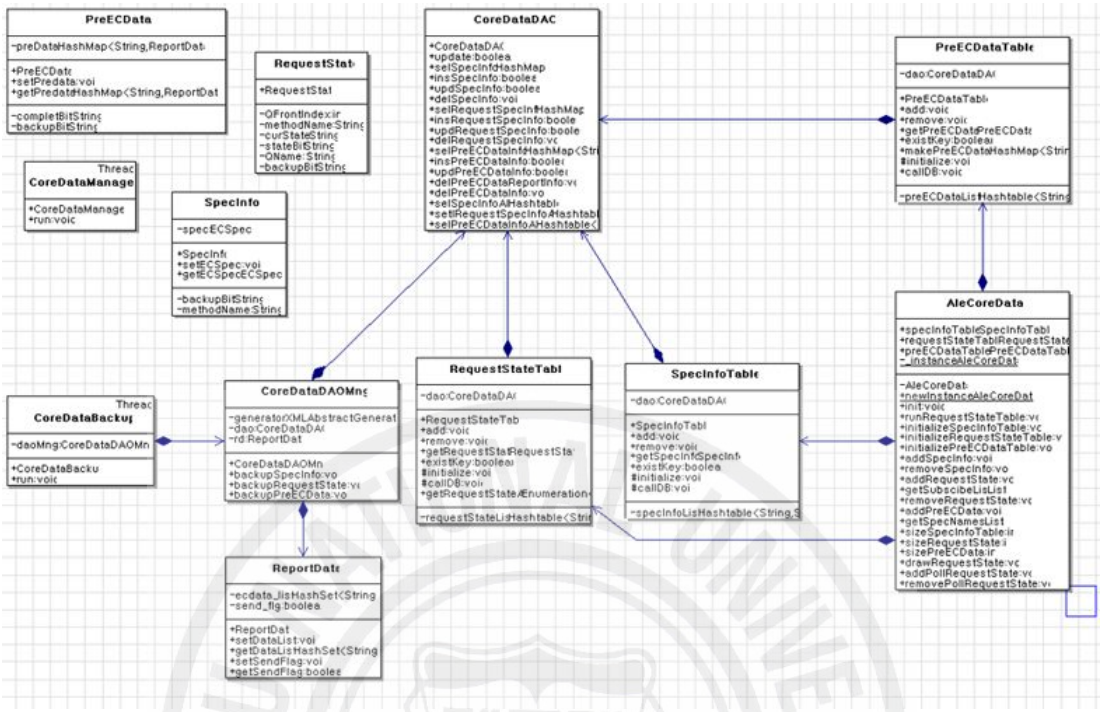

그림 25. CoreData 패키지 포함 클래스 다이어그램

표 17. CoreData패키지에 포함된 클래스

Class Name	설 명
AleCoreData	ALE Core에 사용되는 메모리 구조 테이블을 다루는 클래스. PreECDataTable, ReqeustStateTable, SpecInfoTable를 주로 관리한다.
CoreDataBackup	테이블에 들어 있는 정보를 DB에 백업하는 클래스
CoreDataDAOMng	백업 메소드가 들어 있는 DAO 매니저 클래스
CoreDataManager	테이블에 들어 있는 정보를 삭제 및 수정하는 클래스
CoreDataDAO	SQL 구문이 있어서 실제적으로 DB에 백업하는 클래스
PreECData	이전 이벤트 싸이클 데이터를 지니고 있는 객체. ReportData를 가지고 있다.
PreECDataTable	이전 이벤트 싸이클 데이터를 저장하는 메모리 테이블
ReportData	report 별로 데이터를 지니고 있는 객체
RequestState	subscribe할 경우, 스펙 상태의 정보가 들어 있는 클래스
RequestStateTable	스펙 상태의 정보를 저장하는 메모리 테이블
SpecInfo	ECSpec의 정보가 들어 있는 객체
SpecInfoTable	ECSpec의 정보를 저장하는 메모리 테이블

8) 이벤트 사이클(EventCycle) 패키지

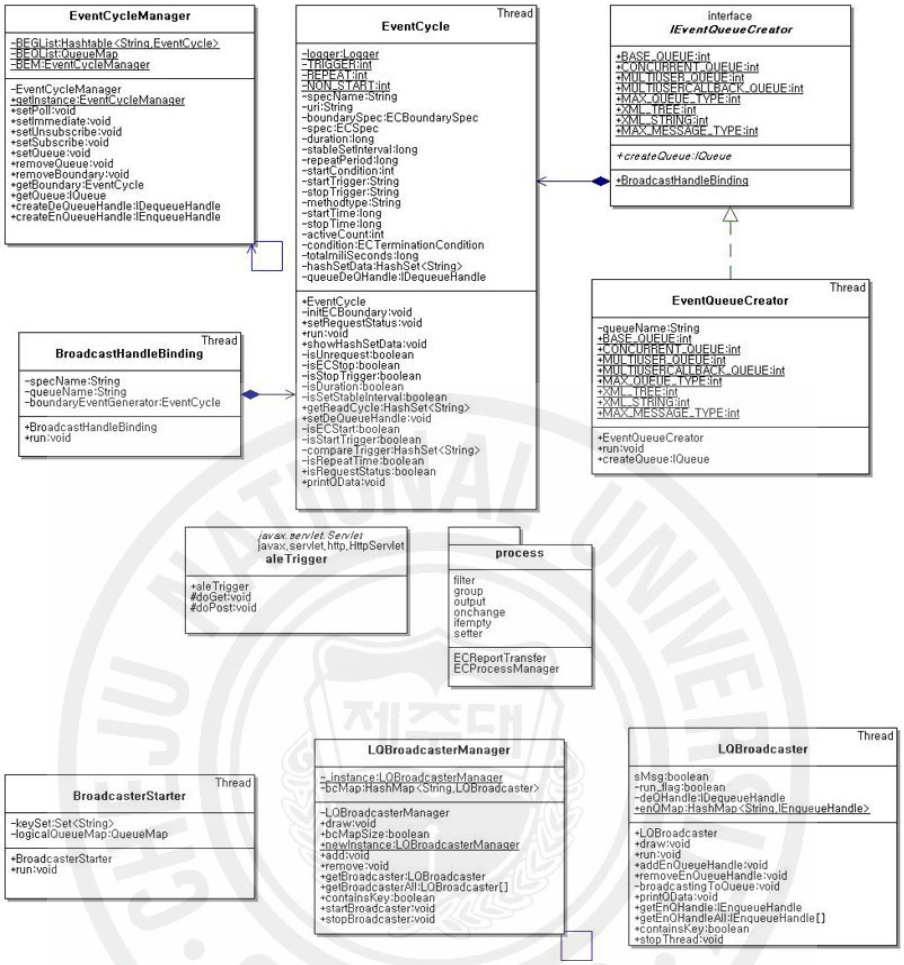

그림 26. EventCycle 패키지에 포함된 클래스 다이어그램

표 18. 이벤트 사이클 패키지에 포함된 클래스

Class Name	설 명
EventCycleManager	EventCycle를 생성 관리, deQHandle&enQHandle관리, 이벤트 큐 생성 및 관리
EventCycleWorker	스펙에 명시된 조건을 기반으로 메시지를 받아 중복처리 후 ECProcess객체에게 넘기는 역할
IEventQueueCreator	EventCycle을 처리하기 위한 이벤트 큐 생성 인터페이스
EventQueueCreator	EventCycle을 처리하기 위한 이벤트 큐 생성
BroadcastHandleBinding	생성된 사용자큐의 enQHandle 과 deQHandle 바인딩을 담당
BroadcasterStarter	logicalQueueMap에 등록된 logicalQueue들의 각각 DeQueueHandle을 생성하고 BroadcastManager에게 넘겨 Broadcatser를 생성하는 역할
LQBroadcaster	연결된 논리큐에서 사용자가 필요로하는 이벤트큐로 메시지 보내기
LQBroadcasterManager	LQBroadcaster를 생성하고 관리하는 목적

