

Head Start Application

1. Migrate Legacy Java App (XWiki)
(User Guide Documentation)

February 2023

Content

Content	1
Glossary	2
1. Overview	2
2. GCP Microservices and API	4
API Enabled	4
GCP Microservices/ Products	4
3. Setup	5
3.1 Prerequisites	5
3.2 Configuration	6
1. Clone XWiki repository	6
2. Enable Service API:	6
3.3 Build XWiki	7
3.4 XWiki	9
3.5 Remove XWiki	10
4. Load Test with JMeter	11
(Optional step for Load Test) Increase C2 CPU quota for autoscaling:	11
5. View Metrics on DataDog	13

Glossary

GCP	Google Cloud Platform
GCE	Google Compute Engine
DATADOG API KEY	DataDog web page navigate to Organization settings -> API Keys -> Create or select a key
DATADOG APP KEY	DataDog web page navigate to Organization settings -> Application Keys -> Create or select a key

1. Overview

Head Start application - Migrate Legacy Java App aims to reduce the entry barrier for Google Cloud Platform developers in migrating a local Java application infrastructure to a cloud Java application infrastructure on Google Cloud Platform.

This project uses XWiki (ver. 14.10.4) as a sample application, in which XWiki WAR files and dependencies are built and deployed on Google Compute Engine, with as few end-user steps as possible with a Terraform designed application.

