OpenCGA Wiki 번역본

<General>

<https://github.com/opencb/opencga/wiki>

: OpenCGA는 수백 테라 바이트의 게놈 데이터 분석을 위한 빅데이터 스토리지 엔진 및 분석 프레임 워크 제공. 생물 정보학 파일 형식 제공. 시각화, 분석 등의 플랫폼.

주요 기능

* + 파일 업로드, 다운로드
  + 정보를 원본 파일 형식과 무관하게 일반적인 방식으로 저장
  + 정보를 효율적으로 검색
  + NoSQL 데이터 베이스 선택 가능
  + 빅데이터 처리 및 저장에는 Apache Hadoop
  + 정보 처리 및 시각화에 HTML5, RESTful 웹서비스
* Platform Overview

: OpenCGA의 인프라구조 파일이 시스템에 업로드될 때, 저장을 위해 파일 시스템에 저장 (hadoop 이나 unix 기반)

상호 질의를 위해 DB에 저장 (MongoDB)

: Dual-Schema -> 파일 시스템에서 원본 파일을 다운로드 할 수도 있고, 파일 읽는 것 보 다 빠르게 DB에서 데이터 회수 가능

R

aw

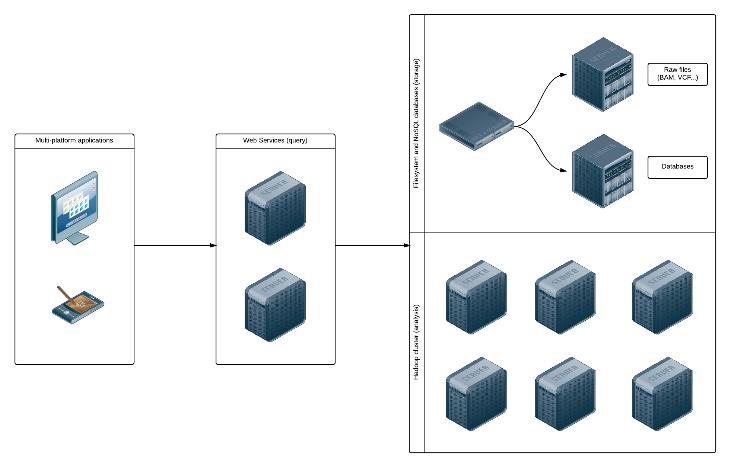
data

Databases

H

adoop

cluster



위는 Storage ( Raw data를 저장하는 File System 과 NoSQL DB), 아래는 분석을 위한

Hadoop cluster

: Apache Hadoop 은 빅데이터 분석 표준. OpenCGA는 하둡으로 파일에 액세스하고 분석 가능. 첫째, HDFS 파일 시스템에 파일을 저장하면 직접 읽을 수 있고, Map/Reduce 작업 돌릴 수 있다. Hbase DB에 데이터 저장되면, 실시간 쿼리도 가능. Hadoop은 네트워크 트 래픽을 줄이고 효율성을 높이기 위해 데이터 저장 및 계산에 동일 시스템 적용

: 사용자는 스토리지나 분석 시스템에 직접 접근은 불가 but 웹 서비스 API를 통해 접근. 정보 접근과 보안 동시에 제공.

