Impact Korea 2012

모바일 메시징 솔루션
MQTT(MQ Telemetry Transport)

발표자 : 지용득 부장

직함 : 웹스피어 기술영업팀, 소프트웨어 사업부

Agenda

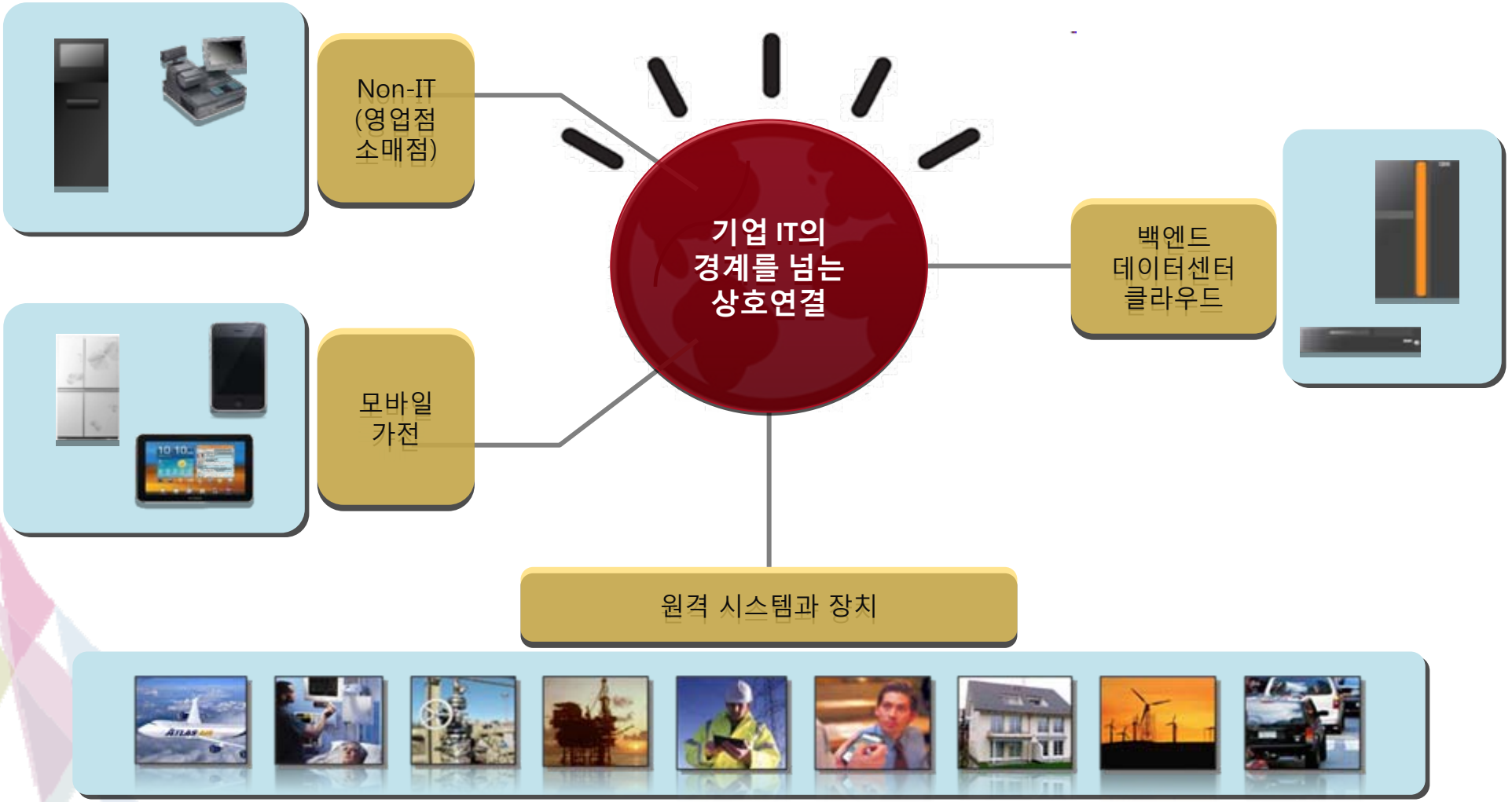
1 모바일의 전개에 따른 웹 환경의 변화

2 모바일 메시징, MQTT 프로토콜

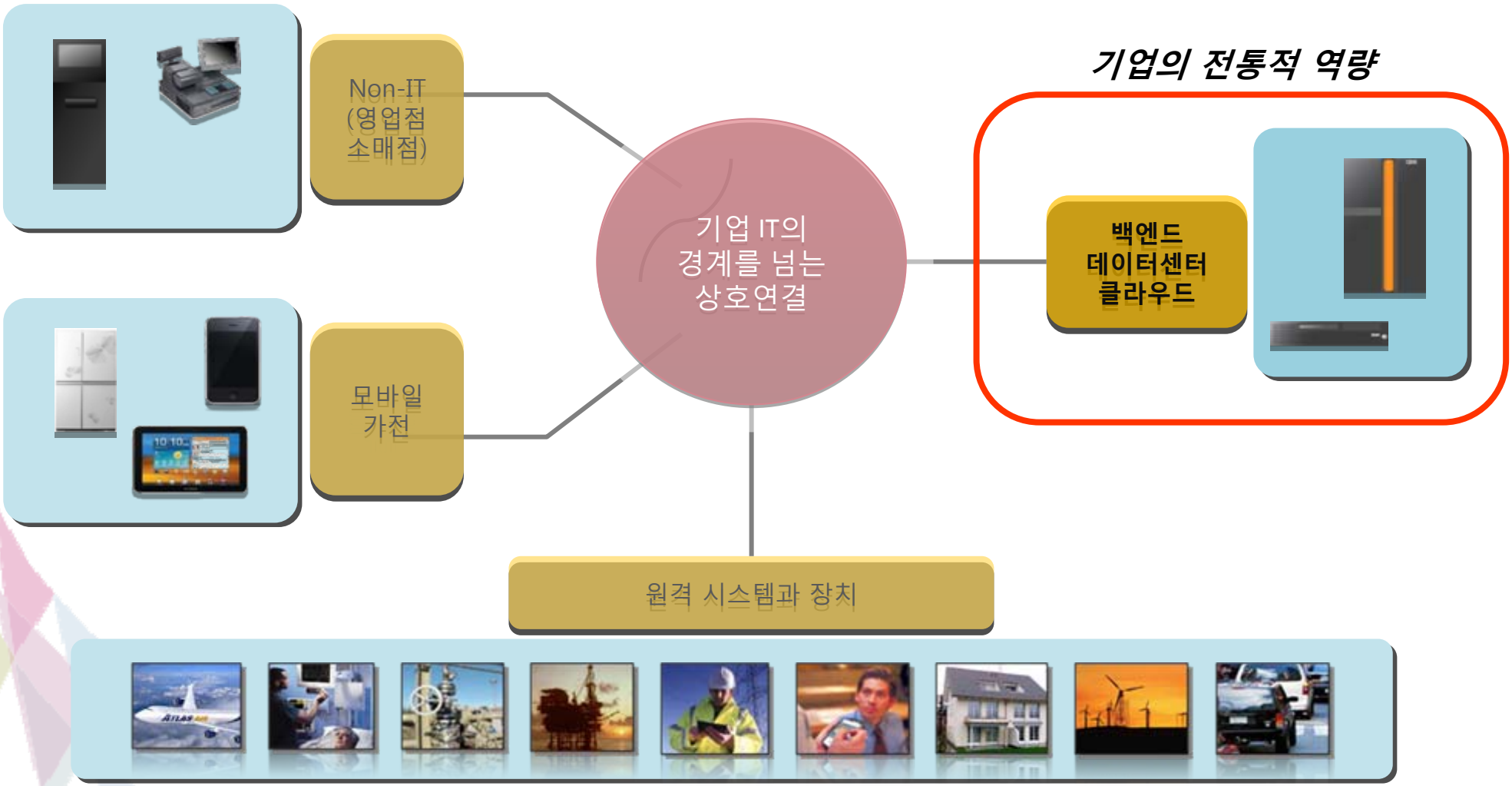
3 MQTT 구현과 적용 모델

1 모바일의 전개에 따른 웹 환경의 변화

IBM의 Smarter Planet Connectivity: "Internet of Things"



기업의 전통적 IT 역량은 내부의 기간 업무에 집중



"Internet of Things"

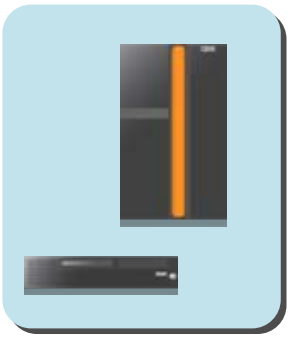
"Smarter Planet 시대에 요구되는 새로운 역량"

Non-IT
(영업점
소매점)