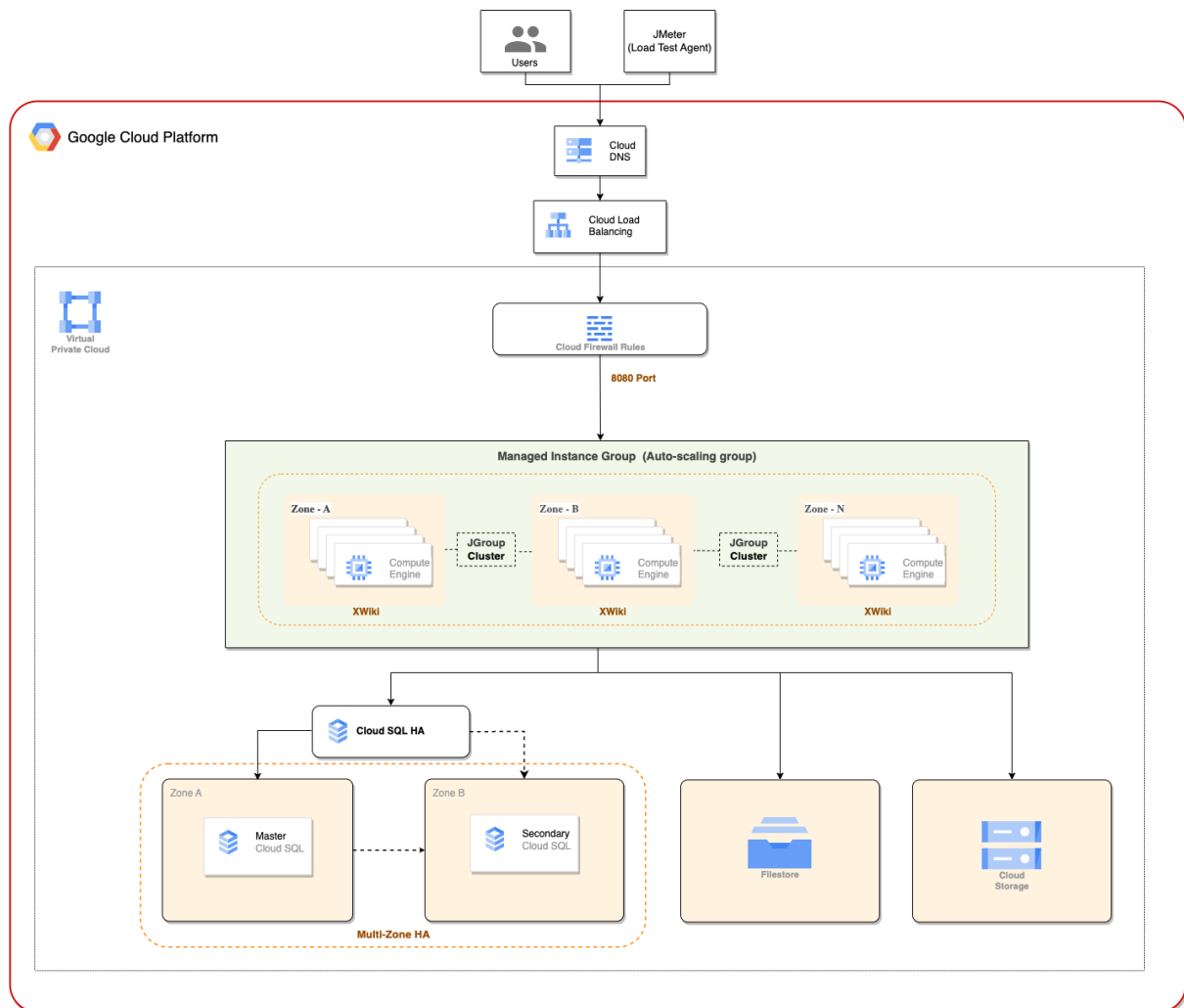


Figure above depicts XWiki-GCE architecture and GCP resources used within the application.

2. GCP Microservices and API

API Enabled

- Cloud Auto-scaling API
- Cloud Build API
- Cloud Filestore API
- Cloud Pub/Sub API
- Cloud Source Repositories API
- Cloud SQL API
- Cloud Storage API
- Container File System API
- Google Cloud APIs
- Google Cloud Storage JSON API
- IAM Service Account Credentials API
- Identity and Access Management (IAM) API
- Compute Engine API
- Secret Manager API
- Service Management API
- Service Networking API
- Service Usage API

GCP Microservices/ Products

- Google Cloud Build
- Google Cloud Storage
- Google Compute Engine
- Google Cloud Source Repositories
- Google Filestore
- CloudSQL

3. Setup

This section contains step by step instructions to setup XWiki through GCP **Cloud Shell Editor**.

3.1 Prerequisites

These requirements are needed to run the XWiki-GCE Terraform build:

- [Google Cloud Platform](#) account.
- Create a New Google Cloud project.
- [DataDog](#) account [Optional]
 - DataDog API Key
 - DataDog APP Key
- Access to Migrate-Legacy-Java repo:
 - For Migrate-Legacy-Java-App-GCE (Choose one)
 - Cloud Source Repositories (requires access to **hsa-testing** project):
https://source.cloud.google.com/hsa-testing/github_hsa-integration_migrate-legacy-java-app-gce/+/main:
 - GitHub Repo (requires collaborator access):
<https://github.com/HSA-Integration/Migrate-Legacy-Java-App-GCE>
- XWiki VM Image:

Please note that the location of the customized public XWiki VM Image is currently hard-coded in the project

This image is currently publicly accessible from CleNet's GCP project, but will be provided in Google's testing project (e.g. **hsa-testing**) in the future.

Url to fetch the image:

[https://www.googleapis.com/compute/beta/projects/\\$%7Bvar.xwiki_img_info.image_project%7D/global/images/\\$%7Bvar.xwiki_img_info.image_name%7D](https://www.googleapis.com/compute/beta/projects/$%7Bvar.xwiki_img_info.image_project%7D/global/images/$%7Bvar.xwiki_img_info.image_name%7D)

