SIP 프레즌스 모델 기반 센서 태그의 상태이력 추적

(Tracking and Tracing the Status Changes of Sensor Tags based on the SIP Presence Model)

김 동 욱 † 홍 진 표 ††
(Dong-uk, Kim) (Jin-pyo, Hong)

요 약 RFID객체 이력추적기술은 EPCglobal에서 제안하는 Discovery Service를 중심으로 다수의 연구가 진행 중이다. 하지만 객체의 상태변화를 모니터링 하는 기능을 제공하지 않으며, 객체의 이력정보를 특정 서버에서 취합하여 관리해야 한다는 단점이 있다. 이 문제를 해결하기 위해서는 객체에 대한 상태정보를 제공하기 위한 모델이 필요하며 RFID 객체의 URL을 나타낼 수 있는 식별자 역시 필요하다. 이러한 문제점을 보완하고자 SIP 이벤트 통지 및 프레즌스 모델을 활용하여 RFID 객체의 상태정보를 Track & Trace하기 위한 새로운 방법을 제안한다. 본 논문에서는 객체의 URL을 얻기 위해서 기존 ONS를 그대로 활용하되 성능저하가 없도록 하여 backward-Compatibility를 충족하도록 하였다. 또한 이렇게 얻어진 URL을 이용하여 RFID 객체 혹은 객체의 상태정보를 관리하고 있는 요소에 접근하기 위해 SIP 프로토콜을 사용하도록 하여 객체관리를 위해 개방형 인터페이스를 사용하였다. SIP 메시지 조합을 통해 객체의 이력정보를 얻을 때 DS와 같은 추가적 요소 없이도 객체의 Track & Trace가 가능하게 하였다. 이 방식을 활용하여 지금까지와는 달리 분산취합구조를 이용하여 객체에 대한 상세이력정보 및 모니터링을 제공할 수 있도록 하였다.

키워드 : RFID 이력추적 및 모니터링, SIP 이벤트 패키지, 프레즌스, 센서태그

Abstract The EPC-Discovery Service (EPC-DS) is a good representative of the RFID Track & Trace. But this mechanism has several problems. EPC-DS uses centralized server scheme which may arise bottle-neck state and that cannot provide detail trace information of a RFID object. In addition, a trace node requires direct access method to a RFID object or an element which has information of the RFID object for Track & Trace. In this paper, we propose a novel RFID Track & Trace mechanism which based on the SIP presence model and SIP event notification. This mechanism can provide detail trace information and monitoring function, and also can rid the bottle-neck section by combination of SIP methods instead of centralized element.

Key words : RFID Track and Trace, SIP Event Package, Presence, Sensor Tag

1. 서 론

RFID 기술은 사물의 고유한 ID를 단순히 인식하는 읽는 기능 중심에서 사물의 이력 정보를 관리할 수 있는 읽고 쓰기 기능, 그리고 센서와 결합하여 온도, 습도, 압력, 위치 등 주변 정보까지 감지하는 센싱 기능을 가지도록 발전하고 있다. RFID 태그는 내부 배터리와 자체 송신 장치 내장 여부에 따라 수동형 태그, 능동형 태그로 구분되며, 반능동형 태그는 전지를 이용하여 대기 상태에 있다가 리더로부터 활성화 명령을 수신했을 때만 깨어나서 전송하는 방식이다.

능동형 태그와 반능동형 태그는 자체전원을 가지고

· 본 논문은 2008년도 한국외국어대학교 학술연구비 지원에 의해서 연구되었음

† 학생회원 : 한국외국어대학교 정보통신공학과
ghost@hufs.ac.kr
†† 종신회원 : 한국외국어대학교 정보통신공학과 교수
jphong@hufs.ac.kr

논문접수 : 2008년 10월 15일
심사완료 : 2009년 3월 11일

있어 수동형 태그의 문제점인 인식률을 크게 개선한 점이 특징이다. 여기에 센싱기능을 추가한 것을 센서 태그라 부르며 식의약품 관리(Cold-chain), 혈액 관리, 자동차 및 교통 관리, 환경 관리 등 다양한 분야에 활용 할 수 있다.

EPCglobal의 RFID 네트워크는 주로 물류 유통에서 활용되어 사업체내 물품관리에 EPC-Information Service(EPCIS)를 활용하고 있으며 사업체간 물품거래와 같은 공급망 상에서 객체의 이동을 EPC-Discovery Service(EPCDS)를 활용하여 어떤 사업체의 IS를 거쳐 왔는지 여부를 추적하는 것으로 정의되어 있으나, 이에 대한 규격과 표준은 제정되어 있지 않다.

본 논문에서는 수동형 RFID 태그를 사용하여 대상의 ID획득을 통한 단순추적이 아니라, 센서와 태그 일체형인 센서태그가 부착된 경우와 태그와 별도로 주변에 존재하는 센서들 또는 sink node를 연관시킴으로써 객체의 상태정보(즉, 센싱 정보)의 변화를 지속적으로 추적하는 Track & Trace 모델을 제시하고자 한다. 이를 위해서 ID와 센서 정보의 연관성을 부여하는 센서-태그 정보와 이 정보의 보고 및 이력 축적을 위한 이력 정보를 수학적으로 표현하여 Track & Trace 정보 모델을 정의하고, 이를 기반으로 논리 평면에서 개념 모델을 정의하고, 분산 평면에서 Track & Trace를 위한 기능적 구성요소와 인터페이스를 정의한다. 다양한 응응 분야에 적용성을 높이고 개방형 인터페이스를 통해 비즈니스간 상호운용성을 확보하기 위해 IETF 표준인 SIP Presence Model을 응용한다.

또한, 객체의 상태정보를 원하는 응용에서 RFID 객체에 접근하기 위해서는 현재 RFID URN 형식이 아닌 객체 혹은 객체의 정보를 갖고 있는 요소에 접근할 수 있는 식별자가 필요한데, 본 논문에서는 RFID 객체의 상태에 대한 이력정보를 원하는 응용에서 객체정보에 접근하기 위한 식별자변환 방법을 제안한다.

객체의 상태정보의 생성량은 시간과 센서 정보의 변화량에 따라 기하급수적으로 증가하기 때문에 객체의 이력정보를 공유하는 서버는 대용량 데이터베이스를 가지며 대규모의 트랜잭션을 처리해야 하며, 통신 대역폭도 막대하게 소요되기 때문에 이에 대한 scalability 확보가 중요하다. 다수의 Track & Trace 서버를 분산 배치하더라도 RFID만을 가지고 서버와 이력정보의 위치를 알아내는 방안을 소개하며, 이력 추적이 가능한 도메인이 한정될 수 있기 때문에 관리 도메인이 달라도 끊김 없이 지속적으로 객체의 이력 추적을 가능하게 하는 메커니즘을 제시한다.

2. 관련연구

2.1 센서태그

기존의 ID 획득만 가능한 태그에 외부 환경 정보를 습득할 수 있는 센서와 자체 전원 공급을 위한 박형 전지(film battery)가 추가된 RFID 태그를 '센서 태그(smart active label 혹은 RFID sensor tag)'라고 부르며, 그 방식은 크게 반능동형 방식(semi-passive 혹은 semi-active)과 능동형 방식(active)으로 구분된다. 반능동형 센서 태그 기술은 기존의 수동형 태그의 단점인 짧은 인식거리, 낮은 인식률과 신뢰도를 보완할 수 있고 온도, 습도 등의 다양한 외부 환경정보를 제공할 수 있다[1].

