Sky Engine

- Cloud 기반의 맞춤형 프로그램 성능 모니터링 서비스 -

1차 중간 보고서

이름	학번	이메일	전화번호
박진성	201011334	bak723@gmail.com	010-5244-7576
이자형	201011354	ljjh6810@gmail.com	010-5107-8838
김은빈	201011322	icecreambku@gmail.com	010-8442-1677

지도 교수 : 하 영 국 교수님 (인)

1. 개요

1.1. 요약

기존 웹서비스를 하고 있는 회사의 70~80% 는 회사의 서비스 모니터링 툴로 제니퍼 소프트 사의 APM솔루션인 'Jennifer'을 사용하고 있다.
이에 APM솔루션에 대한 관심으로 제니퍼 소프트 사의 'Jennifer'을 공부하게 되었고 클라우드 기반의 새로운 웹 서비스에 맞는 새로운 모니터링 툴을 생각해보게 되었다.

Sky Engine은 Runnable service와 Github service에서 동기부여를 받았다.
Runnable service에서는 실제로 코드를 작성하고, 그 코드를 터미널을 통해 실행할 수가 있고, Github service에서는 소스 코드를 호스팅 해주며 그 소스 코드를 찾을 수 있는 검색 서비스를 제공해주고 있다. 본 계획서에서 기획한 Sky Engine은 사용자가 소스 코드를 클라우드 컴퓨팅 서비스에 올려놓고 이때 Sky Engine을 실행하게 되면 서비스 내에서 각 메소드 별로 성능을 측정하여 대시보드로 보여준다. 클라우드 컴퓨팅 서비스에 올려서 사용하는 개인 단위 뿐만 아니라 기업 단위의 enterprise enviroment에서 사용할 수 있는 agent를 직접 다운 받은 후 성능을 측정하여 그 결과를 서버로 보내서 대시보드로 보여준다.

1.2. 새로운 성능 모니터링 서비스의 필요성

지금까지의 성능 모니터링 툴은 단순한 프로그램이지만, 복잡한 설치 과정과 오직 내 컴퓨터 안의 프로그램의 성능 만을 측정할 수 있다는 단점이 있었다.
이러한 단점을 보완하기 위해 우리가 추구하고자 하는 새로운 성능 모니터링 서비스 Sky Engine은 성능을 측정하는 데에 있어 설치 과정이 간단하고 단순히 프로그램을 서버에 올리기만 하면 성능을 측정할 수 있으며 실시간으로 측정 결과를 대시보드에 보여준다. 또한, Enterprise 환경 등에서의 상황을 고려하여 성능 측정 도구를 제공할 수 있으며, 이것을 다운로드하여 직접 측정할 수 있고 실시간으로 모니터링 대시보드를 이용할 수 있는 API도 제공한다.

2. 주요 기능

2.1. 사용자

2.1.1. 프로젝트 추가

사용자가 자신이 프로파일링할 프로젝트를 추가한다.

2.1.2. 프로젝트 관리

사용자가 자신의 프로젝트의 설정을 변경하거나 삭제한다.

2.1.3. 프로젝트의 실행환경 설정

실행환경(자바 버전, 서버 스펙 등…)을 설정한다.

2.1.4. 프로젝트의 프로파일 이력조회

실시간으로 프로파일링된 결과를 포함하여 지난 과거의 프로파일을 조회한다.

2.1.5. 실시간 프로파일링 대시보드

프로젝트를 실행한 후 대시보드를 통해 실시간으로 프로파일링 되는 과정을 볼 수 있다.

2.1.6. OpenAPI를 이용하여 다른 빌드서버에서 실행

프로파일링을 원격에서 실행하여 Jenkins와 같은 곳에서 플러그인 형태로도 사용할 수 있다.

2.2. 관리자

2.2.1. 사용자 관리

사용자들을 관리 할 수 있지만 대체적으로 풀어주고 악성 사용자만 골라낼 수 있다.

2.2.2. 워커 관리

Sky Worker(이하 워커)는 곧 프로파일링을 수행할 서버이기 때문에 제대로 수시로 살아 있는지 확인하고, 그것을 웹에서 볼 수 있다.

2.2.3. 워커 등록 및 추가

워커를 Sky Server(이하 스카이서버)에 추가하여 새로운 워커일 경우 스카이서버에 등록하여 곧바로 사용할 수 있다.