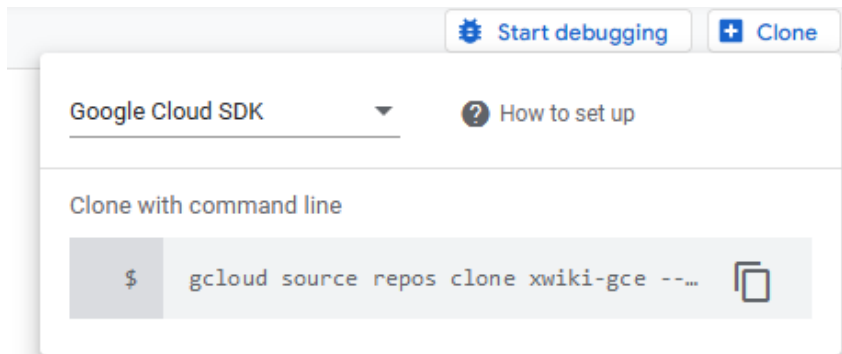
In *tools/prepare_tfvars.sh*, the *image_project* and *image_name* variables are used to detect which image should be retrieved and used in the application. Currently these variables are hardcoded in our implementation.

3.2 Configuration

Sets up the required services and APIs that are needed to build the application on GCE.

1. Clone XWiki repository

Run the following command in GCP Cloud Shell terminal to clone the provided XWiki repository to your own private repository:



For GCE:

- `$ gcloud source repos clone github_newfel-cloud_migrate-legacy-java-app-gce --project=hsa-testing`

You can navigate to **Cloud Shell Editor** and confirm the repository has successfully been cloned to your own private repo.

2. Enable Service API:

Enable the service API through **APIs & Services -> Enable APIs and Services**

- **Cloud Build API -> Enable**
 - Once enabled, go to **IAM & Admin** -> locate principal xxxxx@cloudbuild.gserviceaccount.com -> Edit -> Add Another Role -> Basic -> Owner -> Save.
- **Secret Manager API -> Enable**

3.3 Build XWiki

Utilizing **Cloud Shell Editor** terminal, you can build the application with Cloud Build by running the following command inside your XWiki directory:

```
$ gcloud builds submit . --substitutions
_ZONE1=${ZONE1}, _ZONE2=${ZONE2}, _XWIKI_SQL_USER_PASSWORD=${SQL_PWD}
--region=${REGION}
```

If DataDog is enabled:

```
$ gcloud builds submit . --substitutions
_ZONE1=${ZONE1}, _ZONE2=${ZONE2}, _DATADOG_API_KEY=${DATADOG_API_KEY}, _DATADOG
_APP_KEY=${DATADOG_APP_KEY}, _XWIKI_SQL_USER_PASSWORD=${SQL_PWD}
--region=${REGION}
```

- Replace \${REGION} with a region of your choice e.g. us-west1 or us-central1

Replace \${ZONE1} and \${ZONE2} with the fully-qualified names of the selected zones. E.g. if the fully-qualified names of the selected zones are *us-west1-b* and *us-west1-c*. \${ZONE1} and \${ZONE2} values will be **us-west1-b** and **us-west1-c** respectively.

GCP offers a variety of locations to host your server e.g. *us-west1-b*, please refer to [Available regions and zones](#).

- [If DataDog is enabled] \${DATADOG_API_KEY} and \${DATADOG_APP_KEY} can be found under your DataDog -> Organization settings.
- \${SQL_PWD} will be the password set for the SQL database used by XWiki. If the parameter is not initialized in the command, The default password for the SQL database will be **vuYTCazG0lcEc**.

An example of what the command would look like:

```
$ gcloud builds submit . --substitutions
_ZONE1=us-west1-a, _ZONE2=us-west1-b, _DATADOG_API_KEY=b8be7b83c065196ae9753ds
3
, _DATADOG_APP_KEY=c6528cf345bc3ef94a203ea0c, _XWIKI_SQL_USER_PASSWORD=Xw!kipW
d44123 --region= us-west1
```

Once the command is executed and running, navigate to **Cloud Build -> History** to view the build log. There should be 10 total build steps executed for a successful build.

Once the build is successfully completed, all necessary server, database and application is built and is ready to be launched.