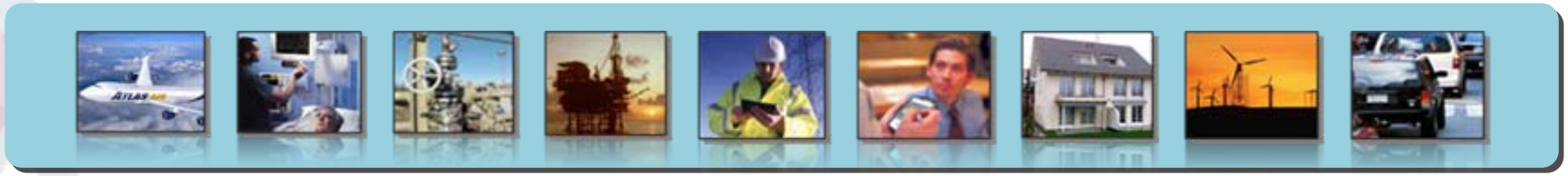
모바일
가전

기업 IT의
경계를 넘는
상호연결

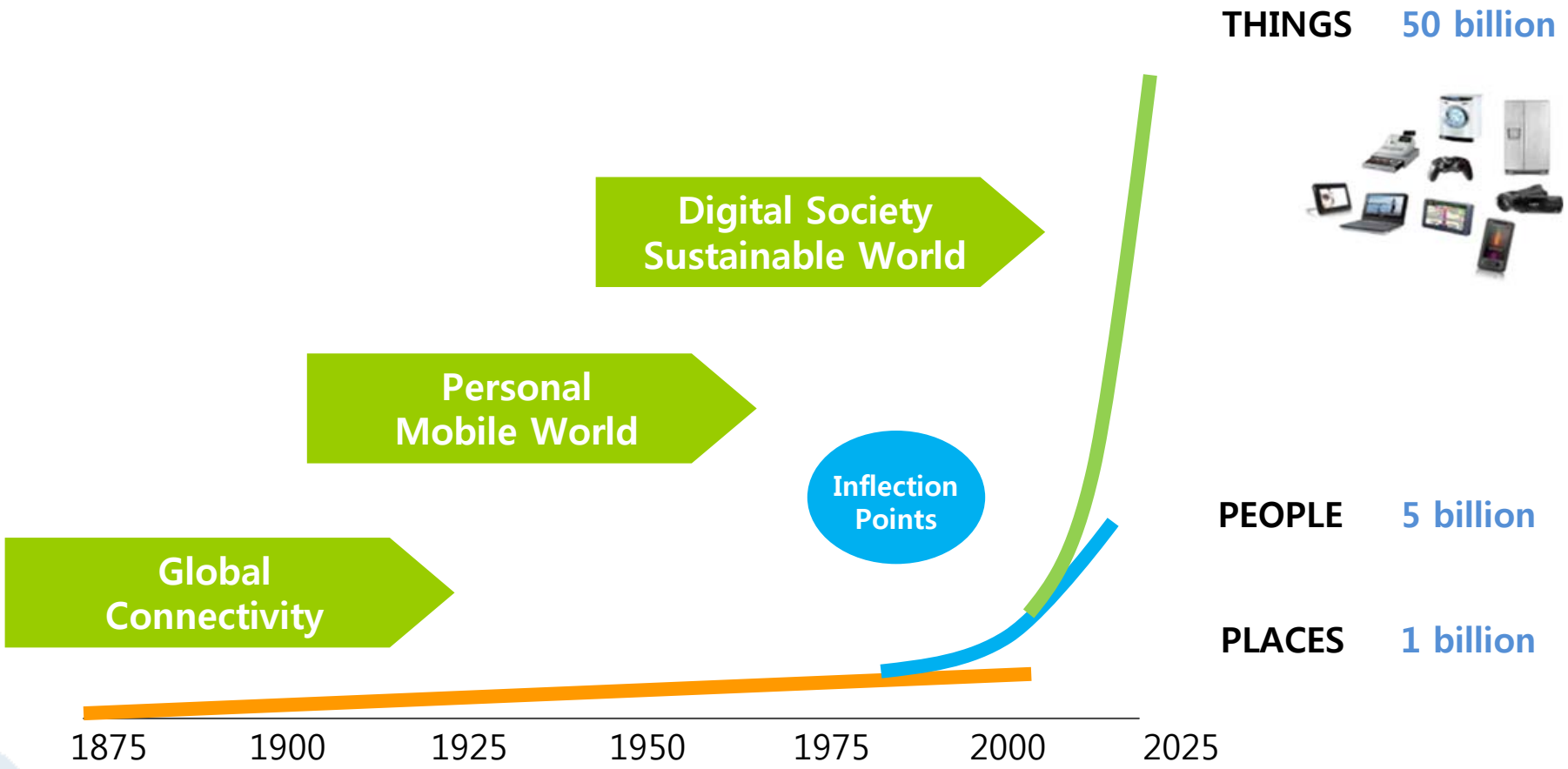
백엔드
데이터센터
클라우드



원격 시스템과 장치



모바일의 성장은 "컴퓨터가 아닌 것들(Things)의 연결"을 촉발



출처: Ericsson AB, "Infrastructure Innovation – Can the Challenge be Met?" Sept 2010

모바일과 기존의 데스크톱 환경의 상이함

데스크톱	모바일
자리에 앉아서 작업	이동 중에 작업
문서 지향	메시지 지향
크고 복잡한 앱	특정 목적을 위한 미니 앱
사용자 상황에 중립적	사용자 상황에 대한 인지
태스크 중심	고지(Notification) 중심
지속적 전력 공급	배터리에 의한 전력 공급
예측 가능한 네트워크 응답	예측할 수 없는 네트워크 응답

다양한 모바일의 요구는 앱/통신 환경에 변화를 초래

HTML 4

- 애플리케이션은 온라인 상태에 있어야 함
- 추가 기능을 위해 브라우저 플러그인 필요
- "콘텐트"

- 오프라인 애플리케이션 가능
- 더 풍부한 웹앱 실행 환경 – 플러그인 불필요
- "애플리케이션 플랫폼"

HTTP

Web Socket

- **자체 메시징 신뢰 메커니즘 결여:**
  - 메시지 유실 가능
- **폴링(Polling) 지향:**
  - 불필요하게 추가적인 커뮤니케이션
  - 과도한 배터리 사용
- **장황한 프로토콜:**
  - 메시지 전송에 많은 통신 왕래
- **단방향 호출 구조:**
  - 항상 클라이언트가 커뮤니케이션을 개시

- **TCP 소켓처럼 양방향 데이터 교환:**
  - 푸시(Push)나 스트리밍이 가능
- **여전히 중요 기능들이 결여:**
  - **완결된 메시징 프로토콜이 아닌** 저수준 TCP 소켓
  - 메시지 **전달에 대한 보장 메커니즘 부재**
  - 모바일에 적합한 **Pub/Sub 커뮤니케이션에 대한 지원이 없음**
  - 메시지 처리에 대한 로직을 프로토콜 위에 구현해야 함

모바일 영역의 통신 방식 진화의 전형: Push Notification

SMS

- 어떤 환경에서든 사용(모든 디바이스/서비스에서 지원)
- 메시지 건 당 과금으로 상대적으로 비쌈
- 활용에 제한이 많은 텍스트 기반 고지(Notification)
- 메시지에 대한 기밀성(Confidentiality)이 없음