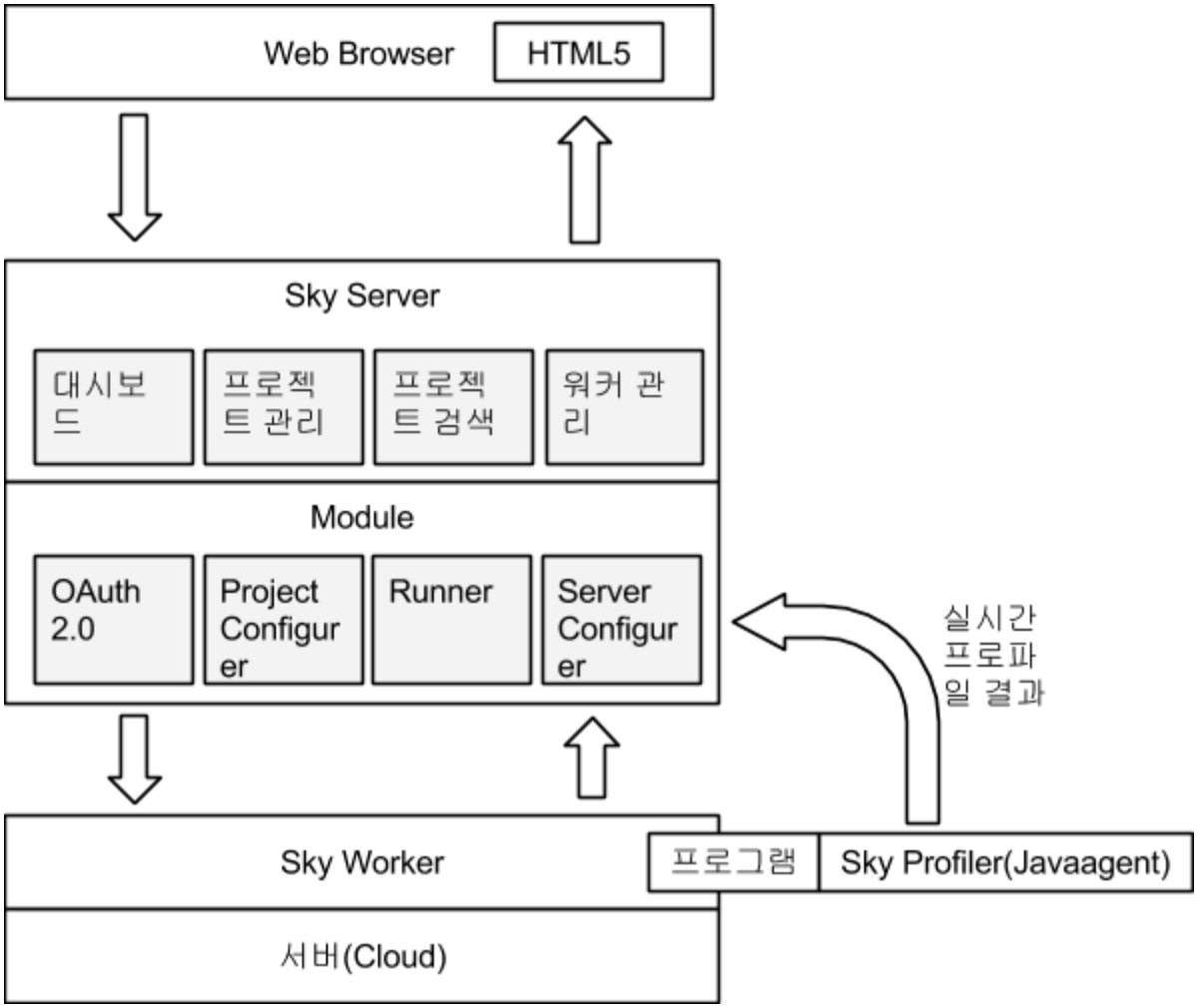
2.2.4. Sky Server 웹 모니터링

JMX를 이용하여 스카이서버를 실시간으로 모니터링 할 수 있고, 또한 각 워커들이 어떤 프로젝트를 프로파일링 하고 있는지도 확인 할 수 있다.

2.2.5. 프로파일러 관리

프로파일러의 버그가 있는지, 또는 프로파일링 도중 오류들을 실시간 피드백을 통해 관리 할 수 있다. 이 경우 관리자에게 이메일이 날아가게 된다.

3. 아키텍처



유저가 웹 브라우저를 통해 프로젝트를 생성하고 실행파일을 등록하거나 빌드시스템을 등록한다. 그 이후 등록한 프로젝트의 프로파일을 요청할 경우 본 서비스에서 Cloud에 있는 서버를 뒤져 워커셋을 형성, 워커를 나눠주어 실행환경을 구축한다.

이렇게 구축된 실행환경에 빌드된 실행파일(.jar)를 넣고 프로파일러에서 실시간으로 각 메소드의 프로파일 결과물을 던져주게 된다. 이를 받아 대시보드에 실시간으로 뿌려준다.

또한 최종 사용자가 웹브라우저를 통해 웹서버에 접속하여 사용하는 것 외에 내부적으로 통신이 필요한 구간에는 전부 thrift를 사용하고 있다. thrift 사용의

이점으로는 TCP/IP 소켓을 사용하지 않아도 되며 RPC의 특성으로 프로토콜이 객체 형태로 결정된다는 것이다.

3.1. Sky Server

3.1.1. 대시보드

Sky Engine에서 돌아간 자바프로그램의 성능 측정한 데이터를 실시간으로 가져와서 대시보드에 차례대로 보여준다. 대시보드에서는 가장 오래걸리는 메소드와 그의 상세한 코드 또는 트리들도 매칭해서 보여주도록 한다.
또한 검색을 통해 샘플 코드를 선택할 때 해당 코드가 수행되어 모니터링된 성능 그래프도 보여주는 역할을 한다.
HTML5와 CSS3를 적극적으로 활용하여 실시간적인 데이터를 받아 그래프를 그려낸다.
Web Browser의 HTML5/CSS3를 기반으로 클라이언트 사이드에서는 대시보드를 만들고 서버에서는 실시간으로 해당 대시보드에 웹소켓을 통해 데이터를 밀어넣어 유저에게 실시간으로 프로파일 과정을 보여주게 된다.

3.1.2. 프로젝트 관리

유저는 로그인을 한 후, 자기 자신의 프로젝트를 만들어 각 설정을 하게 된다.(소스코드 등록, 빌드 등록, 실행 환경 설정 등…)
Project를 설정할 수 있게 해주는 모듈이다. 각 프로젝트의 정보가 디비에 담겨져 있고 해당 디비와 연결해주는 역할이다.

3.1.3. 프로젝트 검색

유저는 자신의 프로젝트 말고 공개되어있는 다른 사람의 프로젝트를 검색할 수 있을 뿐만아니라, Github와 같은 곳에 이 프로젝트는 프로파일링 되었다는 검증마크 또한 제공해 줄 수 있다.

