

Wi-Fi 내 Direct 통신 프레임워크



이름	학번	이메일	전화번호
김세호	201011317	seho00@naver.com	010-9568-8280
박대규	201011329	qkreorb0321@naver.com	010-9552-1624
차금옥	201011369	lovecko123@naver.com	010-9437-0713

지도 교수 : 민덕기 교수님 (인)

목차

1. Wi-Fi 내 Direct 통신 프레임워크(DCOFIN)	4p
1.1. 전체 구조도	4p
1.2. 프레임워크 개요	5p
1.2.1. 전체 프레임워크 클래스 다이어그램	7p
1.3. 세부 구조도	8p
1.3.1. Discovery 모듈	8p
1.3.2. Connection Management 모듈	10p
1.3.3. Network Management 모듈	12p
1.3.4. Security 모듈	14p
1.4. 프레임워크 Console Test 화면	
2. DCOFIN Cloud	17p
2.1. 전체 구조도	17p
2.2. DCOFIN Cloud in Web	18p
2.3. DCOFIN Cloud in Android	27p
3. 장점	28p
4. 기대효과	28p
5. 역할 분담	29p
6. 개발 일정	30p

[그림 목차]

[그림1] DCOFIN 전체 구조도

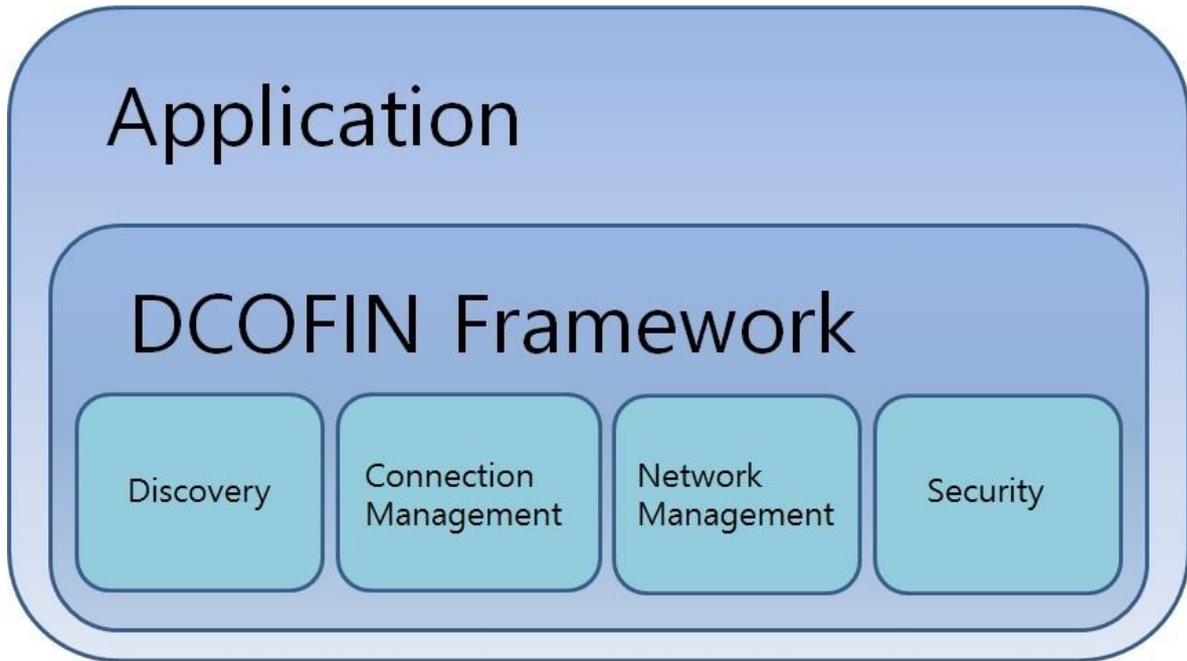
[그림2] DCOFIN Framework 흐름도

[그림3] DCOFIN 개요

- [그림4] Daemon간 통신 구조도
- [그림5] Device간 통신 구조도
- [그림6] Daemon간 메시지 전송
- [그림7] Application Peer Session
- [그림8] DCOFIN Framework 클래스 다이어그램
- [그림9] Discovery Module 구조도
- [그림10] Discovery Module Class Diagram
- [그림11] Connection Management Module 구조도
- [그림12] Connection Management Module Class Diagram
- [그림13] Network Management Module 구조도
- [그림14] Network Management Module Class Diagram
- [그림15] Security Module 구조도
- [그림16] Security Module Class Diagram
- [그림17] Framework Console Test 화면
- [그림26] DCOFIN Cloud Logo
- [그림27] DCOFIN Cloud 전체 구조도
- [그림20] DCOFIN Cloud in Web 동작과정
- [그림21] 역할 분담

1. Wi-Fi 내 Direct 통신 프레임워크(DCOFIN)

1.1. 전체 프레임워크 구조도

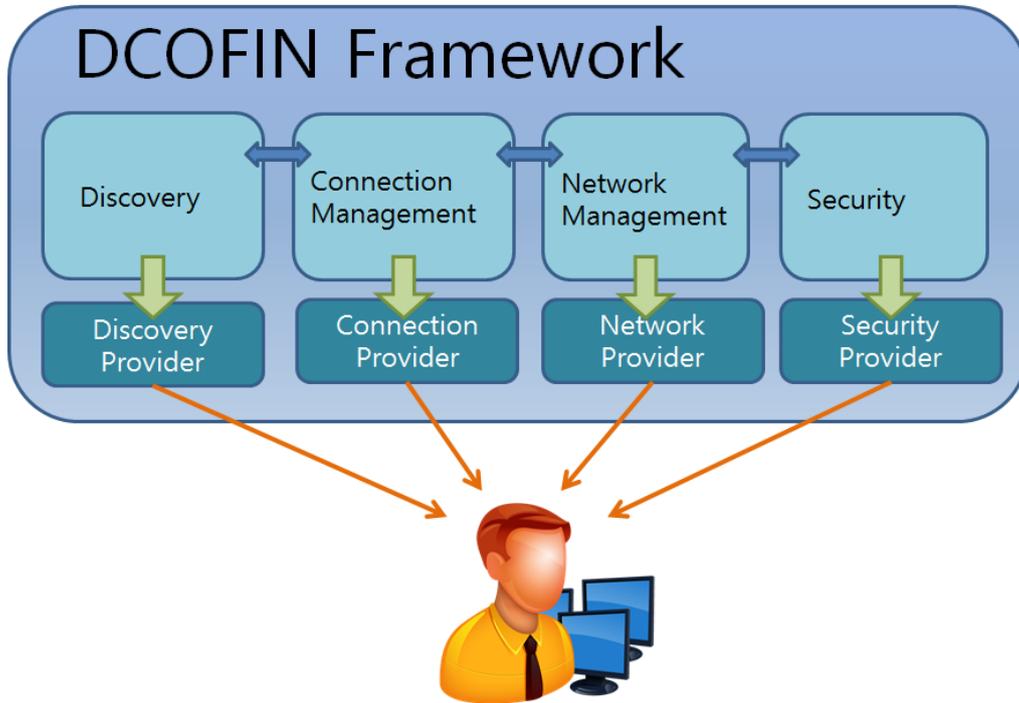


[그림1] DCOFIN 전체 구조도

Direct Communication Framework In wireless Network(DCOFIN)은 크게 네 종류의 모듈들로 구성된다. Discovery, Connection Management, Network Management, Security로 구성되며, 각각의 기능을 담당한다. Discovery module은 Wi-Fi내에 접속했을 시 Daemon의 작동과 정보를 교환하는 기능을 한다. Network Management module은 TCP Socket Communication에 관련된 기능을 담당한다. Connection Manager는 Daemon과 Application과의 연결을 담당한다. Security module은 개체 인증, 암호화 등의 기능을 한다. 각각의 module은 서로 상호작용하며 기능을 수행한다. Programmer는 module내부의 모든 class들을 직접 사용하지 않으며, Provider class를 통하여 보다 더 손쉽게 Framework 기반의 application을 구현할 수 있다.