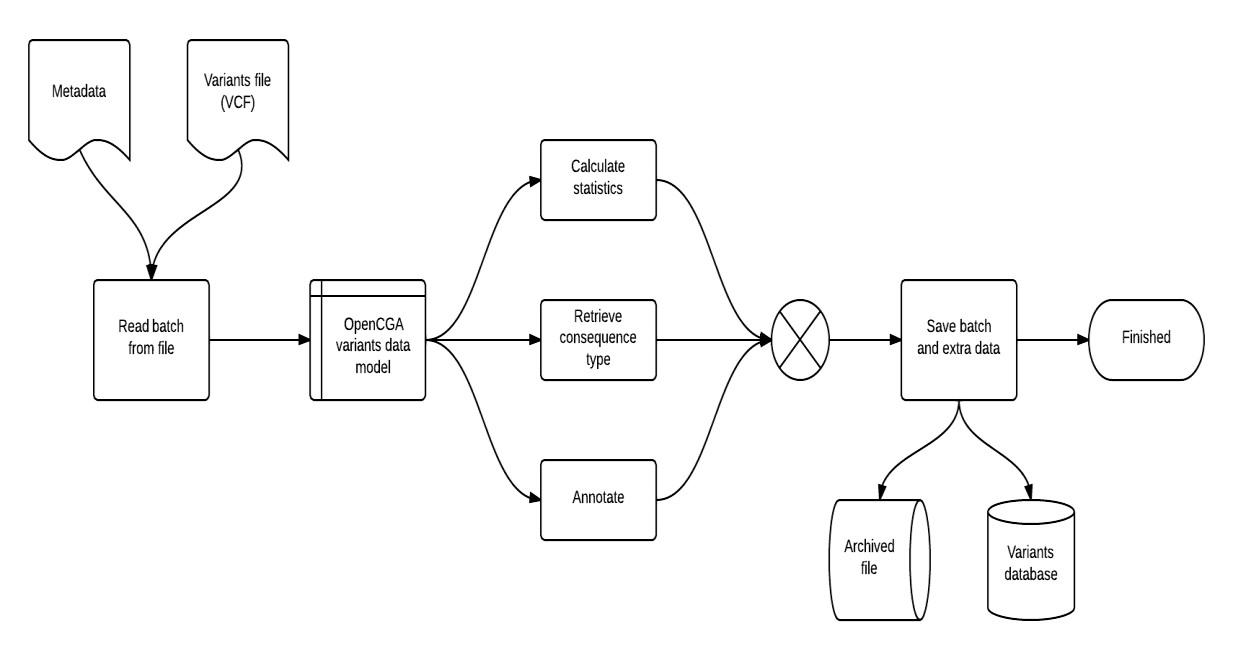
1. Architecture

<https://github.com/opencb/opencga/wiki/Architecture>

* + Submission Process

: 파일은 변이와 alignment를 포함한 메타 데이터와 configuration 파일이 첨부 되어

야 함. 메모리를 제어하기 위해 파일 엔트리는 수천 줄 배치로 읽음



* 흐름도

Read 후 파일 형식 변환. 변이형에 결과 유형, 유전자 명, 통계 정도를 주석으로 달 수 있고 이후 데이터 모델에 추가. 마지막으로 DB에 저장

* + Fault Tolerance

: 메모리 사용 제어를 위해서, 입력 파일을 수천 가지 변이의 배치로 처리하고 저장.

프로세스가 실패할 수도 있으므로 falut tolerance와 rollback 메커니즘 필요

Hbase에 Raw 데이터 삽입하고 Mongo에서 인덱싱 하는 동안 오류 발생 가능. 변이 삽입 실패 시 재시도.