3.1.4. Runner

실질적으로 실행파일을 등록시켜놓고 남는 서버를 찾아 해당 서버에 실행파일을 배포시키고 실행하는 역할이다.

3.1.5. Sky Worker 관리 기능

클라우드 컴퓨팅을 기반으로 하고 있는 다중 서버들을 관리, 추가, 설정하는 역할이다. Administrator권한이 있어야하며, 서버들을 WorkerQueue로 관리해서 Runner에서 쓸 수 있도록 만들어야 한다.

3.2. Sky Worker

Sky Server(이하 스카이서버)에서 사용자로부터 특정 프로젝트의 프로파일링 요청이 들어오면 그것을 먼저 Work의 객체로 만든 후 디비에 있는 WorkQueue에 저장해놓는다. 실질적으로 Sky Worker는 서버의 WorkQueue로부터 Work를 끌어와서 정보를 읽어 들여 해당 프로젝트를 프로파일링을 시작하게 된다.

Worker는 유저가 실행되는 즉시 서버에 등록 및 추가되며 곧바로 사용할 수 있다.

Sky Server는 Worker들을 10분마다 살아있는지 확인하면서 관리하게 되며, 살아 있지 않고 죽어 있는 Worker가 있다면 즉시 Worker Set에서 해당 Worker를 삭제하게 된다.

Sky Server에서 프로파일링을 위해 비어있는 클라우드 기반 서버를 찾을 때, 사용 되는 것이 이 Sky Worker이며 Sky Server는 직접적으로 클라우드 기반 서버에서 프로그램을 실행 할 수 없으며 이 Sky Worker를 통해서만 실행을 시켜야 한다. 이로써 Sky Worker를 분산컴퓨팅으로 관리 할 수 있다.

3.3. Sky Profiler

Javaagent형태로 만들어져 있어 JVM에 붙어 Java Instrument기술을 이용하여 프로파일링을 수행하며, Sky Server의 Dashboard에 실시간으로 결과를 전달하게 된다.

4. 관련 연구

4.1. 관련 서비스

4.1.1. Github

Github(이하 깃허브)는 소스 코드를 호스팅 해주는 서비스이다. 깃이라는 VCS툴을 이용하여 소스 코드를 깃허브에 올리게 되고 각 브랜치와 참여하는 사람들의 커밋 정보도 볼 수 있다. 또한, 한 프로젝트에 대해 버그나 이슈를 올리거나 추가 기능을 붙여 요청을 보낼 수도 있다.

4.1.2. Runnable

Runnable은 웹에서 소스 코드를 작성하게 되고 그 작성한 것을 터미널을 통해 실행해 볼 수 있는 샘플 리스트 웹 서비스이다. 이 서비스는 내가 원하는 샘플을 검색을 통해 찾을 수 있으며 또한 실행해 볼 수 있는 것이 장점이다.

4.1.3. Jennifer

제니퍼 소프트 (이하 제니퍼)에서는 APM 솔루션으로써 제니퍼라는 제품을 내놓고 있다. 이 제니퍼는 위에서 말했다시피 메인 프레임 세대의 하나의 물리적 서버 내에서 돌아가는 WAS를 타겟으로 모니터링을 수행하는 솔루션이다. 이 제니퍼는 이에 맞는 Dashboard, 기술 등을 갖고 있다.

4.1.4. Apache JMeter

Apache JMeter(이하 JMeter) 분석을 위한 테스팅이나 Web Service에 집중되어진 다양한 서비스의 성능을 측정하는 툴이다.

JMeter는 JDBC Database Connection, FTP, LDAP, Web Service, JMS, HTTP, TCP Connection 등을 위한 유닛테스트로서 이용 될 수 있다.

JMeter는 여러개의 파라미터, 주장, 쓰레드 수행, 파라미터 조작과 다양한 레포트들을 지원한다.

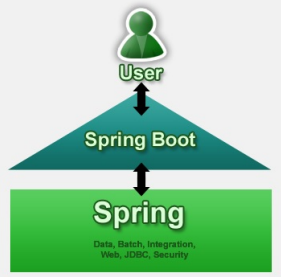
4.1.5. Travis-ci

Travis-ci(이하 트래비스)는 지속적 통합 및 배포 시스템을 호스팅하는 서비스이다. 보통 오픈소스 리포지토리에는 무료로 제공하며 비공개 리포지토리에는 유료로 제공하게 된다. 또한, 많은 오픈소스 프로젝트들이 사용하고 있다.니

4.2. 사용 기술

4.2.1. build

4.2.1.1. Spring Boot



기존에 spring framework 로 프로젝트를 셋팅하려면 프로젝트 환경설정 xml이 java 파일을 작성하고, WAS를 설치하고 WEB.XML을 설정하고, 컨트롤러-서비스-DAO-DOMAIN을 만들어서 기본 셋팅을 했다. 하지만 이 spring boot 프로젝트는 위 과정들을 다 생략해 준다. spring boot 커맨드 라인용 툴을 다운 받으면 매우 간편하게 일련의 작업들을 자동화할 수 있다.

4.2.1.2. Maven

메이븐은 사실 의존 관계 관리 도구라기보다는 프로젝트 관리 도구이며, build lifecycle에 따라 프로젝트 표준을 제공하고 의존 관계를 관리하고 플러그인이 제공하는 부가 기능을 사용할 수 있게 하는 도구이다. 메이븐은 이클립스 연동도

가능하며 이클립스 프로젝트를 만들 수 있으 이클립스에서 메이븐 플러그을 설치하여 메이븐 프로젝를 만들 수도 있다. 메이븐의 기능으로는 빌드, 문서화, 리포팅, 의존관계 관리, 소스코드 관리, 릴리즈, 배포 등이 있다.