Discovery Provider, Connection Provider, Network Provider, Security Provider는 각 모듈의 기능을 편리하게 제공하기 위한 Wrapper Class이다. Programmer는 Provider

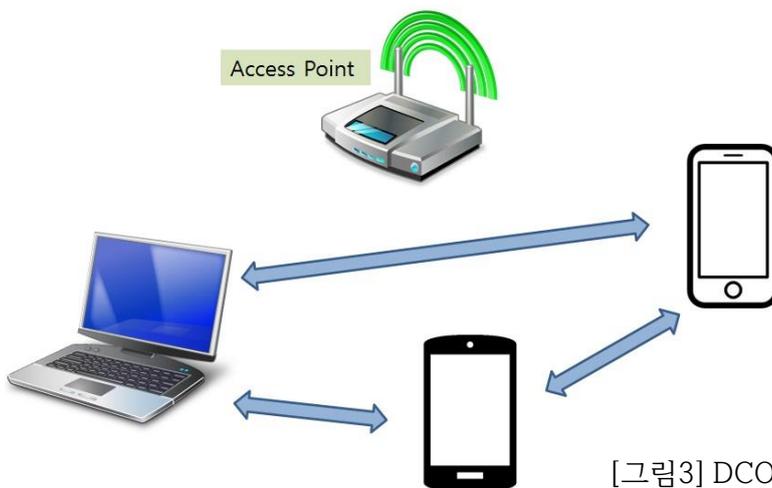
Class를 사용하여 DCOFIN Framework 기반의 Application을 손쉽게 제작할 수 있다.



[그림2] DCOFIN Framework 흐름도

내부의 구조를 모두 알 필요 없이, Provider에서 제공하는 기능만으로 핵심 기능을 사용할 수 있게 한다.

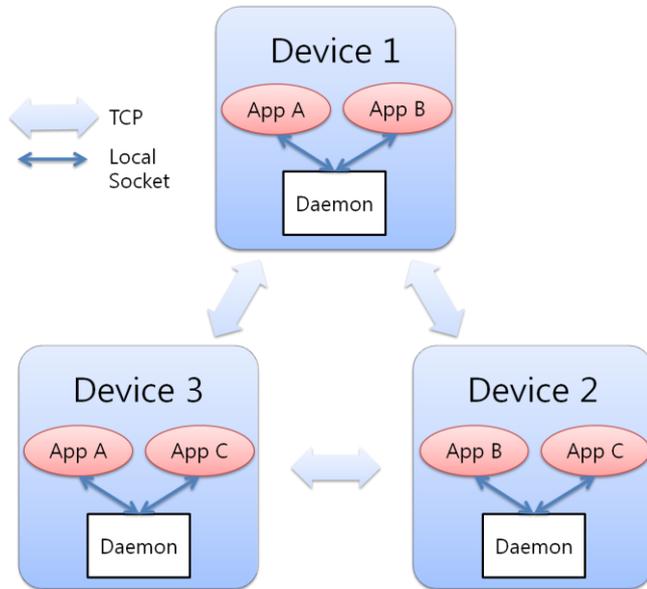
1.2. 프레임워크 개요



[그림3] DCOFIN 개요

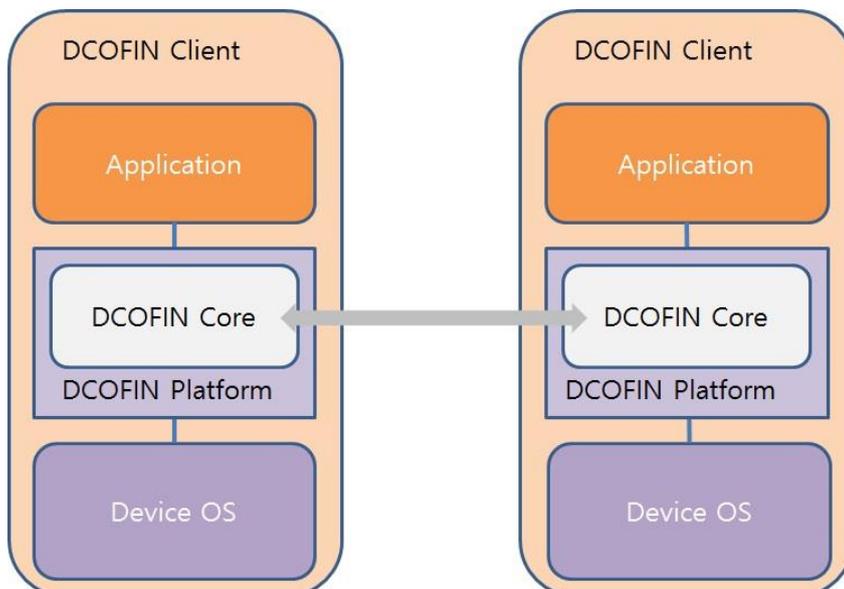
DCOFIN Framework는 동일 Access Point(Wi-Fi 공유기)에 접속한 Device간에 Direct Communication을 가능하게 하는 Framework이다.

각 디바이스, 즉 Wi-Fi 연결이 가능한 Mobile 기기의 응용프로그램들은 해당 플랫폼의 Daemon과 통신을 하게 되고, Daemon은 응용프로그램이 다른 디바이스와 주고 받는 데이터를 Routing하는 역할을 한다. 각 디바이스는 Wi-Fi에 연결된 상태에서 TCP 통신을 하고 응용프로그램과 Daemon은 Local socket 통신을 한다.



[그림4] Daemon간 통신 구조도

Daemon은 디바이스 내 응용프로그램들이 전송하는 데이터 Packet이 거쳐가는 일종의 Gateway 역할을 하는데, 디바이스 간 Direct 통신을 제공하는 주체이다. 각 디바이스의 Daemon들은 같은 네트워크(Wi-Fi)에 접속된 Daemon들 간의 정보를 주기적으로 교환한다. 해당 Daemon이 네트워크 상에 아직 존재하는지 Health checking을 하는 것이다.

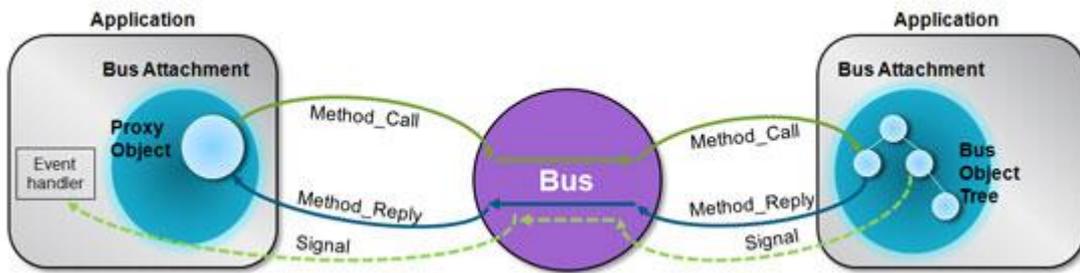


[그림5] Device간 통신 구조도

각 Daemon들은 데이터를 주고 받는 메시지 타입이 4가지로 정의된다.

