MQTT 설치 및 실행 가이드

2016년 12월

경북대학교 컴퓨터학부 통신프로토콜연구실

정중화, 최동규

godopu16@gmail.com, supergint@gmail.com

목 차

1. 서론

사물인터넷 (IoT : Internet of Things) 의 개념은 1999년 케빈 애쉬톤 (Kevin Ashton) 에 의해 최초로 사용되었다. 초기에 케빈 애쉬톤이 사용을 하던 개념은 RFID 태그를 활용한 시스템의 발전을 시작으로 개념이 조금씩 변화되어, 최근에는 유비쿼터스 컴퓨팅을 포함하여 생활 속 유무선 네트워크로 연결 할 수 있는 모든 사물들로 범주가 커졌다. 사물인터넷은 M2M 기술을 인터넷으로 확장하여 사물은 물론, 현실과 가상세계 등 모든 정보와 상호작용 할 수 있는 인프라 개념을 의미한다. 사물인터넷은 기존 네트워크 통신망에만 국한되어 있는 것이 아니라 주변에 다양한 가전제품, 전자기기, 모바일 등 사물 대 사물, 사람 대 사물간의 네트워크를 포괄하는 차세대 패러다임이다. 사물이 인터넷에 연결됨으로 인해 사용자에게 다양한 가치를 제공할 뿐 아니라, 데이터의 수집, 온라인을 통한 실시간 관찰, 원격 제어, 정보 관리 등을 활용하여 사용자에 맞는 맞춤형 서비스 제공도 가능하다.

사물인터넷 인프라가 구축된다면 무선 네트워크, 센서, 스마트 기기 등 기술 발전 및 보급 확산으로 무수히 많은 기기가 네트워크에 접속할 것으로 예상되어 네트워크 혼잡이 예상된다. 따라서 기기 간 통신인 D2D (Device to Device) 통신 지원 기술도 개발되어야 한다. 이러한 사물인터넷의 인프라 구축이 완료된다면 사물인터넷의 영향력은 일상생활 모든 부분까지 확대 될 것이다. 특히 의료, 교통, 제조 등 다양한 분야에서 기존의 프로세스에 큰 변화를 줄 것이다.

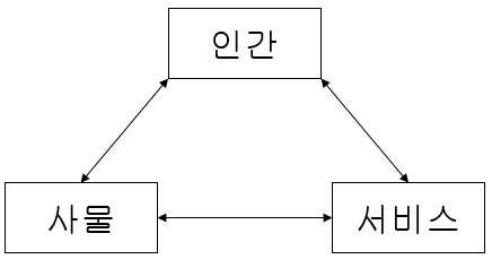

Figure 1. 사물인터넷 3대 구성 요소

인간과 사물, 사물과 사물 간의 통신과 적절한 서비스의 상호작용이 이루어져야 사물인터넷 인프라의 최대한의 효율을 낼 수 있다. 사물의 요소는 우리가 일반적으로 사용하는 전자기기 및 가전제품뿐만 아니라 경량화 된 CPU, 센서 등도 포함 하고 있기에 현재 인터넷 기술인 TCP/IP 프로토콜 스택을 맞추기에는 불가능하다. 사물인터넷은 주로 낮은 성능의 소형 기기나 센서 등을 사용하기 때문에 사용할 수 있는 자원이 한정적이다. 사물인터넷 환경은 낮은 전력, 불안정한 통신 상황, 프로그램의 경량화 등의 특성을 가지기 때문에 기존의 프로토콜들 보다는 다른 경량화 된 프로토콜이 필요하다.

현재 국제적인 획일화 된 표준은 존재하지 않지만 ITU-T, oneM2M, 3GPP, IETF, IEEE 등의 다양한 표준화 기구에서 국제적인 표준을 확립하기 위해 글로벌한 IoT/M2M 서비스 기술을 연구하고 있다. ITU-T와 oneM2M 등은 주로 스마트 헬스케어나 스마트 홈과 같은 서비스 플랫폼 표준 기술을 개발 하고있으며, 3GPP, IETF, IEEE 등의 표준 기구는 네트워크 통신을 위한 프로토콜을 주로 연구한다. 현재 표준화의 선두주자인 3GPP는 기대만큼의 수익이 나오지 않아 소강상태에 머무르고 있으며, IETF는 대표적인 프로토콜로 CoAP을 핵심적으로 사용하고 있다. IEEE는 원활한 M2M 통신을 제공하기 위해 IEEE 802.x 계열의 무선통신 기술을 확장하는 표준 개발이 이루어 지고 있다.

국내에서의 표준 활동은 TTA를 중심으로 M2M 서비스 요구사항, 이동통신 무선 접속 기술, 표준 플랫폼 간의 인터페이스 등을 표준화 하는 작업을 ITU-T, oneM2M 등과의 국제표준화 공동협력을 추진 하고 있다.

2. MQTT란?

2.1 MQTT 소개

MQTT는 1999년 IBM에서 M2M과 사물인터넷에서 사용하기 위해 만들어진 경량의 Publish-Subscribe기반 메시지 프로토콜이다. 사물인터넷의 한계인 낮은 전력, 낮은 대역폭, 낮은 성능 등의 환경에서도 사용 할 수 있도록 설계됐다. 가전기기, 빌딩, 집, 산업 등 다양한 영역에서의 정보를 수집할 수 있다.