4.2.2. Web

4.2.2.1. Spring Framework

spring framework는 java platform을 위한 오픈 소스 어플리케이션 프레임워크로서 간단히 스프링이라고도 한다. 동적인 웹 사이트를 개발하기 위한 여러가지 서비스를 제공하고 있다. 스프링은 경량 컨테이로서 자바객체를 직접 관리한다. 또한 POJO(Plain Old Java Object)방식의 프레임워크이며 제어반전(IoC : Inversion of Control) 방식과 의존성 주입(DI : Dependency Injection)관점 지향 프로그래밍(AOP : Aspect-Oriented Programming)을 지원한다. 스프링은 영속성과 관련된 다양한 서비스도 지원하고 있고, 확장성 또한 높다.

4.2.2.2. Spring Social

Spring Social은 페이스북, 트위터, 링크드 인과 같은 소프트웨어로서의 서비스(SaaS) API 제공 업체와 응용 프로그램을 연결할 수있는 스프링 프레임 워크의 확장이다. Spring Social은 몇 가지 특징을 가지고 있다. 첫째로 호스팅 공급자 계정에 로컬 사용자 계정을 연결하는 과정의 간소화 서비스를 제공한다. 두번째로 Java/Spring Web 응용 프로그램, 서비스 공급자 및 사용자의 권한 부여 흐름을 처리하는 연결 컨트롤러이다. 세번째로 서비스 공급자를 통해 로그인하여 응용 프로그램에 인증 할 수 있게 로그인을 컨트롤러이다.

4.2.3. DB

4.2.3.1. Spring Data JPA

Spring 데이터의 JPA 모듈은 repository beans을 정의 할 수 있는 사용자 정의 네임 스페이스가 포함되어 있다. 또한 JPA에 대한 특별 특정 기능과 요소의 속성이 포함되어 있다. 일반적으로 JPA 저장소는 저장소 요소를 사용하여 설정할 수도 있다. Spring data JPA는 쉽게 JPA 기반 저장소를 구현할 수 있다. 이 모듈은 JPA 기반 데이터 액세스 계층에 대한 향상된 지원을 다룬다. 그것은 쉽게 데이터 액세스 기술을 사용하는 스프링 구동 애플리케이션을 구축 할 수 있다.

4.2.3.2. Hibernate ORM

Hibernate에서 ORM 이란 자바 애플리케이션 내의 객체들을 RDB에 있는 테이블로의 자동화된 영속화이다.
쉽게 설명 하자면 우리가 예전부터 사용해오던 ResultSet 객체를 그 사용 목적에 맞는 형태의 객체로 변환 하는 작업이라고 말할 수 있다. 결국 ORM이란 기존의 행하였던 작업들에 대하여 소프트웨어 공학적으로 정의 해 놓은 것이라고 생각 할 수도 있다. 즉, ORM 이란 가상의 Object DB를 효과적으로 만들어 RDB를 OOP 언어의 개념으로 연계하는 프로그램 기술이다. Hibernate는 이런 ORM을 해주는 툴의 하나라고 할수 있다. 이런 ORM을 사용하는 가장 좋은 방법은 당연히 OODBMS를 사용하는 것이며 아직까지 완벽하게 구현된 상용 DBMS는 나와있지는 않다.

4.2.3.3. QueryDsl

QueryDsl는 자바 JPA, MongoDB의 및 SQL 등 다양한 백엔드의 형식이 안전한 SQL과 유사한 쿼리의 건설을 가능하게 하는 프레임 워크이다. 대신 인라인 문자열로 쿼리를 작성하거나 XML 파일로 이를 구체화의 그들은 유창 API를 통해 구성된다.

4.2.3.4. MariaDB

MariaDB는 오픈 소스의 관계형 데이터베이스 관리 시스템(RDBMS)이다. MySQL과 동일한 소스 코드를 기반으로 하며, GPL v2라이선스를 따른다. Oracle 소유의 현재 불확실한 MySQL의 라이선스 상태에 반발하여 만들어졌으며, 배포자는 몬티 프로그램 AB(Monty Program AB)와 저작권을 공유해야 한다. 이것은 MySQL과 높은 호환성을 유지하기 위함이며, MySQL APIs와 명령에 정확히 매칭하여, 라이브러리 바이너리와 상응함을 제공하여 교체 가능성을

높이고자 함이다. MariaDB에는 새로운 저장 엔진인 아리아(Aria)뿐만 아니라 InnoDB를 교체할 수 있는 XtraDB 저장 엔진을 포함하고 있다. 이것은 트랜잭션과 비트랜잭션엔젠 그리고 미래에 나올 MySQL판에 대응할 것이다.

4.2.4. OpenStack

OpenStack은 IaaS 형태의 클라우드 컴퓨팅 오픈소스 프로젝트이다. 2012년 창설된 비영리 단체인 OpenStack Foundation에서 유지, 보수하고 있으며 아파치 라이선스하에 배포된다. AMD, intel, 캐노니컬, 수세 리눅스, Red Hat, 시스코 시스템즈, Dell, HP, IBM, NEC, VMware, Yahoo!등의 150개 이상의 회사가 이 프로젝트에 참가하고 있으며, 주로 리눅스 기반으로 운용과 개발이 이루어진다.

4.2.4.1. nova

IaaS(infrastructure as a Service) 구축에 필요한 컴퓨트 인스턴스들을 제어 및 관리하기 위한 서비스로 All share 기능을 담당하는 프로젝트이다.

4.2.4.2. swift

블록 스토리지(Block Storage)가 아닌 오브젝트 스토리지(Object Storage) 환경을 구축 및 관리하기 위한 서비스로 별도의 독립적인 구축이 가능한 스토리지 서비스(Storage Service) 프로젝트이다.