9) 데이터 연산처리 패키지

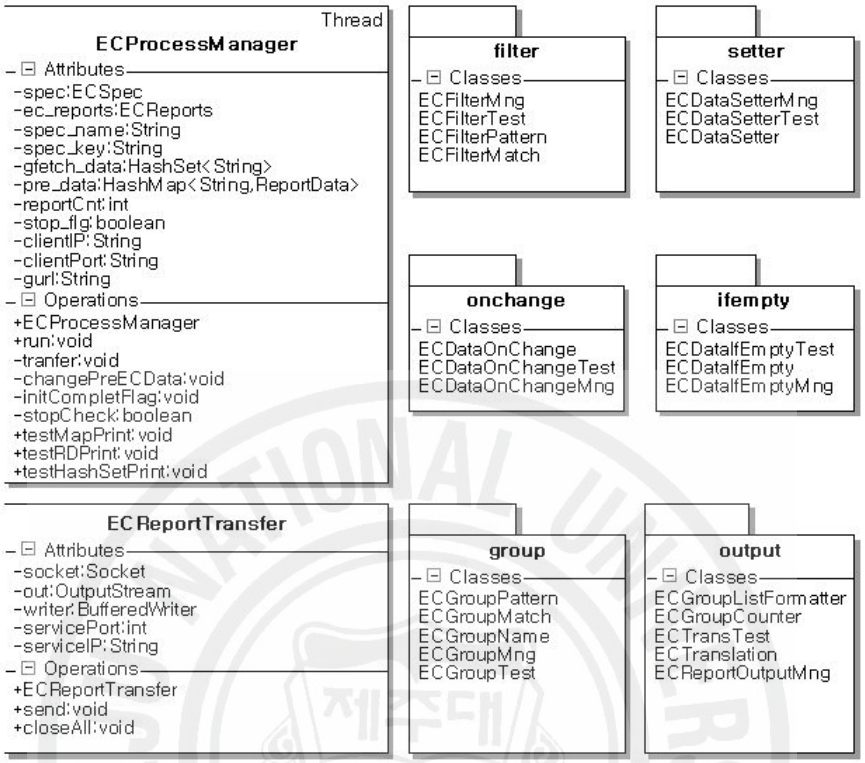

그림 27 데이터 연산처리 패키지에 포함된 클래스 다이어그램

표 19. 데이터 연산처리 패키지에 포함된 클래스

Class Name	설 명
ECProcessManager	EventCycle 동안 수집된 fetch_data를 바탕으로 ECSpec에 명시되어 있는 조건에 의해 수집된 데이터를 가공 처리한다.
ECReportTransfer	ECProcessManager에서 생성된 ECReports를 소켓기반으로 응용 URI 주소로 전송한다.
ECFilterMng	EventCycle 동안 수집된 fetch_data를 바탕으로 ECSpec에 명시되어 있는 필터링 조건에 의해 데이터를 필터링한다.
ECFilterPattern	ECSpec의 필터링 조건 패턴을 분석한다.
ECFilterMatch	ECFilterPattern에서 분석된 패턴과 fetch_data를 매칭하여 필터링한다.
ECGroupMng	ECDataIfEmptyMng 처리가 완료된 settered_data 를 바탕으로 ECSpec에 명시되어 있는 그룹핑 조건에 의해 데이터를 그룹핑한다.
ECGroupPattern	ECSpec의 그룹핑 조건 패턴을 분석한다.
ECGroupName	ECGroupPattern 에서 분석된 패턴과 settered_data를 매칭하여 그룹 이름 리스트를 생성한다.
ECGroupMatch	ECGroupName에서 생성된 그룹 이름을 바탕으로 해당 그룹 이름에 해당하는 settered_data를 그룹핑한다.
ECDataIfEmptyMng	ECDataSetterMng 처리가 완료된 settered_data 를 바탕으로 ECSpec에 명시되어 있는 IfEmpty 조건에 의해 데이터 유무를 체크한다.
ECDataIfEmpty	settered_data Hash_Set 데이터의 자료 유무를 체크하여 데이터에 결과를 셋팅한다.
ECDataOnChangeMng	ECFilterMng 처리가 완료된 filtered_data 를 바탕으로 ECSpec에 명시되어 있는 OnChange 조건에 의해 데이터 변경 유무를 체크한다.
ECDataOnChange	현재 이벤트 사이클 filtered_data 와 이전 이벤트 사이클의 pre_data 를 비교하여 데이터 변경 유무를 셋팅한다.
ECReportOutputMng	ECGroupMng 처리가 완료된 ECReportList를 바탕으로 ECSpec에 명시되어 있는 ReportOut 조건에 그룹 카운트와 최종 그룹 리스트를 생성한다.
ECGroupListFormatter	ECReportList를 바탕으로 ECSpec에 명시되어 있는 TAG, RawHex, RawDecimal, EPC 코드 형태로 변환한다.
ECGroupCounter	ECReportList를 바탕으로 각 그룹 리스트의 카운트 정보를 구한다.
ECTranslation	ECGroupListFormatter의 코드 변환 처리를 수행한다.
ECDataSetterMng	ECDataOnChangeMng 처리가 완료된 filtered_data 를 바탕으로 ECSpec에 명시되어 있는 ReportSet 조건에 의해 전송집합을 구한다.
ECDataSetter	ECSpec의 ReportSet Addition, Deletion 조건을 바탕으로 filtered_data의 전송집합을 구한다.

3. 실험 내용 및 환경

본 실험에서는 ALE 기반 RFID 미들웨어 시스템이 응용서비스가 요청(ECSpec)을 했을 때 서비스를 제공하기 위해 연속적인 EPC 데이터를 받기 위해 Alien 리더 1대, 리더 에뮬레이터 , 모바일 리더 1대 등을 이용하여 실험하였고, 논리 리더 설정은 FRONT_DOOR(정문), BACK_DOOR(후문), CENTER_DOOR(중앙문), OUT_DOOR(외부문)으로 구성하였다. 그림 28은 실제 ALE 기반 RFID 미들웨어 시스템 및 기타 장비의 화면이다.

그림 28. 실제 ALE 기반 RFID미들웨어 시스템 및 기타 장비

1) 리더 등록 및 미들웨어 실행

그림 39은 Alien Reader를 초기 설치 및 셋팅을 위해 터미널 연결하는 화면이다. 장치관리자에서 Serial Bridge의 Port번호를 확인하고 Alien 데모 프로그램에서 COM2로 연결된 iclab 리더기로 연결한다.

그림 29. Alien reader 설정 및 실행

그림 30. Alien reader 셋팅 화면

그림 30은 연결된 Alien Reader의 설정화면으로 기본환경 설정 명령어이며 세팅은 위와 같이 Reader 와 Programmer 가 있는데 Reader의 세팅은 Tag를 쓸 수는 없고 읽을 수 만 있게 세팅하며 Programmer는 쓸 수도 있게 세팅한다.

예) Set/Get Function = Reader / Programmer

안테나 전파의 감소량을 세팅하거나 보여준다. 수치가 크면 클수록 안테나 전파 크기는 작아진다. 예) Set/Get RFAttenuation = (0 - 255) 기본 = 0

NETWORK은 리더기장비의 네트워크 설정으로 DHCP가 Off인 경우는 IP를 설정하고 Gateway, Netmask, DNS, CommandPort, 등을 셋팅한다.

Autonomous Mode Commands 설정은 자동모드를 설정한다. 기본값으로는 OFF로 설정되어 있다. 예) Set/Get AutoMode = ON, OFF

다음은 Notify Mode Commands의 설정으로 Auto 모드에서만 작동되며 태그 리스트 정보를 받을지 여부를 설정하는 NotifyMode, NotifyAddress 등 리더기가 읽은 정보를 보낼 곳의 내용을 설정한다.

그림 31은 리더 에뮬레이터 실행화면이다. 본 논문에서는 동시 여러 리더 장치에서 보내는 데이터 처리를 위해 사용된다.

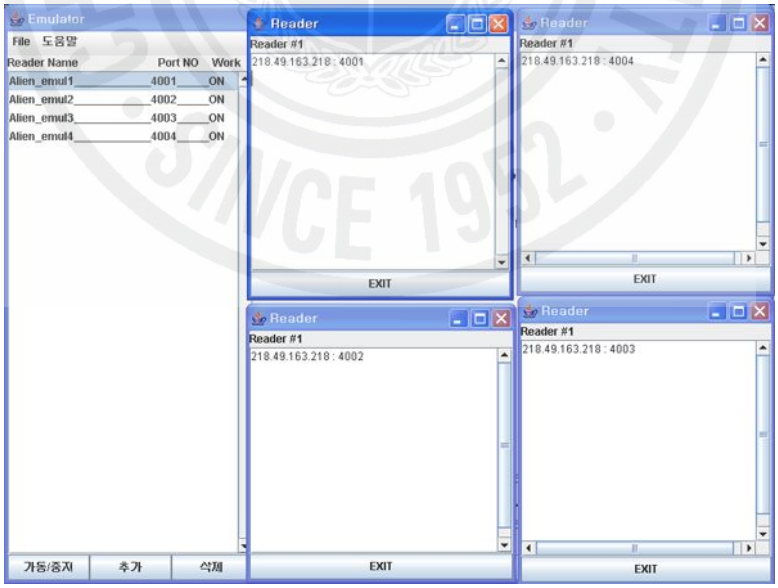

그림 31. 에뮬레이터 리더기 실행 화면

표 20. 물리리더 와 논리리더 매핑 테이블

물리리더 명	논리리더 명	리스너 포트	데이터범위		
			Company	Item	Serial Number
Alien_emul1	FRONT_DOOR	7001	0	0 – 10	0 – 20
Alien_emul2	BACK_DOOR	7002	0	11 – 20	21 – 30
Alien_emul3	CENTER_DOOR	7003	0	21 – 30	31 – 60
Alien_emul4	OUT_DOOR	7004	0	31 – 40	61 – 80
Alien 9640	A_DOOR	7000	–	–	–

표 20은 물리리더와 논리리더 매핑을 테이블로 정리한 것이다. 그림 32는 미들웨어 초기 실행화면이다. 그림 33은 미들웨어가 실행되었을 때 물리리더 Reader 4개를 등록하고 그룹 생성하는 화면과 미들웨어에서 처리 되는 화면이다.