3.4 XWiki

Navigate to http://{LOAD_BALANCER_FRONTENDS_IP}:8080/xwiki , XWiki should have Flavor UI installed and can be used by end-users instantly.

Replace `${LOAD_BALANCER_FRONTENDS_IP}` with the frontend IP address of the load balancer, the IP address can be retrieved using one of the following methods:

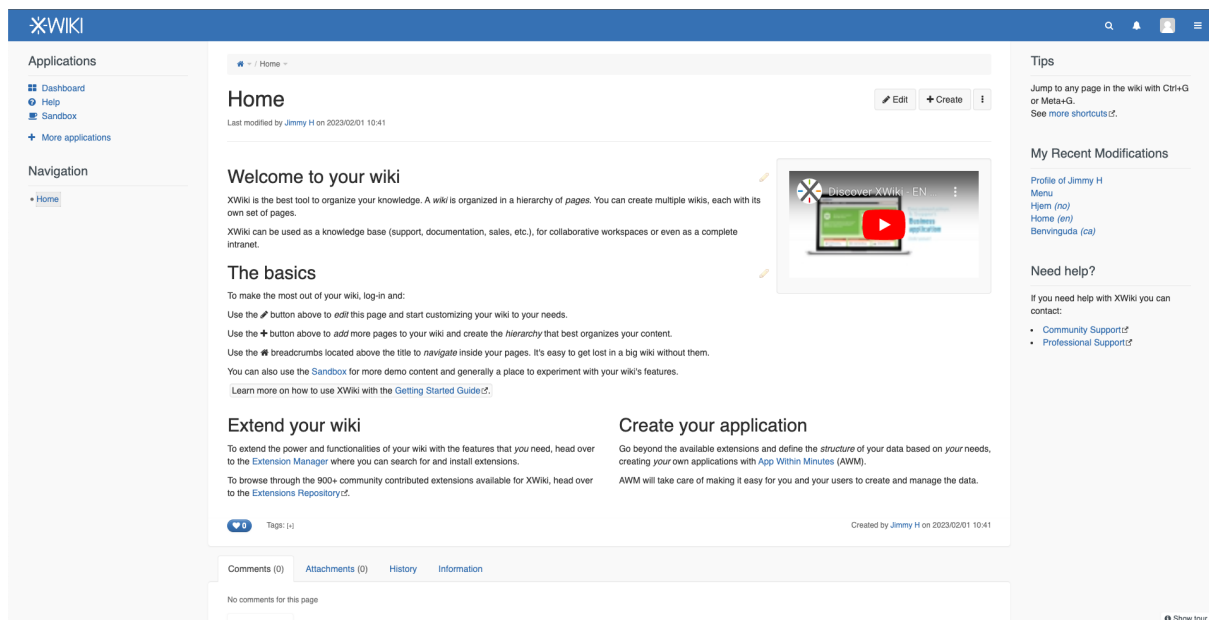
- **From GCP console:**

- GCE:**

- Network services -> Load balancing -> Frontends -> Copy the IP address shown in the console.

- **Run the gcloud command in Cloud Shell Editor terminal:**

- `gcloud compute forwarding-rules list --format="value(IP_ADDRESS)"`



3.5 Remove XWiki

To remove the entire application and GCP products/services created, run the following command under XWiki-GCE directory in the Cloud Shell terminal:

```
$ gcloud builds submit . --substitutions _ZONE1=${ZONE1}, _ZONE2=${ZONE2}  
--region=${REGION} --config=cloudbuild_destroy.yaml
```

If DataDog was enabled, include the Datadog api and app key you had input in the build command:

```
$ gcloud builds submit . --substitutions  
_ZONE1=${ZONE_CODE1}, _ZONE2=${ZONE_CODE2}, _DATADOG_API_KEY=${DATADOG_API_KEY},  
_DATADOG_APP_KEY=${DATADOG_APP_KEY} --region=${REGION}  
--config=cloudbuild_destroy.yaml
```

Replace \${REGION}, \${ZONE1} and \${ZONE2} with the region and zone code you had selected to build XWiki in.