4.2.5. Java Instrument

instrument란 자바소스를 변경하여 컴파일하는 것이 아니라 바이트 코드(byte code)로 되어 있는 자바 클래스를 분석하여 변경하는 것을 말한다. 유명 자바 instrument 라이브러리로는 BCEL, ASM등이 있다.

4.2.5.1. BCEL

The Byte Code Engineering Library(BCEL)은 자바언어에서 생성하는 클래스파일(바이트코드)를 분석하고 변경, 재조립하는 간편한 인터페이스를 제공하기 위해 만들어진 아파치 재단에서 후원하는 프로젝트 중 하나이다.

4.2.5.2. ASM

Byte Code Instrumentation을 지원하는 라이브러리로 Object Web에서 제공한다.

4.2.5.3. CGLIB

CGLIB는 기존의 자바 클래스 파일로부터 자바의 소스코드를 동적으로 생성하는 라이브러리이다.

4.2.6. HTML5/CSS3

'14년에 차세대 웹 표준으로 확정 예정이며, 기존 텍스트와 하이퍼링크만 표시하던 HTML이 멀티미디어 등 다양한 애플리케이션까지 표현, 제공하도록 진화한 "웹 프로그래밍 언어"이다. 예로 오디오, 비디오, 그래픽 처리, 위치정보 제공 등 다양한 기능을 제공함으로써, 웹 자체에서 처리할 수 있는 기능이 대폭 향상 되었다.

4.2.6.1. 2D Canvas

브라우저 위의 그림판으로 HTML5의 많은 새로운 기능 중 가장 자주 언급되어 왔던 것이다. Canvas 위에 선, 도형, 텍스트, 이미지와 같은 그래픽을 표현할 수 있고 색깔, 그림자, 패턴과 같은 여러효과를 적용할 수 있다.

4.2.6.2. JQuery / Ajax Framework

JQuery: JQuery는 브라우저 호환성이 있는 HTML 속 자바스크립트 라이브러리이며 클라이언트 사이드 스크립트 언어를 단순화 할 수 있도록 설계되었다.
Ajax Framework: web 어플리케이션 개발에서의 Ajax framework는 클라이언트 측에서 동적 웹 페이지를 구축하기위한 기술의 컬렉션을 활용하는 Framework이다.

4.2.6.3. Backbone.js

Backbone.js는 RESTful JSON interface를 가진 Java Script library로 model-view-presenter(MVP) application 설계 패러다임을 기반으로 한다.

4.2.6.4. Twitter Bootstrap

Twitter Bootstrap은 website와 web application을 만들기 위한 tool들의 무료 컬렉션이다. 이것은 HTML 및 입력 체계, 양식, 버튼, 탐색 및 기타 interface 구성 요소뿐만 아니라 선택 사양 Java Script 확장에 대한 CSS기반의 디자인 템플릿이 포함되어 있다.

4.2.7. Network

4.2.7.1. Apache Thrift

Thrift는 페이스북이 개발한 규모 가변적인(scalable) 이종 언어 서비스 개발을 위한 소프트웨어 프레임워크이다. 이 프레임워크는 2007년에 페이스북에 의해 개발되기 시작하여 2008년 이후 아피치 재단이 유지 보수를 맡고 있다. Thrift는 C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, Javascript, Node.js, Smalltalk, OCalm 언어 등 다양한 환경을 지원하고 있다. 우리가 사용하는 Perl언어를 비롯해 학계나 산업계가 주목하는 언어들의 대부분을 지원하고 있는 셈이다. 이렇게 Thrift를 이용하면 다양한 환경의 소프트웨어를 쉽게 결합할 수 있다.