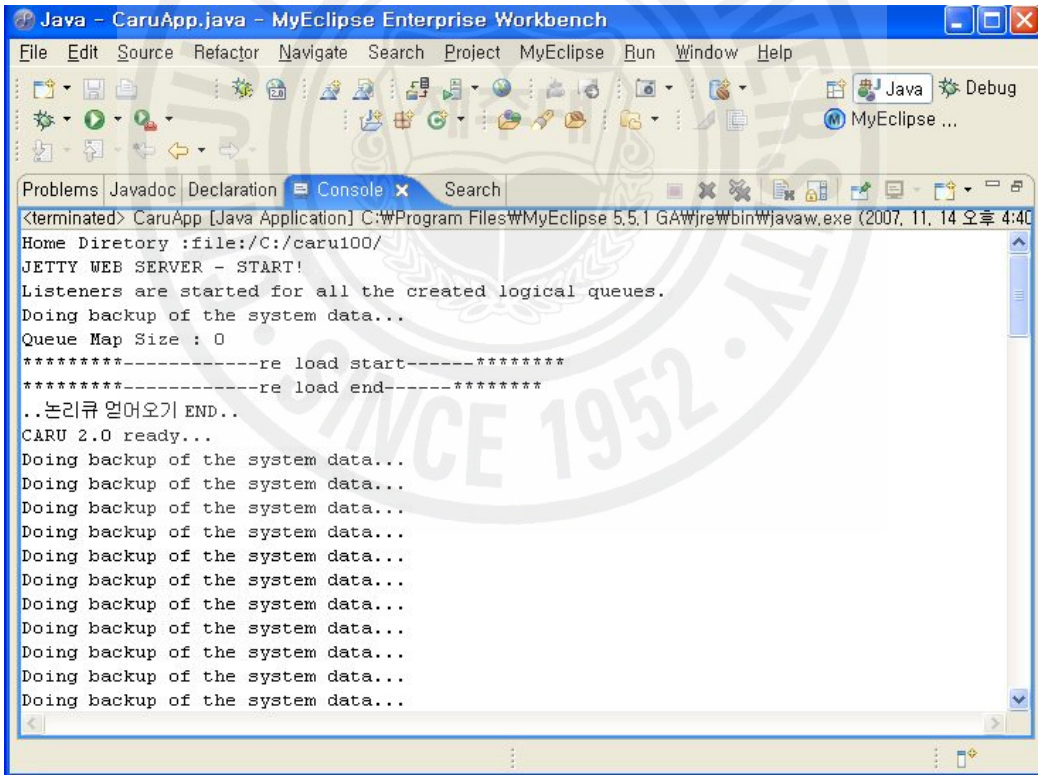

그림 32. 미들웨어 초기 실행화면

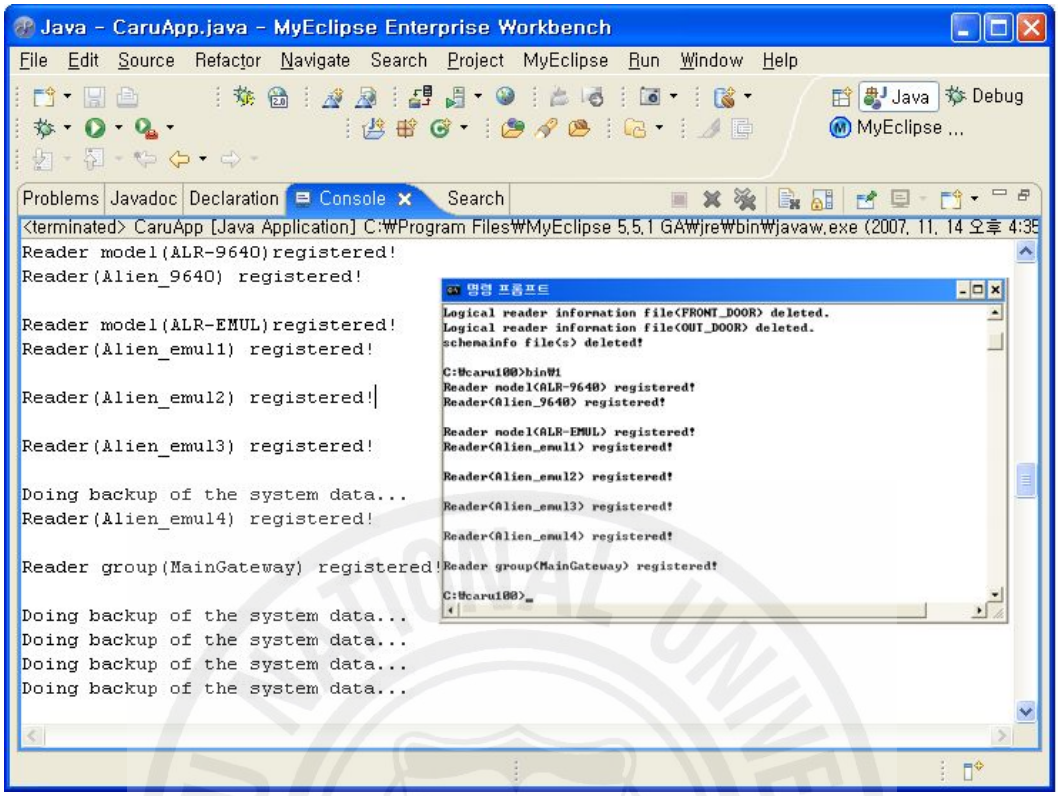

그림 33. 물리리더 등록시 미들웨어 실행 화면

그림 34는 7000, 7001, 7003, 7004 포트로 리스너가 구동된 상태로 응용서비스에게 서비스를 할 준비가 된 실행화면이다. 응용서비스는 FRONT_DOOR, BACK_DOOR, CENTER_DOOR, OUT_DOOR라는 논리 리더를 이용하여 서비스 요청을 할 수 있다.

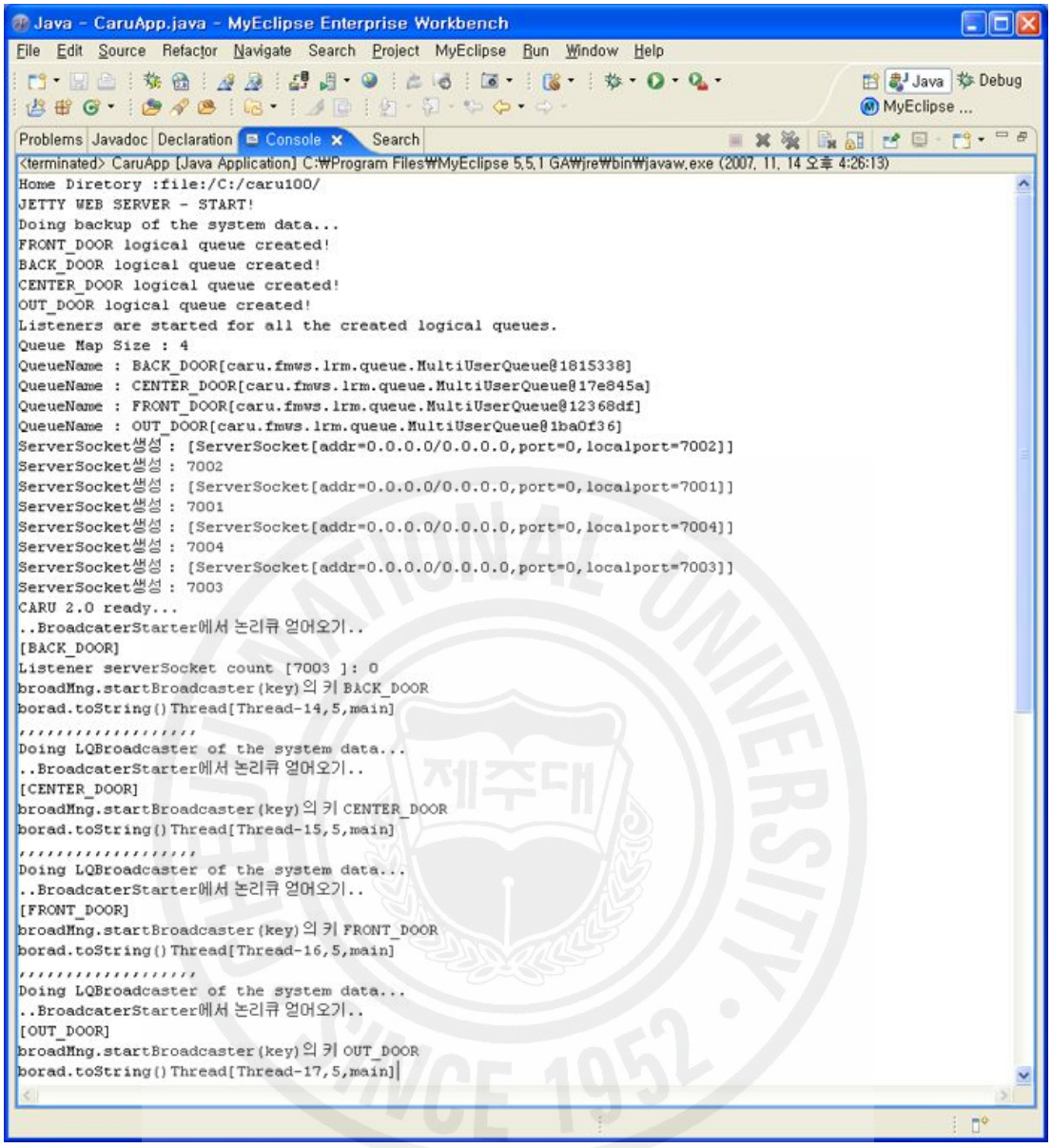

그림 34. 미들웨어 실행화면

2) 테스트에 사용되는 ECSpec 유형 명세

표 21은 ECSpec에 기술된 내용 중 이벤트 사이클의 시작 조건, 종료 조건과 얼마 동안 데이터를 처리할 것인지에 대한 명세로 테스트에서 사용할 타입 A과 B로 정리하였다. 타입 A-1의 경우 시작조건과 종료조건은 Trigger로 동작 되고, 시작과 종료 사이 동안 이벤트 사이클은 수행되고 그 결과를 리포팅하는 경우이다. B-2S의 경우는 시작조건이 repeatPeriod가 3분이고(반복주기) 2분 동안 읽고 이벤트 사이클을 종료하고 레포팅하는 경우이다. B-2의 경우는 종료조건으로 stableSetInterval이 더 추가된 경우이다. stableSetInterval의 경우 1분 동안 새로운 데이터가 읽히지 않는다면 이벤트 사이클이 종료 된다.

표 21. Boundary 조건 명세

타입		Boundary 조건				
		Trigger		repeatPeriod	duration	stableSetInterval
		start	stop			
A	1	○	○	–	–	–
	2	○	○	X	3분	–
B	1	–	–	3분	2분	X
	3	–	–	3분	3분	1분

표 22. ECReportSpec 조건 명세

타입	reportSet	filterSpec	groupSpec	IfEmpty	OnlyOnChange	output
C	CURRENT	Include exclude	○	true	true	EPC Hex Tag Decimal count
D	CURRENT	exclude	–	false	false	EPC Hex Tag Decimal count
E	ADDITIONS	Include exclude	○	○	○	EPC Hex Tag Decimal count
G	CURRENT	–	–	true	true	EPC Hex Tag Decimal count

표 22는 ECReportSpec의 조건 명세 테이블로 이벤트 사이클 수행에서 반환할 리포트를 지정하는 것을 표로 나타냈다.

C타입은 리포트할 EPC 집합이 현재 이벤트 사이클인지 이전 이벤트 사이클의 추가 또는 삭제분인지를 선택하는 요소이다. reportOnlyOnChange==true 이면 이전 데이터와 비교하여 바뀌었을때만 리포트를 전송한다. reportIfEmpty == false로 설정되면 ECReport 인스턴스가 빌 경우 ECReports에 넣어 전송하지 않는다. reportIfEmpty=false이면, 최종 필터된 EPC가 공집합이면 대응하는 ECReport 인스턴스는 이 이벤트 사이클에 대한 ECReports에서 생략된다.

표 23은 미들웨어에서 사용된 ECSpec의 예를 정리한 것이다.

표 23. 테스트에 사용한 ECSpec 예1

스펙유형	조건명	조건값
A1C	논리 리더	FRONT_DOOR, BACK_DOOR
	시작조건	http://218.49.163.218/eventcycle?specname=test_A1D!triggertype=stop!methodtype=su!methoduri=218.49.163.218:5500
	종료조건	http://218.49.163.218/eventcycle?specname=test_A1D!triggertype=stop!methodtype=su!methoduri=218.49.163.218:5500
	reportSet	CURRENT
	Include pattern	urn:epc:tag:sgtin-64:0.0.13.* urn:epc:tag:sgtin-64:0.0.9.* urn:epc:tag:sgtin-64:0.0.10.* urn:epc:tag:sgtin-64:0.0.20.30
	Exclude pattern	urn:epc:tag:sgtin-64:0.0.13.21
	groupSpec	urn:epc:tag:sgtin-64:0.0.9.*
	output	includeCount=true, includeEPC=true, includeRawHex=true, includeRawDecimal=true, includeTag=true
	reportIfEmpty	reportIfEmpty=true ->조건에 맞는 태그가 없어도 보냄
	OnlyOnChange	OnlyOnChange=true ->이전 이벤트 사이클과 비교해 변경되었을 때만 보냄
	IncludeSpec InReports	IncludeSpecInReports=true -> 사용한 스펙정보를 함께 보냄

3) 서비스 시나리오에 따른 데이터 처리

그림 35는 응용에서 defind 메소드를 이용하여 표 23의 ECSpec을 등록하는 모습과 그림 36은 미들웨어서 처리하는 모습으로 스펙을 파싱하고 분석하는 화면이다.