현재 RFID 네트워크를 구성하는데 있어 가장 중요한 역할을 수행하는 EPCglobal은 수동형 태그뿐아니라 (반)능동형 태그도 수용하기 위한 EPC 네트워크 프레임워크 모델을 제시하고 있다. EPCglobal은 다음 그림 1과 같이 수동형/(반)능동형 태그들간의 관계와 네트워크상에 배치될 수 있는 타입을 통해 획득할 수 있는 ID

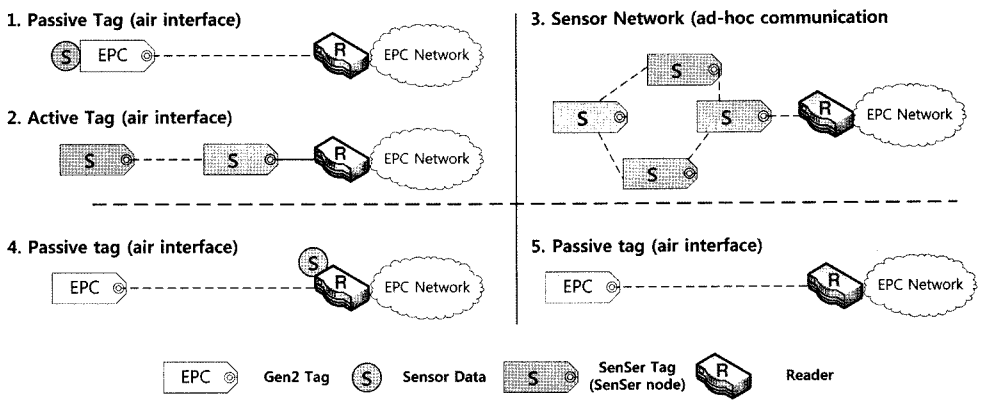

그림 1 센서태그와 EPC network 접속점에 관한 분류

정보, 센싱 데이터에 대해 정의하기 위한 연구와 이를 처리하기 위한 프레임워크 표준에 대해 연구를 진행하고 있다[2].

2.2 객체정보검색기술

EPCglobal 에서 물품정보를 얻기 위해 DNS 기반 구조의 검색 서비스를 하고 있는데 이를 ONS(Object Naming Service)라고 정의하고 있다.

ONS에 질의를 할 때 RFID 태그 URN에서 Serial Code를 제외한 FQDN 형식으로 질의를 한다. 따라서 반환된 접속정보를 이용하여 EPC-IS 접속 후 Serial code를 이용하여 객체에 대한 검색을 하게 된다. RFID 서비스를 위해 RFID 태그를 URN 형식으로 기술하고 객체검색을 위해 도메인 네임 형식으로 변환한다. 일반적으로 RFID URN은 아래와 같이 표시 할 수 있다[2].

urn:<NID>:<Code Indicator>:<CSS> NID: Namespace Identifier, CSS: Code Specific String 예시) urn:epc:id:sgtin.1.10.100

객체정보 조회를 할 때는 상품의 정보에 대한 정보만 제공하기 때문에 객체검색서버에서는 이 serial code는 관여하지 않고 종류 및 세부분류에 관여하는 문자열 까지만 사용하여 domain name으로 변환 한다.

국내외 표준에서 CSS의 마지막 Digit node(숫자, 위 예시의 경우 100)는 각 독립된 객체를 식별하기 위한 Serial code로 활용된다. 객체정보 조회를 할 때는 상품의 정보에 대한 정보만 제공하기 때문에 객체검색서버에서는 이 Serial code는 관여하지 않고 종류 및 세부분류에 관여하는 문자열까지만 사용하여 Domain name으로 변환 한다[3,4]

2.3 객체이력추적기술

RFID로 관리되는 물품의 이력정보, 다시 말해 물품의 상세 정보를 포함하는 유통과정에서 거친 지점의 정보 및 지점간 이동정보를 통합적으로 제공하는 서비스를 EPC Discovery Service(EPC-DS)라는 이름으로 정의하고 있으며, 그림 2와 같이 IBM에서는 EPC-DS의 한 방식을 구현하고 있다[5]. 이 방식의 특징으로 DS를 분산 구성하였으며 분산된 DS를 검색하기 위해 별도의 Naming Server를 두고 있다.

그림 3에서는 이력추적을 위한 PTSP(Product Trace Service Platform)구조를 나타내고 있다[6]. PTSP는 EPCDS를 경량화 하여 구현된 형태로 Discovery Service를 제공한다. 이 방식의 특징으로 IBM방식과는 달리 DS를 찾기 위한 별도의 Naming Server를 두지 않고 ONS 질의를 통해 DS를 알 수 있도록 별도의 NAPTR Resource Record를 두고 사용한다.

그림 3의 ONS 위임 체계에서는 ONS 쿼리의 결과인 "EPCDS+ws" 서비스를 활용하여 물품의 이동에 따른 이벤트를 등록하고 특정물품의 이력정보를 조회하는 메시지를 나타내고 있다. 등록, 조회할 때 물품의 Serial Code를 이용하여 개별 객체를 등록, 조회하고 있으며 ONS질의 결과는 Serial Code까지가 아닌 Item Code임을 알 수 있다.

한편, SRMS(SIP-based RFID Management System)에서는 SIP 프로토콜의 INVITE와 REGISTER method를 이용하여 RFID를 추적하는 방안을 제시하였다[7]. 이 연구는 표준 프로토콜을 채택하였다는데 의미가 있으나, 기본적으로 INVITE method는 향후 통신을 위한 호 설정 용도에 적합하지만 이력정보를 push model로 주기적 또는 상태정보 변화, 관리 도메인의 변경에 따른 추적에는 부적합하다.

3. RFID 센서태그를 이용한 상태정보 Track & Trace 모델과 Framework

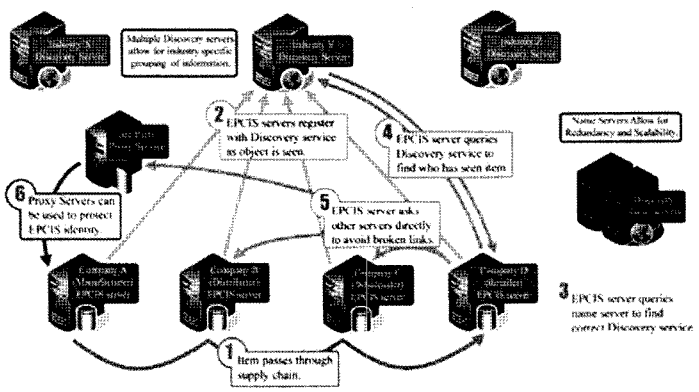

그림 2 IBM의 객체이력추적

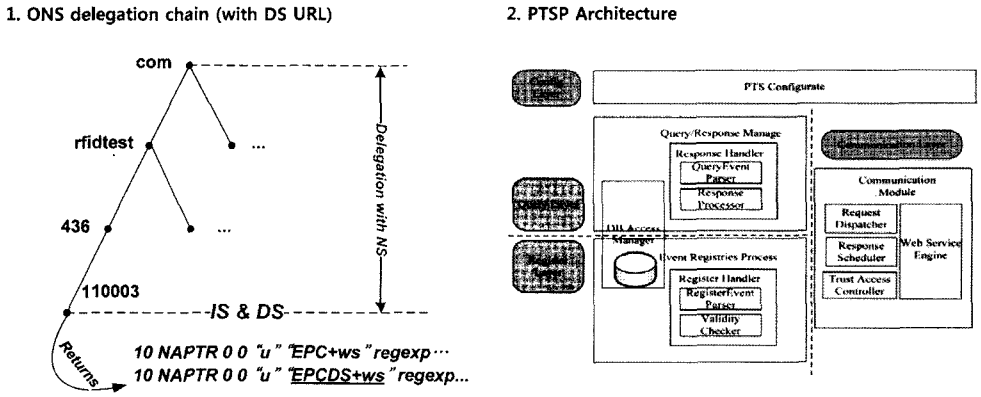

그림 3 PTPS의 객체이력추적

3.1 Track & Trace 정보 모델

태그와 센서들이 일체형인 센서 태그가 어떤 객체에 부착되어 있거나, 혹은 센서가 내장되지 않은 태그 근처에 센서들이 위치하여(예를 들어, 냉동차 내부의 온도 센서와 여기에 실린 식품) 객체 주변 센싱 정보를 얻을 수 있다고 가정한다.