여러 번 시도 후에도 시스템이 복구 불가능 하면 전체 작업 롤백. 따라서 레지스터가 디비에 성공적으로 저장될 때 마다 로그파일에 항상 기록

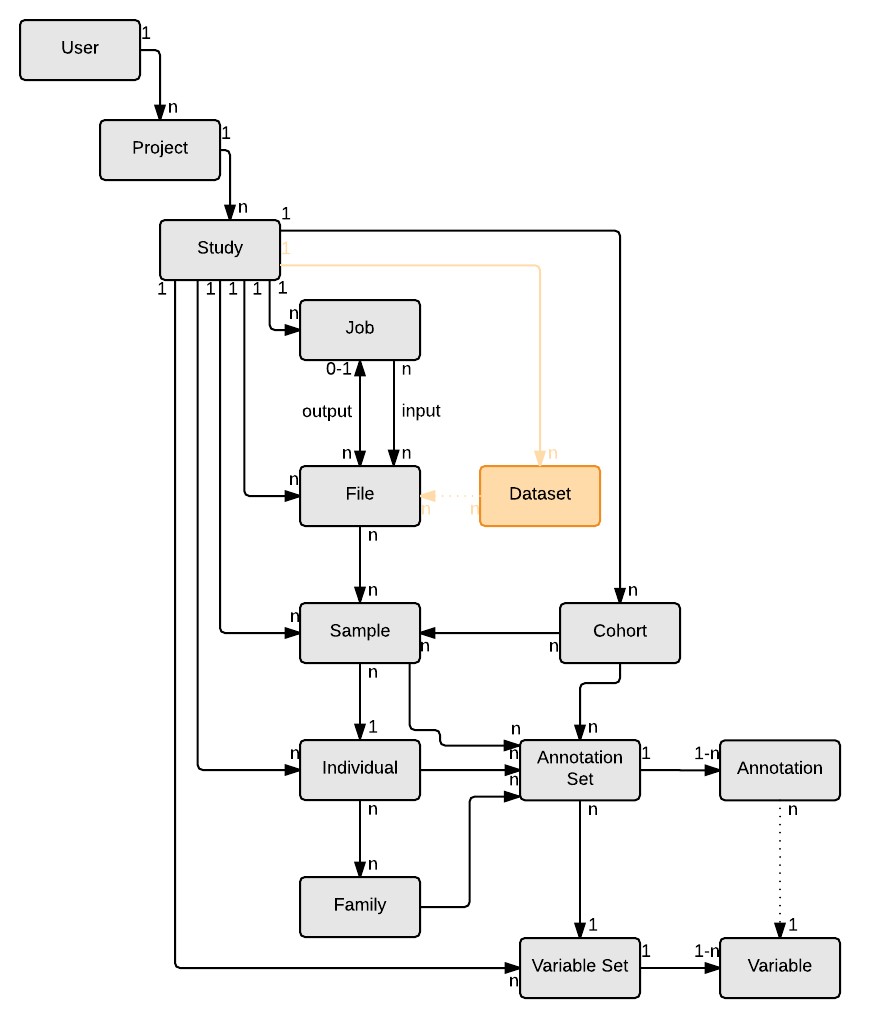
1. 데이터 모델

<https://github.com/opencb/opencga/wiki/Data-Models>

: 원래 데이터 형식을 모르는 게 중요. 파일 전송 등을 위해서만 형식에 대한 참조 저장. 변형과 정렬을 위한 데이터 모델은 Java로 설계 구현. 명시적으로 가장 흔한 필드를 명시, 동시에 특정 형식의 모든 정보를 보존하는 메커니즘 제공. 예를 들어, 변이에 대해 지정 된 필드는 염색체, 위지, 참조 및 대체항목 등. 만약 VCF를 저장하면, INFO 같은 열도 키 밸류 형식으로 저장.

* + 셀베이스 등 OpenCGA 카탈로그

<https://github.com/opencb/opencga/wiki/OpenCGA-Catalog-Data-Models>



* + 1. 공통으로 들어가는 필드

: id ( 숫자, unique, API와 REST 웹서비스에서 사용)

: status ( 객체의 상태 저장. READY, TRASHED, DELETED 있음)

: attributes ( JSON 객체 저장 )

: lastActivity ( 데이터의 마지막 업데이트 기록)

* + 1. Session

: 사용자의 모든 로그인/아웃 기록

* + 1. User , Project

: id – 알파벳 문자열

: status – 디폴트 이외의 추가적 상태 (BANNED)

: account – GUEST( 프로젝트나 스터디 생성 불가), FULL( 다 가능)

: configs – config map에서, 사용자는 자신이 쓴 쿼리 저장 가능, 회수도 가능

{ // 예시

"id": "jsmith",

"name": "John Smith",

"email": "jsmith@do.co",

"password": "a04f2825f2227f70e41d58b16c890661a803b453",

"organization": "ACME",

"account" : {

"type" : "full",

"creationDate" : "20160920090458",

"expirationDate" : "20170920090458",

"authOrigin" : "internal"

},

"status" : {

"name" : "READY",

"date" : "20160920090458",

"message" : ""

},

"lastModified" : "20160920090752582",

"diskUsage" : -1,

"diskQuota" : 200000,

"projects" : [

{

"id" : 1,

"name" : "Default",

"alias" : "default",

"creationDate" : "20160920090458",

"description" : "This is my project description.",

"organization" : "ACME",

"status" : {

"name" : "READY",

"date" : "20160920090458",

"message" : ""

},

"lastModified" : null,

"diskUsage" : 0,

"studies" : [ ],

"dataStores" : { },

"attributes" : { }

}

],

"tools" : [ ],

"sessions" : [

{

"id" : "oQRQcpBRmkCi1rhMJlGi",

"ip" : "localhost",

"login" : "20160920090458",

"logout" : "20160920090458"

}

],

"configs" : { },

"attributes" : { }

}

4. Study

: 메인 객체. 파일, 작업, 샘플의 세트. 스터디의 모든 파일은 위치, Cipher, 공유 옵션을 공 유한다.