5. 프로그램

5.1. 시나리오

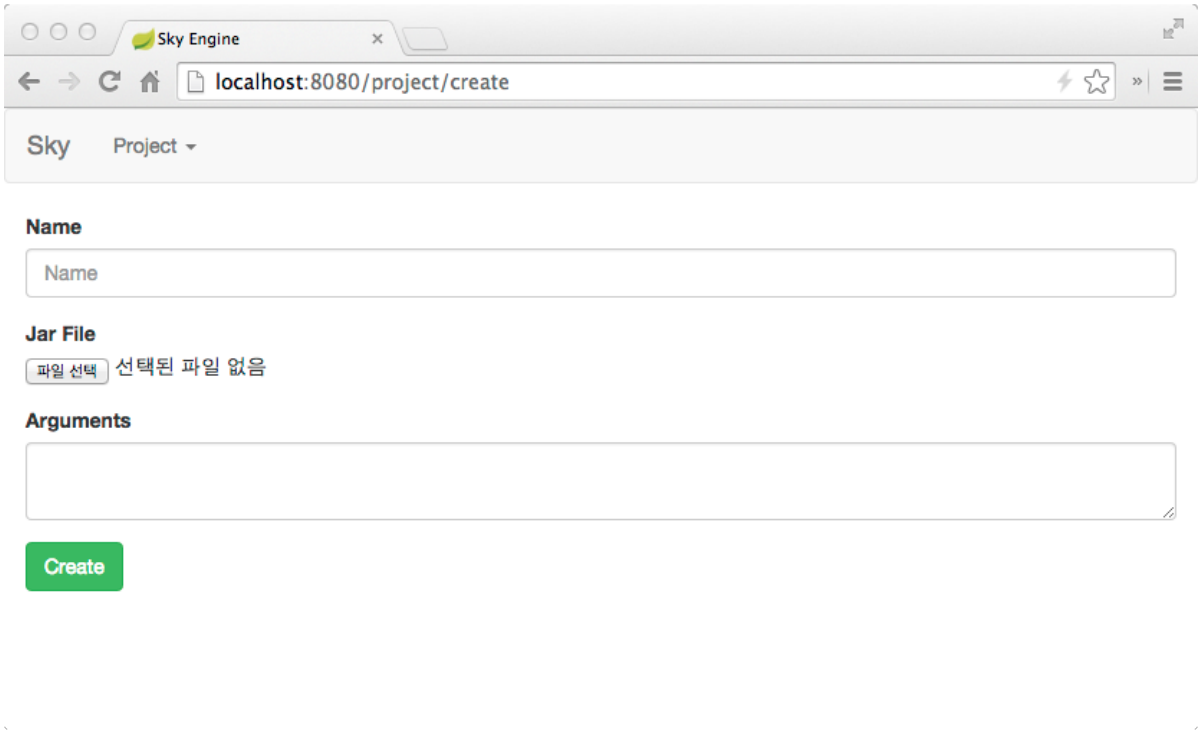
- **프로젝트 만들기**
  사용자가 직접 프로파일링을 하고 싶은 자신의 프로젝트를 만든다.
- **사용자의 모든 프로젝트 보기**
  사용자가 자신에게 속해 있는 모든 프로젝트를 나열한다.
- **프로젝트 검색**
  다른 사용자의 흥미 있는 프로젝트를 검색한다.
- **프로젝트의 프로파일링 요청하기**
  프로젝트의 설정을 끝낸 후 설정이 정상적이라면 프로파일링을 요창한다.
- **모든 과거 프로파일 보기**
  특정 프로젝트에 대해 모든 프로파일을 나열한다.
- **실시간으로 프로파일 결과 보기**
  특정 프로젝트의 프로파일링을 요청한 후 실시간으로 프로파일 결과를 확인한다.
- **프로젝트의 설정을 수정하기**
  프로젝트의 설정을 계속 수정할 수 있다.
- **새로운 서버의 워커 등록하기**
  새로운 클라우드 서버의 워커를 등록 할 수 있다.
- **워커의 상태를 확인하기**
  각 워커들을 나열하고 하나하나의 상태를 확인 할 수 있다.
- **JMX를 통한 서버 모니터링**
  JMX를 이용하여 웹서버를 지속적으로 모니터링 할 수 있다.
- **워커 등록 해제하기**
  서버에 문제가 있거나 접속이 불가능한 워커를 등록 해제할 수 있다.
- **사용자 모니터링 및 관리하기**
  특정 사용자가 의도치 않은 목적을 가졌는지 지속적으로 모니터링하여 삭제하거나 이메일을 이용하여 관리 할 수 있다.
- **워커를 지속적으로 살아있는지 검사하기**
  시스템 내부적으로 워커들이 지속적으로 살아있는지 검사하여 워크를 할당한다.
- **프로파일러 다운로드**
  프로파일러를 직접 다운로드 하기도 하며 시스템 내부적으로 워커에서 다운로드 할 수 있다.

5.2. 진행상황

5.2.1. 사용자

5.2.1.1. 프로젝트 추가

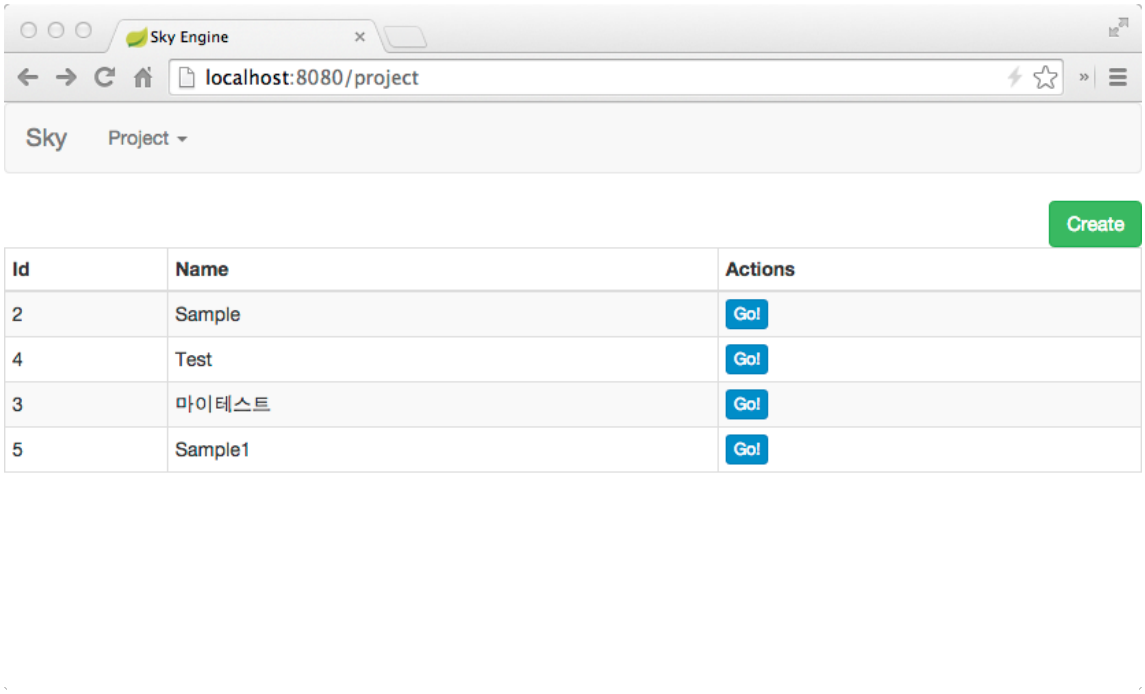
아래와 같이 프로젝트를 추가할 수 있으며, Name, Jar File, Arguments등을 적고 프로젝트를 생성할 수 있다. Jar File이 곧 프로파일러가 프로파일링할 대상이 되며, 아직 구현사항으로는 모든 메소드를 프로파일 하게 되며 특정 메소드를 지정하는 기능은
미구현사항으로 남아있다.