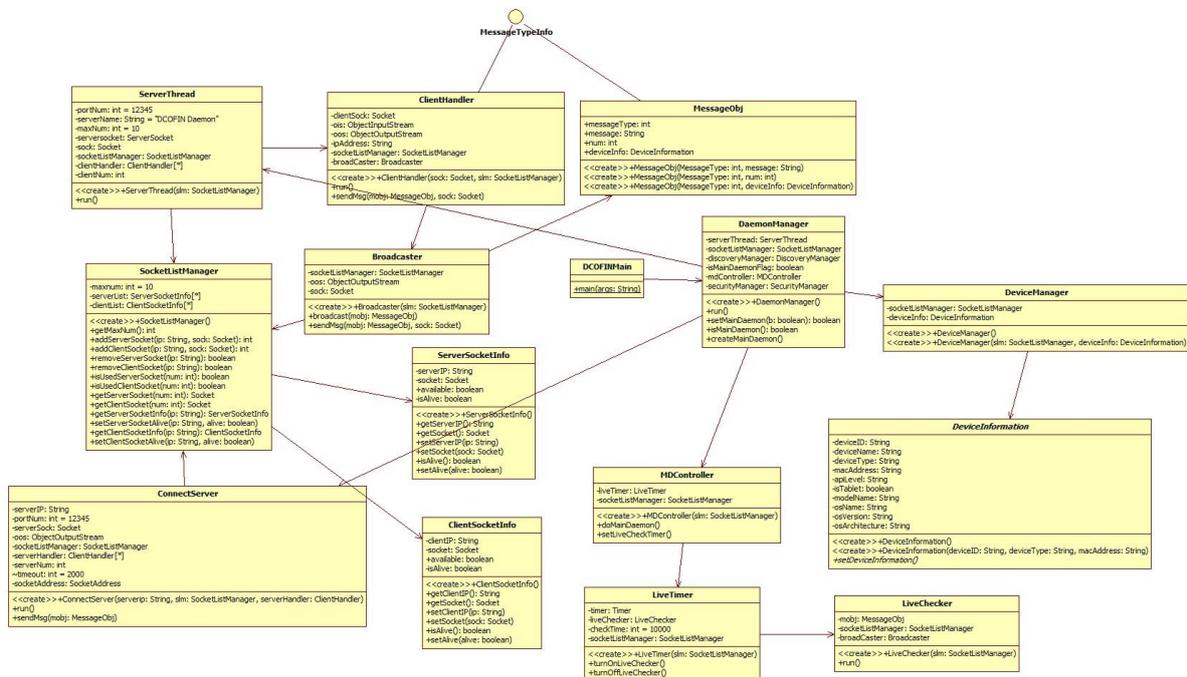
- Signal (Async.) : 시그널은 Broadcast, Multicast, Point-to-Point 전송에 사용한다.
- Method call (Sync.) : RPC(Remote Procedure Call)를 할 때 사용한다.
- Method reply (Sync.) : RPC의 리턴 메시지를 보낼 때 사용한다.
- Error : RPC에 대한 에러 메시지를 리턴할 때 사용한다.

Method call과 reply는 Synchronous 이벤트이므로 response가 올 때까지 Block된다.



[그림6] Daemon간 메시지 전송

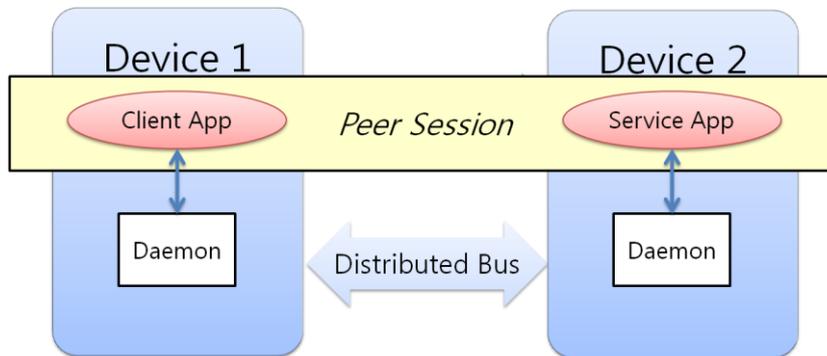
1.2.1. 전체 프레임워크 클래스 다이어그램



[그림7] DCOFIN Framework 클래스 다이어그램

1.3. 프레임워크 세부 구조도

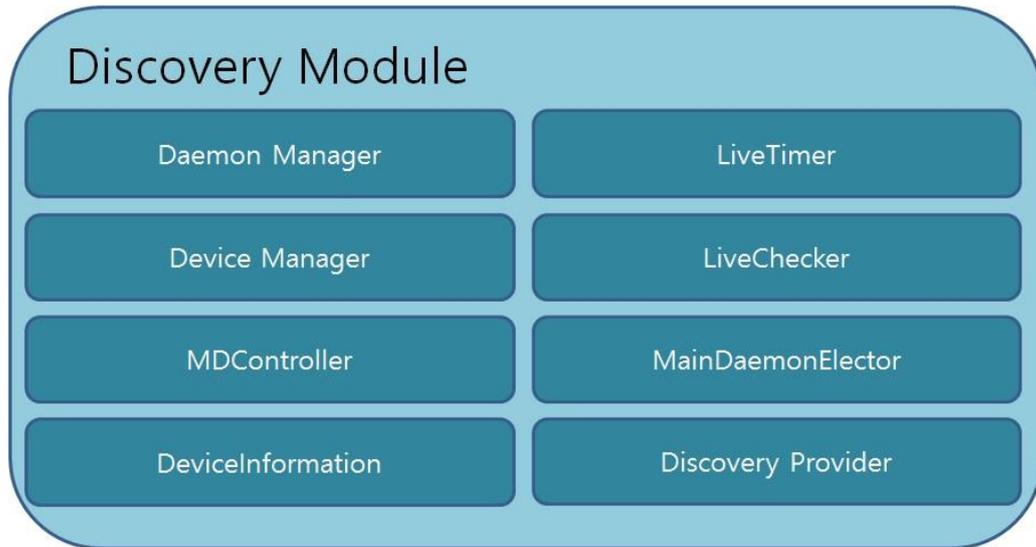
1.3.1. Discovery Module



[그림8] Application Peer Session

Discovery Module은 Wireless Network(Access Point)에 접속했을 경우 Daemon을 실행하고 Daemon의 작업에 관한 정보를 교환하는 역할을 한다. Daemon은 background에서 실행되며, 초기화 단계에서 Wi-Fi내의 Device에 Broadcasting을 통하여 접속을 알린다. Broadcasting의 기능은 Network Management module에서 담당한다. 접속을 알리면, Daemon가운데 대표 역할을 하는 Main Daemon이 새로 접속한 Daemon(Device)과의 인증을 시도한다. 새로운 Daemon의 인증은 Security module에서 담당한다. 만약 Main Daemon이 존재하지 않을 경우(Network에 접속한 첫 번째 Device일 경우) 스스로 Main Daemon이 되며, 그렇지 않을 경우 Main Daemon을 그대로 유지하거나 새롭게 선출한다.