1) 센서와 태그 결합된 정보 모델

시간 t에서 RFID reader R이 RFID 태그를 읽었을 때 얻은 식별자를 ID이라고 하고, 이 태그와 관련된 (associated) n개의 센서들로부터 얻은 sensor data를 S_1, S_2, ···, S_n이라 할 때, ID로 유일하게 식별되는 객체의 주변 정보를 Status(ID)라고 하면, 이는 다음과 같이 정의할 수 있다.

$$Status(ID) = \{R, S_1, S_2, \cdots, S_n, t\}$$

센서가 없이 태그만을 부착했을 경우, 통상적인 RFID를 통한 객체의 상태는 이의 특수한 경우이다.

$$Status(ID) = \{R, t\}$$

통상적으로, Status(ID)는 다음의 경우에 얻게 되며, 이를 추적 감시하는 노드로 보내지거나, 또는 이를 축적하는 서버(즉, Track & Trace Server 혹은 EPCglobal에서 Discovery Server)로 보내 축적하여 이력 정보를 제공할 수 있다.

- Read 명령에 의해(on-demand)
- 주기적으로
- 미리 설정된 센서 데이터의 한계를 초과했을 때

2) 객체의 이력정보 축적

Status(ID)를 m개 얻었다면, ID를 식별되는 객체의 이력은 다음과 같이 정의할 수 있다.

$$History(ID) = \{Status_1(ID), Status_2(ID), \cdots Status_m(ID)\}$$

만일, 새로운 Status(ID)가 발생했다면, 이력은 다음과 같이 축적된다.

$$History(ID) \leftarrow History(ID) \parallel Status(ID)$$

3) 객체 이력정보의 검색 및 모니터링

Track & trace는 보통 함께 쓰는 용어이지만 엄밀히 구분한다면, tracking은 현재 또는 최근의 상태를 알아내는 일이고, tracing이라 하면, 현재까지의 객체의 모든 정보의 변화 즉 History(ID)를 알아내는 일이다.

$$Tracking(ID): Status(ID) \leftarrow \text{tail of } History(ID)$$

$$Tracing(ID): History(ID)$$

3.2 Track & Trace Framework

1) 개념모델

그림 4는 이력정보처리를 위한 모델을 개념적으로 나타낸 것이다. 개념모델은 크게 상태정보를 수집하고 보고하는(Status Capture & Reporting) 부분과 상태정보를 수집, 교환하는 부분(Status Collection & Exchange) 그리고 상태정보를 추적하고 모니터링 하는(Status Track & Trace)부분으로 구분 할 수 있다. 센싱정보를 포함하고 있는 RFID 태그를 리더에서 읽을 경우, 리더에서는 태그를 가진 각 객체의 상태정보 Status(ID)를 보고하기 위한 메시지타입으로 변환한다. 즉, 이진수 형태의 Status(ID)는 유, 무선 네트워크 상에서 프로토콜에 적합한 형태로 메시지가 작성되며, 상태정보를 객체의 상태가 변화할 때 혹은 주기적으로 Status Collector로 송신한다.

Status Collector는 수집하는 모듈과 수집한 정보를 Status Repository에 저장하고, 시간에 따른 상태변화를 저장하여 이력을 축적하며(즉, History(ID)의 갱신) 정보를 원하는 노드로 상태 및 이력정보를 제공한다. 이력정보를 조회하는 Track & Trace Application들은 질의-응답으로 이력을 검색하거나(pull model), 또는 주기적으로 또는 event가 발생할 때 마다 이력정보를 받아볼 수 있다(push model). Object Searcher는 객체에 관련된 정보의 위치나 객체의 이력정보의 위치를 제공

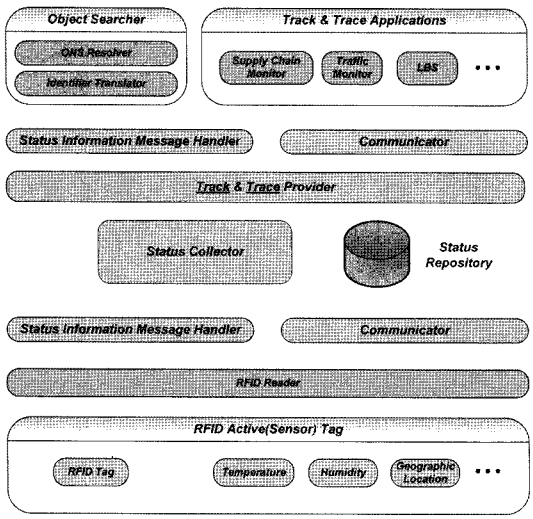

그림 4 Track & Trace 개념모델

해 주며, Status Collector들이 분산되었을 경우 이의 위치도 알려준다.

2) 분산평면에서 기능모델

개념적 모델을 분산평면에서 기능적 요소(functional entity)로 나누어 보면 그림 5와 같이 표현할 수 있다. RFID tag와 sensor들, 상태정보를 수집하고 보고하는 요소(SRN: Status Reporting Node), 상태정보를 수집, 교환하는 요소(STS: Status Track & Trace Server), 그리고 상태정보를 추적하고 모니터링 하는 요소(STN: Status Tracking Node)로 구분할 수 있다.

먼저, 센서 태그와 SRN간의 인터페이스는 ISO/IEC

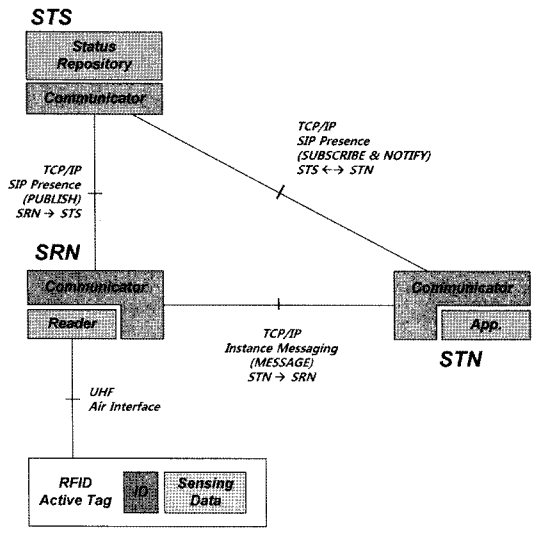

그림 5 Track & Trace 기능모델

표준을 따른다. 다시 말하면 SRN에서는 태그정보와 태그에 부착된 여러 센싱 정보를 동시에 읽어 태그가 부착된 객체의 상태정보를 얻는다. SRN은 복수개의 태그를 읽어 각 태그에 대한 정보를 STS로 보고하게 되는 State Agent[8] 역할을 하게 된다. 수동형 태그의 경우 태그를 읽을 수 있는 최대거리 때문에 태그와 SRN이 시공간적으로 같은 위치에 있다고 가정할 수 있다. 따라서 그림 1의 4와 같이 읽은 태그정보와 SRN에 위치한 센서 혹은 센서코디네이터를 통해 태그 및 상태정보를 취합하여 STS로 상태정보를 전달할 수 있다. SRN, STS, STN 간은 TCP/IP 프로토콜을 지원하는 유, 무선 네트워크로 연결된다.

4. IETF SIP Presence Model 기반 Track & Trace

4.1 개요

SRN, STS, STN간의 프로토콜은 상태정보의 보고와 이력정보의 획득에 적합한 전용 프로토콜을 설계함이 효율적일 수 있지만, 서로 다른 회사간에 Track & Trace 서비스가 제공될 수 있으며, 육상, 해상 운송 등, 수송수단도 다양할 수 있기 때문에 국제표준에 따르는 것이 현실적이며, 쉽게 deploy할 수 있을 것이다. IETF SIP 프로토콜은 이미 VoIP, IMS 등에 널리 쓰이고 있는 프로토콜로서, 특히, 프레즌스 모델(Presence model)은 기능적으로 보아 Track & Trace 응용에 매우 적합한 것으로 판단된다. 이 프레즌스 모델은 RFC 3265[8] SIP Event Package를 통해 SIP 이벤트 처리 패키지로 확장이 가능하다.