: type (to cohort) – CASE\_SET, CONTROL\_SET, CASE\_CONTROL, PAIRED, FAMILY, TRIO

: stats

: status – ACTIVE

: diskUsage – 스터디의 모든 파일의 디스크 용량 사용 총 합

: cipher – NONE( 암호화 X), AES\_256( 구현 X)

: uri – 스터디의 위치, PATH 대신 사용

{ // 예시

"id": 15,

"name": "Study test 1",

"alias": "std1",

"type": "FAMILY",

"creatorId": "jcoll",

"creationDate": "20141215182938",

"description": "",

"status": "ACTIVE",

"lastActivity": "20141215182938",

"diskUsage": 0,

"cipher": "NONE",

"acl": [ ],

"experiments": [ ],

"files": [ ],

"jobs": [ ],

"samples": [ ],

"uri": "hdfs:///data/opencga/catalog2/users/jcoll/projects/14/15/",

"datasets": [

{

"id": 0,

"name": "bam\_test\_files",

"creationDate": "20141215182938",

"description": " ... ",

"files": [ 26, 27, 28, 29, 35, 36, 38],

"attributes": { }

}

],

"cohorts": [ ],

"variableSets": [ ],

"stats": { },

"attributes": { }

}

1. File

: type – FILE(파일 시스템에 있는 진짜 파일), FOLDER(폴더)

: format – PLAIN, GZIP, BINARY, IMAGE, EXECUTABLE

: bioformat – VARIANT, ALIGNMENT, SEQUENCE, NONE

: status ( 파일 상태 ) – STAGED(생성됨), READY(사용가능), MISSING(물리적 파일이 손실), TRASHED(삭제 대기중), DELETED(삭제됨)

: jobid , experimentld ( 파일의 근원을 명시, job이나 experiment에서 생성됨)

|  |
| --- |
| { // 예시  "id" : 3, "name" :  "chr14.phase1\_release\_v3.20101123.snps\_indels\_svs.genotypes.refpanel.AMR.vcf.gz",  "type" : "FILE",  "format" : "GZIP", "bioformat" : "VARIANT", "path" :  "data/vcf/chr14.phase1\_release\_v3.20101123.snps\_indels\_svs.genotypes.refpanel.AMR. vcf.gz",  "ownerId" : "jcoll",  "creationDate" : "20141215162449",  "description" : " ... ",  "status" : "READY",  "diskUsage" : 24276833,  "experimentId" : -1,  "sampleIds" : [ ],  "jobId" : -1,  "acl" : [ ],  "stats" : { },  "attributes" : { }  } |

1. Job

|  |
| --- |
| {  "id" : 138,  "name" : "Test job",  "userId" : "jcoll",  "toolName" : "network-miner",  "date" : "20141031151537",  "description" : " ... ",  "startTime" : 1415632245213,  "endTime" : 1415632258708, "outputError" : "",  "commandLine" : "/opt/opencga/analysis/network-miner/babelomics/babelomics.sh -tool network-miner --seedlist 150140.chrom20.ILLUMINA.bwa.CHM1.20131218.bam.bai -  -significant-value 0.05 --list  HG00096.mapped.ILLUMINA.bwa.GBR.low\_coverage.20120522.bam --list-tags gene -intermediate 1 --outdir /home/cafetero/opencga/catalog/jobs/J\_KrOrWfEwkx/ --order ascending --interactome hsa --randoms 500 --components false --group all --o-name result",  "visits" : -1,  "status" : "READY", "outDirId" : "6",  "tmpOutDirUri" : "file:///home/cafetero/opencga/catalog/jobs/J\_KrOrWfEwkx/", |