1.3.1.1. Discovery module 구조도



[그림9] Discovery Module 구조도

-DaemonManager class : Daemon의 실행과 Live 상태 체크, Main Daemon역할 수행 여부 등 Daemon의 기능에 관한 전반적인 역할을 담당한다. 위의 기능을 담당하는 Class등의 객체를 생성하여 기능을 수행한다.

-DeviceManager : Main daemon과 나머지 Device들의 정보를 저장하고 관리하는 역할을 수행한다.

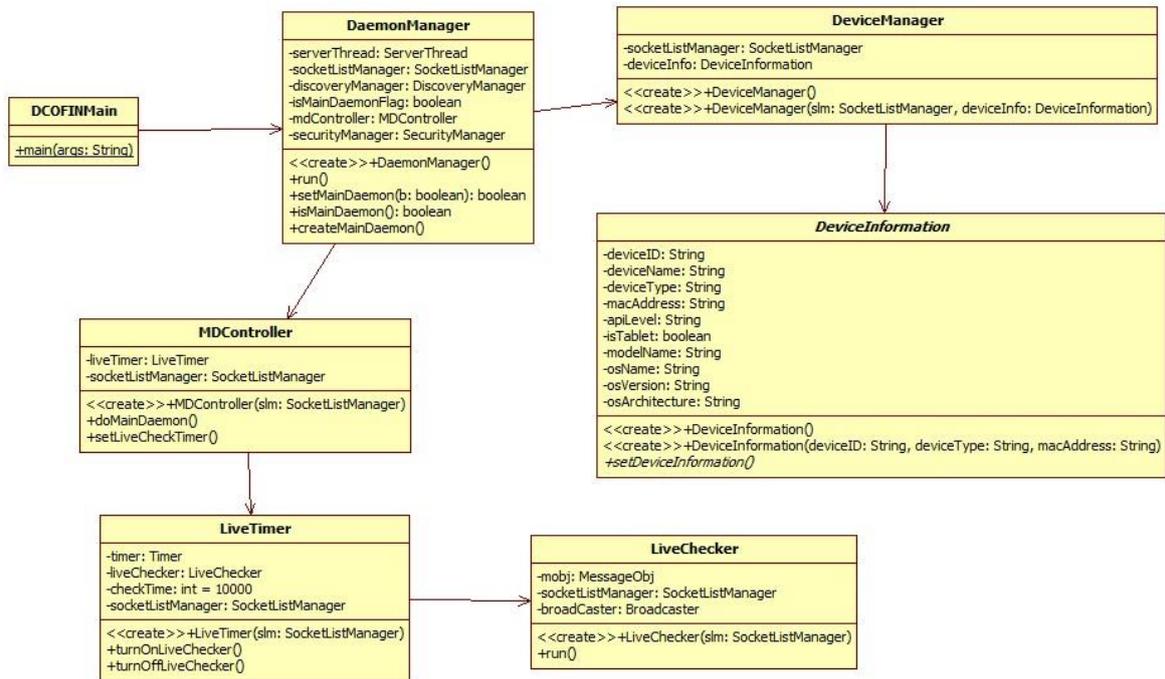
-DeviceInformation : Wi-Fi Network에 있는 Device들의 정보를 저장하는 Class이다. 객체 하나당 하나의 Device정보를 저장한다.

-MDController : Main Daemon으로 선정될 시, Main Daemon의 기능을 수행한다. 새로운 Device가 접속하여 Broadcasting message를 보낼 시 Main Daemon이 답을 하여 통신하는 기능, 새로운 Device를 Authenticate하는 기능, 새로운 Device에게 Network내의 Device정보들을 넘겨주는 기능들을 한다.

-LiveChecker : Network내의 Device들의 접속여부를 확인하기 위하여 주기적으로 Main Daemon에게 live Message를 보내는 역할을 한다.

-LiveTimer : 주기적으로 Live message를 보내기 위하여 해당하는 시간을 측정하는 역할을 한다.

1.3.1.2. Discovery module Class Diagram



[그림10] Discovery Module Class Diagram

1.3.2. Connection Management Module

Connection Management module은 Application과 Daemon간의 연결을 담당한다. 앞서 보았듯이 Daemon간의 Communication은 TCP Socket을 이용하고 Daemon과 Application간의 Communication은 local socket을 이용한다. Connection Management module은 Daemon과 연결된 Application을 namespace로 관리한다. 즉, 각 Daemon들은 응용프로그램에 대한 namespace를 유지하고 있다가, 세션이 연결되면 각 Daemon들은 namespace를 공유한다. 세션이 유지되는 동안 디바이스 간 통신 상태는 살아있음을 의미한다. Multicast의 경우 세션에 연결된 모든 Peer에게 보낼 수 있다.

1.3.2.1. Connection Management Module 구조도



[그림11] Connection Management Module 구조도

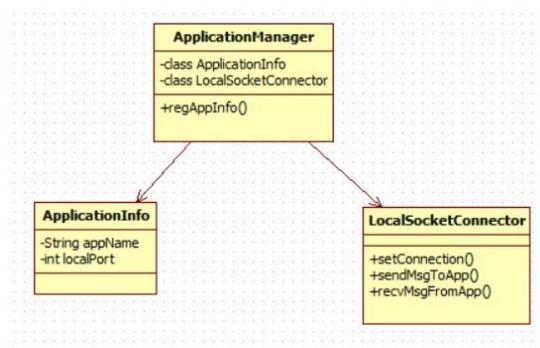
-ApplicationManager : Daemon과 연결되는 Application을 통합적으로 관리한다. Application class와 LocalSocketConnector class의 객체를 생성하여 관리한다.

-ApplicationInfo : 객체를 생성하여 Application 정보를 저장한다. 해당 Application의 namespace와 local에서 사용할 port number를 저장한다. 앞서 보았듯이, device내의 통신은 local socket 통신을 통하여 이루어진다. 따라서 해당 application의 port 번호를 저장한다.

-LocalSocketConnector : Local socket을 통하여 Application과 Daemon간에 연결을 설정하며 Message를 주고받는다.

-ConnectionProvider : 프로그래머에게 손쉽게 사용할 수 있도록 Connection Manager module을 wrapping한다.