MQTT는 메시지를 Publish 하고, 관심 있는 주제를 Subscribe 하는 것을 기본 원칙으로 한다. Publish-Subscribe 모델은 센서들 가운데 Broker가 필요하다. Broker는 관심있는 Topic을 기반으로 클라이언트들에게 메시지를 나눠주는 역할을 한다. 이 때 클라이언트는 Publisher와 Subscriber 모두를 지칭한다. Publish와 Subscribe는 Topic을 기반으로 작동한다. Topic은 '/'를 이용해서 계층적으로 구성할 수 있어 대량의 센서 기기들을 중복 현상 없이 효율적으로 관리 할 수 있다.

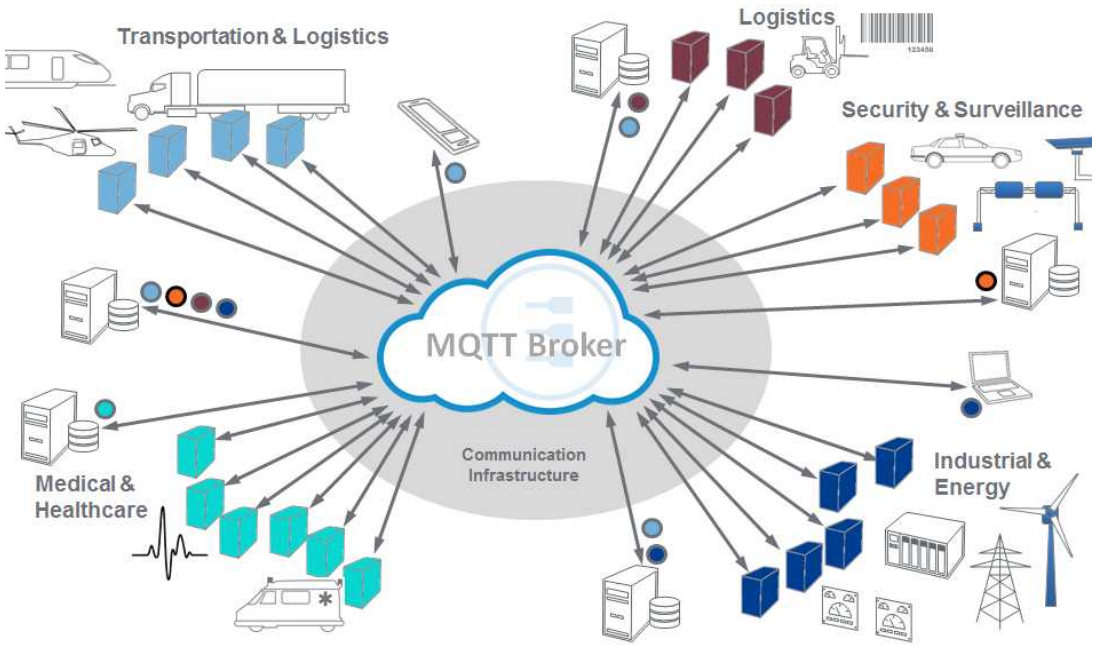

Figure 2. MQTT System 개요도

MQTT로 통신을 하기 위해서는 우선 Publisher가 브로커에 연결을 요청해야한다. 연결이 완료 된 후 Publisher는 관심 있는 주제로 브로커에게 정보를 전송 할 수 있다. 만약 연결을 끝내고 싶을 때는 Publisher가 브로커에게 연결 해제 요청을 전송하면 된다.

MQTT는 신뢰성을 위해 3가지의 QoS(Quality of Service) 레벨을 정의한다. Qos 0 레벨은 메시지를 최대 1번만 전송 하는 방식으로 메시지가 잘 도착 했는지 중간에 손실 되었는지 알 수 없다. QoS 1 레벨은 메시지를 적어도 한 번 이상 보내는 방식으로 메시지가 잘 도착하면 수신 측에서 ACK 메시지를 전송 한다. 이 ACK 메시지를 보고 Publish/Subscribe가 잘 되었는지 확인 할 수 있다. QoS 2 레벨은 메시지를 정확히 1번 전송 한다. Publish를 하게 되면 수신 측에서 잘 받았으면 PUBACK를 송신 측으로 보내게 되고, 송신 측이 PUBACK까지 잘 받았다면 다시 PUBREL 메시지를 보낸 후 수신 측으로부터 PUBCOMP를 받게 되면 메시지 전송이 완료 된다. QoS 2는 가장 신뢰성이 높은 방식이지만 큰 오버헤드 때문에 낮은 성능의 기기나 좋지 못한 네트워크 환경에서는 사용하기 힘들다.

2.2 MQTT Broker Mosquitto 소개

Mosquitto는 이클립스에서 만든 MQTT 프로토콜 버전 3.1과 3.1.1을 구현한 오픈 소스 메시지 브로커입니다. Mosquitto는 publish/subscribe 모델을 이용한 경량화 통신 기능을 제공합니다. 이 프로그램은 저전력 센서, 모바일 기기, 임베디드 컴퓨터 사이의 통신과 같은 IoT 환경에서의 통신에 적합합니다.

3. Mosquitto broker 설치

MQTT 프로토콜을 사용하기 위해 오픈소스로 제공되는 Mosquitto 브로커를 설치합니다.