5.2.1.2. 프로젝트 관리

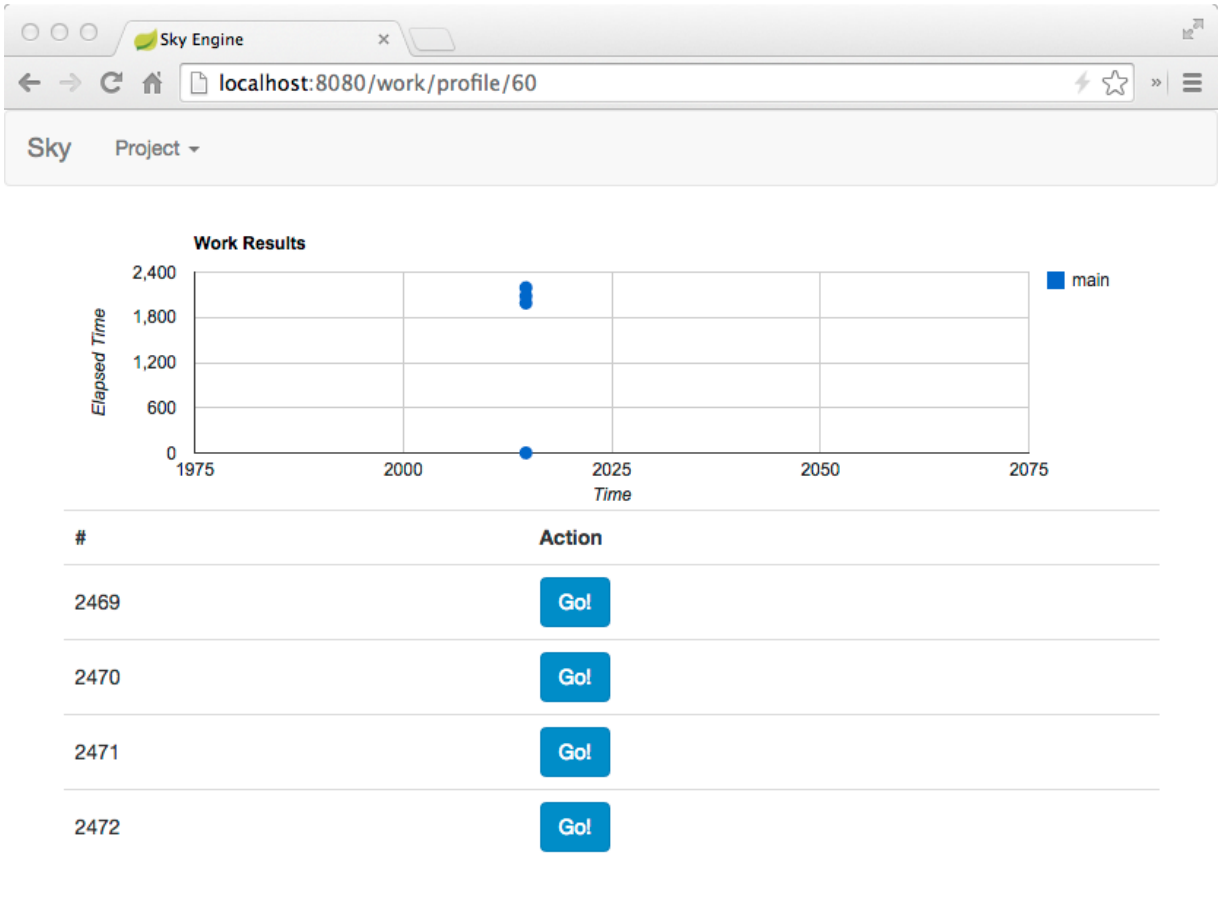
실제 내가 만든 프로젝트들을 볼 수 있다. 프로젝트는 사용자가 만들며 사용자에 종속 된다. RestFul로 웹이 설계되어 있어 후에 다른 사용자가 프로젝트를 검색하여 볼 수도 있다.

현재는 프로젝트의 설정을 바꾸거나 할 수는 없고 Go! 버튼을 통해 프로파일링을 실행 하게 된다.



5.2.1.3. 실시간 프로파일링 대시보드

프로젝트에서 Go! 버튼을 눌렀을 경우 실시간으로 프로파일링이 실행된다. 해당 화면은 프로파일이 완료되고 나서의 화면이다. 아직 구현중에 있으며 아래에 각 메소드들이 나타나고, 해당 메소드마다 다시 버튼을 누른다면 Stack Trace가 나온다.



5.2.1.4. OpenAPI를 이용하여 다른 빌드서버에서 실행

실제로 서버 내부적으로 Rest-ful로 설계되어 있으며 웹브라우저에서 디비에 접속 할 때 웹 서비스의 RestFul API를

이용하여 접속 되고 있다. 똑같이 외부에서 Rest-ful API를 OpenAPI로 활용 할 수 있다.
빌드 서버에서는 해당 소스코드의 빌드가 끝난 후 결과로 나온 Artifact를 OpenAPI를 통해 Project에 업데이트시키고 프로파일링을 요청하여 빌드서버에서 플러그인 형태로 만들 수 있다.

5.2.2. 관리자

5.2.2.1. 워커 관리

서버에 한번 등록된 워커는 죽더라도 지워지지 않는다. 하지만 문제가 있는 워커나 사용되지 않는 워커는 관리자에 의해 삭제 할 수 있다. 또한 해킹의 목적으로 워커를 등록하거나 추가할 경우도 있기 때문에 해당 IP를 밴 시킬 수 있다.

5.2.2.2. 워커 등록 및 추가

Sky Worker프로그램을 이용하여 워커를 등록 및 추가할 수 있다. Sky Worker를 실행할 경우 자동으로 웹서버에 인식되며 새로운 Worker일 경우 디비에 등록이 되어 그 이후로 계속 사용된다.

5.3. 미구현사항

5.3.1. 프로젝트 관리

프로젝트를 삭제하거나 기타 설정들을 바꾸어 프로젝트를 끊임없이 관리할 수 있어야 한다.

5.3.2. 프로젝트의 실행환경 설정

프로젝트가 실행 될 수 있는 환경(JVM 버전, OS, Memory, CPU 등)을 선택할 수 있는 기능이 있어야 한다.

5.3.3. 프로젝트의 프로파일 이력조회

프로젝트의 지난 프로파일들의 이력을 조회할 수 있어야 한다.

5.3.4. 실시간 프로파일링 대시보드

각 메소드들의 Call 관계와 Stack Trace들을 더 구현해야 한다. 이를 통해 사용자가 실제 프로파일링 결과를 보고 성능 튜닝을 할 수 있게 된다.

5.3.5. 사용자 관리

관리자가 의도치 않은 사용자(해킹을 목적으로 한..)를 삭제하거나 밴시킬 수 있어야 한다.

5.3.6. 워커 관리

실제 워커들을 분산컴퓨팅 형태로 관리 할 수 있어야 한다. 워커들을 디비에서 직접 지우는 것이 아닌 웹페이지를 통해 워커의 상태를 체크하고 직접 지울 수 있어야한다.

5.3.7. Sky Server 웹 모니터링

JMX를 적용하여 Sky Server를 모니터링 해야 한다.

5.3.8. 프로파일러 관리

프로파일러를 버전별 또는 JVM별로 갖고 관리해야 하며, 프로파일러 내에서의 버그나 피드백에 대한 프로토콜을 정의내리고 구현해야 한다.

6. 개발환경

6.1. 개발 언어

6.1.1. HTML5/CSS3

대시보드를 통해 결과를 실시간으로 웹브라우저에 띄워주기 위한 GUI

6.1.2. Javascript

HTML5/CSS3를 이용하고 Web Browser에서 동적인 화면 구성

6.1.3. Java

Sky Server를 자바를 기반으로 웹서버를 만들고, 프로파일러의 자바에이전트를 통한 구성

6.1.4. MySQL

Sky Server의 데이터 저장 및 관리

6.1.5. XML

프로젝트의 설정 및 서버의 설정, 실행환경의 설정 등을 XML로 관리하여 검증이 쉬워지고 편리함

6.2. 시스템 환경

- **Language: Java 1.7**
- **IDE: Intellij IDEA**
- **OS: Ubuntu Linux Server**

7. 향후 결과 및 기대효과

현재 클라우드 형태로 CTIP 환경을 제공하는 기본적인 항목이 코드 커버리지인 coveralls.io와 유닛 테스트 빌드시스템인 Travis-ci 인데 이는 수행하기가 쉽고 프로젝트를 설치하기 쉽기 때문이다. 우리는 Sky Engine을 이용하여 성능 모니터링의 클라우드 컴퓨팅 서비스를 제공하는 것이 가장 최종 목표이다. 이를 통해 Github나

Bitbucket과 같은 소셜 오픈소스 커뮤니티에서 이 프로젝트는 성능 모니터링이 우수하다는 워터마크와 같은 검증 서비스로써의 역할을 수행할 수 있다.

7.1. Github와 같은 소셜 오픈소스 개발 환경에서 동적 프로파일링 마크 제공

Github와 같이 오픈소스를 상대로 하는 소스코드 호스팅 환경에서 Travis-ci, coveralls, codeclimate, gemnasium 등과 같이 해당 프로젝트가 동적 프로파일링을 통해 어느정도의 속도가 나왔는지 마크로 표시해주고 해당 웹페이지로 링크를 걸어준다. 이를 통해, Github에서 오픈소스를 개발하고 사람들에게 홍보할 때 객관적인 자료를 제시해줌과 동시에 홍보에도 영향을 미치게 된다.

7.2. Github

우리 프로젝트도 Github를 기반으로 개발하여 외부의 다양한 개발자들이 사용하면서 어려웠던 점이나 개선하고 싶은 사항을 적극적으로 반영하여 같이 나아가는 방식으로 하겠다.

7.3. 설치형과 호스팅형으로 나눠서 오픈소스의 특성으로 커스터마이징 가능

Sky Engine을 제공할 때 Atlassian의 제품 패키징을 모티브로 하여 설치 형과 호스팅 형으로 나눠서 제공한다. 처음은 먼저 호스팅 형으로 제공하고 나중에 설치 형으로 나아가는 방향으로 설정한다. 설치형으로 하게 될 경우 자신의 입맛에 맞게 커스터마이징이 가능하도록 한다.

8. 일정

일련 번호	주요 내용	추진일정						기간 (주)	상태
		5	6	7	8	9	10		
1	자료조사 및 분석	◀▬▬▶						4	(완성)
2	1차 계획서 작성	◀▬▬▶						4	(완성)
3	개발환경 구성		◀▬▬▬▬▬▶					8	(완성)
4	시스템 설계		◀▬▬▬▬▬▶					8	(완성)
5	중간보고서 작성1			◀▬▬▶				4	(완성)
6	시스템 구현			◀▬▬▬▬▬▶				8	(진행)
7	중간보고서 작성2				◀▬▬▶			4	(계획)
8	테스트 및 보완				◀▬▬▬▬▬▶			8	(계획)
9	최종 보고서 작성					◀▬▬▶		4	(계획)
10	데모 시연 및 발표					◀▬▬▶		4	(계획)