

자기주도 PJT

IoT 통신 프로토콜 MQTT 이해와 실습

목차

| | |
|-----------------|---|
| 1. 과제 개요 | 3 |
| 2. 참고 자료 | 3 |
| 3. 기능 명세 | 5 |
| 4. 산출물 제출 | 9 |

1. 과제 개요

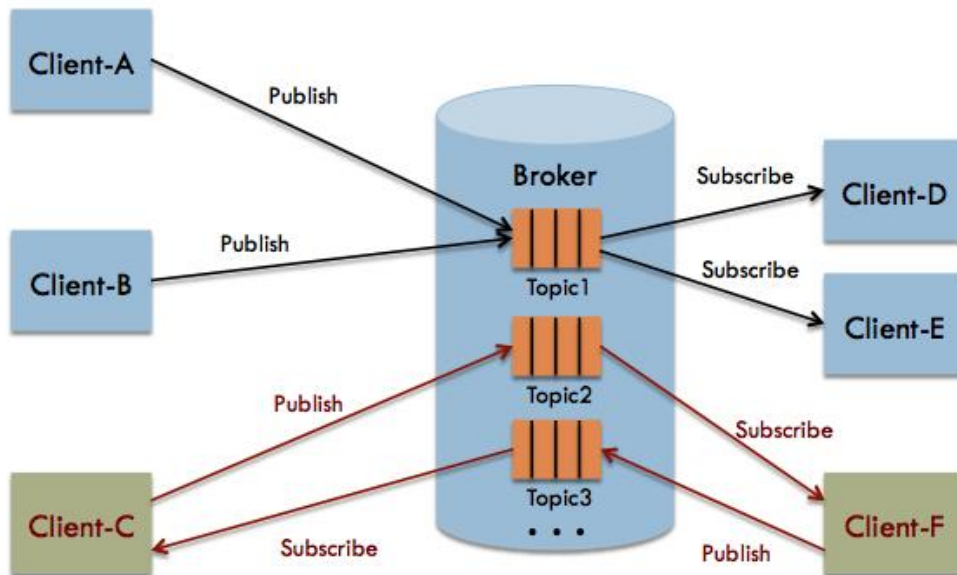
MQTT(Message Queuing Telemetry Transport)는 M2M(Machine to Machine), IoT(Internet of Things) 통신에 사용되는 표준 기반 메시징 프로토콜입니다. 일반적으로 리소스 제약이 있는 네트워크를 통해 제한된 대역폭으로 데이터를 전송하는데 사용하며, 구현이 쉽고 IoT 데이터를 효율적으로 전달할 수 있기 때문에 널리 사용되고 있습니다.

또한 이러한 특징으로 인해 초기 페이스북 메시지 앱에서 메시지 전송 속도 개선을 위해 MQTT 프로토콜 방식이 도입되었습니다. 또한 우아한형제들에서도 배달의 민족 주문중계채널의 기존 설계 구조 방식(API Poling)의 한계로 이용자 수가 많아지면서 운영비용 상승이 발생하게 되었고, 이를 해결하기 위해 MQTT를 적용하여 중계시스템을 개선하였습니다.

이처럼 MQTT는 M2M, IoT 뿐만 아니라 다양한 곳에서 사용되고 있습니다. 본 과제는 이러한 MQTT를 이해하고, 간단한 MQTT 클라이언트를 구현해 보는 실습 과제입니다.

2. 참고 자료

본 과제를 진행하기 위해서는 먼저 일반적인 서버-클라이언트 구조와는 다른 MQTT 통신 방식을 이해해야 합니다. MQTT는 브로커(Broker)라고 불리는 서버를 거쳐서 통신을 하게 됩니다. 이때 주고 받는 메시지는 토픽(Topic)을 기준으로 작동합니다. 발행자(Publisher)는 토픽을 발행하기 위한 목적으로, 구독자(Subscriber)는 토픽을 구독하기 위한 목적으로 브로커 서버에 연결합니다. 하나 이상의 발행자와 구독자가 브로커 서버에 연결해서 토픽을 발행하거나 구독할 수 있습니다. 즉 일대일, 또는 일대다 통신이 모두 가능합니다.