3.1 Broker 다운로드

Mosquitto 브로커를 다운로드 하기 위해 http://mosquitto.org/download/ 로 이동하여 windows 버전의 설치파일을 다운로드 받습니다.

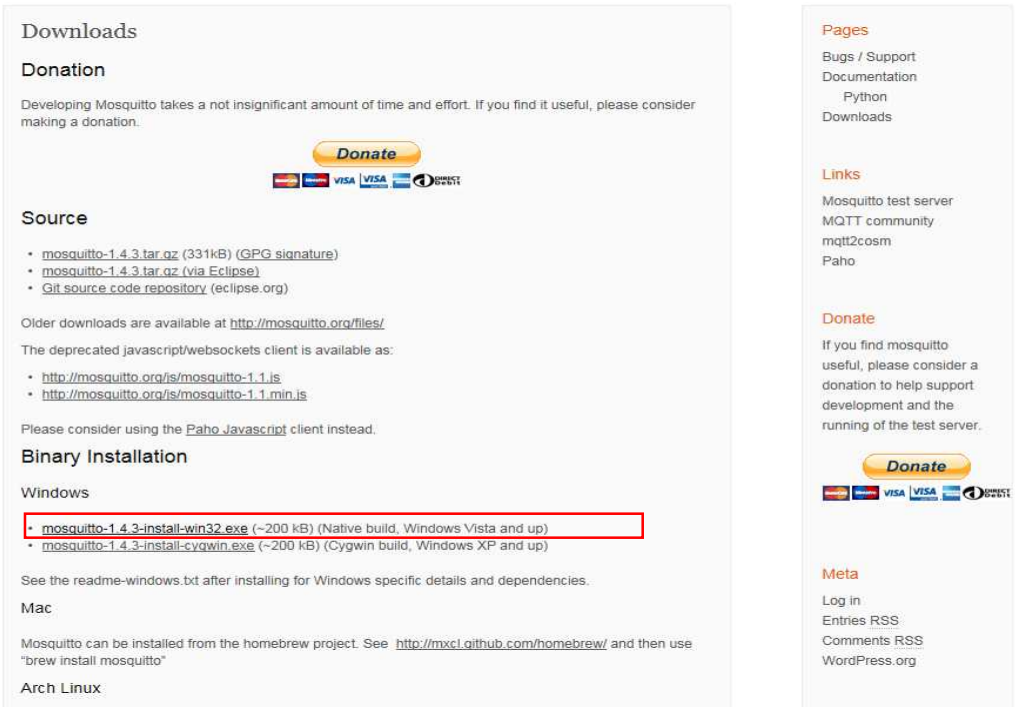

Figure 3. Mosquitto 설치 파일 다운로드

3.2 Broker 설치

다운로드 받은 설치파일을 실행시킵니다. Mosquitto는 내부적으로 보안을 위한 OpenSSL과 Pthread 라이브러리를 사용하기 때문에 OpenSSL을 설치하고 Pthread dll 파일을 다운로드 하여야 합니다. 설치 중 설치 파일과 dll파일을 다운로드 할 수 있는 링크를 알려주니 해당 링크로 이동하여 OpenSSL 설치 파일과 Pthread dll파일을 다운로드 합니다. 나머지는 기본 설정으로 두고 설치를 계속 진행합니다.