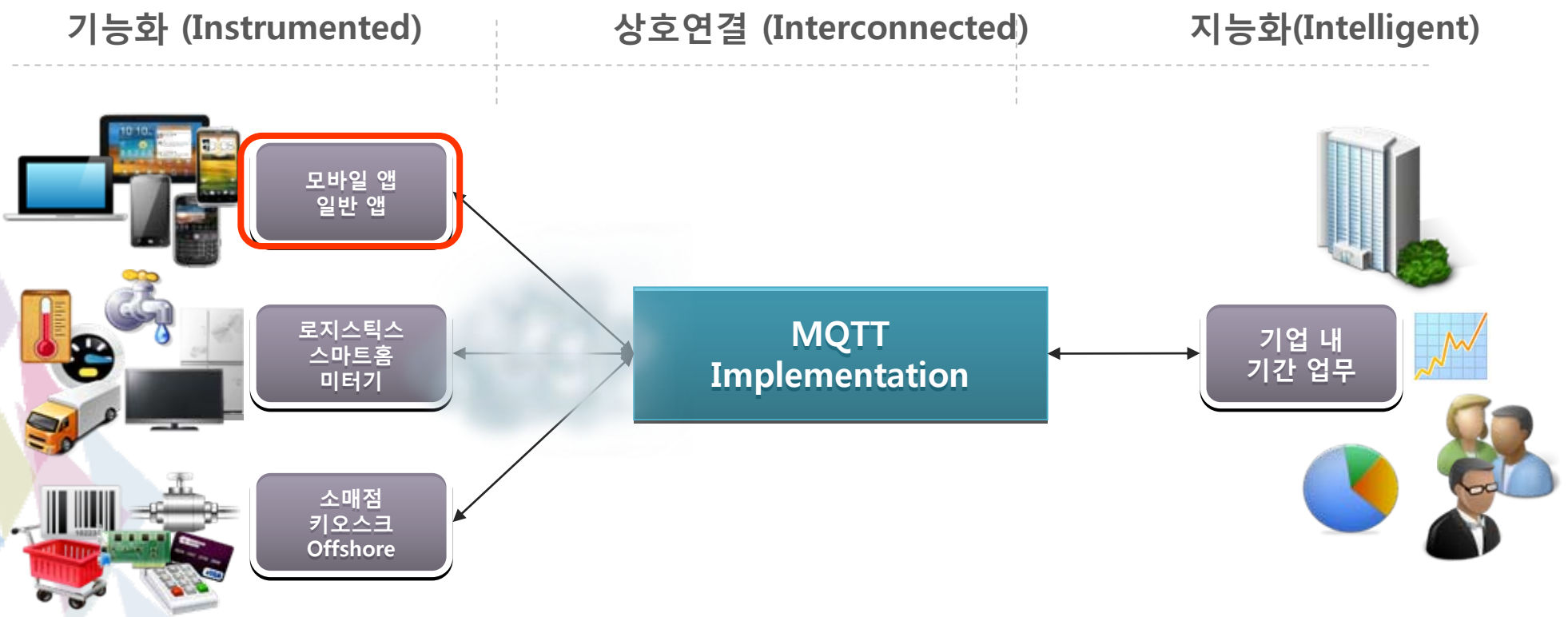
모바일 벤더 서비스

- 모바일 OS 별로 각기 다르며 상호 호환성 없음
  - APN(Apple), C2DM(Google), BBM(RIM)
- 데이터 플랜이 필요
- 활용에 제한이 있는 텍스트 기반 고지(Notification)
- 기밀 데이터에 적합하지 않음
  - 각 메시지는 Apple이나 Google, RIM을 경유함
- APN/C2DM은 단방향 통신만 허용됨
- 사용료는 엄밀히 무료가 아님
  - 서비스 이용 정책 변화에 따라 달라질 수 있음

2 모바일 메시징, MQTT 프로토콜

MQTT: Smarter Planet 프로토콜

- 텔레메트리 장치, 센서 그리고 **모바일 기기**에 최적화된 메시징 프로토콜
- 기업 외부의 이벤트를 토대로 한 지능적인 의사결정을 가능하게 하는 기술
- 정적/동적인 자산, 인력, 업무공간에 대한 원격 관리의 근간이 되는 메시징 인프라



페이스북 메신저: 350,000,000 모바일 사용자를 MQTT로 연결

"...성능의 향상을 통해 **메시징에 있어서의 경험을 크게 개선시키도록 디자인**되었습니다. **배터리를 덜 소모하면서도 이를 달성**하기 위해 **우리는 MQTT라는 프로토콜을 사용**했습니다... **네트워크 대역과 배터리를 적게 쓰도록 설계**되었습니다... 기기와 기기 간에 몇 초가 걸리는 게 아니라 **수백 밀리 초에 메시지 전달이 가능**합니다."

- Lucy Zhang, 페이스북 엔지니어 2011년 10월 www.facebook.com/lucyz
  (MQTT used by their 350M mobile users, 475 mobile operators)

MQTT, 개방형 프로토콜: Eclipse Paho 프로젝트

Open Messaging for Machine-to-Machine(M2M) and "Internet of Things"

http://eclipse.org/paho

...Paho는 **장치 연결에 있어서 태생적인 물리적, 비용적인 제약을 반영**하였습니다. 프로젝트의 목표 중에는 장치와 애플리케이션 간에 효과적인 수준의 디커플링이 포함되어 있으며... Paho는 **임베디드 플랫폼에 MQTT의 Pub/Sub 클라이언트 구현을 탑재**하고 있습니다...

MQTT is Open and Proven!!!

MQTT 프로토콜 설계의 의도

- 프로토콜이 차지하는 모든 면의 **리소스 점유(footprint)를 최소화**
- 느리고 품질이 낮은 **네트워크의 장애와 단절에 대비**
- 클라이언트 애플리케이션 동작에 **자원 활용이 극히 제한적임을 고려**
- 다수의 클라이언트 연결에 적합한 **Publish/Subscribe 네트워크 채용**
- **신뢰성 있는 메시징**을 위한 QoS 옵션 제공
- **개방형 표준 메시징 프로토콜을 지향** – 제3자(3rd Party) 기기 제조업체와 소프트웨어 개발업체의 용이한 도입, 적용을 유도

MQTT 프로토콜 Key Features: Open, Simple

- **IBM과 Eurotech(Arcom)에 의해 1999년 최초 개발**
  - 발표 후 10년 이상 유지/발전
- **센서/장치 + 모바일 기기들의 연결을 위한 프로토콜**
- **MQTT 프로토콜 스펙은 오픈 소스로 공개**
  - www.mqtt.org에 스펙 및 클라이언트 라이브러리 공개
  - 2012년 Eclipse 오픈소스 M2M Working Group 내 M2M 프로젝트(Paho)에 MQTT 클라이언트 라이브러리 공여
- **단순하고 미니멀한 Pub/Sub 메시징 체제**
  - 기업 경계 밖의 Edge 네트워크 장치와 기업 내의 백엔드 애플리케이션 간 메시지 교환에 적합
  - 간편한 메시징을 위한 직관적 verb set(connect/disconnect publish/subscribe) 제공
- **오버헤드를 최소화**
  - 가장 작은 메시지 사이즈는 2 byte: 가변 길이 MQTT 헤더 + 애플리케이션 Payload
  - Payload 데이터에 중립적: 별도의 다른 애플리케이션 헤더 불필요
  - 클라이언트 라이브러리: C 버전은 30Kb, Java 버전은 100Kb 내외

MQTT 프로토콜 Key Features: Reliable

- **Pub/Sub에 있어서 세 가지 메시징 신뢰성을 위한 QoS 레벨 제공**:
  - **반드시 전달되어야 하는 중요 메시지에 대한 전달 보장**
  - 0 – 메시지가 최대 1번 전달, 유실 가능성 있음
  - 1 – 메시지가 최소 1번 전달, 중복 전달 가능성 있음
  - 2 – 메시지가 단 한 번, 정합성 있게 전달
- **클라이언트와 서버 간의 연결을 잃었을 때 이를 보정하기 위한 자체 기능**:
  - **Last will and testament**: 클라이언트가 예고 없이 연결을 잃을 경우 이벤트가 서버에서 발생, 서버 측에서 연결의 유실 여부를 인지
  - **Durable subscription**: 서버에 클라이언트의 구독(subscription) 정보 저장됨, 세션 종료 후 재접속 시에도 재작업 없이 Pub/Sub 유지
  - **Clean session 기능**: 연결 해제 후 다시 연결되었을 때의 이전 세션 유지/삭제 선택

*대규모 기업을 위한 MQTT 서버 구현이 WebSphere MQ V7.1에 기본 포함되어 있습니다: **"WebSphere MQ Telemetry"***