MQTT 이해와 과제를 위한 참고 자료

| 구분 | 제목 | 링크 |
|----|-------------------------|---|
| 이해 | MQTT | https://mqtt.org |
| 이해 | MQTT 적용을 통한 중계시스템 개선 | https://techblog.woowahan.com/2540 |
| 이해 | Facebook 메신저와 MQTT | https://d2.naver.com/helloworld/1846 |
| 이해 | Eclipse paho | https://www.eclipse.org/paho |

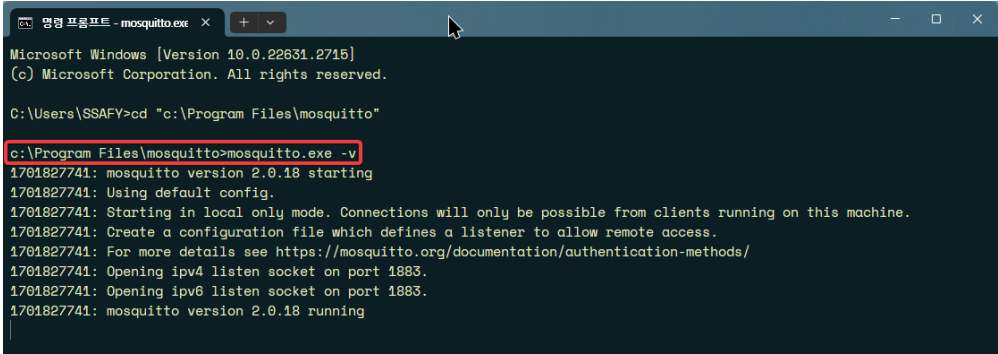
3. 기능 명세

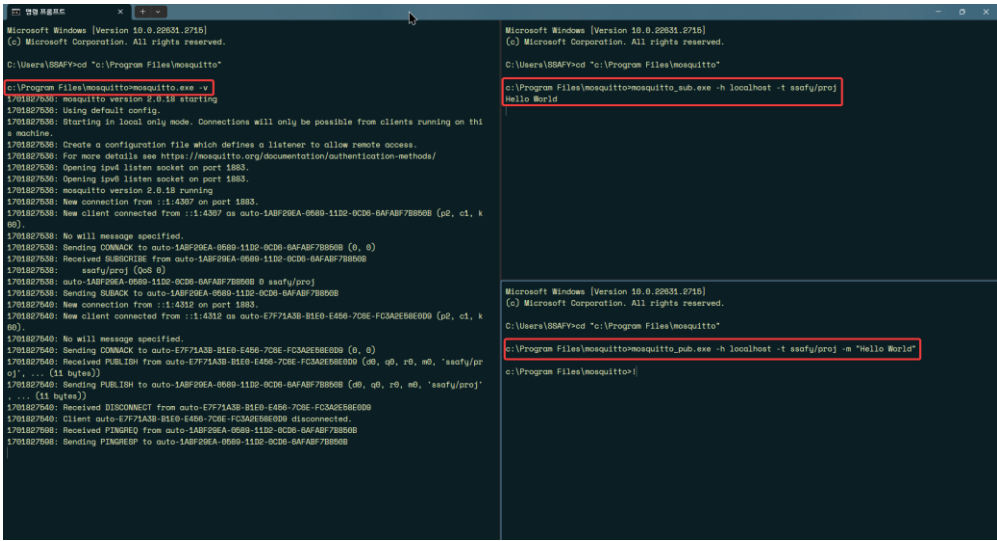
1. 기능/과제 목록

| Req. | Category |
|------|--------------------------|
| 1 | MQTT Broker 설치 및 동작 확인 |
| 2 | MQTT 를 이용한 IoT 통신 시스템 구현 |

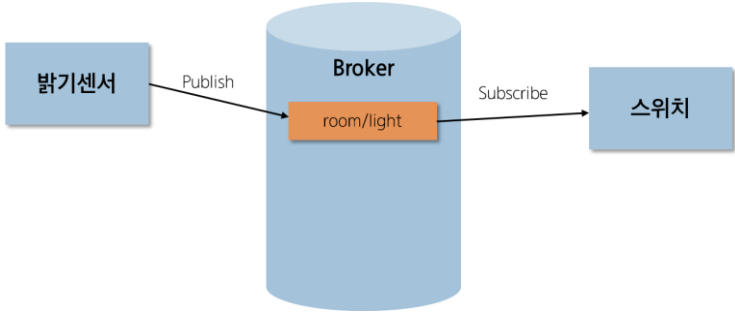
2. 기능/과제 상세

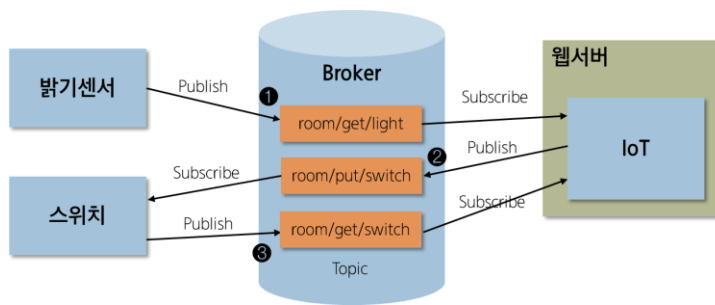
Req. 1. MQTT Broker 설치 및 동작 확인

| | |
|----------|--|
| Req. 1-1 | MQTT Broker 설치 및 구동 |
| 기능 상세 | <p>본 과제에서는 MQTT Broker 로 오픈소스 프로젝트로 널리 사용되고 있는 Mosquitto 를 사용합니다. (설명에서는 윈도우 기준으로 설명)</p> <p>1. Mosquitto 설치 아래 사이트에서 운영체제에 맞는 설치파일을 다운 받아 설치합니다. https://mosquitto.org/download</p> <p>2. Mosquitto 구동 cmd 실행한 후 Mosquitto 설치 폴더로 이동, 그리고 <code>mosquitto.exe -v</code> 명령어를 입력합니다.</p>  |

| Req. 1-2 | MQTT Broker 동작 확인 | | | | | | | | | | | | |
|----------------------|--|-------|----|--------------------|---------------------------|----------------------|----------------------|------------------|---|-------------------|-------------------------|-------------------|---------------------------|
| 기능 상세 | <p>Mosquitto 는 설치 시 간단하게 Publish/Subscribe 를 테스트할 수 있는 CLI 명령어가 포함되어 있습니다. Mosquitto 설치 폴더에 가시면 다음과 같은 실행파일을 확인할 수 있습니다.</p> <table border="1"> <thead> <tr> <th>실행 파일</th><th>설명</th></tr> </thead> <tbody> <tr> <td>mosquitto_ctrl.exe</td><td>Mosquitto broker 초기화 및 설정</td></tr> <tr> <td>mosquitto_passwd.exe</td><td>Mosquitto 비밀번호 파일 관리</td></tr> <tr> <td>mosquitto_rr.exe</td><td>Mosquitto v5/3.1.1 request/response 클라이언트</td></tr> <tr> <td>mosquitto_pub.exe</td><td>Mosquitto publish 클라이언트</td></tr> <tr> <td>mosquitto_sub.exe</td><td>Mosquitto subscribe 클라이언트</td></tr> </tbody> </table> <p>이 중 mosquitto_pub.exe 와 mosquitto_sub.exe 클라이언트를 이용하여 간단하게 메시지를 주고 받을 수 있습니다.</p> <p>Mosquitto 가 구동중인 상태에서 다음과 같이 publish/subscribe 클라이언트 명령어를 각각 cmd 창에서 실행합니다. 명령어의 자세한 내용은 Mosquitto man 페이지에서 확인하세요. (https://mosquitto.org/documentation)</p> <pre>> mosquitto_sub.exe -h localhost -t ssafy/proj > mosquitto_pub.exe -h localhost -t ssafy/proj -m "Hello World"</pre>  <p>Publish 를 통해 보낸 메시지가 Subscribe 에 잘 나타난다면 Mosquitto Broker 가 정상 구동 상태입니다.</p> | 실행 파일 | 설명 | mosquitto_ctrl.exe | Mosquitto broker 초기화 및 설정 | mosquitto_passwd.exe | Mosquitto 비밀번호 파일 관리 | mosquitto_rr.exe | Mosquitto v5/3.1.1 request/response 클라이언트 | mosquitto_pub.exe | Mosquitto publish 클라이언트 | mosquitto_sub.exe | Mosquitto subscribe 클라이언트 |
| 실행 파일 | 설명 | | | | | | | | | | | | |
| mosquitto_ctrl.exe | Mosquitto broker 초기화 및 설정 | | | | | | | | | | | | |
| mosquitto_passwd.exe | Mosquitto 비밀번호 파일 관리 | | | | | | | | | | | | |
| mosquitto_rr.exe | Mosquitto v5/3.1.1 request/response 클라이언트 | | | | | | | | | | | | |
| mosquitto_pub.exe | Mosquitto publish 클라이언트 | | | | | | | | | | | | |
| mosquitto_sub.exe | Mosquitto subscribe 클라이언트 | | | | | | | | | | | | |

Req. 2. MQTT 를 이용한 IoT 통신 시스템 구현

| | |
|----------|---|
| Req. 2-1 | 기본 통신 구현 (Publish/Subscribe) |
| 기능 상세 | <p>앞서 Mosquitto Broker 동작 확인을 위해 사용하였던 mosquitto_pub.exe, mosquitto_sub.exe 와 유사하게 평소에 사용하던 친숙한 언어를 이용하여 CLI(Command Line Interface)로 구현합니다. 참고 자료 중 Eclipse paho 는 MQTT 클라이언트 오픈소스 프로젝트로 C/C++, Java, Python, JavaScript 등 많은 언어의 클라이언트 라이브러리를 제공합니다. 예제도 제공하고 있으니 참고하여 구현합니다.</p> <p>가상 IoT 시스템은 다음과 같이 밝기센서와 전등 스위치로 구성되어 있습니다.</p>  <pre> graph LR A[밝기센서] -- Publish --> B[(Broker room/light)] B -- Subscribe --> C[스위치] </pre> <p>밝기센서와 스위치 모두 클라이언트로 각각 다음과 같이 구현합니다.</p> <p>밝기센서</p> <ul style="list-style-type: none"> ● 밝기값(integer)을 인자로 받음 ● "room/light" 토픽으로 밝기값을 발행 <p>스위치</p> <ul style="list-style-type: none"> ● "room/light" 토픽 구독 ● 전달 받은 밝기값에 따라 스위치 제어(해당 메시지 출력) <ul style="list-style-type: none"> - 밝기 ≤ 20 : 스위치 ON - 밝기 > 20 : 스위치 OFF <div style="display: flex; justify-content: space-between; margin-top: 20px;"> <pre style="background-color: #2e3436; color: #eeeeec; padding: 10px;"> D:\03_Personal\exam_mqtt\Req1>python client_light.py 19 D:\03_Personal\exam_mqtt\Req1>python client_light.py 20 D:\03_Personal\exam_mqtt\Req1>python client_light.py 21 </pre> <pre style="background-color: #2e3436; color: #eeeeec; padding: 10px;"> D:\03_Personal\exam_mqtt\Req1>python client_switch.py rc: 0 Subscribed: 1 room/light b'19' Turn on light room/light b'20' Turn on light room/light b'21' Turn off light </pre> </div> |
| Req. 2-2 | 복합 IoT 시스템 통신 구현 |
| 기능 상세 | <p>Req. 2-1 에서 구독과 발행을 좀 더 복잡하게 구성한 시스템입니다. 밝기센서 값에 따라 IoT 서버에서 판단하여 스위치를 제어하도록 하였으며, 스위치는 동작에 대한 결과값을 IoT 서버에 다시 반환하도록 하였습니다.</p> |



- ① 밝기센서 > IoT
 - "room/get/light" 토픽 사용
 - IoT 는 전달 받은 밝기값 출력
- ② IoT > 스위치
 - "room/put/switch" 토픽 사용
 - 밝기값에 따라 스위치 제어 명령어 전달
 - * 밝기 ≤ 20 : "ON" 메시지 발행
 - * 밝기 > 20 : "OFF" 메시지 발행
 - 스위치는 전달 받은 제어 명령어 출력
- ③ 스위치 > IoT
 - "room/get/switch" 토픽 사용
 - 스위치 제어 결과 전달
 - IoT 는 전달 받은 결과 출력

```

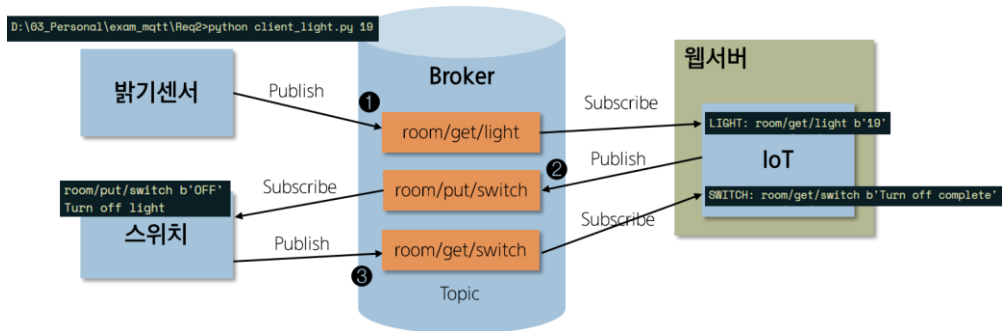
exam_mqtt\Req2>python client_light.py 19
exam_mqtt\Req2>python client_light.py 20
exam_mqtt\Req2>python client_light.py 21
  
```

```

D:\03_Personal\exam_mqtt\Req2>python server_iot.py
rc: 0
Subscribed: 1
LIGHT: room/get/light b'19'
SWITCH: room/get/switch b'Turn off complete'
LIGHT: room/get/light b'20'
SWITCH: room/get/switch b'Turn off complete'
LIGHT: room/get/light b'21'
SWITCH: room/get/switch b'Turn on complete'
  
```

```

D:\03_Personal\exam_mqtt\Req2>python client_switch.py
rc: 0
Subscribed: 1
room/put/switch b'OFF'
Turn off light
room/put/switch b'OFF'
Turn off light
room/put/switch b'ON'
Turn on light
  
```



4. 산출물 제출

<https://lab.ssafy.com/s10-study/self-project/> 의 “산출물 제출 가이드”.docx 참조