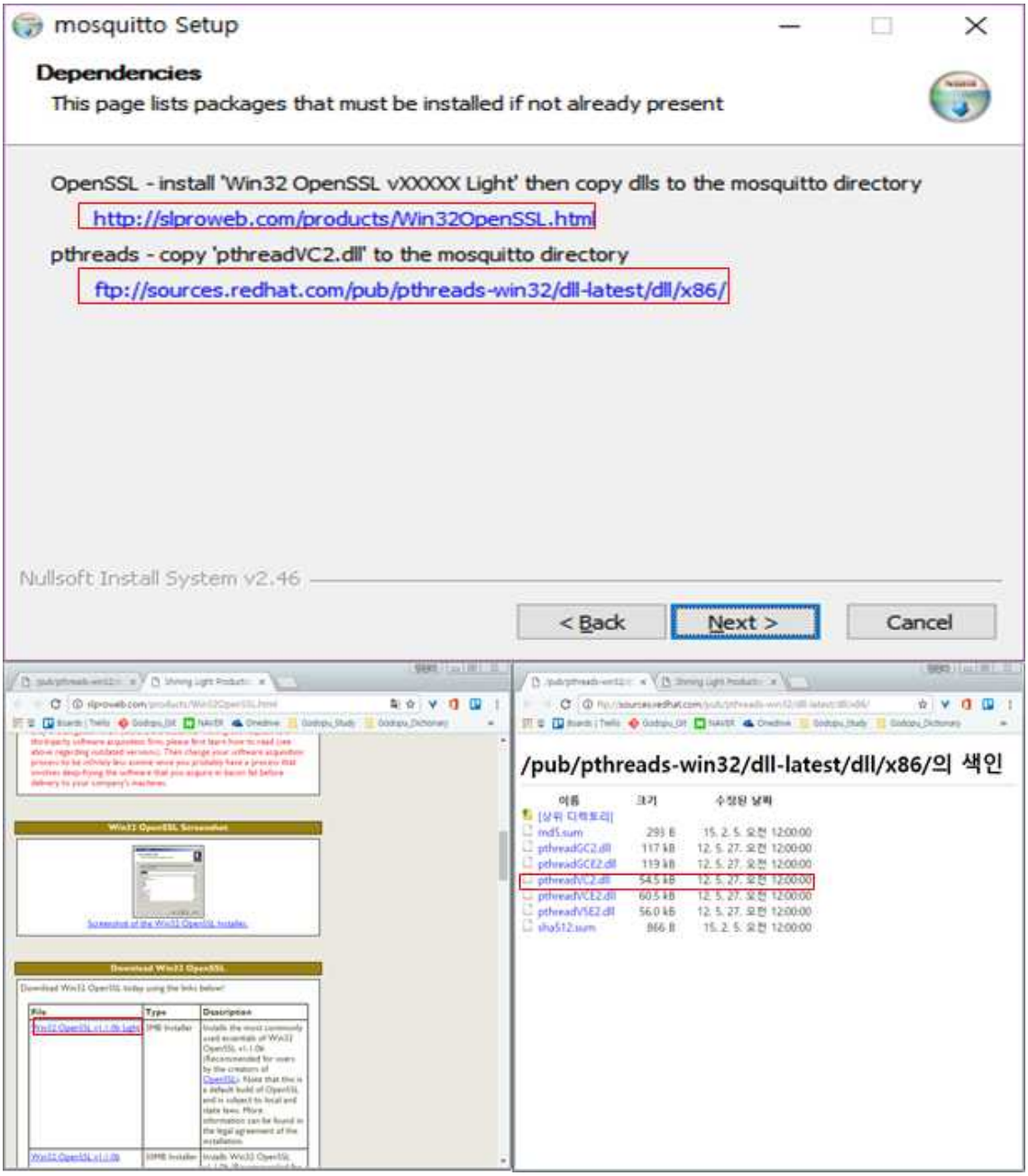

Figure 4. OpenSSL & Pthread dll 다운로드

Mosquitto 설치가 끝나면 설치 중 다운로드한 PthreadVC2.dll 파일은 Mosquitto 디렉터리(C:₩Program Files (x86)₩mosquitto)에 넣어주고, 다운로드한 설치 파일을 이용하여 OpenSSL을 설치합니다. OpenSSL 설치 시 OpenSSL dll 설치 디렉터리를 그림과 같이 변경시켜줍니다. 다른 설정은 변경하지 않고 설치를 진행합니다.

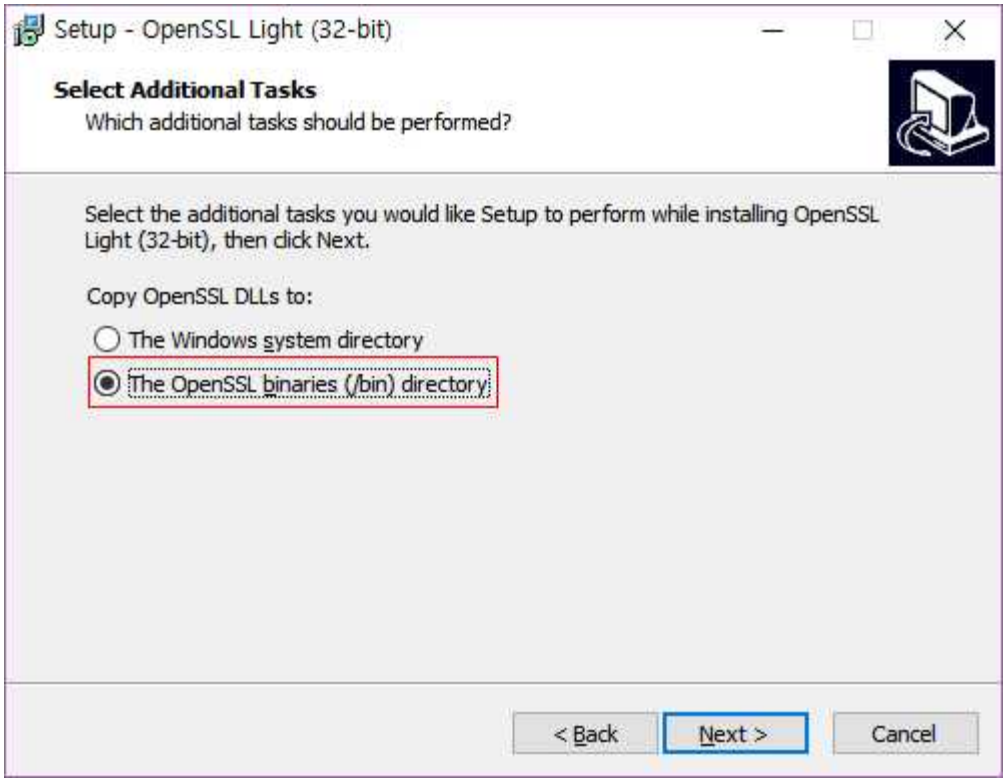

Figure 5. OpenSSL 설치

OpenSSL 설치가 완료되면 OpenSSL이 설치된 디렉터리(C:₩OpenSSL-Win32₩bin)에 있는 모든 dll 파일들을 Mosquitto 디렉터리(C:₩Program Files (x86)₩mosquitto)에 복사합니다.