MQTT 프로토콜 Key Features: Scalable

최근 국내의 한 고객을 위한 성능 및 확장성 테스트를 IBM Hursley 연구소에서 수행한 결과

1 개의 WebSphere MQ Queue Manager*에

240,000 개의 MQTT 클라이언트를 동시에 연결

초당 **480 개**의 메시지(200 bytes)를 송수신*

5% 미만의 CPU 사용*

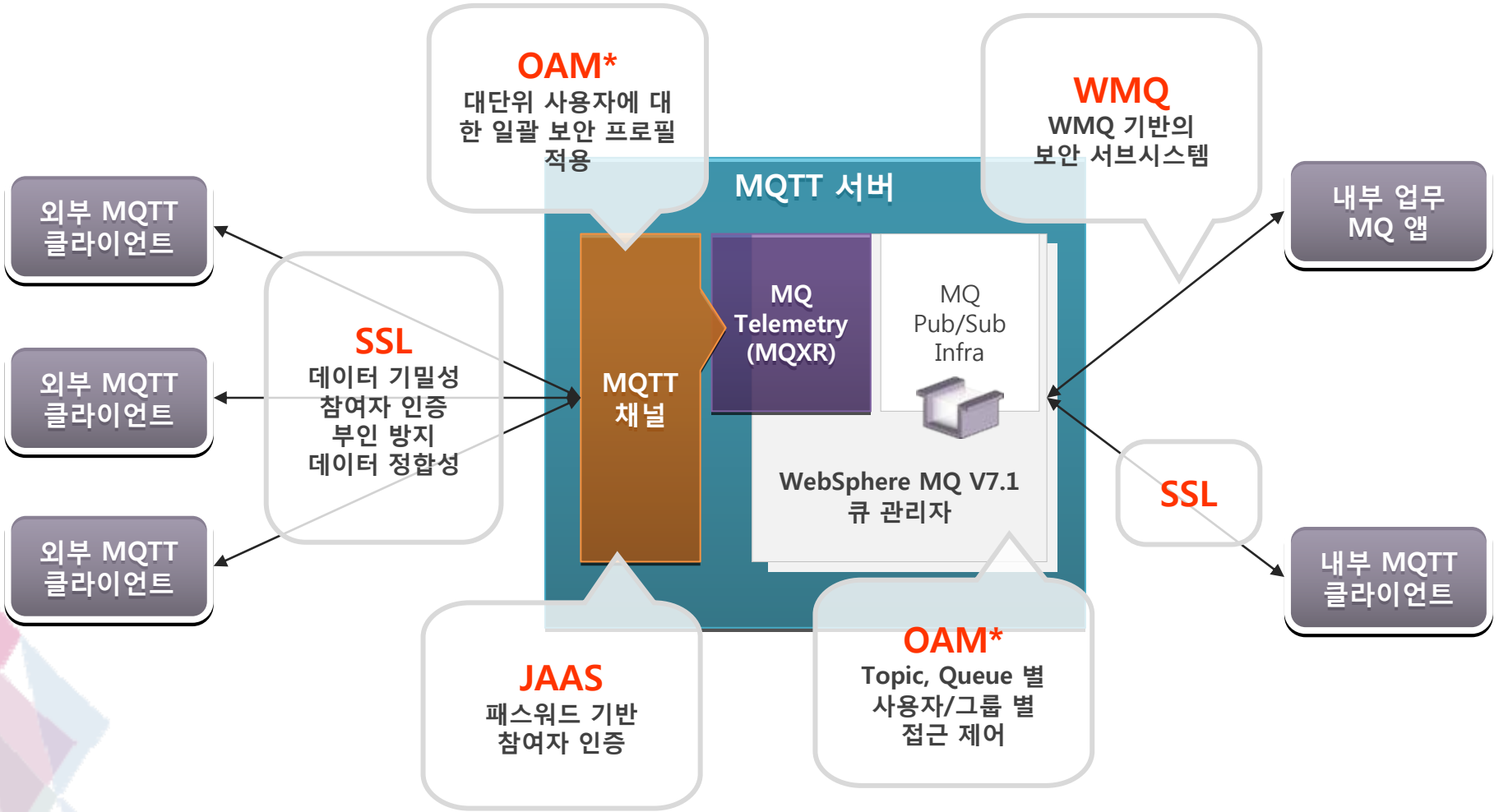
더 많은 클라이언트를 동시에 연결할 수 있었으나 실행 시스템 부족으로 240,000 개의 동시 연결까지만 테스트 수행

비교: Apache 웹 서버는 25,000 개의 동시 연결이 상한

* System x3650 M3 Xeon® 5660 2.80GHz x 12Core with 32GB RAM on 10Gbit Ethernet

MQ Telemetry 내의 MQTT 보안 지원

IBM에서 구현한 엔터프라이즈용 MQTT 서버 기능이 WebSphere MQ V7.1에 포함되어 있으며 기업 내부로 유통되는 MQTT 접근에 대한 보안성을 제공



*WebSphere MQ에 포함된 권한부여 체제로 Object Authority Manager의 약자임

HTTP 프로토콜과의 비교 – 데이터 사용 측면

Web API가 활성화되면서 모바일 내 다양한 영역에서 광범위하게 쓰이고...

- 프로토콜 자체가 Stateless한 성격: **다른 특성을 갖는 통신을 위해 더 장황(Verbose)해짐**
- Request/Reply 또는 Push Notification 모델의 경우: **이벤트에 대한 폴링(Polling)을 의미**

IBM Hursley 연구소 테스트

시나리오	HTTP	MQTT
1. Get a single piece of data from the server	302 bytes	69 bytes (~4 times)
2. Put a single piece of data to the server	320 bytes	47 bytes (~7 times)
3. Get 100 pieces of data from the server	12600 bytes	2445 bytes (~5 times)
4. Put 100 pieces of data to the server	14100 bytes	2126 bytes (~7 times)

유럽의 한 완성차 생산업체

차량 텔레매틱스

모바일 네트워크 데이터 사용료(추정)
Data Costs/Vehicle/Year*

HTTPS	MQTT(SSL)
220€/vehicle /year	23€/vehicle /year

*비교 1일 200 Byte 길이의 100 개의 메시지 Payload를 기준으로 1-2€/100MB TCP 전송 가격을 기준으로 한 것임

HTTP 프로토콜과의 비교 – 배터리 사용 측면

- HTTP와 MQTT는 프로토콜 성격이 다름; Stateful vs. Stateless – 이를 보완하여 비교*
- 전체 비교 결과와 상세 설명: http://stephendnicholas.com/archives/1217