1.3.2.2. Connection Management Module Class Diagram

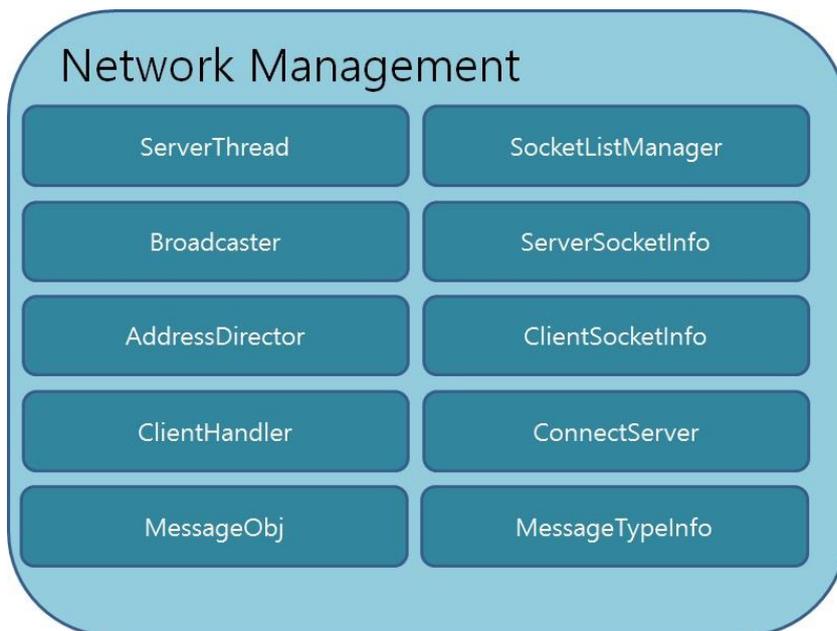


[그림12] Connection Management Module 구조도

1.3.3. Network Management Module

Network Management Module은 TCP Socket Communication과 관련된 전반적인 기능을 담당한다. DCOFIN Framework에서 Daemon들은 TCP Socket을 이용하여 서로 통신한다. 따라서 Wi-Fi내에서 TCP Socket Communication을 담당하는 module이 따로 필요하다. Network Management Module은 Wi-Fi 내에서 Private IP를 이용하여 Device를 구분하고, Message를 Transfer한다. Application Level에서 Private IP를 고려할 필요 없이, DCOFIN Framework를 이용하여 Device의 Private IP에 쉽게 접근이 가능하다. Network Management Module은 Private IP를 통한 Device간의 Communication 기능을 제공하기 위하여 해당 Network의 Gateway Address, Subnet Mask등을 이용하여 Network Address를 계산하는 기능도 제공한다.

1.3.3.1. Network Management Module 구조도

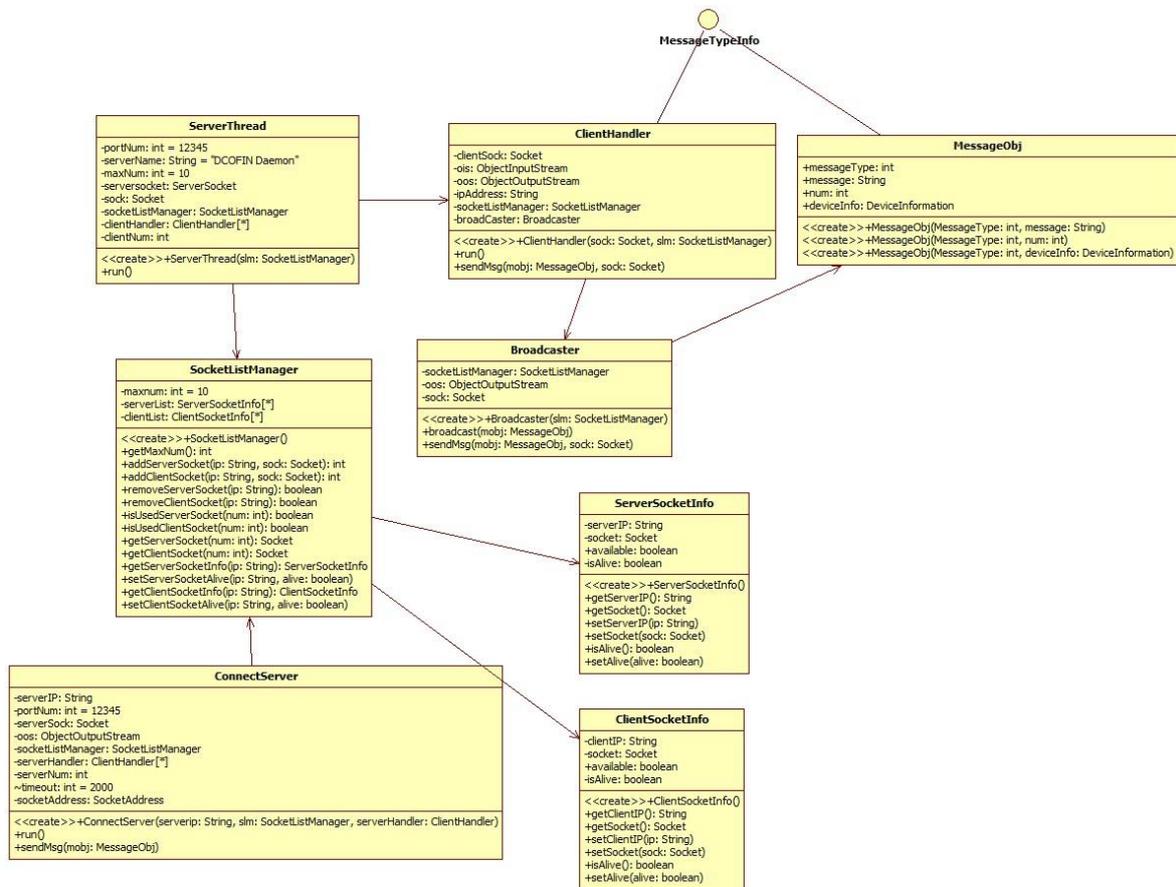


-ConnectServer : Daemon 실행 시 가장 먼저 생성되어 실행되는 객체이다. 하나의 ConnectServer는 스레드로 실행되며 계산된 주소 값에 대하여 TCP connect 요청을 한다.

-AddressDirector : device의 gateway address, subnet mask를 얻어와서 network주소를 계산한다. [그림13] Network Management Module 구조도

- Broadcaster : DCOFIN Framework에서는 device간의 존재유무를 서로 판단하기 위하여 Wi-Fi내의 private address를 통하여 Broadcasting을 한다. Broadcaster class는 AddressManager class로부터 private address정보를 받아와 broadcasting한다.
- ServerThread : 소켓통신을 위하여 서버소켓을 여는 클래스이다. 클라이언트가 접속되면 ClientHandler 객체를 생성하여 해당 소켓으로 오는 메시지를 관리한다.
- ClientHandler : 해당하는 Socket에 대하여 ObjectInputStream으로 메시지를 받아들이는 역할을 한다. 메시지에 따른 Operation을 정의하고 처리한다.
- MessageObj : device간의 메시지를 주고받을 때 형식에 맞추어 보낼 수 있도록 정해진 객체이다. 생성자를 통하여 다양한 종류의 메시지를 정의할 수 있다.
- MessageTypeInfo : Interface로서 다양한 Message Type에 대하여 정의되어있다. Implement하여 사용한다.
- SocketListManager : ServerThread, ConnectServer를 통하여 접속된 Server, Client의 소켓들을 관리하기 위한 클래스이다.
- ServerSocketInfo : ServerSocket의 정보를 저장하고 관리하는 클래스이다.
- ClientSocketInfo : ClientSocket의 정보를 저장하고 관리하는 클래스이다.

1.3.3.2. Network Management Module Class Diagram



[그림 14] Network Management Module Class Diagram

1.3.4. Security Module

Security Module 은 Device 의 Authentication 과 Communication 의 Encryption 에 관련된 전반적인 기능을 담당한다. Wi-Fi Network 내에서의 Authentication 은 Security 측면에서 굉장히 중요한 이슈이다. Daemon 을 통하여 Application 간의 Direct Communication 을 하는 DCOFIN 에서의 Authentication 은 더욱 중요하고 치명적이라 할 수 있다. 따라서 Security module 에서는 기존의 EAP-TLS, EAP-TTLS, EAP-AKA, PEAP 방식 등을 응용하여 안전한 Device Authentication Mechanism 을 개발한다.