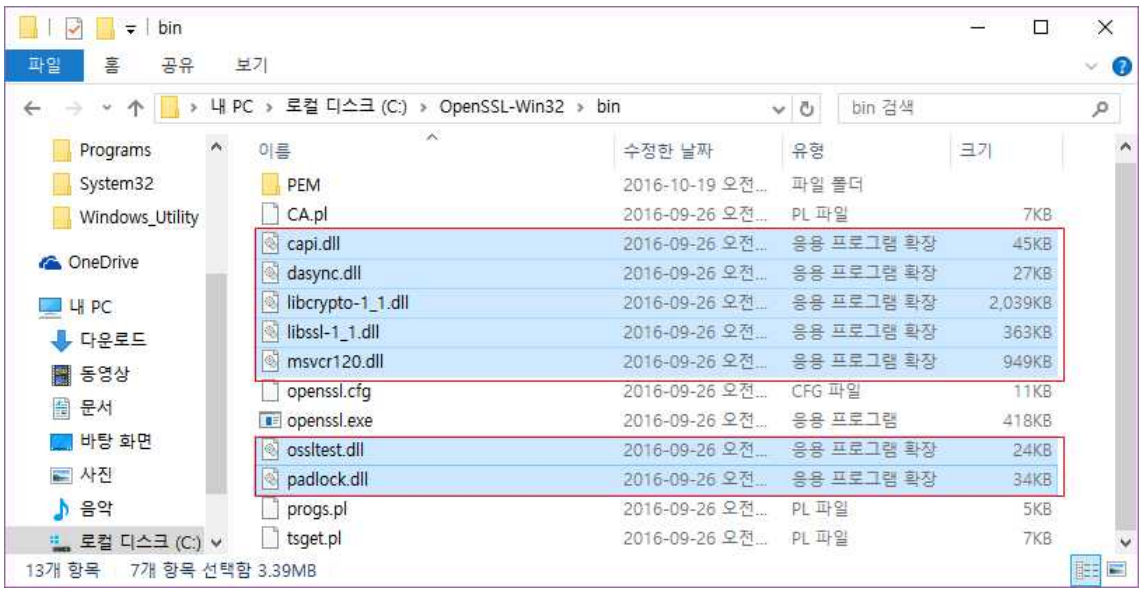

Figure 6. OpenSSL dll 파일들

3.3 Broker 실행

다음과 같은 순서로 Mosquitto를 실행합니다.

1) CMD를 실행합니다.

2) Mosquitto 디렉터리 (C:₩Program Files (x86)₩mosquitto)로 이동합니다.

   → cd C:₩Program Files (x86)₩mosquitto

3) mosquitto.exe를 실행합니다.

   → mosquitto.exe -v

Mosquitto 브로커 실행

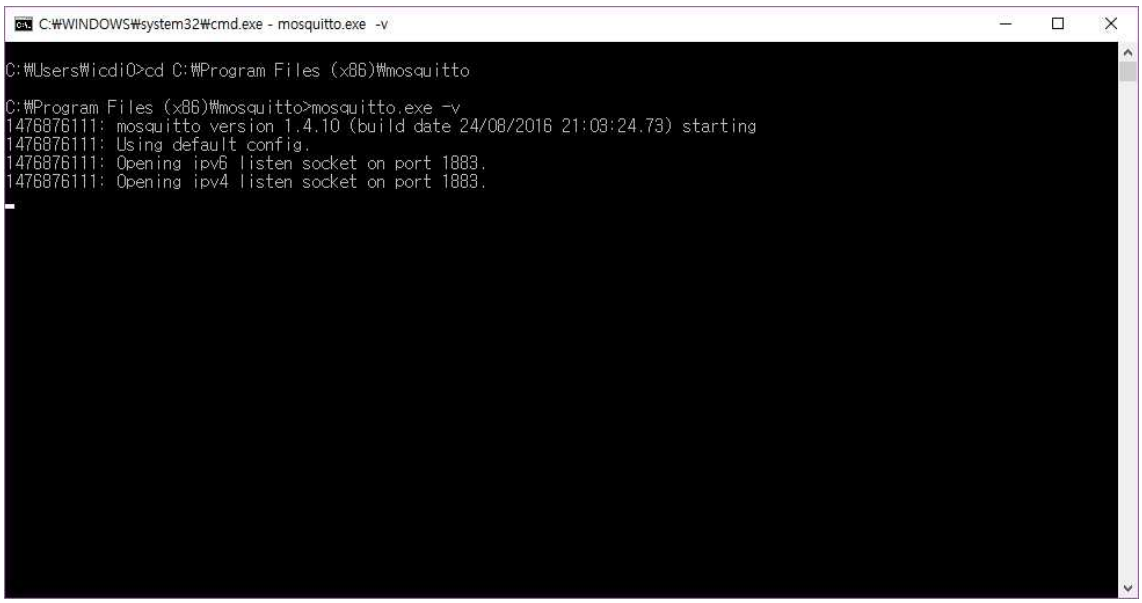

Figure 7. Mosquitto 실행 화면

3.4 Broker 테스트

Mosquitto가 제대로 설치되었는지 알아보기 위해 다음과 같은 순서로 mosquitto를 테스트 합니다. 먼저 mosquitto_sub 프로그램을 simpletest 토픽을 구독하도록 실행합니다.

1) CMD를 실행합니다.
2) Mosquitto 디렉터리 (C:₩Program Files (x86)₩mosquitto)로 이동합니다.

   → cd C:₩Program Files (x86)₩mosquitto
3) mosquitto_sub.exe 를 실행합니다.

   → mosquitto_sub.exe –h localhost –t "simpletest"

mosquitto_sub.exe 실행

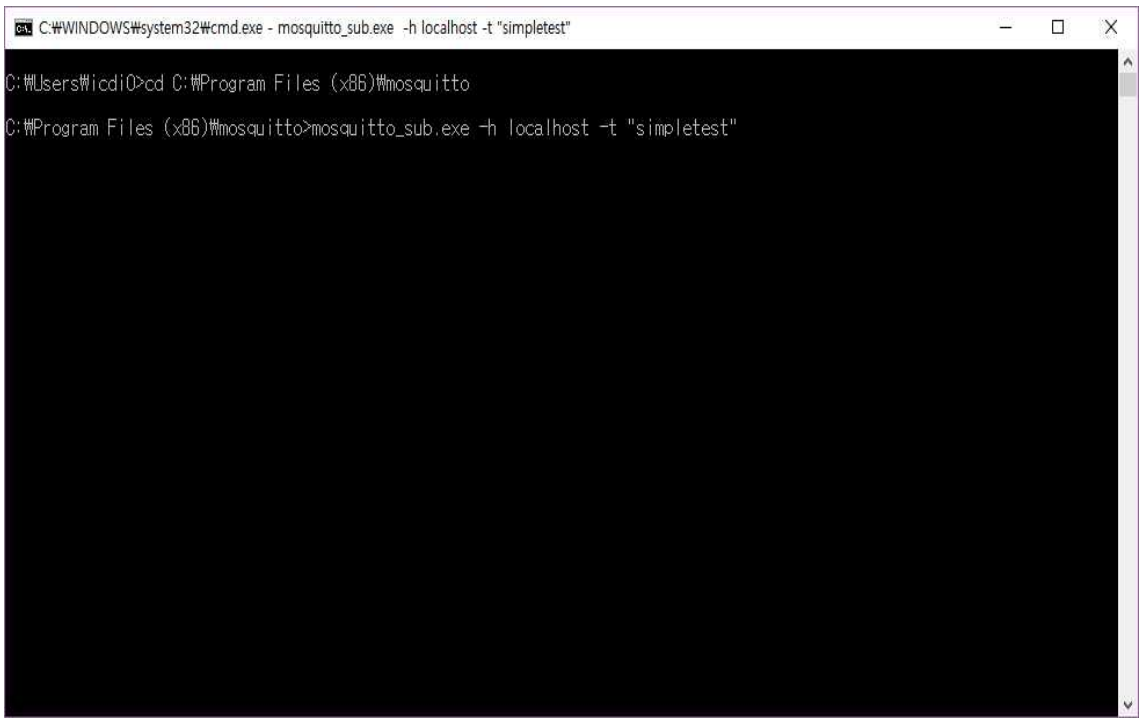

Figure 8. mosquitto_sub.exe 실행화면

이제 다음과 같은 순서로 mosquitto_pub프로그램을 simpletest 토픽을 구독하고 있는 클라이언트들에게 메시지를 전달하도록 실행 후 메시지가 전달 되는 것을 확인합니다.

1) CMD를 실행합니다.
2) Mosquitto 디렉터리 (C:₩Program Files (x86)₩mosquitto)로 이동합니다.
   → cd C:₩Program Files (x86)₩mosquitto
3) mosquitto_pub.exe 를 실행합니다.
   → mosquitto_pub.exe –h localhost –t "simpletest" –m "Hello Mosquitto"

mosquitto_pub.exe 실행

4. Paho를 이용한 Pub-Sub 구현

MQTT를 쉽게 사용할 수 있도록 도와주는 JAVA 기반의 Paho 라이브러리가 존재합니다. Paho 라이브러리를 이용하여 Mosquitto에게 데이터를 보내는 publisher와, publish된 데이터를 받는 Subscriber client를 구현해 보도록 하겠습니다.

4.1 JRE 7 설치

MQTT Paho 라이브러리는 자바로 구현되어 졌기 때문에 사용하기 위해서는 JDK가 필요합니다. JAVA는 Oracle이 소유하고 있으므로 Oracle 홈페이지로 이동합니다.
http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html 주소로 접속하여 JDK 설치 파일을 다운로드 합니다.