"input" : [

66

],

"tags" : [ ],

"output" : [

658,

659

],

"attributes" : { },

"executionAttributes" : {

"type" : "analysis",

"jobExecutionId" : "268",

"executionManager" : "SGE",

"qname" : "normal.q",

"group" : "cafetero",

"jobname" : "network-miner\_Test\_job",

"end\_time" : "Wed Dec 10 11:10:06 2014",

"jobnumber" : 268,

"failed" : 0,

"start\_time" : "Wed Dec 10 11:10:06 2014",

"hostname" : "host001",

"qsub\_time" : "Wed Dec 10 11:10:00 2014",

"mem" : "0.000",

"cpu" : "0.049",

"exit\_status" : 0

}

}

7. Sample

{

"id" : 19,

"name" : "SMP00096",

"source" : "",

"individual" : null,

"description" : " ... ",

"annotationSets" : [

{

"name" : "Basic annotation",

"variableSetId" : 21,

"annotations" : [

{ "id" : "NAME", "value" : "Glennie the platypus" },

{ "id" : "BORN-DATE", "value" : "20071000000000" },

{ "id" : "GENDER", "value" : "FEMALE" }

{ "id" : "PHEN", "value" : "CASE" }

{ "id" : "WEIGHT", "value" : 25.38 }

],

"date" : "20141216135957",

"attributes" : { }

}

]

}

1. Restful Web Services

<https://github.com/opencb/opencga/wiki/RESTful-Web-Services>

- URL의 이해

OpenCGA RESTful URL :

[http://HOST\_URL/webservices/rest/{version}/{resource}/{id}/{endpoint}?{options}](http://host_url/webservices/rest/%7bversion%7d/%7bresource%7d/%7bid%7d/%7bendpoint%7d?%7boptions%7d)

HOST\_URL : [**http://bioinfodev.hpc.cam.ac.uk/opencga**](http://bioinfodev.hpc.cam.ac.uk/opencga)

 인자 : version – opencga의 버전 표시, resource - id에서 회수하고 싶은 데이터 타입 을 명시, id – 질의를 하고 싶은 리소스 id, endpoint – 입력 데이터에 따라 명시됨.

만약 특정 스터디의 모든 파일에 질의 하고 싶으면 그 스터디와 파일을 엔드포인트 로 설정. options – exclude, include, limit, skip and count 등의 설정

|  |  |  |
| --- | --- | --- |
| **Category** | **Description** | **Main points** |
| ***users*** | Different methods to work with *users* | info, create, login, logout, ... |
| ***projects*** | *projects* are defined for each user and contains *studies* | info, create, studies, ... |
| ***studies*** | *studies* are the main component of catalog, the can be shared and contain *files*, *samples* and *jobs* | info, create, files, samples, jobs, variants, alignments, groups, ... |
| ***files*** | *files* are added to the *study* and can be indexed to be queried | info, create, index, share, ... |
| ***jobs*** | *jobs* are tool executions that can be queued | info, create, ... |
| ***individuals*** | *samples* come from the *individuals* | info, create, ... |
| ***samples*** | *samples* are each of the experiment samples, typically matches a NGS BAM file or VCF sample | info, create, annotate, share, ... |
| ***variables*** | *variables* annotate *samples* with different information useful for data analysis | info, crate, ... |
| ***cohorts*** | these model a group of *samples* that share some common properties, these are used for data analysis | info, create, stats, samples, ... |
| ***tools*** | bioinformatics *tools* installed for data analysis | info, help, ... |

< Resource 종류 > <http://bioinfodev.hpc.cam.ac.uk/opencga/webservices/> -> 더 알아보기

1. Configuration

<https://github.com/opencb/opencga/wiki/Configuration>

(Configuration File) 4 가지

: configuration.yml – 기본 설정

: catalog-configuration.yml – 메타데이터 등의 OpenCGA catalog 설정

: storage-configuration.yml – 인덱싱과 질의를 위한 스토리지 관련 설정

: client-configuration.yml – user CLI 나 Restful web services clients를 위한 설정

<https://github.com/opencb/opencga/blob/develop/opencga-core/src/main/resources/configuration.yml> 참고

| **Variable** | **Maven property** | **Description** |
| --- | --- | --- |
| **rest** | - | these determine the REST server properties |
| **rest > port** | *OPENCGA.SERVER.REST.PORT* | set the REST server port |

1. Tutorials
   * 1. **Indexing VCF files (v0.7.0)**

[https://github.com/opencb/opencga/wiki/Indexing-Variant-Data-v0.7.0](https://github.com/opencb/opencga/wiki/Indexing-Variant-Data-v0.8.0)

: VCF는 CLI 파이프라인 또는 JAVA API를 사용해 인덱싱 가능. 목적은 인덱싱 된 데이터를 통한 쿼리 허용. 첫째 변환 단계에서는 VCF 데이터가 정규화 되어 내부 변형 데이터 모델로 변환. 두번째 로드 단계 동안 정규화 되고 유효성 검사 된 파일은 active storage engine plugin에 로드. 자세한 내용은 openCGA storage 에서.