Push를 위한 연결 유지 및 대기

	% Battery / Hour			
	3G		Wifi	
Keep Alive (Seconds)	HTTPS	MQTT (SSL)	HTTPS	MQTT (SSL)
60	1.11553	0.72465	0.15839	0.01055
120	0.48697	0.32041	0.08774	0.00478
240	0.33277	0.16027	0.02897	0.00230
480	0.08263	0.07991	0.00824	0.00112

Push 메시지 수신**

% Battery Used			
3G		Wifi	
HTTPS	MQTT (SSL)	HTTPS	MQTT (SSL)
0.34645	**0.27239**	0.04817	**0.00411**

*Push Notification 시나리오를 가정하여 Push 메시지 수신을 위해 메시징 서버와 연결을 유지하는 것이며 HTTP는 특성 상 이러한 유형의 커뮤니케이션이 불가하여 애플리케이션 로직으로 처리한 경우임

** 1 시간에 1 byte x 6 차례 수신하는 시나리오로 구성했으며 메시지 QoS는 HTTP와 MQTT를 동일하게 하기 위해 QoS=1로 설정

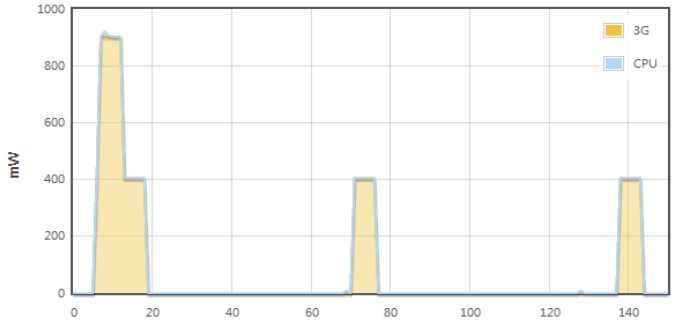
MQTT의 안드로이드 기기 상의 배터리 사용 테스트

- MQTT 클라이언트의 연결 대기, 메시지 송수신 시의 배터리 사용 내역을 테스트*
- 전체 테스트 결과와 상세 설명: http://stephendnicholas.com/archives/219

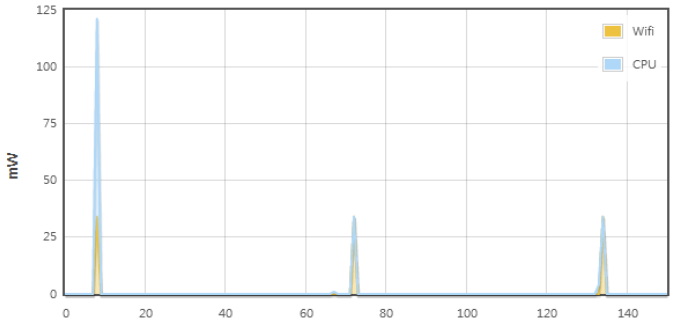
Push를 위한 연결 유지 및 대기

	% Battery / Hour	
Keep Alive (Seconds)	3G	Wifi
60	0.77641278	0.0119021
120	0.38884457	0.0062861
240	0.15568461	0.00283991
480	0.07792208	0.00134018

연결/대기 CPU/3G 전력 사용 추이



연결/대기 CPU/WiFi 전력 사용 추이

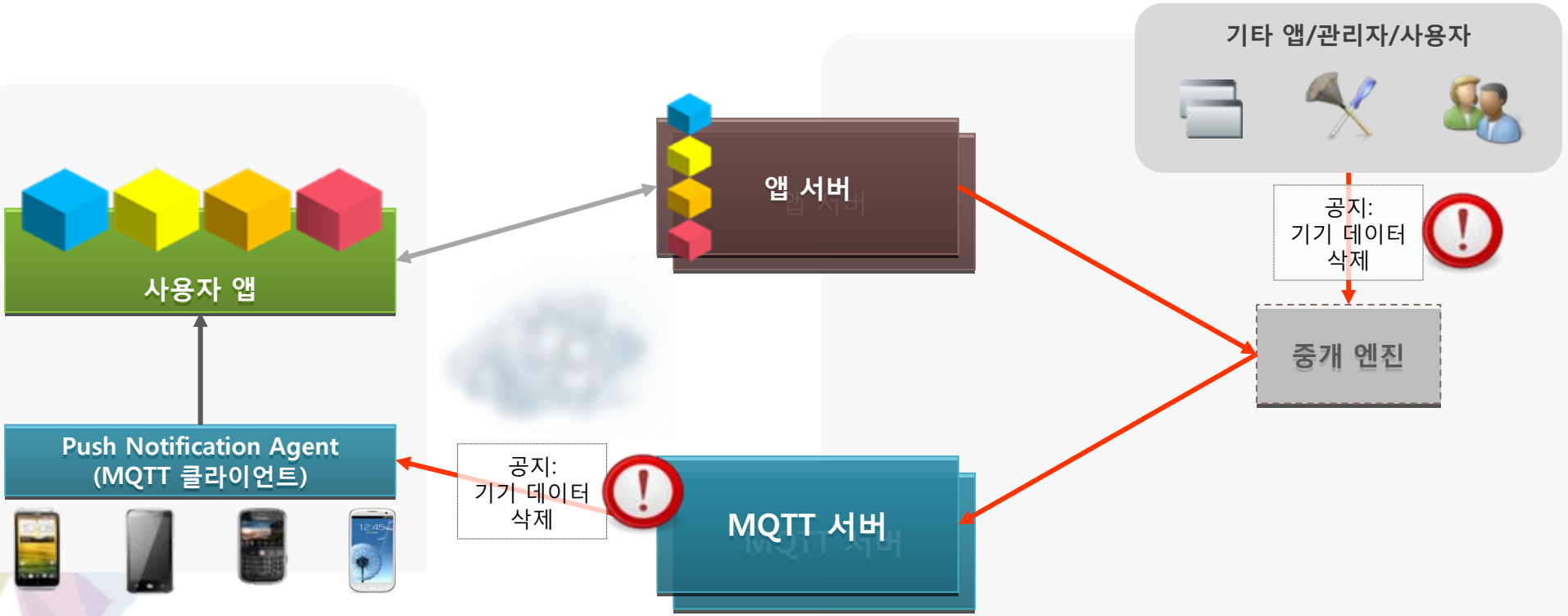


*테스트는 안드로이드 기반 스마트폰 기기인 HTC Desire™에서 수행되었음

3 MQTT 구현과 적용 모델

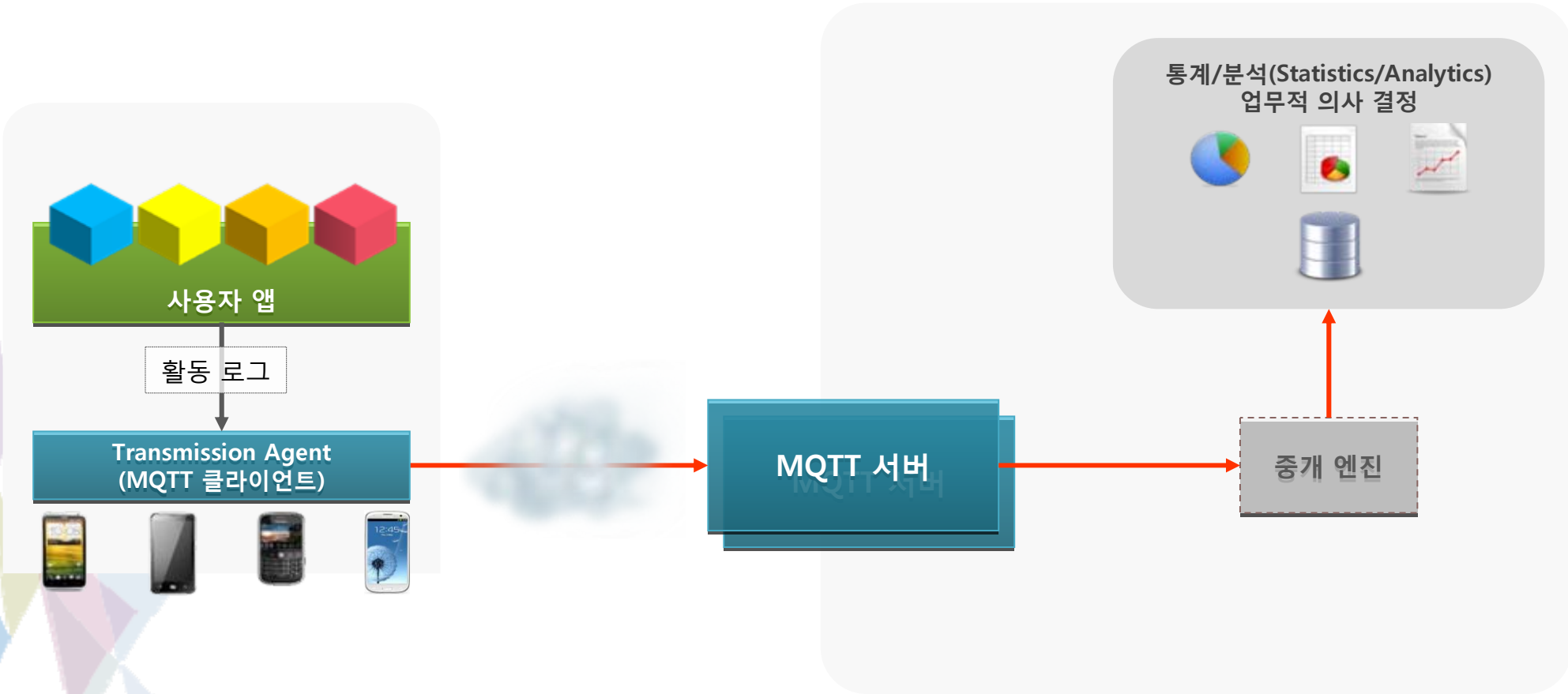
모바일 앱을 위한 Push Notification 인프라 구축

- 자체적인 Push Notification 메시징 인프라 구축을 위한 구현 모델
- Push Notification 메시징의 모든 부분을 제어하려는 On-Premise 요건 충족
- 모바일 기기의 사용시간 보장과 중요 Notification에 전송의 신뢰성을 확보:
  예)MDM 요건(기기의 원격 잠금/해제/위치탐지/사용자 데이터 삭제/기타 제어)



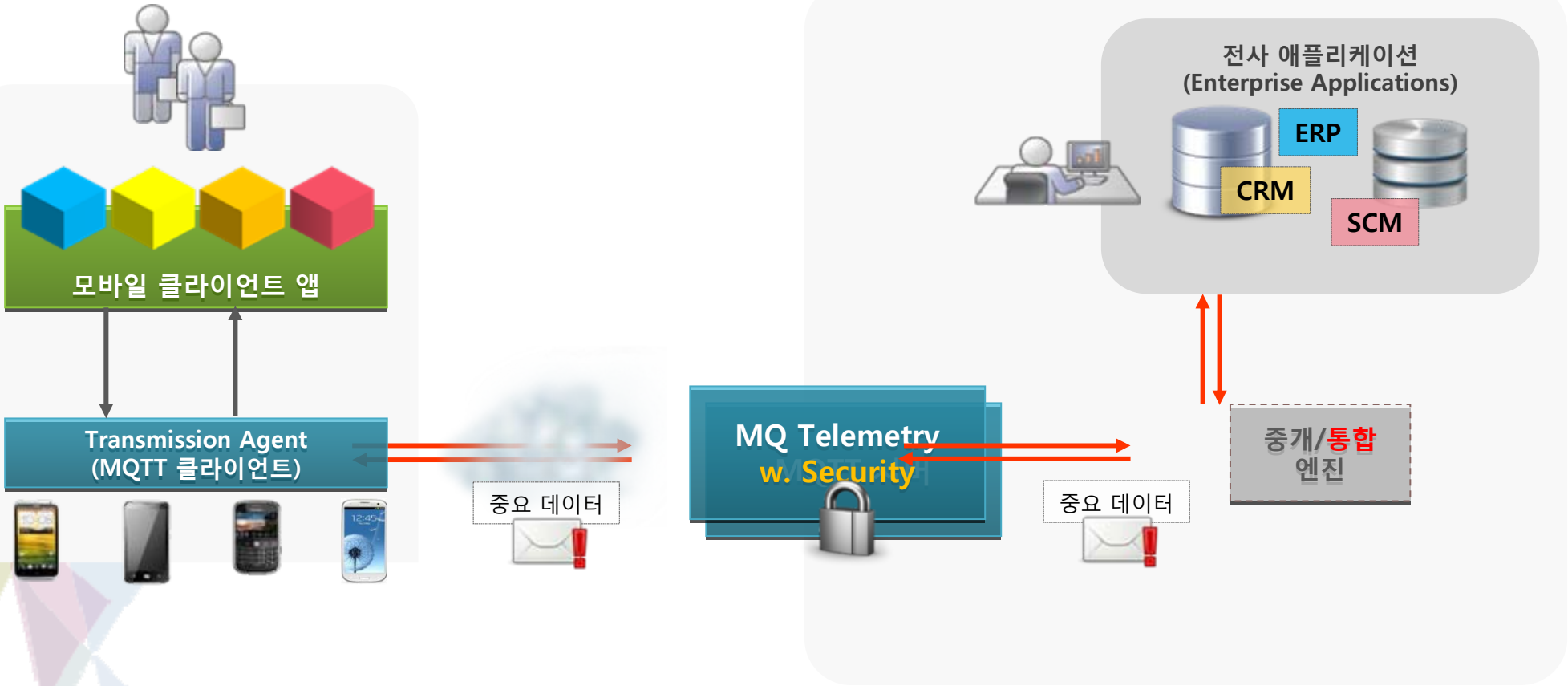
모바일 앱 분석(Analytics)를 위한 데이터 수집

- 기업의 의사 결정 기반 데이터를 수집하기 위한 메시징 백본 구축
- 대량 데이터를 제한된 환경에서 수용하기 위한 경량화된 메시징 프로토콜 마련



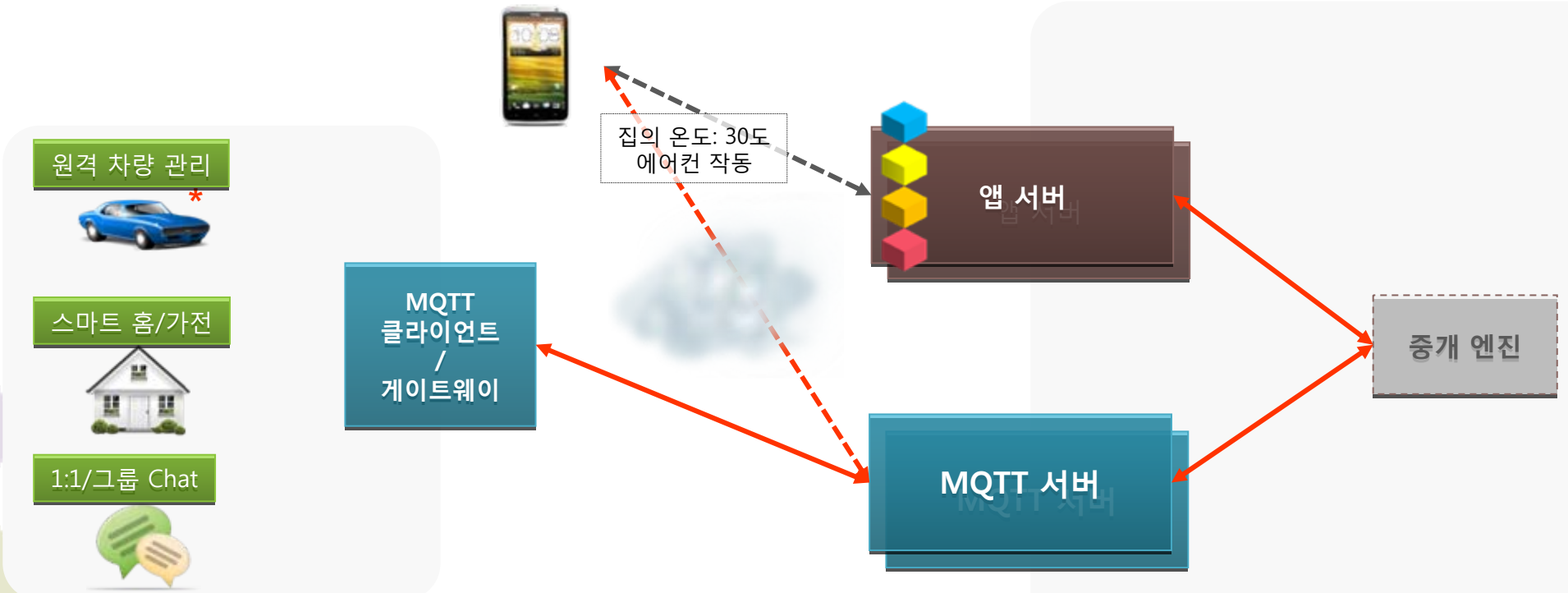
모바일 Workforce를 위한 기간 업무 통합

- 모바일 인력들의 클라이언트 앱을 통한 기간 업무에 대한 신뢰성 있는 접근
- 기간 업무 통합에 적절한 메시징 인프라로서의 MQTT 프로토콜과 WebSphere MQ Telemetry 서버



Machine-to-Machine 커뮤니케이션

- M2M 커뮤니케이션, N 디바이스 등의 이동통신과 스마트 기기를 활용한 다양한 적용 서비스 모델에 있어 적절한 프로토콜과 서버 측 구성을 위해 MQTT를 채택
- 적용 서비스는 무궁무진하지만 항상 프로토콜 효율성이 추구되어야 함



*The icon is used here by courtesy of Cemagraphics (http://cemagraphics.deviantart.com)

IBM Universal Information Framework(UIF) 앱 비디오 데모

http://www.youtube.com/watch?v=URweWTr3-vw

요약

1 모바일의 전개에 따른 Web 환경의 변화

더 다양한 앱과 서비스의 등장으로 HTTP 등의 기존 프로토콜만으로는 커뮤니케이션의 다양한 요구사항들을 수용할 수 없습니다.

2 모바일 메시징, MQTT 프로토콜

제한된 통신 환경을 고려하여 디자인된 MQTT 프로토콜은 모바일 영역에 진화에 따라 최적의 프로토콜로 주목받고 있습니다.

3 MQTT 구현과 적용 모델

MQTT의 Pub/Sub 메시징 패러다임을 기반으로 모바일/Edge 네트워크와 기업의 자산의 연계를 위한 인프라를 구축합니다.

감사합니다