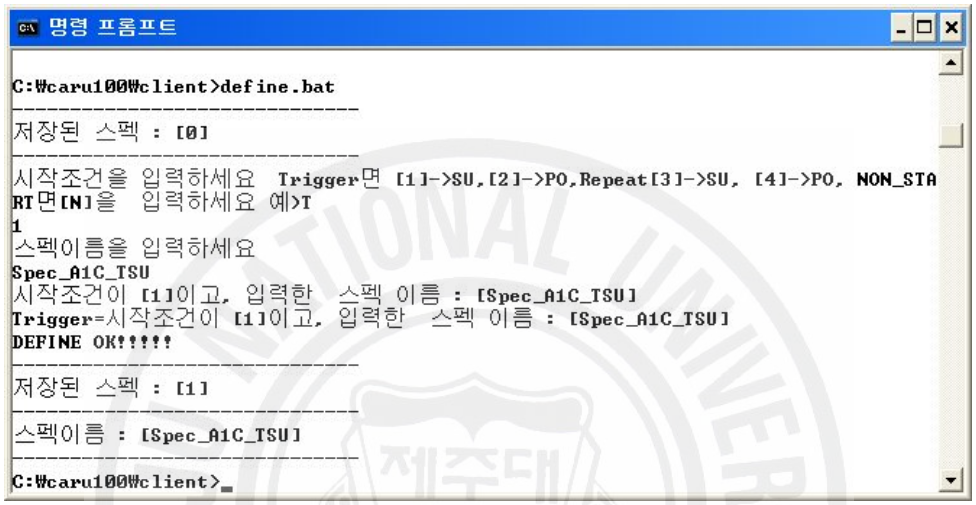

그림 35. 응용에서 ECSpec을 등록하는 화면

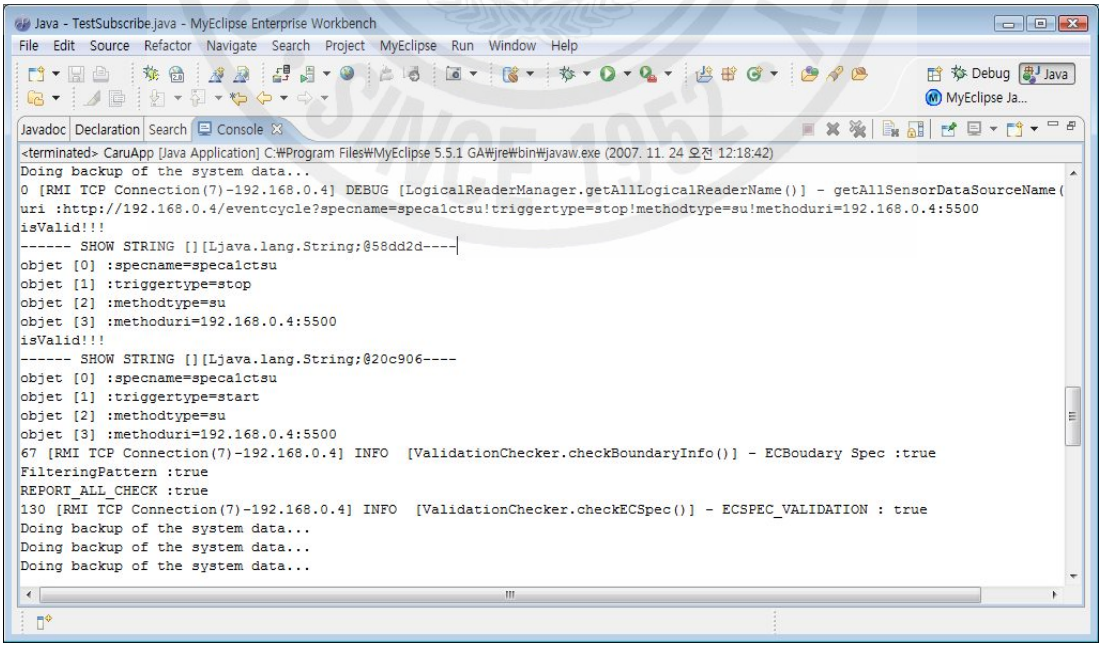

그림 36. Spec A1C가 미들웨어에서 처리되는 화면

그림 37은 ALE 인터페이스로 제공되는 메소드중 subscribe 메소드를 호출하는 것으로 미들웨어에 등록되어 있는 Spec_A1C_TSU라는 스펙을 이용하여 서비스를 요청하는 화면이다.

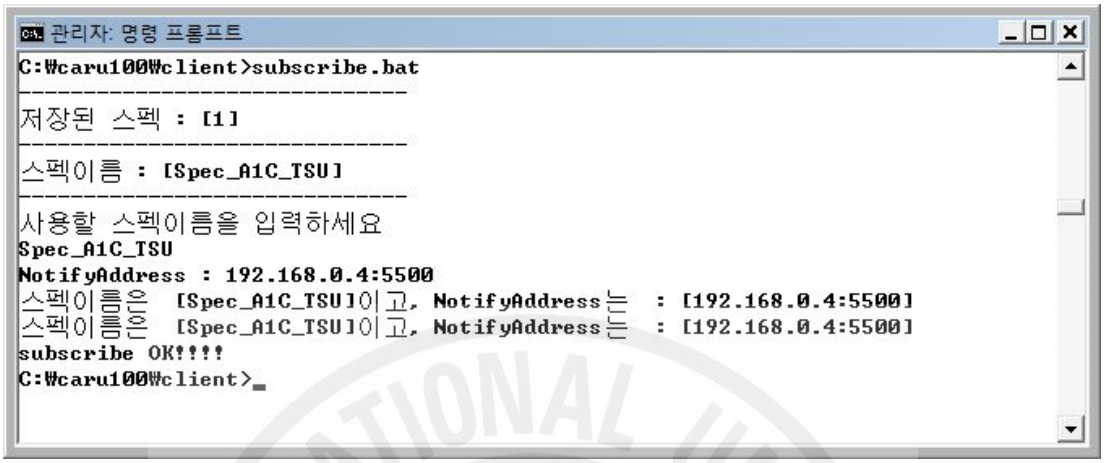

그림 37. Subscribe 메소드를 호출하여 서비스를 요청하는 화면

그림 38은 응용에서 subscribe 메소드를 호출했을 때 처리되는 것으로 ECSpec에 기술된 시작트리거를 기다리고 있는 화면이다.

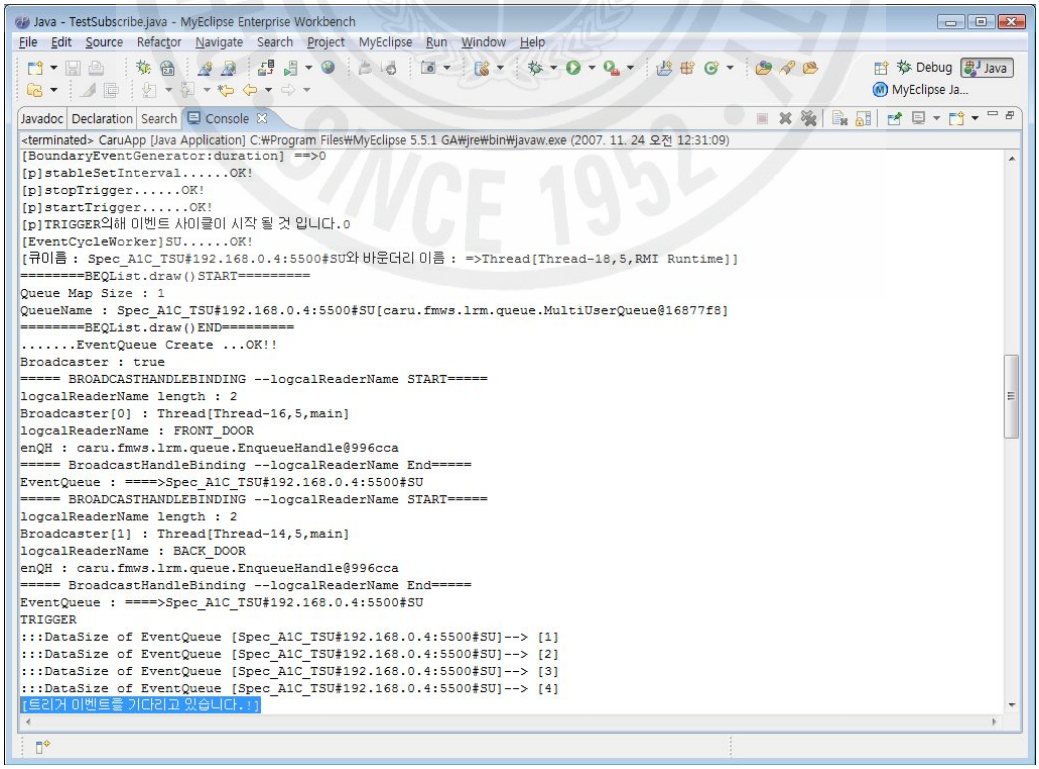

그림 38. subscribe를 처리하는 미들웨어 모습

그림 39는 트리거 조건을 기다리는 미들웨어에게 시작 트리거를 발생시키는 화면이다.

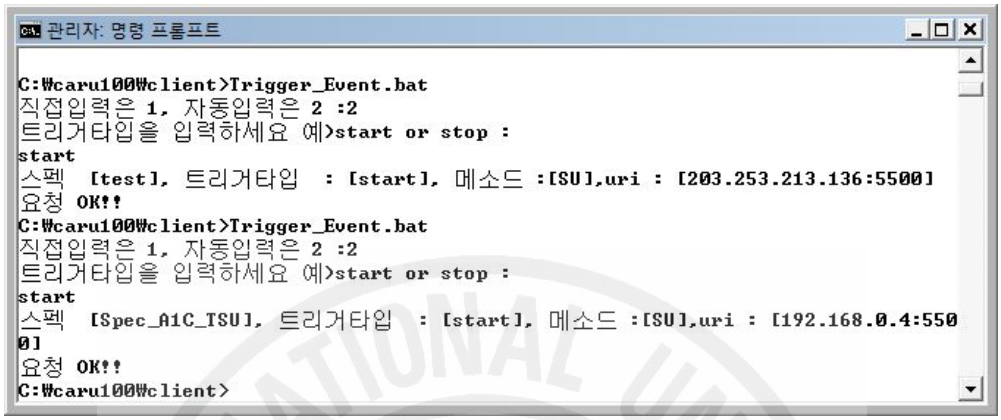

그림 39. 트리거 이벤트를 보내는 응용

그림 40은 트리거 이벤트를 받고 이벤트 사이클이 시작된 화면이다.