프레즌스 서버를 이용한 모델은 앞에서 제시한 기능모델에서 SRN, STS, STN 간의 인터페이스를 구현하는데 매우 적합하다. 아래 그림에서 보는 바와 같이 SRN은 프리젠티티, STS는 프레즌스 서버, STN은 와쳐의 기능을 가지면, 상태정보 Status(ID)는 PUBLISH method에 실려 보고하고, 이력정보 History(ID)는 NOTIFY method로 전달된다.

4.2 상태정보 생성 및 보고: Status Reporting Node (SRN)

SRN은 프레즌스 모델에서 프리젠티티의 성격을 가지며 RFID reader를 비롯한 상태정보수집을 위한 센싱 기능 혹은 센싱 정보를 필터링하고 취합하기 위한 Coordinator 모듈을 가질 수 있다. SRN은 정보수집대상이 사람일 경우 모바일 RFID 단말이 될 수 있으며 사물의 경우 RFID 리더에 각 기능을 추가한 경우가 될 수 있다. SRN은 자신이 읽은 RFID 테그정보 및 센싱 정보를 보고하기 위해 자신이 속한 도메인의 STN과 미리 논리적 연결을 구성하고 있어야 한다. SIP 프레즌스 모

델의 경우 Presentity가 PUBLISH 메시지를 보낼 프레즌스 서버의 위치를 기본 값으로 구성하고 있는 경우를 예로 들 수 있다. 프레즌스는 RFC 3859[9]를 통해 "pres" 라는 URI scheme을 정의하고 있다.

프레즌스 모델에서 프리젠티티를 알리고 와쳐에서는 프리젠티티를 확인하기 위해 프레즌스 정보 데이터 포멧(PIDF)을 사용하고 있다. 이는 IETF IMPP WG에서 RFC 3863표준으로 제정하였다. PIDF는 MIME 타입으로 "application/pidf+xml" 형식으로 콘텐츠 타입이 정의되어 있다. 본 연구에서는 SRN이 보고하는 상태정보 Status(ID)를 PIDF 포멧으로 encoding하여 PUBLISH method로 송신하는 방식을 채택하고 있다.

PUBLISH <presence URI>
Contents-Type: application/pidf+xml
<encoded Status(ID)>

RFC 2822에서는 URI scheme에서 사용되는 addr-spec를 local-part "@" domain으로 정의하고 있다. 이를 본 논문에 응용할 경우 객체의 상태정보는 객체가 속한 도메인(혹은 사업자)의 상태정보서버로 전달되도록 하기 위해 domain part를 SRN이 속한 도메인 주소로 정의할 필요가 있다. 또한 local-part의 경우 객체를 식별하기 위해 RFID 태그를 RFC 2822에서 허용하는 형식으로 변환할 필요성이 있다.

따라서 SRN에서 보고하는 상태정보의 주체를 나타내는 식별자는 "pres:<URI form of RFID>@<management domain>"와 같은 객체를 정확히 식별할 수 있는 presence URI 형식이 되어야 한다.

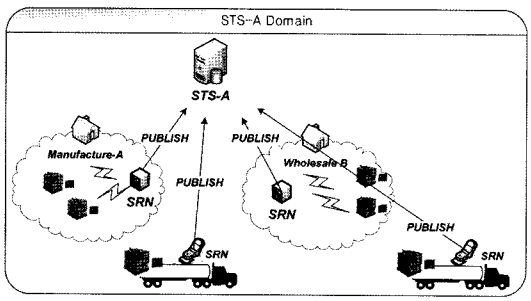

그림 6 하나의 관리 도메인에서의 상태정보 보고

예를 들어 상품 "p"가 "A.com" 회사에서 "B.com"으로 이동한 경우 "pres:p@A.com"라는 식별자와 "pres:p@B.com"이라는 식별자로 이동 전, 후에 사용될 것이다.

4.3 상태정보의 수집 및 축적: Status Track & Trace Server(STS)

STS는 프레즌스 서버와 같이 상태정보를 제공하는 단말(즉, SRN)로부터 정보를 수집하고 특정 객체의 이력정보를 원하는 단말(즉, STN)에게 이력정보를 제공하는 역할을 한다. 다시 말해, PUBLISH 메시지를 수신하여 repository에 이력으로 축적하고, 미리 가입한 STN들에게 NOTIFY method로 이력정보를 송신한다. 이력정보는 상태정보와 마찬가지로 NOTIFY 메시지 body 부분에 PIDF 포맷을 따라 encoding된다.

STN의 경우 객체의 이력정보를 얻고 모니터링 하기 위해서 그 객체에 대한 정보를 알고 있는 STS로 접근해야 하며 STS는 STN 에게 객체의 현재 상태 및 축적된 정보와 객체가 다른 도메인(혹은 사업자)로 이동하였을 경우 이동에 관한 정보를 제공하여야 한다.

STN에서 STS "a.com"으로 요청한 객체 "p"에 대한 정보는 "pres:p@a.com"이라는 식별자로 정의할 수 있다. STN이 요청한 객체가 요청을 받은 STS내에 머물러 있다면 "pres:p@a.com"에 대한 정보를 NOTIFY 메시지를 이용하여 STN으로 전달할 수 있다. 만일 요청을 받은 STS에서 객체가 "b.com"으로 이동한 정보를 STN으로 알려야 할 경우 REFER method를 활용하여 전달할 수 있을 것이다. SRN에서 PUBLISH하는 메시지의 경우 SRN-STS가 속한 도메인에 따라 객체의 URI 가 달라진다. 다시 말해 식별자에서 domain 부분이 바뀌게 되는데 이동한 정보의 "a.com"에서 "b.com"으로 이동한 경우 "pres:p@b.com" 식별자를 REFER 메시지를 통해 전달해야 한다.

4.4 상태정보 추적: Status Track & Trace Node (STN)

STN은 프레즌스 모델에서 Watcher의 성격을 가지며 프레즌스 정보, 이력정보를 수집하기 위한 역할을 한다. STN이 특정객체에 대해 Track & Trace 서비스를 받기 위해서는 STN이 객체를 테깅하거나 어플리케이션에서 수동으로 입력하여 RFID 코드를 식별한 것을 가정한다.

첫 번째로 식별한 RFID 코드를 바탕으로 첫 번째로 어떤 STS가 객체에 대한 상태정보를 보관하고 있는지 알아야 하며 두 번째로 STS에 접근하기 위해 인증절차가 필요하며 세 번째로 어떤 방식으로 상태정보를 수신할지에 대한 고려가 필요하다.

첫 번째 조회하고자 하는 객체를 어떤 STS가 관리하는지 조회하는 부분에 대해서는 4.5 식별자 변환에서 설명하며 이 조회의 결과는 객체에 대한 프레즌스 식별자이다. 프레즌스 식별자의 Address-of-Record(AoR)을 활용하여 RFC 3263[10]과 같은 방법으로 STS의 접근정보를 얻을 수 있을 것이다.

두 번째로 STS에 접근하기 위해서는 인증절차가 필요하다. SRN - STS의 경우 두 요소간의 논리적 연결이 필요하며 같은 도메인에 속한 것을 가정하지만 STN

의 경우에는 정보를 획득하기 위해 다른 도메인에서 조회하는 경우이기 때문에 네트워크에 참가하기 위한 인증절차를 필요로 한다. 이때 HTTP, SIP에서 사용되는 RFC 2617과 같은 HTTP 인증방식을 사용할 수 있다.

세 번째로 정보수신을 받을 주기를 협상하여야 한다. STN 에서 STS로 객체의 상태정보를 얻기 위해 SUBSCRIBE 메시지를 송신할 수 있으며 이 메시지를 이용하여 정보획득 형태에 대해 기술할 수 있다. 예를 들어 SUBSCRIBE 메시지의 "Expires:" 기간에 따라 한번의 NOTIFY만 수신할 것인가(Pull model) 혹은 기술된 기간 내에 주기적 혹은 이벤트 발생에 따라 수신을 할 것인지(Push model) 선택을 할 수 있다.

지금까지 기술한 내용 이외에 STN에서는 객체의 이동여부를 판단할 수 있어야 한다. 객체의 이동여부는 STS 로 보낸 SUBSCRIBE 메시지에 대해 NOTIFY 메시지 대신 이동된 도메인에 따라 객체의 변화된 식별자를 포함하는 REFER 메시지를 수신하였을 때 객체가 이동하였음을 알고 이동된 도메인의 STS를 찾아 SUBSCRIBE를 송신한다. 이는 객체가 도메인간 이동이 발생했을 때 STN이 객체를 지속적으로 모니터링 하도록 해 주기 때문에 별도의 이력정보서버 없이도 객체의 추적이 가능해 지도록 한다.

4.5 RFID에서 Presence URI로 변환

1) Domain Name Space에서 객체 단위의 위임과 레코드 축약

EPCglobal의 ONS 표준에는 serial code를 삭제한 객체 클래스(즉 item code까지만)에 대해 URI로 변환하는 것을 다루고 있으며, 객체 하나하나에 대한 처리가 필요한 discovery service에 대해서 현재는 논리적 개념을 Verisign에서 제공하는 서비스와 여러 논문을 통해 소개 되고 있으나 EPCglobal에서 표준으로 정의된 것은 없다. ONS가 객체 단위로 URI 변환을 다루지 못하는 이유는 객체 당 하나의 DNS NAPTR RR를 DNS에 수록해야 하기 때문에 현실적으로 불가능하기 때문이다.

ONS 내에 각 태그에 매칭되는 SIP 주소를 등록할 때 일반적인 방법으로는 태그의 개수와 NAPTR RR 개수가 같이 유지되어야 한다. 그러나 이 방법은 태그의 증가에 따라 데이터베이스 용량이 지나치게 커짐으로 효율적이지 못하다. 이 문제는 표 1과 같은 방법으로 NAPTR RR을 기술하면 각 도메인당 하나의 RR을 유지하는 것 만으로 객체 별 다른 URI로 변환하는 것이 가능하다.

NAPTR RR의 Service field에 프레즌스 타입의 서비스 필드를 필요로 한다. 예를 들어 ENUM 서비스에서는 프레즌스 서비스 타입을 "E2U+pres", 프레즌스 서비스를 제공하기 위한 URI Scheme으로 "pres:"를 정의하고 있다. 따라서 RFID 프레즌스 응용에서는 "EPC+pres" 혹은 "C2U+pres"와 같은 서비스 타입이 가능하다.

2) DDDS 알고리즘을 활용한 객체의 Presence URI 획득

표 1에서와 같이 AUS를 시리얼 번호까지 포함하였다. 이는 DDDS 데이터베이스 검색 후 정규표현식 필드를 AUS에 매칭하여 URI를 리턴 받을 때 필요한데, AUS에 시리얼 번호까지 포함함으로 리턴되는 SIP URI를 각 개별 태그를 구분하여 만들어지도록 할 수 있다.

4.6 STS의 Scalability와 다른 도메인간 끊김 없는 추적방식 연구

1) STS 서버의 Scalability 이슈

예를 들어 어떤 물류회사는 1,000대의 냉동차량을 보유하여, 차량 한대에 태그가 부착된 1,000개의 식품을 운반하며, SRN 한 대가 설치되어 있으며, 냉동차 컨테이너에는 온도 센서가 있고, GPS 수신 센서도 설치되어 있다고 하자. 또한 STN이 회사 내부에 50대, 고객에게 50대, 총 100대가 각기 운송 중인 차량의 위치와 컨테이너 내부 온도를 추적하는 하고 있다고 하자. 만일, 각 차량은 10초에 1번 식품의 위치와 온도를 STS에 보고한다고 가정하면, 1시간 동안 STS가 상태정보 Status (ID)를 받는 메시지 수는 1,000 × 1,000 × 0.1 = 600,000 messages/sec가 될 것이다. 더욱이, 이 STS를 10개의

표 1 DDDS 알고리즘을 적용한 RFID에서 presence URI 변환

DDDS 단계	객체-1	객체-2
태그 ID	Binary code-1	Binary code-2
태그 URN	urn:epc:id:sgtin:1.10.100	urn:epc:id:sgtin:1.10.200
AUS	urn:epc:id:sgtin:1.10.100	urn:epc:id:sgtin:1.10.200
FWKR	urn:epc:id:sgtin:1.10.100	urn:epc:id:sgtin:1.10.200
First Key	100.10.1.sgtin.id	200.10.1.sgtin.id
Key	100.10.1.sgtin.id.onsepc.com	200.10.1.sgtin.id.onsepc.com
Valid Database	*.10.1.sgtin.id.onsepc.com NAPTR "u" "E2U+pres" !^urn:([^\:]+):([^\:]+):([^\.:+)\.([0-9]+)\.([0-9]+)\.([0-9]+)$!pres:\1.\2.\3.\4.\5.\6@STA-A.com!	
Output	pres:epc.id.sgtin.1.10.100@STS-A.com	pres:epc.id.sgtin.1.10.200@STS-A.com

물류회사가 끊김 없는 Track & Trace 서비스를 위해 공유하고 있다고 하며, 발생되는 상태정보의 량은 6M messages/sec가 되며, 이를 다시 cold supply chain 상의 제조사, 도매상, 소매상, 소비자 등에 적용시키면 기하급수적으로 증가하여 중앙집중형 STS 서버 하나로 처리할 수 없다. 또한, 수집되는 정보를 지속적으로 축적해야 하기 때문에 데이터베이스 량은 상당히 증가하게 될 것이다. 즉, Track & Trace 서비스를 지원하려면 송수신하는 메시지들을 수신하여 처리할 수 있는 대역폭을 필요로 하며, 자료 처리속도 및 데이터베이스 구조를 가지지 않으면 안될 것이다.

STS는 다수개의 SRN이 상태정보를 보고하는 하나의 논리적 집합이며 이를 STS관리도메인(이하 STS도메인)이라고 한다. 하나의 도메인에서 상태보고를 받는 SRN 및 RFID 객체의 개수가 많을수록 많은 부하를 가질 것이며 이때 STS의 부하경감을 위해 분산구성 하는 방법이 있다. 이때 STN에서 상태정보를 얻기 위해 보낸 SUBSCRIBE 메시지는 Request URI로 AoR(Address of Record) 형식을 사용하며 STS도메인의 인입 Proxy에서 FQDN(Fully Qualified Domain Name) 형식으로 변환되어 요청한 객체의 상태정보를 갖고 있는 Proxy로 전달된다. 그러나 이렇게 분산구성 하였을 경우 STN에서 STS 사이 hop 개수가 증가하게 되며 이는 전달지연의 원인이 된다[7].

2) 다른 관리 도메인간 Seamless Track & Trace

하나의 supply chain은 동일한 STS도메인에 속하면 Track & Trace가 용이할 것이며, 이러한 정보도 상호 협약에 따라 공유되는 게 자연스러운 일이다. 일반적으로 보면, 제조사에서 도매상으로, 도매상에서 소매상으로 물류가 유통될 때, 중간에 하나 이상의 운송회사에 의뢰하여 운반하게 될 것이다. 만일, 이 운송회사들이 다른 STS도메인에 속한다면, 운송 중간의 물품 상태를 추적하기가 불가능하게 된다. 이 문제를 해결하기 위해서는 다른STS도메인 간에도 중단되지 않고 Track & Trace가 유지되어야 한다.

그림 7의 예와 같이 STS-A도메인에서 STS-B도메인을 거쳐 STS-C도메인으로 물건이 이송되었고, 이때, 외부에 있는 고객의 STN이 그 이력을 추적하려 한다고 가정하자. 첫 번째 아이디어는 SRN에서 보고하는 상태정보 중 객체가 다른 STS로 이동하였다는 상태정보를 수신하여 객체가 이동한 것을 STS에서 알 수 있기 때문에, 이동된 도메인 내에서 책임진 STS가 새로운 도메인으로 이동되었음을 이전 도메인의 STS에게 알려주록하게 하는 방법이다. 이와 같이 하면, 이동된 경로상에 있는 도메인내의 STS간에 체인이 형성된다.

STN은 RFID만을 가지고 조회해야 하기 때문에

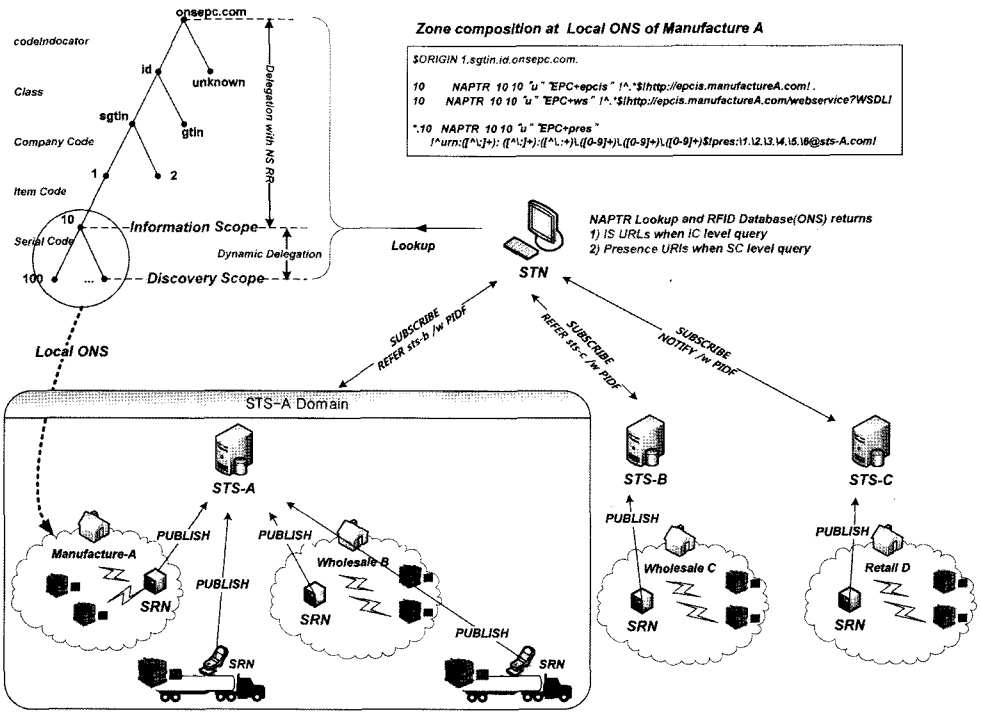

그림 7 STS domain간 이동한 RFID 객체의 Track & Trace

SUBSCRIBE 메시지를 보내 추적 세션 개시를 알리면, 최초 domain내에서 이 물품을 제조하여 RFID tag를 부착했던 제조사의 ONS resolution에 따라 이동 여부와 상관없이 최초 domain으로 SUBSCRIBE 메시지가 전달된다. 만일, 조회하는 객체가 다른 domain으로 이동했다면, NOTIFY 메시지 대신 "pres:<URI form of RFID>@<moved domain>" 형식 URI를 포함하는 REFER 메시지를 송신하여 다른 관리도메인으로 전환하도록 하여 객체 추적이 끊어지지 않도록 한다. 그러나, 여전히 남은 문제는 SUBSCRIBE/REFER 메시지 교환이 이동된 domain 경로를 거치면서 현재 관할하는 도메인을 찾아내는 방식이기 때문에 중간노드인 STS-B가 응답을 하지 않을 경우 추적이 불가능하다.

5. 프레임워크의 프로토타이핑과 분석

5.1 프레임워크의 프로토타이핑

본 논문에서는 ONS name space에서 Item Code 계층 하부에서 Serial Code로 위임하되, 서로 다른 Serial Code에 대해 각기 다른 NAPTR RR를 수록하지 않고 Item Code가 같은 물품의 태그에 대해서 하나의 NAPTR RR을 사용하되 wildcard notation 기법을 도입함으로써 NAPTR Resource Record의 증가를 방지하였다.

그림 8에서는 모두 1000번 가량 테스트를 통해 Wildcard 표기를 이용하여 Serial-code까지 DNS 영역파일에 구성하여 질의한 경우(SC Query) 평균 8.4380ms가 소요 되었으며 일반적인 방법인 Item-code까지 구성하여 질의한 경우(IC Query) 8.4ms가 소요되어 근소한 차이로 SC Query 방법이 늦음을 보여주지만 DNS 영역파일에 기록되는 RR의 개수가 증가할 때 Resolving 속도가 늦어짐을 감안할 때[11] 많은 수의 Item-code를 갖고 있는 실 사용 환경에서는 차이가 없거나 wildcard 표기를 했을 때 성능이 우수하다고 판단할 수 있다.

Wildcard 표기를 이용한 ONS 구성을 통해 RFID 객체별 프레즌스 식별자 변환을 하고 이를 이용하여 STN은 객체상태정보를 저장하고 있는 STS를 찾아 SUB-

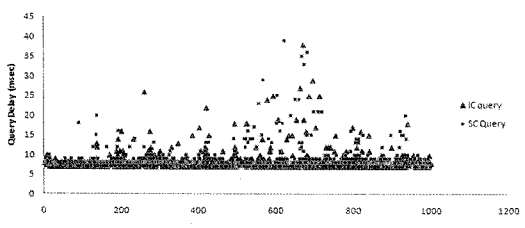

그림 8 Wildcard Resource Record 구성에 따른 지연 비교

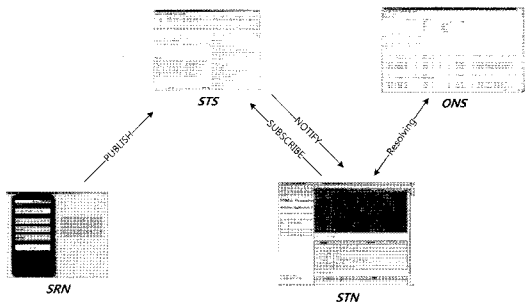

그림 9 서비스 프로토타이핑

SCRIBE 메시지를 전달한다. 메시지를 받은 STS는 SRN이 PUBLISH하고 있는 상태정보데이터를 NOTIFY 메시지를 이용하여 STN으로 전송한다. STN은 XML(PIDF) 형식의 상태정보데이터를 변환하여 출력해준다. 그림 9에서 STN에서는 객체가 이동할 때 SRN에서 GPS 좌표 값과 현재 온도를 STS로 전송하고 이때 변화된 상황에 따라 STN에서 객체의 현재 위치 및 온도를 출력해주는 서비스를 나타내고 있다.

5.2 타 연구와의 비교 분석

관련 연구와 본 논문에서 제안한 이력 표준을 비교하면 표 2와 같이 요약할 수 있다. IBM의 EPCDS와 PTSP의 경우 EPCglobal의 구조를 따른다. 다시 말해 EPCIS와 EPCDS는 논리적으로 별도의 접속정보를 갖고 있다. 이력정보 조회를 원하는 응용은 이력정보 서버를 조회하기 위해서 ONS 가 아닌 별도의 조회서비스를 이용하여야 한다. EPC-DS의 경우 통합적인 Discovery Server (DS)를 사용하여 이 DS에서 각 물품의 이력정보를 취합 후 한번의 쿼리로 IS 목록을 제공한다.

EPCDS 및 PTSP에서는 일반적으로 HTTP/XML을 이용한 SOAP 방식을 사용하며 SRMS 및 본 논문에서 제안한 방식은 SIP 프로토콜을 기반으로 동작한다. SOAP 프로토콜은 웹서비스를 정의한 XML 문서인 WSDL을 이용하여 Compile후 Stub을 생성, 각 Peer간 XML 문서교환으로 이루어진다. 따라서 ONS 검색 혹은 DS를 위한 네임서버 검색 후 이력정보 검색을 위해서는 Stub 생성을 위한 WSDL Compile과정을 거쳐야 한다.

이력정보를 검색하고자 하는 노드에서 Stub을 생성할 수 없을 경우 WSDL을 미리 알고 Stub이 생성되어 있는 별도의 어플리케이션 서버를 이용해야 하는 단점이 있다. 그러나 제안한 구조에서는 SIP 프로토콜을 사용함으로 사전협약 없이도 접근이 가능하다. 또한 보안에 관련한 요소를 고려했을 때 XCAP이나 Watcher.info [12]와 같은 프레즌스 이벤트를 처리함으로 접근제어 목록을 생성할 수 있으며 SIP 네트워크에서 일반적으로 사용되는 HTTP-Digest 인증방식을 사용할 수 있다.

표 2이력추적 방식의 비교

구분	PTSP[6]	EPCDS (IBM)[5]	SRMS[7]	제안한 방식
태그의 URI	EPC 표준	EPC 표준	SIP/SIP	EPC 표준/ SIP
이력추적 서버	중앙집중형	중앙집중형	분산형	분산형
관리 도메인간 이력추적	DS 조회	DS 조회	지원하지 않음	STS 간 분산
통신 프로토콜	SOAP/HTTP	SOAP/HTTP	SIP	SIP
망 개방 여부	Closed	Closed	Open	Open
보안	EPC 표준	EPC 표준	HTTP Digest	XCAP,HTTP Digest
거점기록	제한 없음	제한 없음	2개(추정)	제한 없음
이력정보검색 혹은 DS 조회	ONS의 RR에 DS 명시	별도 NS 구성	이력정보 조회 지원하지 않음	STS 간 분산구성

5.3 제안한 이력정보추적 및 모니터링 프로토콜 장·단점 분석

본 논문에서는 EPC 네트워크에서 사용하는 EPC-IS와 EPC-DS를 동시에 대체 할 수 있는 Status Track & Trace Server (STS)를 제안하였으며 이는 SIP 프로토콜을 사용하며 특히 SUBSCRIBE, REFER 그리고 PUBLISH method를 특징적으로 사용하였다. 또한 객체 별 이력추적을 위해서는 ONS 쿼리에서 Item Code 혹은 Item Category Code까지 포함하는 결과값을 반환하여야 하며 제안한 ONS 방식에서는 Serial Code까지 포함하는 SIP 식별자를 반환하도록 하였다.

그림 10에서는 기존 EPC 네트워크, PTSP 방식과 제안한 방식에서 객체의 이력정보추적방식에 대해 비교한 내용이다. 그림의 각 흐름도는 아래와 같은 가정을 갖고 있다.

- 하나의 위치는 하나의 EPC-IS(STS)에서 관리
- 객체는 EPC-IS(STS) 1부터 두 번 이동하여 EPC-IS (STS) 3까지 이동
- TN은 조회를 원하는 단말 혹은 제 3의 EPC-IS (STS)
- 상품군의 최초 생산으로 ONS 등록하는 과정은 생략

그림 10의 A) 방법의 경우 객체가 EPC-IS 1에서 생산하여 2 3으로 이동하는 과정에서 상품의 입출고에 관한 Timestamp 정보 등을 DS로 전송한다. 상품의 이동 후 TN에서 객체의 이력정보를 얻기 위해 우선 특정 상품군의 정보를 갖고 있는 DS를 별도로 구성된 NS를 통해 질의한다. NS 질의를 통해 DS 정보를 얻어 DS 질의 후 DS에서는 상품의 입출고 및 IS접속정보를 얻는다. 각 IS로 질의 후 객체의 정보를 취합한다. 이와는 달

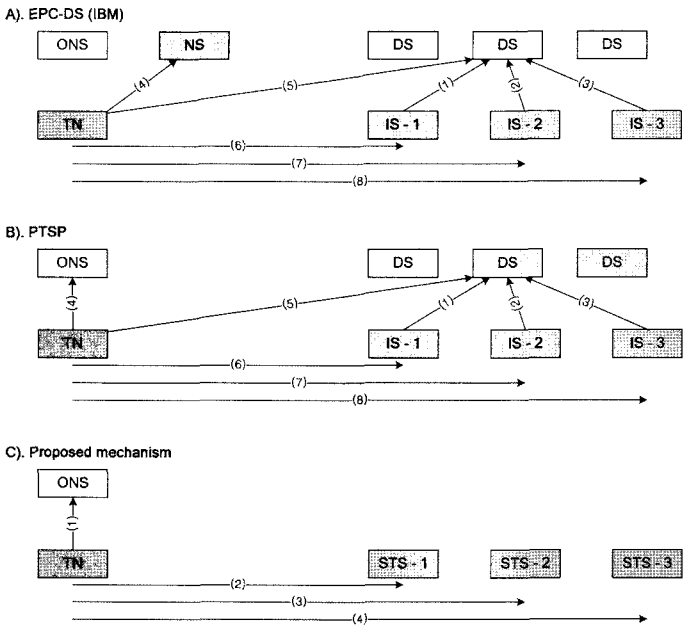

그림 10 객체이력추적의 각 방식 비교

리 B) PTSP에서는 특정 상품군을 관리하고 있는 DS를 알기 위해 별도의 NS를 두지 않고 ONS에 같이 위치시켜 이력정보추적에 필요한 네트워크 요소를 줄였다.

제안한 방식에서는 ONS 질의 후 반환되는 식별자를 이용하여 상품을 최초로 생산했던 STS-1로 질의한다. 이동하였다는 응답을 받고 STS-2, STS-3으로 순차적으로 질의하여 객체의 이력정보를 취합한다.

이와 같이 객체가 각 IS를 이동해온 구간별 상태정보를 획득하기 위한 방법에서 질의횟수를 줄일 수 있으며 네트워크를 단순하게 유지 할 수 있다. 객체가 각 IS(STS)를 N번 이동할 경우 기존 방법에서는 2N+2번 소요하는데 반해 제안한 방법은 N+1번으로 정보를 취합할 수 있다. 또한 A), B) 방법에서 발생할 수 있는 문제점으로 분산된 각 DS를 하나의 관리기관에서 통합, 관리해야 한다는 단점이 있다. 다시 말해 TN, EPC-IS, DS가 모두 협약된 구조에서만 가능하다. 예를 들어 LCD 패널 객체가 하나의 컴퓨터패키지로 포함되는 경우와 TV 패키지로 포함되어 판매되는 경우가 있을 수 있는 이 때는 상품군이 달라질 것이며 이에 따라 두 개의 DS에 질의하는 경우가 발생할 수 있다.

하지만 IP Layer에서 라우팅이 이루어지는 SOAP과는 달리 Application Layer에서 라우팅을 해야 하며 프록시등을 필요로 하는 SIP 특성상 그림 10에서 보이지 않는 요소들이 필요하기 때문에 실제 시스템을 구축하였을 때는 TN에서 STS까지 사이 존재하는 프록시의 개수에 따라 지연이 늘어나는 단점이 있다[7].

EPC 네트워크에서 DS는 일반적으로 객체가 각 지점에 입, 출고된 시간을 비롯한 이동정보를 저장하고 있다. TN에서 객체의 지점간 이동정보 및 시간 등 단순한 정보만을 알고자 할 때는 EPC-DS 및 PTSP 방식에서는 간단히 DS 질의로 알아낼 수 있지만 제안한 방식에서는 TN에서 이동한 횟수만큼 질의를 해야 하기 때문에 적합하지 못하며 이동한 지점 중 중간 STS로 질의가 실패하는 경우 객체의 이후 정보획득에 실패한다.