여기서는 1000 genomes project 샘플 VCF 사용. (아래 복사)

ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/release/20101123/interim\_phase1\_release/ALL.chr22.phase1.projectConsensus.genotypes.vcf.gz

* + 커맨드 라인 인터페이스 사용

VCF 인덱싱 두가지 방법

1. 분석 : OpenCGA catalog를 사용하는 high level CLI 사용

./opencga.sh --version

Version 0.7.0

git version: master bc312857972c90ded9448b251674f04afd2bcf74

user account 필요. Catalog에 저장.

./opencga.sh users create -u myuser -p mypass -e my@e.mail -n "my name"

로그인.

./opencga.sh users login -u myuser -p mypass

여러 개의 프로젝트나 각 프로젝트의 여러가지 스터디에 데이터를 정렬할 수 있다.

./opencga.sh projects create -a myproject -n "Default project" -d "First project created."

./opencga.sh studies create -a mystudy --project-id myuser@myproject -n "Default study" -d "First study created."

실제 인덱싱 파트. 일단 Catalog에 file 위치 전달

./opencga.sh files create -i ALL.chr22.phase1.projectConsensus.genotypes.vcf.gz --study-id myuser@myproject/mystudy --bioformat VARIANT

인덱싱 요청.

./opencga.sh files index -id myuser@myproject/mystudy/ALL.chr22.phase1.projectConsensus.genotypes.vcf.gz

변이 annotation 과 통계 계산.

./opencga.sh files index -u myuser -p mypass -id myuser@myproject/mystudy/ALL.chr22.phase1.projectConsensus.genotypes.vcf.gz --calculate-stats --annotate

스터디에서 변이 annotation

./opencga.sh studies annotate-variants -id myuser@myproject/mystudy

인덱스 된 데이터 불러오기

./opencga-storage.sh fetch-variants -r 20:16050050-16050100 --database opencga\_myuser\_myproject

더 알아보려면,

./opencga-storage.sh fetch-variants --help

1. 저장 : OpenCGA catalog에 의존적인 low level CLI 사용, 이미 메타데이터서버가 있거나 인덱싱 스토리지가 필요할 때 사용.

기본 인덱싱

./opencga-storage.sh index-variants --studyId 1 --study-name default -i ALL.chr22.phase1.projectConsensus.genotypes.vcf.gz --database chr22\_test\_db

2단계로 쪼갠 인덱싱 ( load를 미루기 )

./opencga-storage.sh index-variants --study-id 1 --study-name default -i ALL.chr22.phase1.projectConsensus.genotypes.vcf.gz --transform

./opencga-storage.sh index-variants --study-id 1 --study-name default -i ALL.chr22.phase1.projectConsensus.genotypes.vcf.gz.variants.json.gz --database chr22\_test\_db --load

데이터 불러오기 쿼리

./opencga-storage.sh fetch-variants --database chr22\_test\_db --limit 10

더 알아보려면,

./opencga-storage.sh fetch-variants --help

**2. Querying Variants using CLI**

<https://github.com/opencb/opencga/wiki/Querying-Variants-using-CLI>

: CLI를 사용하여 OpenCGA 스토리지에서 데이터를 불러오는 방법은 2가지.

Opencga.sh : opencga catalog를 사용하여 데이터를 불러오는 top-level CLI

Opencga-storage.sh : region,gene 등의 변이 속성을 이용해 불러오는 low-level

둘다 비슷한 기능과 인자를 가짐. 가장 큰 차이는 top-level CLI 는 opencga catalog를 사용. 따라서 family나 샘플 annotation에 의해 복잡한 질의를 할 수 있음

OPENCGA\_HOME/bin 폴더에 있음

* + Opencga-storage.sh 사용

: 유전체 regions이나 특성 id

: 변이 주석

: 샘플 유전형

: 스터디의 변이 stats

: 랭크 등의 기본 집합

**이외의 튜토리얼은 현재 공사중**

**<OpenCGA Catalog>**

: