**개량된 MQTT 노드 관리 시스템을 이용한 학교 알림 서비스**

2012104094 서민석

2013104074 박나연

2013104091 용찬호

**요 약**

최근 ICT 트렌드의 화두는 단연 모바일과 IoT라 해도 과언이 아니다. 그러나 이를 통합적으로 관리, 제어하는 시스템은 여전히 존재하지 않는다. 또한 본교에는 수업 정보나 학교 정보 등 각종 학생들에게 유익한 정보를 일괄적으로 구독, 전달받을 수 있는 시스템이 없다. 본 문서는 이를 위한 시스템 개발 및 학교에 적합한 Group, Topic based Messenger의 개발을 제안하며, 이에 대한 기반 기술과 필요한 요구사항을 정의한다.

**1. 서론**

**1.1. 연구배경**

현재 IoT, 모바일 기기들을 통합적으로 관리하고 메시지를 주고받을 수 있는 시스템은 존재하지 않는다. 이를테면 밭에 설치한 온, 습도 측정 IoT 장치에서 보낸 메시지를 모바일 기기에서 구독하고, 농장의 관리자가 PC에서 등록한 공지 사항을 모바일 기기에서 받아보며, 자신이 속한 그룹의 소식을 시스템에서 적절하게 전달해주는 통합적인 멀티플랫폼 메시징 시스템이 필요하다고 느꼈다.

또한 현재 경희대학교 교내에는 학생들이 원하고 필요로 하는 정보를 통합적으로 받아볼 수 있는 서비스가 존재하지 않는다. 즉, 식단을 보는 서비스 따로, 학생이 과제나 시험 등의 수업 정보를 볼 수 있는 서비스 따로 존재하는 식이다. 따라서 상기한 메시징 시스템을 이용하여 학생이 듣는 수업의 공지사항을 알림 받고, 오늘의 식단 등 교내 소식을 구독하며, 교내에 설치된 IoT 장비로부터 교정 온도, 습도 등을 구독해 받아볼 수 있는 등, 그룹형 통합 구독 서비스를 개발할 필요가 있다.

**1.2. 연구목표**

퍼블리셔 확장의 여지를 남겨둔 채 서비스 상에서 구독자들이 필요로 하는 여러가지 정보들을 하나의 통합된 어플리케이션에서 지원하기 위해서는 계층적, 일관적인 프로토콜의 확립이 필요하며, 캡슐화된 퍼블리싱 라이브러리를 사용함으로서 개발자가 손쉽게 IoT나 모바일 노드를 퍼블리셔로 만들 수 있어야 한다. 또한 퍼블리셔를 손쉽게 확장시킬 수 있는 기반을 만드는 것을 목표로 함과 동시에 허가되지 않은 노드는 퍼블리싱할 수 없도록 authentication, authorization 기능 또한 요구될 것이다.

그리고 IoT 노드를 포함한 다양한 플랫폼의 노드들이 End-User가 구독한 정보를 퍼블리싱할 수 있어야 한다. 따라서 Linux, iOS, Android 등의 다양한 플랫폼을 지원해야 한다.

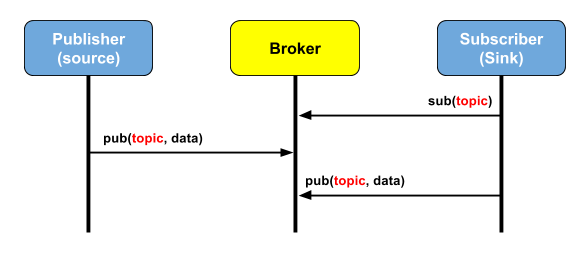
또한, IoT 노드가 퍼블리셔로서 동작한다는 특성상, 저전력으로 작동하는 시스템이 유리하다.

마지막으로 정보를 구독하되, 사용자가 속한 그룹에 해당하는 정보는 구독하지 않아도 시스템에서 알아서 전달할 수 있는 기능 역시 요구된다. 예를 들어, “고급 객체지향 프로그래밍” 수업을 수강 중인 학생은 식단 등의 부가적인 정보와 달리 “고급 객체지향 프로그래밍” 수업에 대한 구독 요청을 하지 않아도 해당 수업 관련 정보를 받을 수 있어야 할 것이다. 이와 같은 그룹 기반 메시징 구독은 시스템을 적용하는 환경에 따라 조건이 다르다. 예를 들어, 소규모 공장에 시스템을 적용하여 생산라인에 대한 메시징 서비스를 제작한다고 하자. 이 서비스는 본 프로젝트에서 제작하는 ‘학생을 위한 메시징 서비스’와 차이점이 존재한다. 각 서비스에는 어떤 그룹이 존재하는지, 또 그 그룹마다 무슨 메시지를 받아야하는지가 다르다는 점이다. 즉, 환경이 다를 때에도 확장이 용이하고, 이를 유연하게 대처하기 위해선 그룹에 따른 구독 정보들을 각각 Group Rule로서 관리자가 추가, 삭제 및 수정이 가능해야 한다.

**2. 관련 연구**

**2.1 모바일 및 IoT 노드 메시징 기술**

**2.1.1 MQTT**



MQTT는 publish-subscribe 구조를 갖는 ISO 표준(ISO/IEC PRF 20922) 메시징 프로토콜로서 경량화 및 저전력에 초점을 맞춰 개발되었다. MQTT는 네트워크 대역폭이 제한 되어있는 상황에서의 원격 메시징을 위해 설계된 점, 저전력 및 경량화된 프로토콜이라는 점 등 때문에 각종 임베디드 IoT 기기들 사이에서 메시지를 주고받을 때 주로 사용되고 있다. 현 2016년을 기준으로 IBM에서 MQTT v3.1 을 출시하여 오픈소스로 활성화 되어있다. 가장 대표적인 SNS인 페이스북의 메신저와 국내 통신사 Push 서버도 MQTT를 사용하고 있다.

MQTT는 Broker, Subscriber, Publisher 로 구분된다. Subscriber는 메시지를 받는, Publisher는 메시지를 보내는, Broker는 중간에서 메시지를 전달해주는 Medium Layer 역할을 한다. 이때 MQTT에서 메시지를 보낼 시 어떠한 내용의 데이터인지를 나타내는 ‘주제’, ‘제목’과 유사한 것이 바로 ‘토픽’이다. Publisher가 특정 토픽에 대해서 Broker에 메시지를 보내면 Broker가 그 토픽을 구독(Subscribe) 하고 있는 Subscriber들에게 메시지를 전송하는 구조로 되어있다. 또한 TLS/SSL 및 X.509 인증서를 이용한 양방향 인증도 지원한다.

**2.1.2 블루투스**

블루투스(Bluetooth) 는 1994년에 최초로 개발된 개인 근거리 무선 통신을 위한 산업 표준(IEEE 802.15.1)으로서 오픈 라이선스이기에 누구나 자유롭게 사용할 수 있다. 2014년에 블루투스 4.2 가 발표되었으며 IoT를 위한 연결성이 강화되었다. IPv6이나 6LoWPAN을 통해 인터넷에 직접 접근할 수 있으며 128비트 AES 암호화를 통한 보안을 지원한다. 또한 사용환경에 따라 최대 60m, 일반적으로 20m 수준의 반경에 걸친 연결 지원과 저전력 블루투스인 BLE(Bluetooth Low Energy) 의 결합으로 비콘 등 각종 무선 IoT 기기들에서 활용되고 있다.

**2.2 클라이언트 그룹 기반 메시지 전송**

사용자가 애플리케이션 등을 이용해 서비스에 접속하면 클라이언트의 고유 키를 서버에 등록한 뒤 사용자를 여러 기준에 맞게 분류하여 저장해야 알맞은 메시지를 클라이언트에게 보낼 수 있다. 이는 리스트, 트리 등의 자료구조를 이용해 여러 종류의 하위 주제로 나뉘어져 논리적으로 구분될 수 있어야 한다. 특정 그룹에 해당하는 사용자들의 클라이언트 단말에 메시지를 보내려 할 때에는 그 그룹에 배정되어 있는 사용자들의 고유 키 값을 이용해 전송하게 된다.

**2.3 메시지 알림**

클라이언트를 그룹핑하여 분류한 뒤, 메시지를 사용자의 단말으로 전송하기 위한 방법으로서는 GCM(Google Cloud Messaging Service), SMS, MQTT, 카카오톡과 같은 서드파티 애플리케이션을 통한 메시지 전달 등이 있다. GCM은 구글 개발자센터의 클라우드 서비스를 통하여 사용자의 단말에 푸시를 보내는 형식이며 안드로이드 단말에 한정되는 서비스라는 단점이 있다. SMS는 사용자에게 직접 문자메시지를 보내는 형식이지만 이는 애플리케이션이 아닌 사용자의 전화번호를 통해서 전송하게 된다. 카카오톡과 같은 서드파티 애플리케이션을 통한 메시지 전달 방식은 해당 서드파티의 API가 개발자들에게 공개되어있지 않을 경우 사용할 수 없다. MQTT는 트리 형태를 가지는 토픽 기반의 저전력 메시지 전달 프로토콜로서 다양한 언어로 바인딩된 라이브러리를 오픈소스로 사용할 수 있기 때문에 대부분의 플랫폼에 종속되지 않는다.

**2.4 기존 메시지 알림 서비스에 관련된 애플리케이션**

기존의 경희대학교 정보지원처의 종합정보시스템을 이용한 SMS 기반 학교 알림 서비스가 대부분의 학생에게 제공되고 있지만 이는 학생들이 원하는 정보가 아닌 정해진 과목 및 단과대학 별로 분류되어 메시지가 전송된다. 또한 각 과목 혹은 단과대학마다 문자, 이메일, 홈페이지 공지 등, 알림 방식이 달라, 한눈에 확인하기 힘들거나 알림을 그냥 지나치는 경우도 허다하다. 수강하는 과목, 소속된 단과대학뿐만 아닌 여러 종류의 메시지를 그룹핑하여 학생들이 원하는 정보를 구독하고 메시지를 전달해주는 애플리케이션은 현재 개발되어 있지않다.

**2.5 기존 연구의 문제점 및 해결 방안**

위에서 언급한 서비스를 구현한 MQTT 메시징 프로토콜을 기반으로 하는 IOS, 안드로이드 애플리케이션 개발에 초점을 맞추고 있다.

**2.5.1 연구의 문제점**

사용자는 애플리케이션에서 학번만을 입력해 자신이 원하는 정보 및 학교에서 제공해주는 메시지를 전달받아야 하지만 MQTT는 Publish-Subscribe 구조를 가지고 있어 일반적으로 사용되는 오픈소스 MQTT 클라이언트 및 브로커 툴들을 사용해서는 본 서비스를 구현할 수 없다. 이는 사용자의 단말이 MQTT 브로커에게 각종 토픽(Topic) 에 대한 Subscribe 요청을 하지 않고 학번만으로 Connection을 맺음으로써 사용자에게 해당하는 토픽을 전달받아야 하기 때문이다.

또한 기존의 MQTT 브로커 및 클라이언트 툴은 사용자에게 MQTT를 통한 메시지 알림이 정상적으로 도착했는지 확인하는 것을 지원하지 않고, Connection 관리도 제공하지 않아 사용자의 단말이 MQTT 브로커와 정상적으로 연결되어 있는지, 어떠한 상황에서 Connection이 끊기는지, Keep-alive 주기는 얼마나 하는 것이 바람직한지 등에 대한 정보를 알 수 없다.

**2.5.2 해결 방안**

**2.5.2.1 메시지 전달**

사용자의 단말이 특정 토픽을 Subscribe를 하지 않은 상태에서 직접 메시지 전송이 가능해야 하므로 홀 펀칭(Hole Punching) 과 같은 우회 기법을 사용해 간접적으로 메시지를 전달하거나 MQTT 브로커에서 Connection을 맺을 때 클라이언트 ID를 따로 보관하여 토픽 Subscribe에 관계 없이 메시지를 전송하는 방법을 사용할 것이다. 이는 오픈소스인 MQTT 브로커와 클라이언트 소스를 수정하거나 랩핑하는 형식을 통해 가능할 것으로 예상된다.

**2.5.2.2 Connection 관리**

사용자의 단말이 최초로 MQTT 브로커와 연결되었을 때, 연결 정보를 데이터베이스에 저장하여 Keep-alive를 유지한다. 연결이 자주 끊어지는 단말의 경우 Keep-alive 주기를 짧게 하거나, Connection Loss로 인해 메시지 전송에 실패하는 경우 우선순위 큐에 넣어 재전송을 할 수 있도록 한다.

**2.5.2.3 사용자 정보 그룹화 저장**

한 번이라도 연결에 성공한 사용자라면 연결 시 사용한 학번을 기존에 가지고 있던 데이터에 따라 분류시켜 저장해야 한다. 사용자로부터 입력 받은 학번과 기존의 시스템에 저장되어 있던 사용자가 수강한 정보, 소속된 대학 등을 매칭시켜 사용자의 단말로 보낼 토픽들을 그룹화하여 저장한다. 이는 위와 같은 기본적인 정보들뿐만 아니라 사용자가 원하는 바에 따라 식단, 온도, 습도에 대한 구독 정보 등도 포함된다.

**3. 프로젝트 내용**

**3.1. 시나리오**

**3.1.1. design concept**



Figure 1 디자인 컨셉 - 메인 화면

기본적인 메인 화면 디자인 컨셉은 Figure 1과 같이 사용자가 구독한 정보들이 한 눈에 들어오는 구조를 지향한다. 본 시스템에서 각각의 정보는 계층적 구조를 띄므로, “마인드 맵”과 유사한 모양의 배치가 될 것이다.

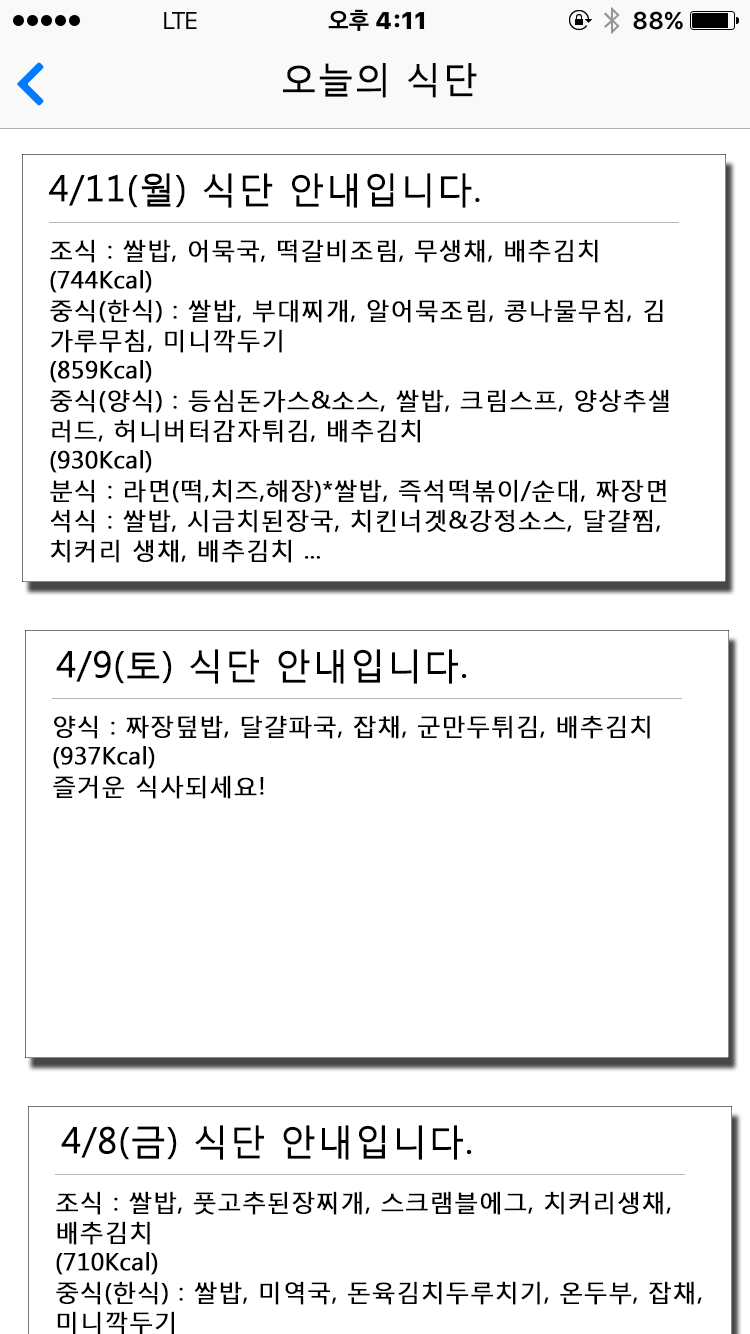


Figure 2 디자인 컨셉 - 세부 메시지 정보

Figure 1에서 구독한 정보에 대해 선택하면 Figure 2와 같이 메시지가 카드, 혹은 테이블 형태로 나열되어 표기된다.