Security module에서 제공하는 다른 기능은 Encryption이다. Daemon간에 전송되는 Message들을 암호화함으로써 안전한 정보 전송을 제공한다. 즉, 각 응용프로그램들은 중요한 정보에 대한 암호화 키를 유지하고 있으며 Packet에 대한 필요한 정보를 추출하기 위해 유지하고 있던 키로 인증을 한다. 보안 및 비 보안 Method call 인터페이스를 제공한다.

1.3.4.1. Security Module 구조도



[그림15] Security Module 구조도

-SecurityProvider : Security module의 기능을 Wrapping하여 programmer에게 제공한다. Authenticator, Decryptor, Encryptor의 객체를 생성하여 기능을 사용한다.

-Authenticator : Wi-Fi network에 새로 접속한 Device의 인증을 담당한다. device 각각의 daemon들은 application마다 다른 key값을 가진다. Authenticator class는 이러한 key값을 KeyRepository class로부터 전달받아 KeyComparator 객체의 기능을 통해 인증을 수행한다.

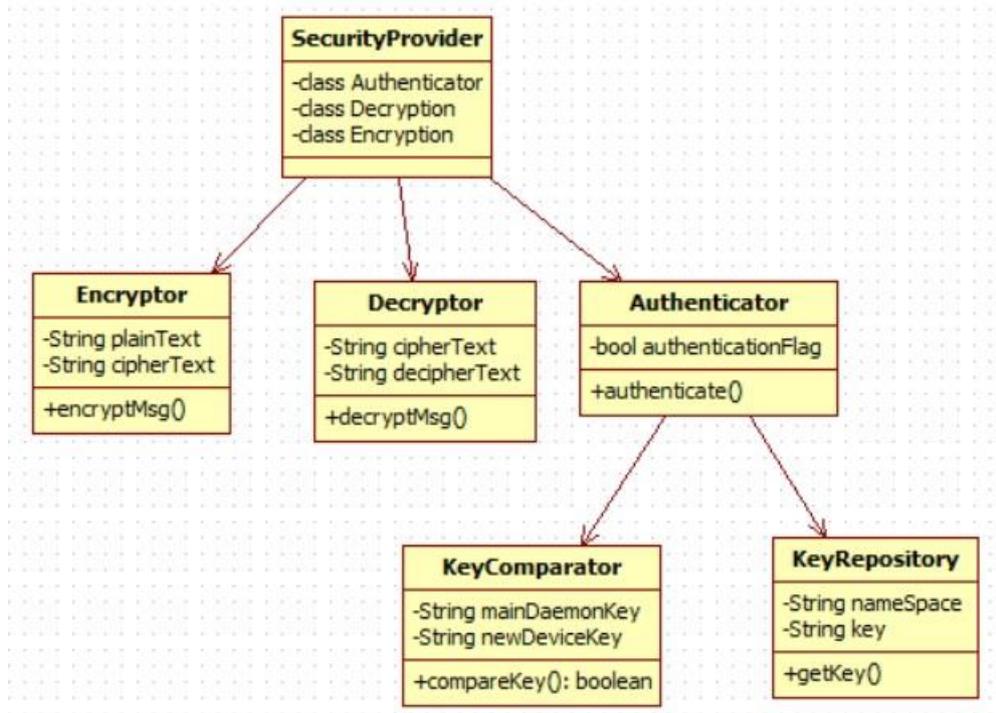
-KeyRepository : Application의 namespace와 key값을 저장하는 class이다. 객체 하나당 하나의 key값 정보를 저장한다.

-KeyComparator : MainDaemon으로부터의 key값과 새로운 Device로부터의 Key값을 비교하여 결과를 return한다.

-Encryptor : plain text를 입력 받아 cipher text로 변환하는 역할을 한다.

-Decryptor : cipher text를 입력 받아 decipher text로 변환한다.

1.3.4.2. Security Module Class Diagram



[그림16] Security Module Class Diagram

1.4. Framework Console Test 화면

```
Problems @ Javadoc Declaration Outline Task List Data Source Explorer Console x
calculating network address :172.16.39.
This is Main Daemon!!
Ready DCOFIN Daemon Server...
Serversocket : ServerSocket[addr=0.0.0.0/0.0.0.0,port=0,localport=12345]
Waiting Client...
start main daemon's function...
```

[그림17] Framework Console Test 화면

2. DCOFIN Cloud

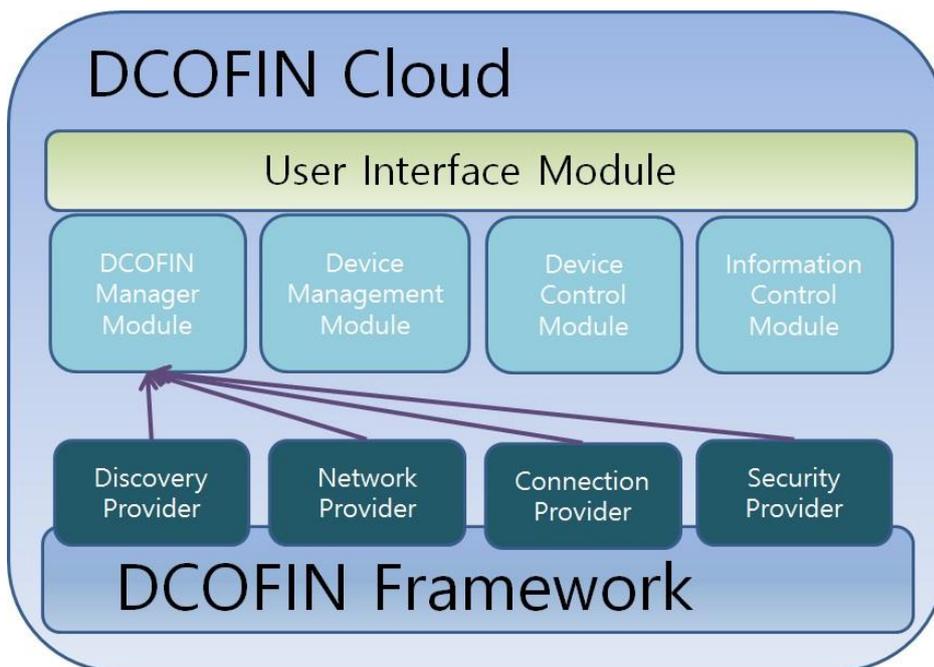


[그림18] DCOFIN Cloud Logo

-DCOFIN Cloud는 DCOFIN Framework를 기반으로 개발하는 통합 Device 관리 system이다.

-기존의 Device 관리 system은 최소 하나 이상의 server가 필요한 반면에, DCOFIN Cloud는 DCOFIN Framework를 사용하여 Network에 접속한 어떤 Device든지 다른 Device를 관리할 수 있도록 한다.

2.1. Application 전체 구조도



[그림19] DCOFIN Cloud 전체 구조도

-DCOFIN Cloud는 다양한 Platform상에서의 User Interface를 제공한다. Windows(java) 기반과 Android기반의 UI를 구현한다.