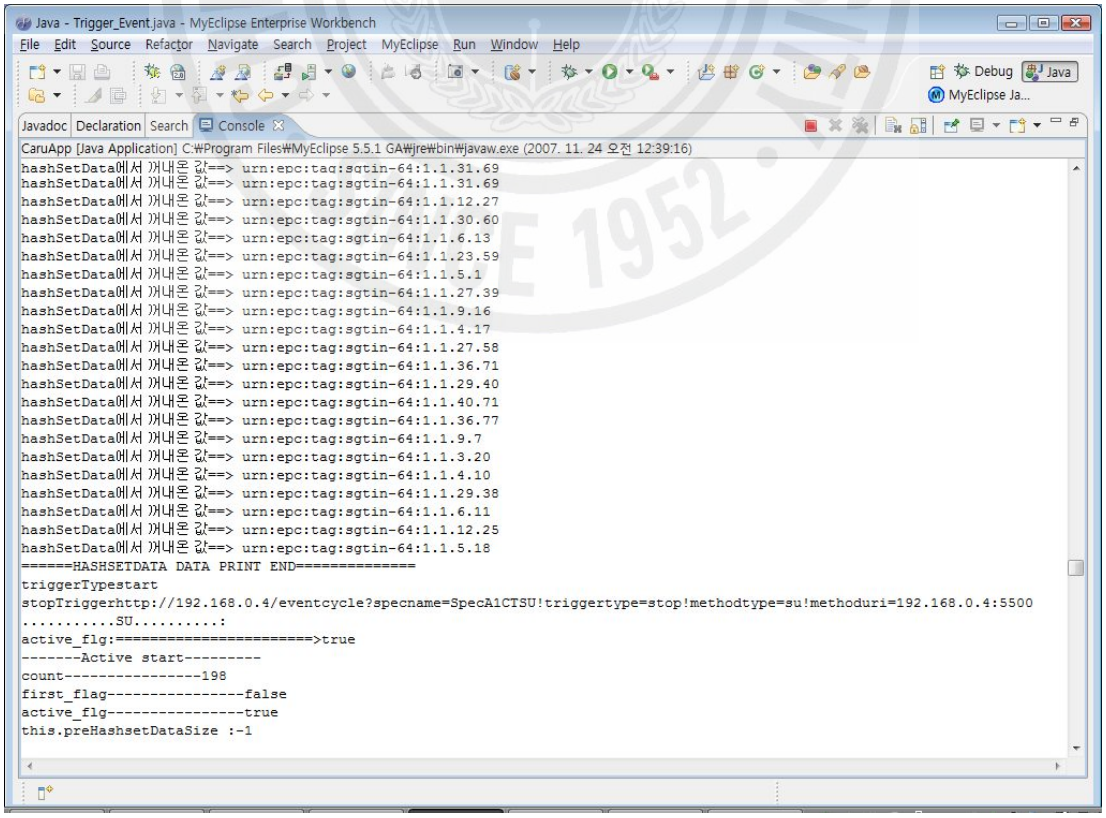

그림 40. 트리거 이벤트를 받고 이벤트 사이클이 실행된 화면

그림 41은 응용에서 정지 트리거 이벤트를 보내는 화면이다.

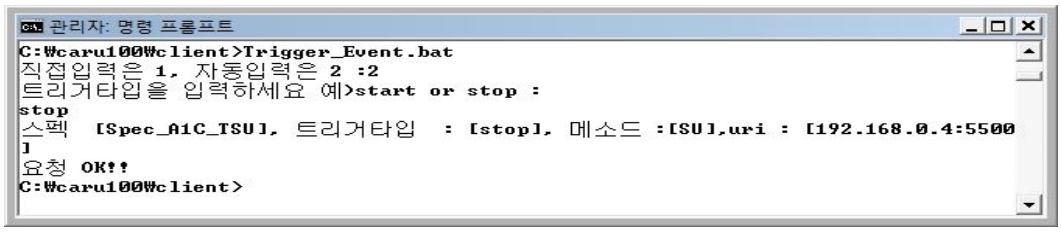

그림 41. 정지 트리거 이벤트를 보내고 있는 응용

그림 42는 정지 트리거 이벤트를 받고 이벤트 사이클이 종료되는 화면이다.

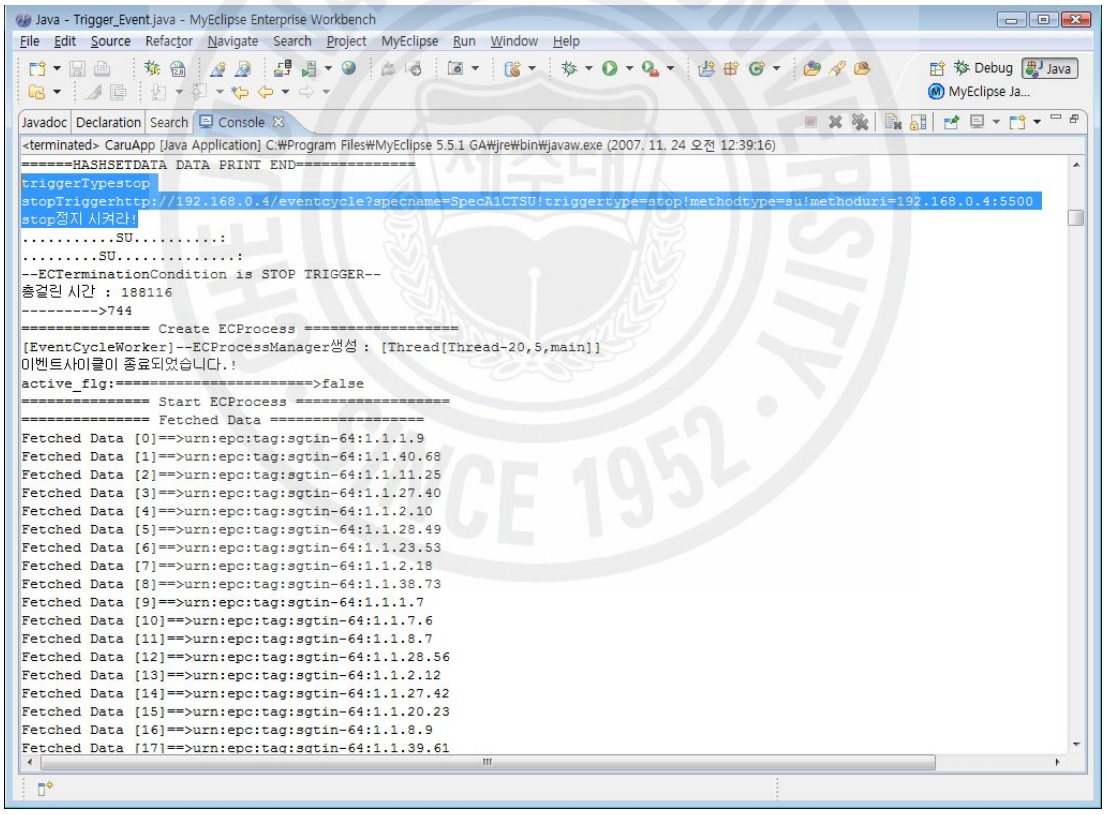

그림 42. 정지 트리거 조건을 전달 받고 이벤트 사이클이 종료되는 화면

그림 43은 이벤트 사이클 동안 읽어온 데이터를 연산 처리하는 화면이다.

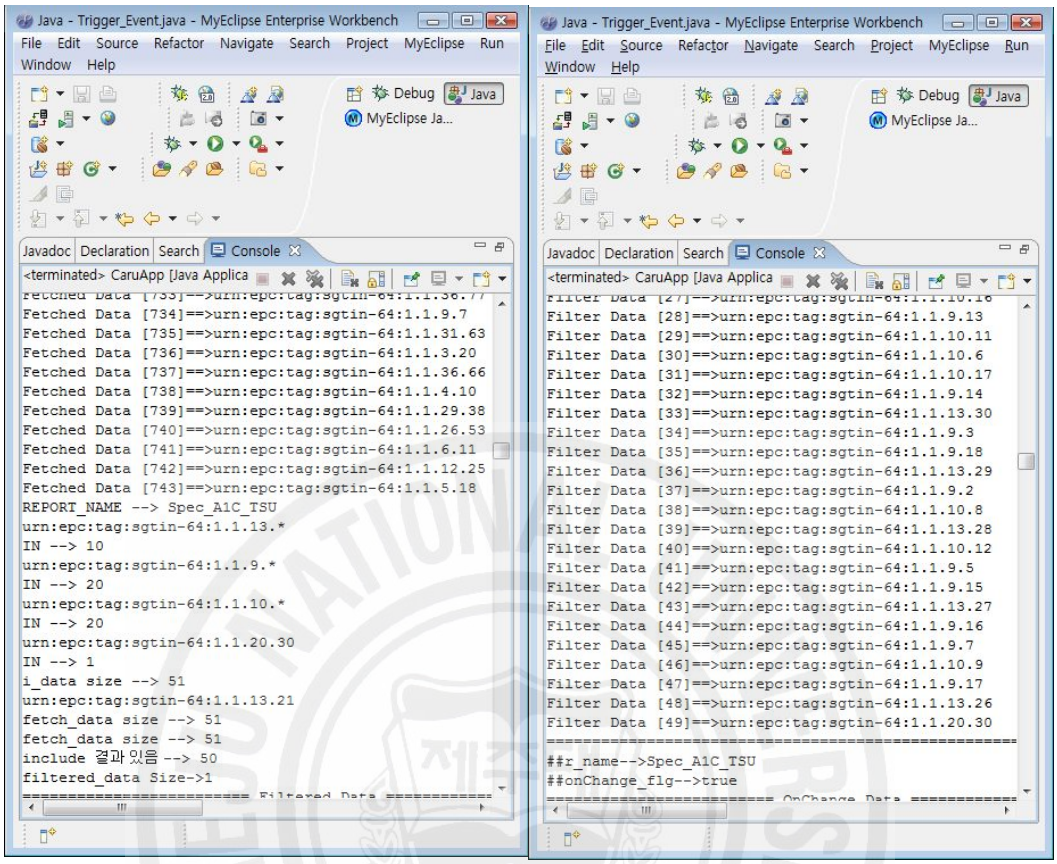

그림 43. 패치 된 데이터(fetched data) 와 필터 된 데이터(filter data)

이벤트 사이클 동안 총 743개의 읽어온 데이터의 결과는 아래와 같다. 리포트 이름은 Spec_A1C_TSU이고 필터 조건 urn:epc:tag:sgtin-64:1.1.13.에 만족하는 개수는 10개, 필터 조건 urn:epc:tag:sgtin-64:1.1.9.*에 만족하는 개수는 20개, 필터 조건 urn:epc:tag:sgtin-64:1.1.10.*에 만족하는 개수는 20개, 필터 조건 urn:epc:tag:sgtin-64:1.1.20.30에 만족하는 개수는 1개 총 51개의 데이터가 include pattern에 만족한다.

하지만 exculde pattern의 조건 urn:epc:tag:sgtin-64:1.1.13.21의 값이 1개 포함되었으므로 응용이 최종 받게 될 데이터는 50개임을 알 수 있다.

50개의 데이터 중 그룹핑 연산을 통해 다음과 같은 결과를 얻을 수 있다. 그룹핑 조건 urn:epc:tag:sgtin-64:1.1.9.*의 그룹이름은 1.1.9.*이며 그 그룹에 속한 개수는 20개이고, 그 나머지는 default 그룹에 속하며 개수는 30개이다.

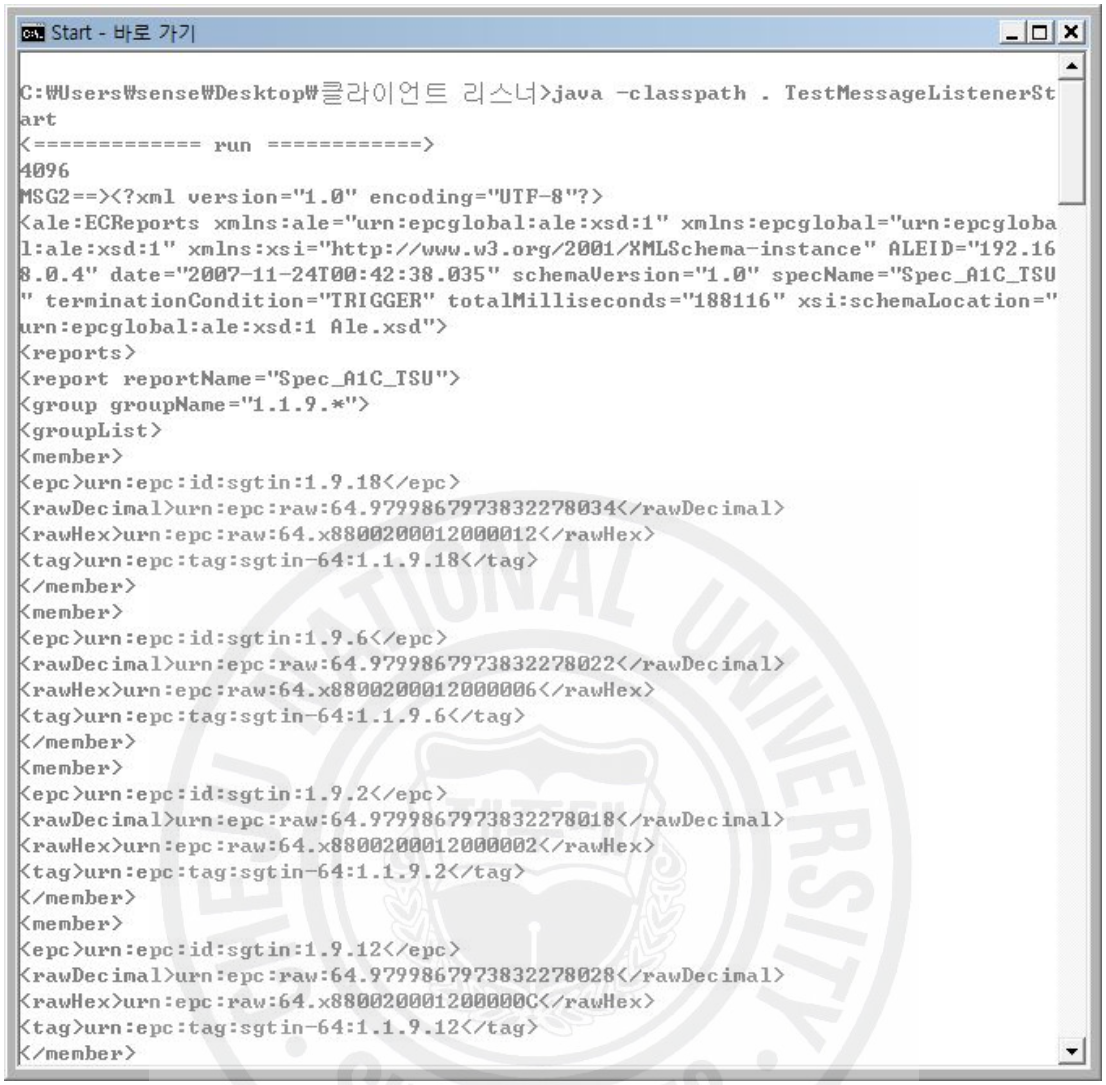

그림 44. ECReports

그림 44는 응용에서 받은 리포트를 XML로 출력한 화면으로 epc, tag, rawDecimal, rawHex 값으로 각각 출력된 것을 볼 수 있다.

4) 시스템 성능 분석

본 논문에서 제안하는 ALE 기반 RFID 미들웨어의 성능 분석은 기존에 연구된 다른 RFID 미들웨어들과 성능을 비교하지 않았으며, 실질적으로 구현한 미들웨어가 ALE 스펙에 준하였는지 확인하고 RFID를 이용하는 환경에서 운영될 수

있는 가능성을 보이기 위하여 미들웨어 자체의 실행 상황을 분석하였다.

표 24는 응용이 ALE 인터페이스를 이용하여 각 메소드를 호출할 때 미들웨어에서 처리되는 결과를 정리 하였다.

표 24. ALE 인터페이스 호출 테스트 결과

스 펙 명	메소드명	비 고	결 과
Spec_A1C_TSU	define	스펙 리스트에 추가 처리	ok
	undefine	스펙 리스트에서 삭제 처리	ok
	subscribe	서비스 요청 리스트에 추가 처리 이벤트 큐 생성 이벤트 사이클 워커 생성 리포트 생성 및 전송	ok
	unsubscribe	서비스 요청 리스트에 삭제 처리 이벤트 큐 삭제 처리 이벤트 사이클 워커 삭제 처리	ok
	poll	서비스 요청 리스트에 추가 및 삭제 처리 이벤트 큐 생성 및 삭제 처리 이벤트 사이클 워커 생성 및 삭제 처리 리포트 생성 및 전송	ok
	immediate	서비스 요청 리스트에 추가 및 삭제 처리 이벤트 큐 생성 및 삭제 처리 이벤트 사이클 워커 생성 및 삭제 처리 리포트 생성 및 전송	ok
	getECSpecNames	-	ok
	getECspec	스펙 리스트에서 ECSpec 목록 불러오기	ok
	getSubscribers	서비스 요청 리스트에서 불러오기	ok

제안한 ALE 스펙 기반 RFID 미들웨어의 효율성 및 미들웨어의 자체의 성능을 확인하기 위하여 미들웨어가 차지하는 리소스의 점유율을 정량 측정 하였으며, 리소스 점유율 측정은 Java 1.5 SDK 버전에서 지원하는 JConsole을 사용하여 로드된 클래스와 쓰레드, 메모리의 사용량을 확인하였다[17].

그림 45-48은 초기 미들웨어 실행시 Summary로 미들웨어를 실행하기 위한 기본 스레드와 메모리 사용량, 로드된 클래스 등을 나타낸 화면이다.