본 논문에서 제안한 방식을 이용하여 SRN에서 STS로 상태정보를 송신하는 경우 여러 개의 SRN에서 각 객체의 상태정보를 하나의 논리적 STS로 보고해야 하기 때문에 많은 양의 트래픽이 몰릴 수 있다. 이를 해결하기 위해 STS를 여러 개의 물리적 서버로 분산 구성하였을 경우 하나이상의 프록시 서버를 필요로 한다. 따라서 분산구성을 함에 따라 지연이 늘어나는 단점이 있다.

또한 UDP가 일반적으로 사용되는 SIP 프로토콜 특성상 다량의 상태정보를 읽어올 때 문제점을 갖고 있다. SIP 프로토콜에서는 path MTU 이상의 데이터를 한번에 전달하지 않도록 권고하고 있으며 이것 이상의 데이터를 전달할 때는 TCP를 사용하도록 하고 있다[13].

5.4 응용 가능한 서비스

프레즌스 기반 Track & Track 모델을 활용하여 각 객체에 접근할 수 있고 이력사항을 수집할 경우 객체모니터링 같은 서비스가 가능하다. 예를 들어 각 창고의 온도를 모니터링 하고 보관중인 물품이 특정 온도에서 보관되어야 할 경우 상태변화를 Track & Trace하고 특정온도이상 온도가 상승할 경우 STN으로 NOTIFY하여 알리는 기능을 가진다. SRN에서는 창고 내 온도 변화가 일어날 때 마다 혹은 주기적으로 STS로 상태정보를 전송한다. 또한 STN 에서 조회하고자 하는 물품이 ONS 질의 결과 그림에서 나타난 STS가 관리한다고 할 때 STS로 SUBSCRIBE하여 상태정보변화가 발생할 때 NOTIFY 받을 수 있도록 한다. 이와 같은 환경에서 창고의 온도가 상승하였을 경우 프레즌스 변화로 STS에서는 STN에 임계값을 초과했다는 NOTIFY를 보내 물품의 관리를 용이하게 하는 서비스이다.

또한 프로토콜 특성상 모바일 단말기를 이용한 객체 추적 역시 가능하다. SRN이 원하는 객체가 물품이 아닌 사람일 경우 SRN을 모바일 RFID 단말로 가정할 수 있으며 환자관리나 미아 찾기와 같은 서비스가 가능하다. 환자의 신체에 부착된 센서를 통해 사람의 체온, 맥박수, 호흡상태 등을 수집하여 모바일 RFID 단말(SRN)에서 STS로 상태정보를 전송하게 된다. STN은 환자의 상태정보를 관리할 수 있으며 필요에 따라 환자 상태확인을 위해 INVITE 메시지 송신과 같이 SRN으로 명령어를 전송하여 영상 어플리케이션과 같은 다른 멀티미디어 처리를 할 수 있다.

6. 결 론

RFID 태그가 부착된 객체의 상태변화에 대해 모니터링하고 이력을 추적하기 위해서는 RFID 태그의 식별자로 사용되는 URN 형식 이외에 개별 객체마다 URL을 할당해 준다면 그 URL 을 이용하여 객체에 직접 접근하여 상태변화를 지속적으로 모니터링 하는 것이 가능할 것이다. 본 논문에서는 RFID 객체에 URL을 할당하되 RFID 객체는 복합적 상태정보를 전달하기 위한 통신환경을 갖지 못하기 때문에 상태정보를 효율적으로 관리할 수 있는 SIP 이벤트 통지 및 프레즌스 모델을 RFID 네트워크에 응용하였다.

RFID 객체의 이름을 관리하는 서버인 ONS 에 모든 객체에 대해 이름을 부여하기 위해서 기존과 같은 방법으로 접근하는 것은 Resource Record의 개수의 증가에 따른 과부하 및 관리의 문제점이 있다. 이것을 wildcard 표기를 이용한 DDDS 알고리듬을 적용하여 local ONS 당 하나의 RR만 유지하는 것으로 제안한 모델을 사용 가능하도록 하여 기존 ONS를 그대로 사용하면서 성능

저하가 없음을 보였다.

물품의 이력을 얻기 위해서 DS에 질의, 반환 받은 리스트를 바탕으로 IS 에 질의, 응답하는 Transaction에 기반한 방법의 경우에 객체의 변화를 지속적으로 모니터링 하는 것이 불가능 하며 이력을 통합적으로 관리해야 하는 단점이 있다. 하지만 제안한 STS-STN에서는 STN이 알고자 하는 RFID 객체의 정보를 주기적 혹은 변화가 있을 때 마다 NOTITY할 수 있도록 조회를 원하는 노드와 조회대상간 Dialog를 생성하여 지속적인 모니터링이 가능하게 하였으며 물체가 이동하여 관리하는 STS가 바뀌어도 REFER method를 통해 바뀐 STS로 유연하게 정보를 재 요청할 수 있도록 하였다.

그러나 SIP 프로토콜은 접근이 용이한 범용 프로토콜이지만 이 때문에 접근제어를 위한 요소를 필요로 한다. 그리고 여러 개의 SRN에서 STS로 보고하는 방대한양의 상태정보를 효율적으로 관리하고 요청한 프리젠티티에 대해 빠르게 응답할 수 있는 메커니즘 역시 필요하며 SIP 프로토콜을 통해 PIDF 형식으로 상태정보를 전달 할 경우 일반적으로 UDP 프로토콜을 사용하는 특성으로 인해 전달해야 하는 정보량에 대한 제한이 있어 이러한 문제에 대한 연구가 필요하다.

참 고 문 헌

[1] 정재영, "센서 태그 기술 동향", ETRI 전자통신동향분석 제22권 제3호, pp. 38-45, 2007년 6월.

[2] EPCglobal, "Architecture Development for Sensor Integration in the EPCglobal Network," AutoID, July. 2007.

[3] RFID 검색 시스템 구축 및 운영 지침서 1.1, NIDA, December, 2005.

[4] EPCglobal Object Name Service (ONS) 1.0, EPCglobal, 4 October 2005.

[5] Steve Beier, "Discovery Service-Enabling RFID Traceability in EPCglobal Networks," COMAD 2006, Dec. 2006.

[6] Yingping Cao, "PTSP: a lightweight EPCDS platform to deploy traceable services between supply-chain applications," RFID Eurasia, 2007 1st Annual, pp. 1-5, Oct. 2007.

[7] Kideok Cho, "SRMS: SIP-based RFID Management System," IEEE ICPS'07, pp. 11-18 July. 2007.

[8] A. B. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification," IETF RFC 3265, June 2002.

[9] J. Peterson, "Common Profile for Presence(CPP)," IETF RFC 3859, Aug. 2004.

[10] J. Rosenberg, "Session Initiation Protocol (SIP): Locating SIP Servers," IETF, RFC 3263, June. 2002.

[11] 김동욱, "VoIP 네트워크간 효율적인 상호접속을 위한 Infrastructure ENUM 구축방안", KCC2007, Vol.34, No.1 (A), pp. 188-189, 2007.6.

[12] J. Rosenberg, "A Watcher Information Event Template-Package for the Session Initiation Protocol (SIP)," IETF, RFC 3857, Aug. 2004.

[13] J. Rosenberg, "SIP: Session Initiation Protocol," IETF RFC 3261, June. 2002.

김 동 욱

2006년 2월 한국외국어대학교 정보통신공학과 학사. 2008년 8월 한국외국어대학교 석사. 현재 ㈜재너시스템즈 연구원. 관심분야는 멀티미디어 통신, VoIP

홍 진 표

1977년 2월 서울대학교 계산통계학과 학사. 1979년 2월 KAIST 전산학과 석사 1983년 8월 KAIST 전산학과 박사. 1983년 8월~1995년 2월 한국전자통신연구원 개발환경연구실장, 소프트웨어공학연구실장, 종합정보통신망연구부장, 지능망연구부장, 정보통신표준센터장 역임. 1995년 3월~현재 한국외국어대학교 정보통신공학과 교수. 관심분야는 멀티미디어통신, 모바일 RFID, 네트워킹