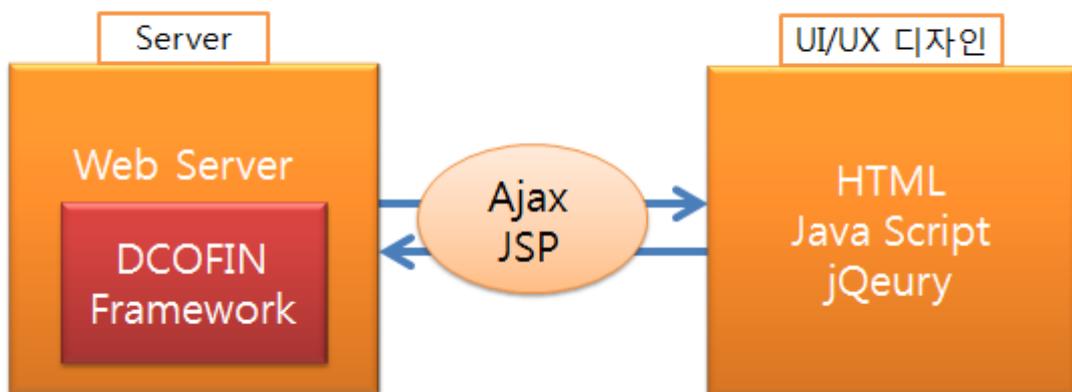
-DCOFIN Framework를 통한 Application 구현으로 Wi-Fi내의 Device의 정보를 direct로 전송 받는다.

-기존 device cloud와의 차이점은 서버가 따로 존재하는 것이 아닌 어떠한 device에서든지 직접 다른 device의 정보를 direct로 전송 받고, direct로 제어할 수 있다는 것이다

2.2. DCOFIN Cloud in Web

-DCOFIN Cloud in Web은 DCOFIN Framework를 기반으로 개발된 웹 어플리케이션이다.

2.2.1. DCOFIN Cloud in Web 동작과정



[그림20] DCOFIN Cloud in Web 동작과정

- DCOFIN Framework를 Server에서 사용
- Ajax 비동기화 방식을 사용하여 효율적인 통신 가능
- server의 데이터를 사용자에게 JSP로 서비스와 함께 제공
- jQuery 라이브러리를 통해 최적화된 사용자 친화적인 UI 제공

2.2.1. DCOFIN Cloud in Web 사용기술

2.2.1.1. Spring Framework



스프링 프레임워크란?

Spring Framework 는 엔터프라이즈급 애플리케이션을 만들기 위한 모든 기능을 종합적으로 제공하는 경량화된 솔루션이다. JEE(Java Enterprise Edition)가 제공하는 다수의 기능을 지원하고 있기 때문에, JEE 를 대체하는 프레임워크로 자리잡고 있고 실제로 Spring 프레임워크는 JEE 가 제공하는 다양한 기능을 제공하는 것 뿐만 아니라, DI(Dependency Injection)나 AOP (Aspect Oriented Programming)와 같은 기능도 지원 한다.

2.2.1.2. AJAX

jQuery.ajax()

AJAX 란 비동기 JavaScript 와 XML 을 말한다. 간단히 말하면, 서버측 Scripts 와 통신하기 위한 XMLHttpRequest 객체를 사용하는 것을 말한다. 서버측으로 다양한 형식(JSON, XML, HTML 및 일반 텍스트 형식등)의 정보를 주고 받을 수 있다. AJAX 의 강력한 특징은 페이지 전체를 리프레쉬 하지 않고서도 수행 되는 "비동기성" 이다. 이러한 비동기성을 통해 사용자의 Event 가 있으면 전체 페이지가 아닌 일부분만을 업데이트 할 수 있게 해준다.

다시 말해 아래와 같이 두 가지로 정리된다.

- 페이지 일부분을 업데이트 하기 위한 정보를 서버에 요청할 수 있다.
 - 서버로부터 받은 데이터로 작업을 한다.
- ① HTML, JavaScript, CSS



- HTML

HTML은 하이퍼텍스트 마크업 언어(HyperText Markup Language, 문화어: 초본문표식달기언어, 하이퍼본문표식달기언어)라는 의미의 웹 페이지를 위한 지배적인 마크업 언어다. HTML은 제목, 단락, 목록 등과 같은 본문을 위한 구조적 의미를 나타내는 것뿐만 아니라 링크, 인용과 그 밖의 항목으로 구조적 문서를 만들 수 있는 방법을 제공한다. 그리고 이미지와 객체를 내장하고 대화형 양식을 생성하는 데 사용될 수 있다. HTML은 웹 페이지 콘텐츠 안의 꺾쇠 괄호에 둘러싸인 "태그"로 되어있는 HTML 요소 형태로 작성한다. HTML은 웹 브라우저와 같은 HTML 처리 장치의 행동에 영향을 주는 자바스크립트와 본문과 그 밖의 항목의 외관과 배치를 정의하는 CSS 같은 스크립트를 포함하거나 불러올 수 있다. HTML과 CSS 표준의 공동 책임자인 W3C는 명확하고 표상적인 마크업을 위하여 CSS의 사용을 권장한다

- JavaScript

자바스크립트(JavaScript)는 객체 기반의 스크립트 프로그래밍 언어이다. 이 언어는 웹브라우저 내에서 주로 사용하며, 다른 응용프로그램의 내장 객체에도 접근할 수 있는 기능을 가지고 있다.

자바스크립트가 썬 마이크로시스템즈의 자바와 구문(syntax)이 유사한 점도 있지만, 이는 사실 두 언어 모두 C 언어의 기본 구문을 바탕으로 했기 때문이고, 자바와 자바스크립트는 직접적인 관련성이 없다. 이름과 구문 외에는 자바보다 셸프와 유사성이 많다.

- CSS(Cascading Style Sheets)

기존의 HTML 은 웹 문서를 다양하게 설계하고 수시로 변경하는데 많은 제약이 따른다.

그래서 이를 보완하기 위해 만들어진 것이 스타일 시트(주 1)이고 스타일 시트의 표준안이 바로 CSS 이다.

HTML 을 이용해서 웹 페이지를 제작할 경우 전반적인 틀에서 세세한 글꼴 하나하나를 일일이 지정해 주어야 하지만, CSS 를 이용해 웹 페이지의 스타일을 미리 저장해 두면 웹 페이지의 한 가지 요소만 변경해도 관련되는 전체 페이지의 내용이 한꺼번에 변경되므로 **작업시간 단축, 일관성 유지**를 할 수 있다.

2.2.1.3. jQuery



Query(제이쿼리)는 브라우저 호환성이 있는 HTML 속 자바스크립트 라이브러리이며 클라이언트 사이드 스크립트 언어를 단순화 할 수 있도록 설계되었다.

jQuery 는 다음과 같은 기능을 갖고 있다:

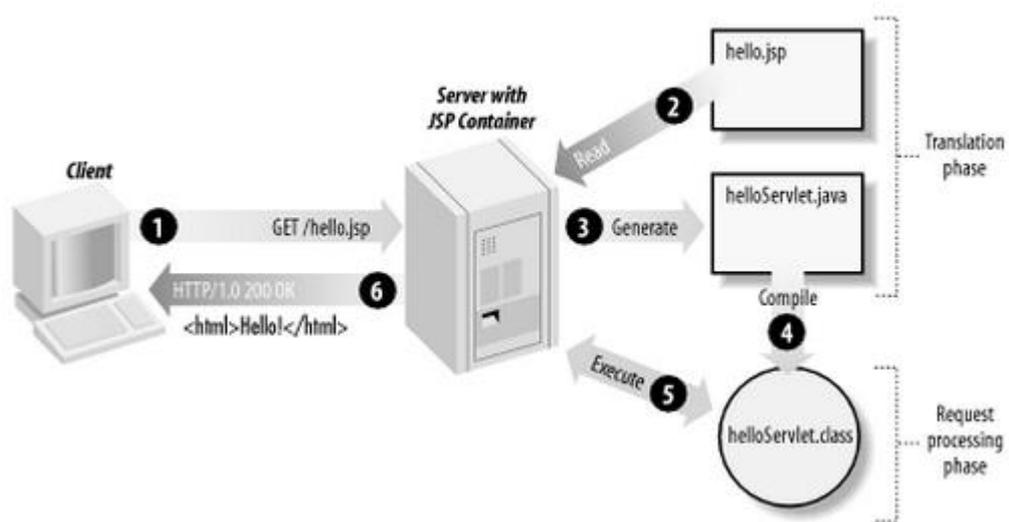
- DOM 요소 선택 기능. 이는 멀티브라우저 오픈 소스 셀렉터 엔진인 *시즐(Sizzle)*을 이용한 것으로, 시즐은 jQuery 프로젝트의 파생 프로젝트이다.^[1]
- DOM 탐색 및 수정 (CSS 1-3 지원)
- CSS 셀렉터에 기반한 DOM 조작. 노드 요소 및 노드 속성(아이디 및 클래스)을 셀렉터 생성을 위한 기준으로 사용.
- 이벤트
- 특수효과 및 애니메이션
- AJAX
- JSON 파싱
- 플러그인을 통한 확장성
- 유틸리티
- 호환성 메소드 (inArray(), each() 함수 등)
- 멀티브라우저 지원 (크로스브라우저와는 다름)

2.2.1.4. JSP



JSP 는 Sun Microsystems 사(oracle 사에 인수됨)의 자바의 서블릿(Servlet) 기술을 확장시킨 기술로써 웹 환경 상에서 100% 순수 자바만으로 서버측 모듈을 개발하기 위한 기술이다. JSP 는 자바 언어 기반으로 작성되었기 때문에 자바언어의 특징을 가지고 있다.

<JSP 의 동작 과정>



- 1 단계 - 클라이언트가 브라우저에서 hello.jsp 를 요청한다.
- 2 단계 - 웹 서버는 hello.jsp 를 찾아서 jsp 컨테이너에게 해당페이지를 처리할 것을 요구한다.
- 3 단계 - jsp 파일을 서블릿으로 변환(파싱)한다.
만약 이전에 이미 요청되었었던 페이지라면 다시 파싱할 필요가 없으므로 5 단계로 넘어간다.
한번만 컴파일하면 되기에 효율적이다.
- 4 단계 - 서블릿 파일을 자바에서 실행 가능한 상태인 클래스 파일로 컴파일한다.
- 5 단계 - 클래스 파일은 메모리에 적재되어 실행된다. 이 후 웹서버로 결과가 넘겨진다.

6 단계 - 웹 서버는 브라우저가 인식할 수 있는 HTML 형태로 결과를 클라이언트에게 응답한다.

2.2.3. DCOFIN Cloud in Web 기능

2.2.3.1. Device

DCOFIN Cloud를 통하여 Wi-Fi Network에 접속한 Device를 통합적으로 관리할 수 있다.

- Network 접속한 Device들을 볼 수 있다.
- 선택한 Device 정보를 불러올 수 있다.
- 선택을 취소하고 다른 Device를 선택할 수 있다.

2.2.3.2. Alarm

Device Information을 관리하여 상황에 맞게 Alarm를 설정할 수 있다.

- Network Disconnected
- Battery
- Datapoint condition

2.2.4. DCOFIN Cloud in Web UI Screen shot

① Main



Device // Operation // Alarm // Contact



② Device

<Add Device>

- Enter a device's MAC address and click Add for each device you want to add.
- If the device doesn't have a MAC address then add it by IMEI number of device ID.
- To find devices on the local network click the Discover button.
- When you are done click OK.

MAC Address

MAC Address	Device ID	Check
		<input type="checkbox"/>

Device List

MAC Address	Device ID	IP Address	Device Type	OS	OS Version	Architecture
18-67-B0-33-C6-48	00000000-00000000-00409DFF-FF4A4D9D	172.16.43.246	Smartphone	Android	4.12	x86

③ Alarm

Alarm Type
Name
Description

Alarm List

Alarm ID	Enabled	Name	Description	Group Path	Status	Check
						<input type="checkbox"/>

2.2.5. DCOFIN Cloud in Web UI 구현 소스

① View 페이지들

-  DcofinAlarm.jsp
-  DcofinContact.jsp
-  DcofinDevice.jsp
-  DcofinMain.jsp
-  DcofinOperation.jsp

② DcofinDevice Device 추가 부분

```
<script type="text/javascript">
<!--jQuery-->

$(function(){

    $("#addDevice").click(function(){
        $.ajax({
            url: "DcofinAddDevice.do",
            type: "post",
            dataType: "json",
            //data: , //요청할 때 보낼 parameter 정보
            success: function(result){
                alert(result);
                $("#headline_contact").append(result);
            },
            error: function(err){
                alert(err + "오류발생");
            }
        });
    });

});

function addDevice(){
    window.open("addDevice.html", "open",
        "height=500px width=700px location=no" +
        "menubar=yes resizable=no status=yes toolbar=yes");
}

</script>
```

③ Dcofin Alarm jQeury 부분

```

<script type="text/javascript">
<!--jQuery-->

$(function(){

    $("#select").click(function(){
        $.ajax({
            url: "deviceSelect.do",
            type: "post",
            dataType: "text",

            success: function(result){
                alert(result);
                $("#headline_contact").append(result);
            },
            error: function(err){
                alert(err + "오류발생");
            }
        });
    });

    $("#create").click(function(){
        $.ajax({
            url: "alarmCreate.do",
            type: "post",
            dataType: "text",
            success: function(result){
                alert(result);
                $("#headline_contact").append(result);
            },
            error: function(err){
                alert(err + "오류발생");
            }
        });
    });
});

```

2.2. DCOFIN Cloud in Android

3. 장점

3.1. 편의성

원격제어, 파일전송 등 제공하는 기능을 통해 사용자의 편의성을 높여준다.

3.2. 범용성

운영체제(OS)나 제조사에 구애 받지 않고 쉽게 사용 할 수 있다.

3.3. 재사용성

프레임워크를 재사용 할 수 있어, 설계 비용이 절감 된다.

3.4. 활용성

Wi-Fi Network를 사용하는 다양한 Application에 활용이 가능하다.

4. 기대 효과

4.1. DCOFIN Framework

- DCOFIN Framework를 사용하여 Application을 구현함으로써 Wi-Fi내의 Device간의 보다 더 효율적인 Communication이 가능하다.
- Application Level에서 Device의 Network Address에 관하여 직접 관여하지 않아도 되므로 프로그래머에게 편의성을 제공한다.
- Wi-Fi 내에서 Socket Communication을 사용하는 Application(예를 들면 원격제어, 파일 전송)의 개발에 유용한 Tool이 될 수 있다.
- windows, android 등 여러 플랫폼에서 사용할 수 있는 framework를 제공하여 개발자와 사용자에게 편의성을 제공한다.

4.2. DCOFIN Cloud

- DCOFIN Cloud를 통하여 Wi-Fi Network에 접속한 Device를 통합적으로 관리하여 사용자에게 손쉽게 IoT Service를 제공한다.
- Device Information을 관리하며 원격제어 등의 서비스를 제공, 특정 Device가 아닌 Wi-Fi Network에 접속한 임의의 Wireless Device를 통하여 Network 내의 Device를 관리할 수 있게 한다.