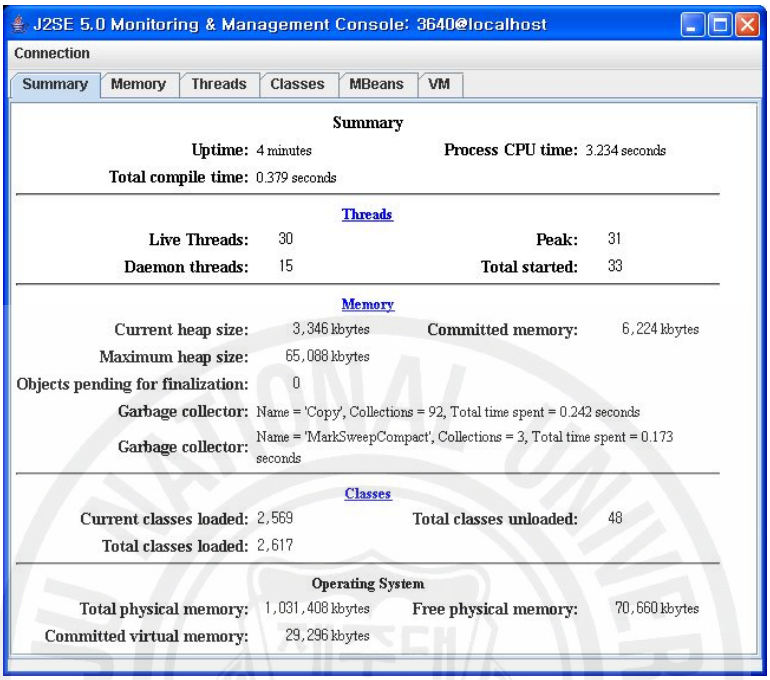

그림 45. 초기 미들웨어 실행시 Summary

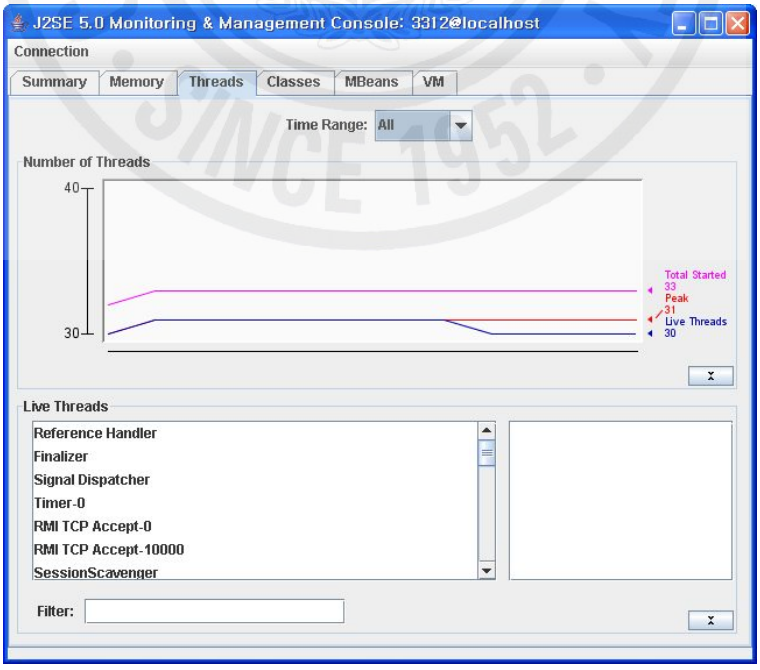

그림 46. 미들웨어 실행시 스레드 현황

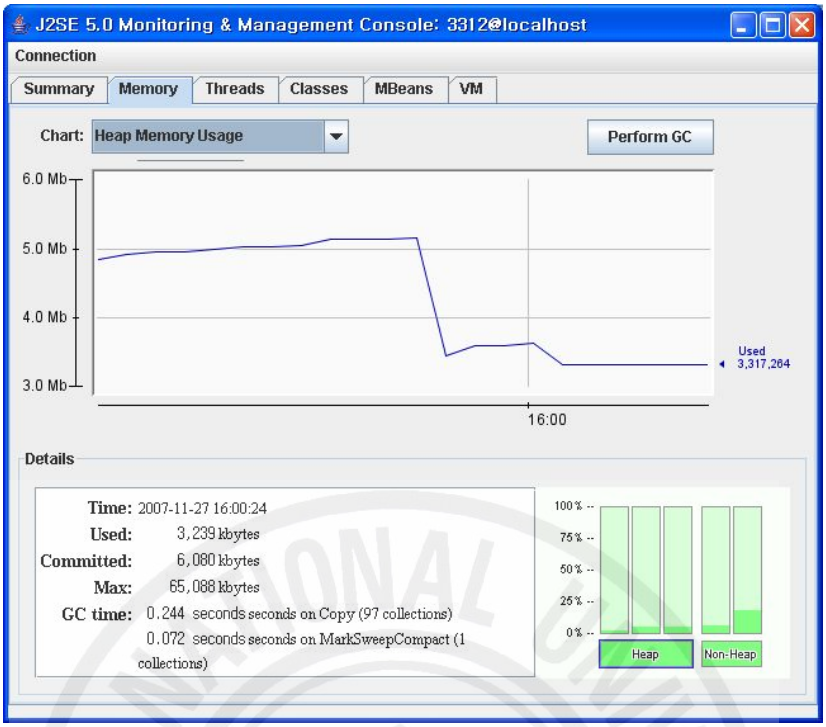

그림 47. 초기 미들웨어 실행시 메모리 사용량

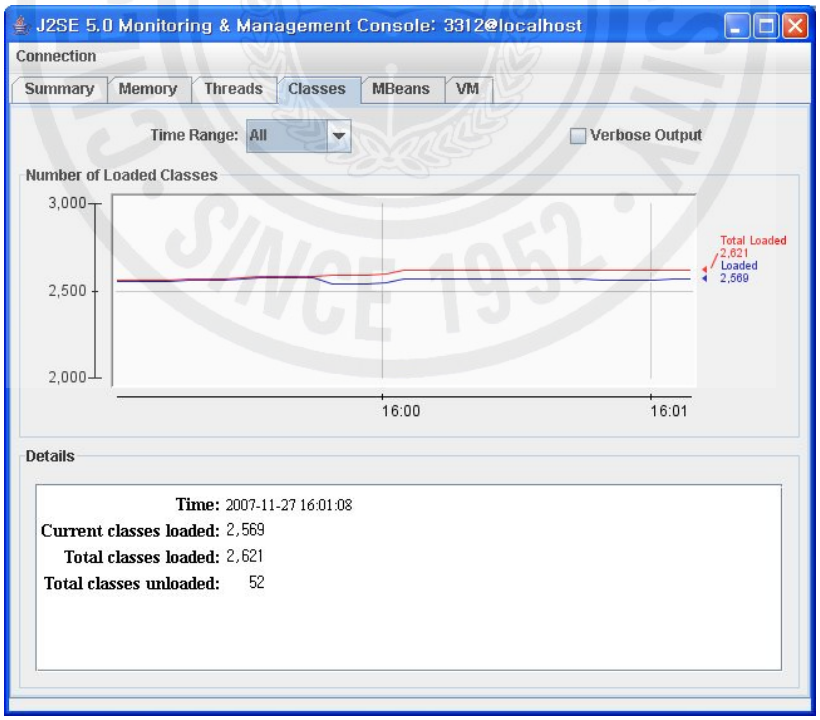

그림 48. 초기 미들웨어 실행시 클래스 로드량

그림 49와 그림 50은 여러 개의 Subscribe를 호출 했을 때 미들웨어의 실행 화면으로 서비스를 처리하기 위해 일시적으로 생성되는 스레드와 지속적인 일을 처리하는 스레드을 한 눈에 알아볼 수 있었다. 동시에 여러 개의 요청 처리시 시스템에는 큰 차이가 없는 것으로 확인 되었다.

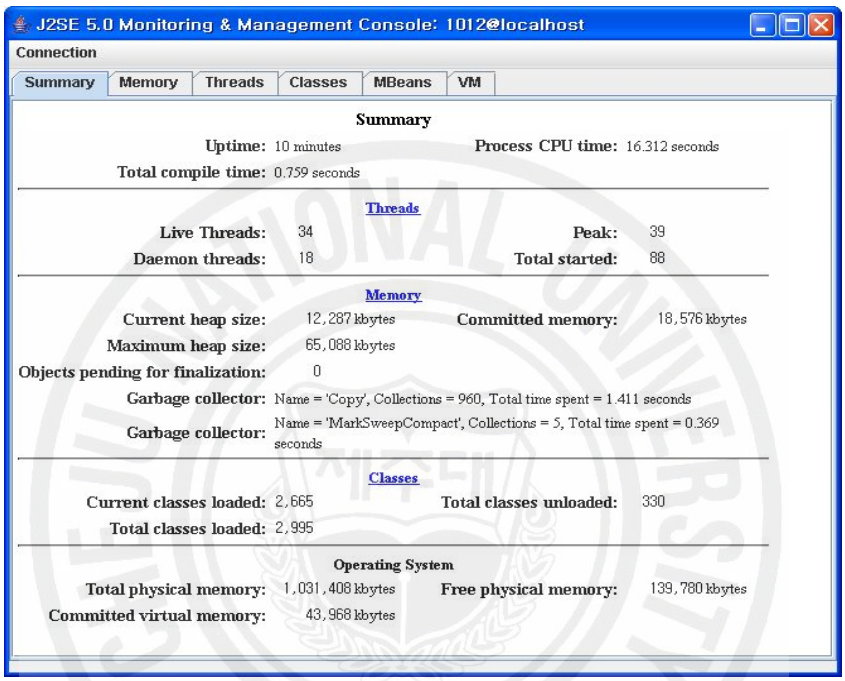

그림 49. 여러 개의 Subscribe 처리시 미들웨어 Summary

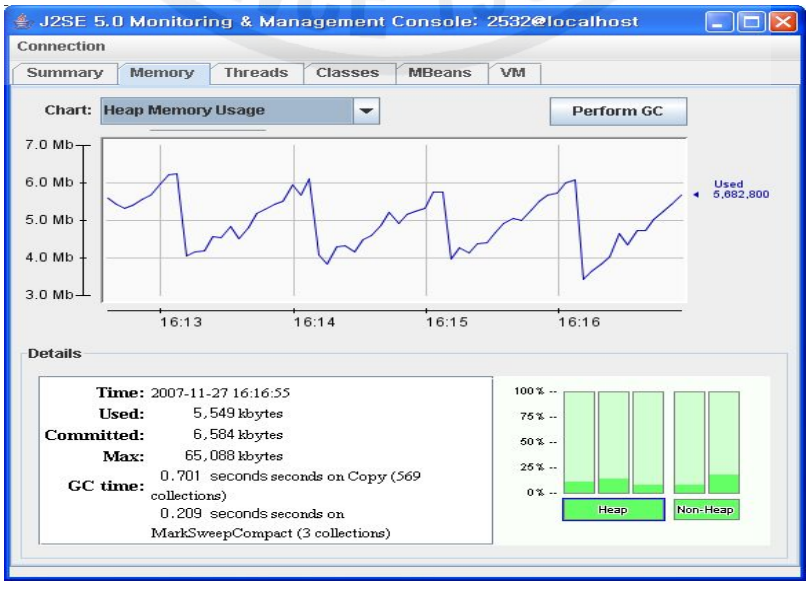

그림 50. 여러 개의 Subscribe를 처리시 메모리 사용량

그림 51은 UnSubscribe를 처리시 스레드 사용량을 나타낸 것으로 정상적으로 실행됨을 알 수 있었다.

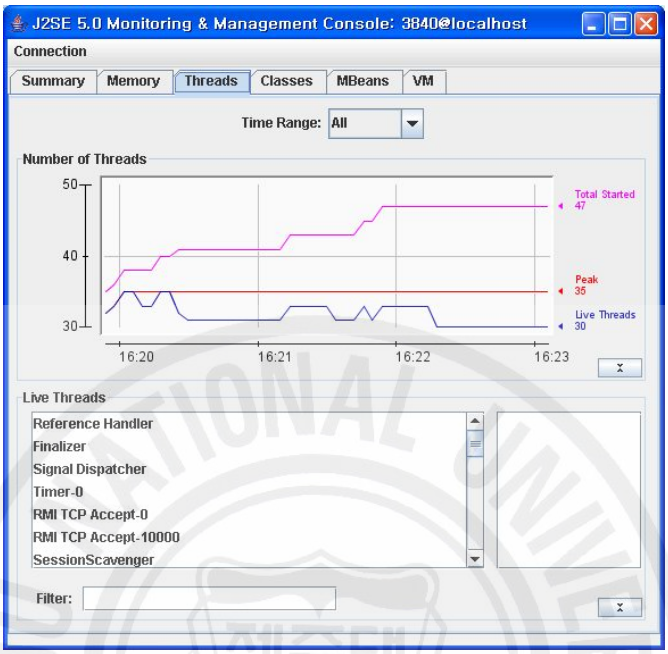

그림 51. UnSubscribe를 요청 했을 때

4. RFID 미들웨어 현장 적용 사례

1) OS 임베디드 항공물류용 복합단말기 개발사업 목적

RFID 기반 항공물류용 OS 임베디드 탑재 복합단말기 시험 및 서비스 기술 개발을 통한 항공물류 기반의 RFID 환경구축은 동북아 경제중심에 위치하고 있는 우리나라의 지리적 특성상 동북아 물류 HUB 화를 위한 국제 경쟁력을 확보할 수 있는 전략 사업이다. 이를 통해 RFID 기반 H/W 와 복합물류 S/W 시스템의 조기 상용화 기회를 선점할 수 있으며 제품의 상용화시 한.중 물류의 50% 이상의 시장점유가 가능하고 일본 및 동남아 진출 시 10배 이상의 확산 효과가 기대된다. 특히 우리나라의 국제항공화물 운송에 있어 상당부분을 차지하고 있는 한-미간 항공화물 운송에서의 실증시험을 통해 '21 세기 동북아 물류 중심 국가 실현'이라는 국가적 과제의 실현에도 크게 일조하게

될 것이다. RF 통신 관련 사전 시뮬레이션 및 현장분석을 통하여 개념적 응용기술과 실 현장시험 응용기술과의 차이점을 해소하고 항공화물 SW 의 개발 시점에 ㈜한진의 국제 택배 업무에 시범 적용하여 SW 패키지의 실제 현장 검증을 통해 문제점과 개선점을 도출하고 SW 의 구축 레퍼런스를 확보하며 나아가 SW 패키지 상품화 하고자 한다.

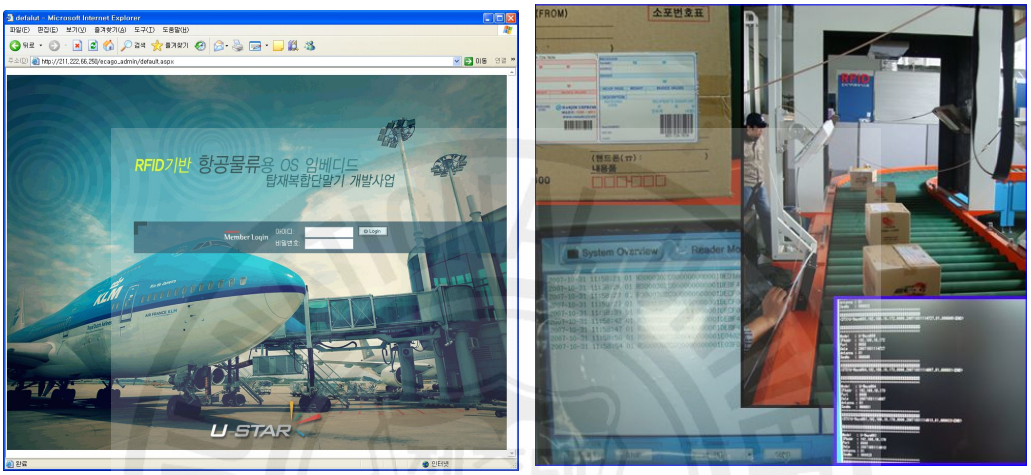

그림 52. 인천공항 수하물 관리 미들웨어시스템에 적용

2) 현장 통합테스트 및 시범운영 개요

'RFID 기반 항공물류용 OS임베디드 탑재복합 단말기 개발사업' 2차 년도 사업을 통해 각 중과제의 개발 결과물을 국제 택배 업무(LA-국내착)에 통합 운영한다.

(1)테스트 대상 : 1중 과제 결과물 항공화물용 H/W 적용
2중 과제 결과물 항공화물용 S/W 적용
3중 과제 결과물 통합시험결과 및 서비스 적용

(2)적용대상 : LA발 국내착 실제 국제 택배 화물에 부착 적용
400 Tags(시범운영 차수별 각 200 Tags)

3) 현장 통합테스트 및 시범운영 일정

시범 사업 테스트는 현장통합테스트와 1차, 2차 시범운영을 거쳐 진행을 하였고 그 결과는 표 26, 표 27, 표 28과 같이 정리하였다.

(1) 기간 : 2007. 11. 5 ~ 2007. 12. 6

그림 53. 현장통합테스트 및 시범운영 일정

(2) 현장 통합 테스트

표 25 . 현장 통합 Test 분석표

날짜	접수 및 발권	Build Up	항공사 반입	항공사 반출	물류센터 반입	Break Down	통관 완료	비고
11.15	10	10	정상인식	정상인식	정상인식	10	10	
11.16	28	28	정상인식	정상인식	정상인식	28	26	S/W 처리문제
11.17	54	54	정상인식	정상인식	정상인식	49	54	태그가 상단을 향하지 않는 경우 리더 읽지 못함.
11.19	30	30	정상인식	정상인식	정상인식	16	27	단말기에서 정확한 자료 집계가 안 되었음
11.22	54	49	정상인식	정상인식	정상인식	13	41	인식률 저하
11.23	98	87	정상인식	정상인식	정상인식	86	92	

본 현장통합테스트는 시범운영에 앞서 단위 테스트, 통합 테스트, 현장 테스트를 통해 문제점을 도출하고 개선하는 과정을 마친 S/W, M/W 및 H/W에 대해 LA의 ㈜한진 물류센터에서의 접수(발권)부터 ㈜한진 인천 물류센터 내의 통관완료과정에 이르기까지 실제 국제 택배 업무 현장에 적

용, 업무 process 진행 간 발생할 수 있는 사항을 사전 점검하고 문제점을 해결하였다.

그림 54. 현장에서 점검하는 화면

(3) 1차 시범운영 결과

표 26 . 1차 시범운영 결과 분석표

날짜	접수 및 발권	Build Up	항공사 반입	항공사 반출	물류센터 반입	Break Down	통관 완료	비고
11.28	100	90	정상인식	정상인식	정상인식	88(1)	86	웹페이지 통계화면 합산 누락
11.29	100	90	정상인식	정상인식	정상인식	94	93	

(4) 2차 시범운영 결과

표 27 . 2차 시범운영 결과 분석표

날짜	접수 및 발권	Build Up	항공사 반입	항공사 반출	물류센터 반입	Break Down	통관 완료	비고
11.28	100	82	정상인식	정상인식	정상인식	83	77	
11.29	100	94	정상인식	정상인식	정상인식	90	91	조업간 1개 통관 완료 미 통과

현장테스트 및 현장통합테스트를 통해 각종 문제점을 도출하고 이를 개선하여 원활하게 시범운영을 수행할 수 있었다. 인식률을 통하여 장비의 성능을 본다면 일부 태그의 인식률 저하로 인하여 전체 인식률영향을 주었

으나, 오류태그로 추정되는 데이터를 제외 하고 인식률을 계산하면, 업계에서 요구되는 통상적인 인식률 수준을 유지할 것으로 예상된다.

4) 시범 사업 테스트의 결론

현장테스트를 통해서 개발된 S/W, M/W 및 복합단말기의 문제점을 도출, 개선시키고 현장에 맞게끔 Configuration 하였다. 계속된 6일차에 걸친 현장통합테스트를 통해 현업적용에서의 문제점을 도출해 냈으며, 대략 18가지의 조치를 취하여 최적의 시범운영이 가능토록 하였다. 2개차에 걸친 시범운영을 통해 'RFID 기반 항공물류용 OS임베디드 탑재 복합단말기 개발사업'의 산출물을 직접 항공물류 현장에 적용해 보았다. 이번 시범 운영에서는 EPC Network 에 기반한 전체 시스템을 구축 하였으며, ㈜한진의 전산망으로부터 정보를 수집, 이용하는 Legacy System도 적용하였다. RFID 리더기에서의 Tag 인식률은 만족할 만한 수준이었으며, 전체 EPC Network 모델상의 RFID리더, Middle Ware, EPCIS, DB, 관리자용 Web Application 등 각 구성요소간의 Interface는 완벽한 수준을 보여주었다. 이러한 결과는 RFID 시스템 도입의 가장 큰 걸림돌로 여겨지는 기술적 한계가 상당부분 해소되었음을 증명하고 있다. 한편으로, 본 복합단말기는 디스플레이 장치가 결합된 OS 임베디드형으로서 첫째, 소형화를 통한 현장 적용이 용이하며, 둘째, 디스플레이 장치를 활용한 다양한 응용 가능성을 갖고 있고, 셋째, OS 임베디드로써 범용적 기능의 구현이 가능함 등의 다양한 장점을 가지고 있다. 이는 기존의 단순한 RFID 리더의 수준에서 벗어나 OS 임베디드 복합단말기라는 신개념의 제품이 무한한 발전 가능성이 있음을 보여준다. 하지만 항공 물류 시스템의 특성상 100%의 신뢰성이 없으면 현장에서 적용이 어렵다. 이를 해결하기 위해서는 현 항공물류 조업 시스템을 RFID 기반 항공물류 조업 시스템으로 수정할 필요가 있다고 판단된다. 현 항공물류 조업 시스템에서는 작업자가 직접 Bar Code를 Scan함으로써 장애를 바로 인식, 그 물건에 대해 재 Scan 작업하여 신뢰성이 100%에 가깝다. 이와 마찬가지로 RFID 기반 항공 물류용 OS 임베디드 탑재 복합단말기를 적용할 때도 각각의 화물이 정확하게 인식이 되었는지를 시스템 및 작업자가 인지할 수 있는 방안이 지속적으로 연구 되어야 할 것으로 판단된다.

V. 결 론

본 논문에서는 RFID 표준화 동향 및 현황에 대해 알아보고 먼저 다수의 RFID 미들웨어를 분석하여 기존 미들웨어의 문제점을 정리하고 현재 사실상 RFID 기술의 국제 표준을 이끌고 있는 EPCglobal의 ALE 스펙을 분석하여 RFID 미들웨어를 설계하였고 그것을 구현한 RFID 미들웨어의 효율성 및 미들웨어의 자체의 성능을 확인하기 위하여 미들웨어가 차지하는 리소스의 점유율을 JConsole을 이용하여 정량 측정 하였다. 또한 다수의 서비스 요청 처리할 때 발생되는 큐의 오버플로 현상 및 데이터 손실을 예방하였고 시스템의 부하 없이 효율적으로 처리할 수 있음을 확인하였다.

기존의 RFID 미들웨어들은 다양한 기능 및 데이터 처리 방법을 제공함으로써 데이터 처리의 효율성 면에서 뛰어나지만 응용 서비스와 미들웨어 간 독립성을 제공하기 위한 사실상의 표준을 따르지 않음으로써 향후 다양한 비즈니스 로직과의 상호 운영이 필요할 경우 서로 다른 제약으로 인하여 문제점이 발생할 수 있다.

본 논문에서 구현한 RFID 미들웨어 시스템은 ALE 스펙을 기반으로 설계하여 응용 개발자에게 ALE 인터페이스를 제공하므로써 어떠한 응용서비스라도 동적으로 쉽게 ECSpec을 등록하여 서비스를 요청할 수 있고 그 결과를 언제 어디서나 다양한 코드 형식으로 제공 받을 수 있다.

또한 RFID 미들웨어에서는 각 응용서비스가 요청한 의미 있는 이벤트 데이터로 생성하는 일에만 관심 있을 뿐 응용서비스와는 밀접한 관계가 없기 때문에 응용서비스는 미들웨어에 종속적이지 않으므로 응용서비스는 보다 효율적으로 RFID 시스템 운영이 가능하게 되었다.

본 논문에서 구현한 ALE 기반 RFID미들웨어는 현재 여러 시범사업에 적용하여 현장 테스트를 통해 실제 환경에 적용될 수 있도록 고도화 작업을 진행중에 있다.

참고문헌

[1] Intel, “Building the Digital Supply Chain: An Intel Perspective,” Intel Corp., 2005.

[2] METRO, “The METRO Group Future Store Initiative,“ http://www.futurestore.org.

[3] http://www.infoworld.com/article/04/07/23 /HNrfidimplants1, 2004.

[4] The EPCglobal Architecture Framework. EPCglobal Final Version of 1 Jul. 2005.

[5] 김명준, 진성일, 이미영, “UbiCore : XML 기반 RFID 미들웨어 시스템,” 한국정보과학회 논문지, 제33권 제6호, 2006.

[6] Josef Preishuber-Pflugl, “Sensor Mapping in ISO/IEC 18000,” ISO/IEC JTC1/SC31/WG4/SG3 Paris Meeting, Aug.2006

[7] “Enterprise Information Architecture for RFID and Sensor-Based Services,” Oracle White Paper, 2006.

[8] “TagsWare: Agile RFID Solutions,” http:// www.tagsware.com.

[9] Sun Microsystems, “The Sun EPC Network Architecture,” Technical White Paper, 2004.

[10] Oat Systems & MIT Auto-ID Center, "Technical Manual: The Savant Version 0.1(Alpha)," Feb. 1, 2002.

[11] EPCglobal, "EPC Tag Data Standards Version 1.1 Rev.1.27", 2005

[12] EPCglobal, "EPCglobal Tag Data Translation 1.3", 2006.

[13] 최용식, 전영준, 박상현, 한수, 신승훈, “RFID 미들웨어 환경에서 센서 노드의 생존성 향상과 효율적인 프로토콜 설계를 위한 연구”, 한국정보과학회, 제 33권 제 2호, pp.68-73, 2006. 10.

[14] 홍연미, 변영철, “ALE 기반 RFID 미들웨어 설계 및 구현”,

한국해양정보통신학회논문집, 제11권, 제4호, pp.648-655, 2007.

[15] EPCglobal, “The Application Level Events (ALE) Specification Version 1.0,” September 2005.

[16] TDS-EPCglobal Tag Data Standards, http://www.epcglobalinc.org/standards/uhfc1g2/uhfc1g2_1_1_0-standard-20071017.pdf 'Standards'

[17] Using JConsole to Monitor Application : http://java.sun.com/developer/technicalArticles/J2SE/jconsole.html

[18] Alien Technology. “RFID Readers,' http://www.alientechnology.com/products/rfid_readers.php

[19] Kun-Lung Shyh-Kwei Chen, Philip S. Yu, "Processing Continual Range Queries over Moving Objects Using VCR-Based Query Indexes," MobiQuitous 2004, pp. 226-235, 2004.

[20] A Basic Introduction to RFID technology and Its use in the supply chain, http://www.printronix.com/uploadedFiles/Laran_WhitePater_RFID.pdf, January 2004.

[21] H.K. Launches RFID Supply Chain Project, RFID journal, http://www.rfidjournal.com/article/articleview/1630/1/1, June 2005.