**3.2 요구사항**

**3.2.1 애플리케이션에 대한 요구사항**

사용자가 학번을 통해 애플리케이션에 로그인하며 성공했을 시, 기존에 존재하던 학생의 수업 수강 정보, 단과대학 정보 등을 학번과 매칭시켜 알림이 갈 수 있도록 자동으로 MQTT 브로커가 사용자의 정보를 그룹 정보에 등록하여 관리해야 한다. 그 외에 학생회관 식단, 각종 학생회 행사 등 사용자가 원하는 정보의 토픽을 구독할 수도 있어야 하며 이에 해당하는 구독 정보 또한 그룹화되어 관리되어야 한다. 만약 MQTT 브로커로부터 메시지가 도착했다면 단말 상단에 알림(Notification) 의 형태로 사용자에게 노출한다.

**3.2.2 MQTT Connection 관리에 대한 요구사항**

사용자의 단말이 MQTT 브로커와의 연결이 성공했다면 연결 정보를 데이터베이스에 저장하고 주기적으로 Keep-alive 를 체크한다. 만약 일정 횟수 이상 Connection Loss가 일어난다면 그 단말에 대해서는 더 자주 Keep-alive를 체크하여 Connection Loss를 최대한 회피할 수 있어야 한다.

**3.2.3 메시지 전송 클라이언트에 대한 요구사항**

특정 그룹의 사용자에게 메시지를 보내고자 하는 교수, 행정실 직원 등은 웹 혹은 CLI 인터페이스를 통해서 메시지를 전송할 수 있어야 한다.

**3.2.4 관리자 모듈에 대한 요구사항**

본 시스템의 관리자는 학생들의 수강 내역, 단과대학 정보 등을 가지고 있어 애플리케이션으로부터 받은 학번 정보와 이를 매칭시킬 수 있어야 한다. 학생들의 정보를 추가하거나 삭제하는 기능이 요구된다.

**4. 결론 : 기대효과 및 예상 일정**

학생들이 여러 앱을 전전하며 필요한 정보를 찾는 것이 아닌, 통합된 메시지 서비스를 이용할 수 있게 된다. 뿐만 아니라 이는 확장성을 가지므로 어떠한 플랫폼의 퍼블리셔 노드를 추가하더라도 기존 시스템의 변형 없이 단순히 퍼블리싱 라이브러리를 이용함으로서 확장이 가능하다. 이러한 퍼블리셔를 허가, 혹은 불허하는 시스템이 존재하지 않고 자신이 구독하지 않은 정보는 받을 수 없는 기존 MQTT 프로토콜과 다르게 본 시스템은 관리자에 의해 슈퍼바이징된 그룹으로 묶인 사용자들이 적절한 정보를 배포 받을 수 있다. 더불어, 사용자가 직접 구독하는 방식과 다르게 Hole punching 등을 이용하여 사용자에게 직접 정보를 배포할 수 있는, 개량된 MQTT 시스템을 완성할 수 있다.

그리고 이를 구축하는 메시징 시스템 또한 프레임워크로서 배포한다면, 유사한 환경의 시스템을 유연하게 구축하는 것이 용이해질것이다. 이를테면 농장 자동화 시스템, 중소기업의 연락망 등이 이에 해당된다. 이와 같은 환경에서는 배포된 프레임워크에 사용자가 그룹에 대한 적절한 정보를 입력하고, 그에 대한 퍼블리셔만 구현한다면 본 프로젝트에서 제작하는 서비스와 유사한 서비스를 간단히 실현할 수 있게 된다.

**5. 참고문헌**

- 사물인터넷과 MQTT 기술 / 심승현, 김학범 / 정보보호학회지 제24권 제6호, 2014.12, 37-47 (11 pages)

- Design and Implementation of a Collaboration Messenger System Based on MQTT Protocol / Hwang, Hyun Cheon , Park, JiSu , Shon, Jin Gon / LECTURE NOTES IN ELECTRICAL ENGINEERING, Vol.2015, No.373 [2015], 513-520

- IoT 미들웨어 경량 메시지 프로토콜 연구 동향 / 김병준, 조성현 / 2014년도 대한전자공학회 정기총회 및 추계학술대회, 2014.11, 674-676 (3 pages)

- 모바일 통합 SNS 게이트웨이의 상위 구조 및 MQTT 기반의 푸시 알림 프로토콜 설계 / 이신호, 김현우, 주홍택 / 한국통신학회논문지 제38권 제5호(네트워크 및 서비스), 2013.5, 344-354 (11 pages)