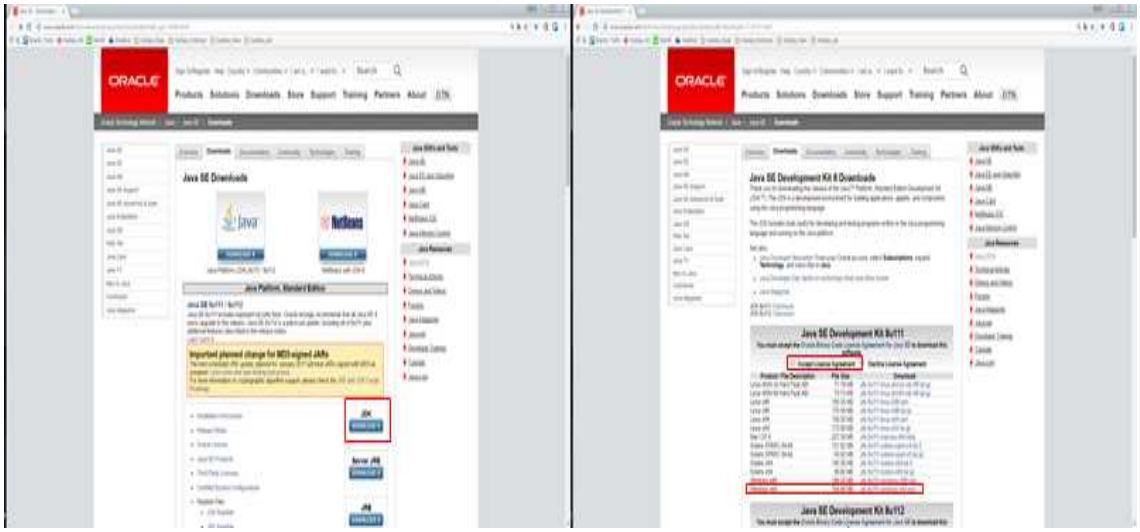

Figure 9. JDK 다운로드

4.2 Paho library 다운로드

Paho library를 사용하기 위해 Paho JAR (JAVA Archive) 파일을 다운로드 합니다.

http://mvnrepository.org/artifact/org.eclipse.paho/org.eclipse.paho.client.mqttv3/1.1.0 로 이동하면 Paho 라이브러리 JAR 파일을 다운로드 할 수 있습니다.

Figure 10. Paho library 다운로드

4.3 Paho library 예제

Paho library를 편하게 사용할 수 있도록 간단한 예제를 만들었습니다. Eclipse를 이용하여 프로젝트를 Import한 후 실행시켜 봅니다. 예제는 http://protocol.knu.ac.kr 에 방문하시면 다운로드 받을 수 있습니다. 다운로드한 파일의 압축을 풀어주세요. 압축을 푼 디렉터리를 Workspace로 하여 이클립스를 실행한 후 아래 그림의 절차를 따라하여 프로젝트를 import 합니다.

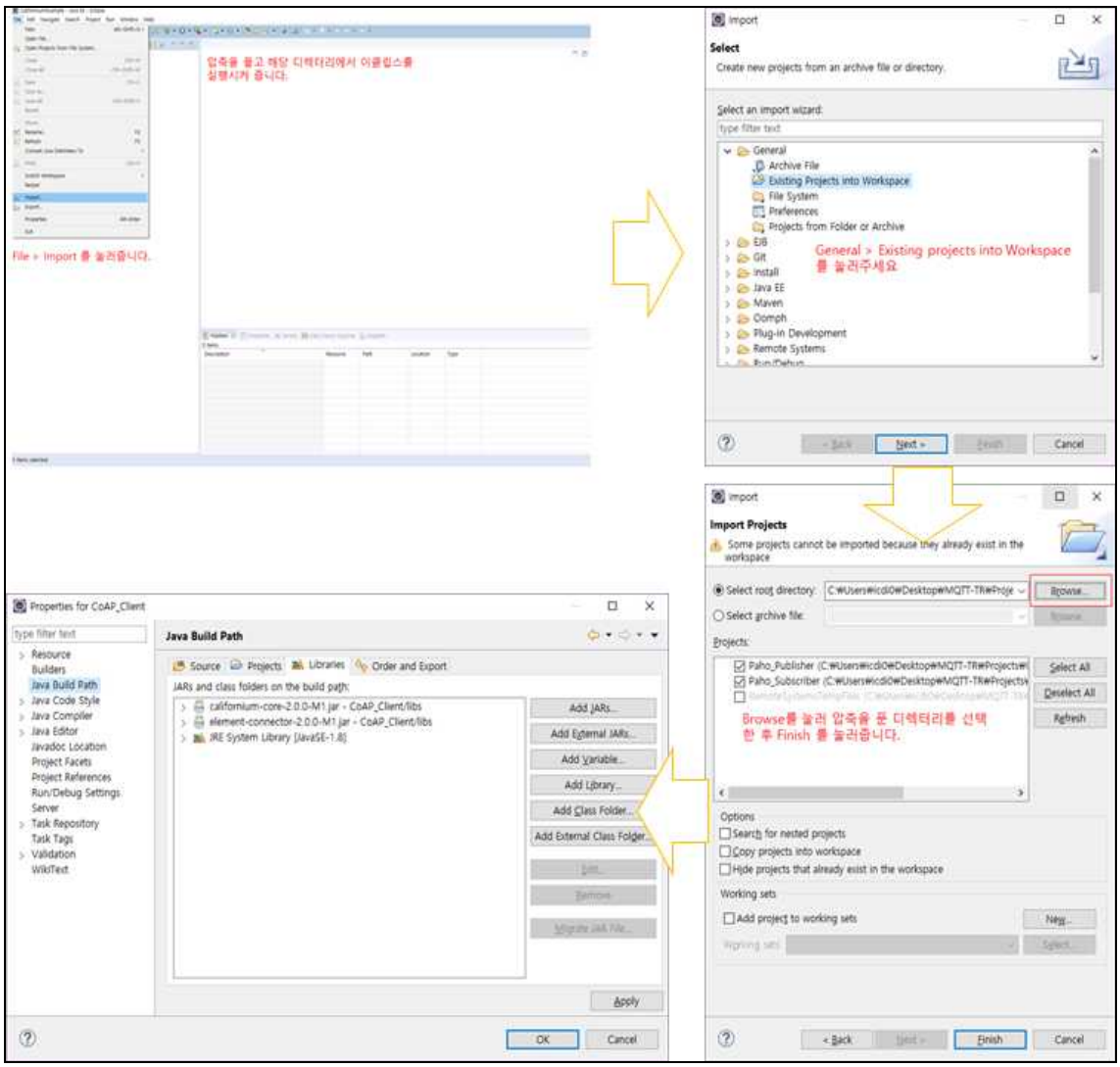

Figure 11. import 프로젝트 절차

4.4 테스트

프로젝트 import가 완료되었으면, 앞의 "3.3 Broker 실행" 페이지를 참고하여 mosquito.exe 프로그램을 실행한 후 아래의 그림을 참고하여 프로젝트에 Broker IP Address를 Arguments 로 추가한 후 Paho_Subscriber , Paho_Publisher 순으로 실행합니다.

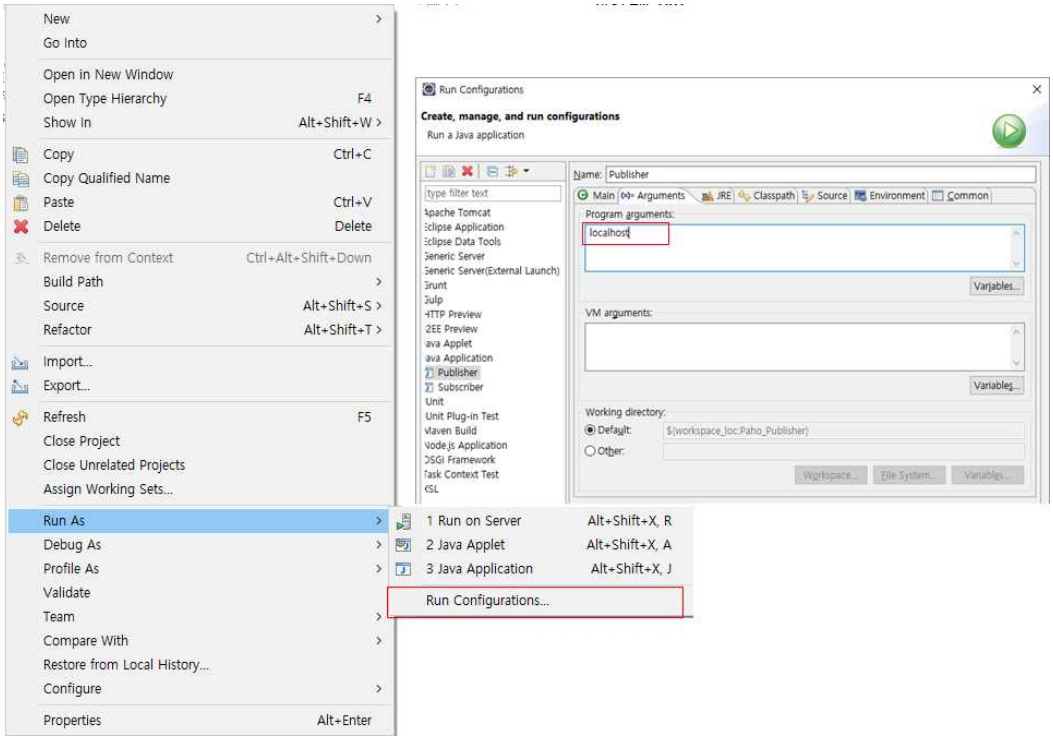

Figure 12. 프로젝트 실행 환경 구성

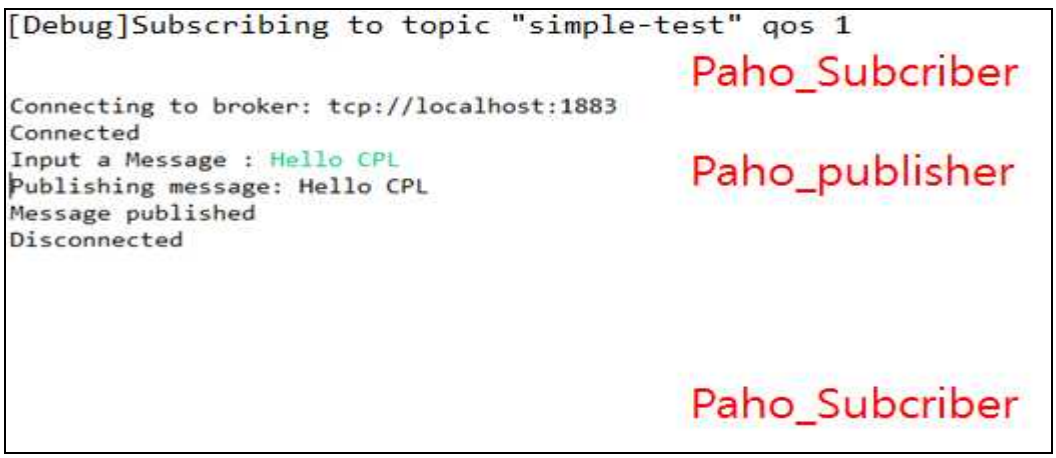

Figure 13. 프로젝트 실행화면