All services and products built in [3.3](#) should all be removed from your GCP project.

4. Load Test with JMeter

Perform a load test on the built XWiki server simulated through Apache JMeter, expecting the server to autoscale and handle the load balance. To run the load test, navigate to your XWiki-GCE folder that is in your Cloud Shell Environment and perform the following steps:

(Optional step for Load Test) Increase C2 CPU quota for autoscaling:

- Navigate to IAM & Admin -> Quotas
https://console.cloud.google.com/iam-admin/quotas?project={PROJECT_ID}
- Increase the C2 CPUs in your selected region to at least **30** if needed.

Service	Quota	Dimensions (e.g. location)	Limit	Current usage percentage	Current usage	7 day peak usage percentage
Compute Engine API	C2 CPUs	region: us-central1	8	100%	8	100%
Cloud Filestore API	Basic HDD (Standard) capacity (GB) per	region: us-central1	2,048 GB (2.048 TB)	50%	1,024 GB (1.024 TB)	50%

*Note: Project quota restrictions may affect the performance of XWiki load balancing.

1. Open Cloud Shell terminal.
2. Create a folder "*load-test*" (or any custom name your prefer) and change current directory to *load-test* folder
 - `$ mkdir load-test`
 - `$ cd load-test`
3. Copy the JMeter load test script to the above folder
 - `$ cp {PATH_TO_XWIKI_REPO}/tools/start_load_test_gce.sh`
4. Run **start_load_test.sh** script to start the load test
 - `$ sh -x start_load_test_gce.sh`
 - The script will get GCE to expose the load balancer ip and download:
 - `Apache-jmeter-5.5.zip`: JMeter software zip file
 - `xwiki_load_test_30.jmx`: JMeter plan file
 - The load test should be running after a few seconds.

```

summary = 18962 in 00:10:26 = 30.3/s Avg: 966 Min: 159 Max: 30202 Err: 26 (0.14%) Active: 30 Started: 30 Finished: 0
summary + 1328 in 00:10:30 = 44.3/s Avg: 675 Min: 350 Max: 1446 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 20290 in 00:10:56 = 30.9/s Avg: 947 Min: 159 Max: 30202 Err: 26 (0.13%) Active: 30 Started: 30 Finished: 0
summary + 1305 in 00:11:00 = 43.5/s Avg: 692 Min: 356 Max: 1477 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 21595 in 00:11:26 = 31.5/s Avg: 931 Min: 159 Max: 30202 Err: 26 (0.12%) Active: 30 Started: 30 Finished: 0
summary + 1405 in 00:11:30 = 46.8/s Avg: 641 Min: 350 Max: 1364 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 23000 in 00:11:56 = 32.1/s Avg: 914 Min: 159 Max: 30202 Err: 26 (0.11%) Active: 30 Started: 30 Finished: 0
summary + 1401 in 00:12:00 = 45.7/s Avg: 642 Min: 350 Max: 1293 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 24401 in 00:12:26 = 32.7/s Avg: 898 Min: 159 Max: 30202 Err: 26 (0.11%) Active: 30 Started: 30 Finished: 0
summary + 1323 in 00:12:30 = 44.1/s Avg: 679 Min: 345 Max: 1493 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 25724 in 00:12:56 = 33.2/s Avg: 887 Min: 159 Max: 30202 Err: 26 (0.10%) Active: 30 Started: 30 Finished: 0
summary + 1392 in 00:13:00 = 46.4/s Avg: 645 Min: 347 Max: 1296 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 27116 in 00:13:26 = 33.6/s Avg: 874 Min: 159 Max: 30202 Err: 26 (0.10%) Active: 30 Started: 30 Finished: 0
summary + 1340 in 00:13:30 = 44.7/s Avg: 673 Min: 348 Max: 1378 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 28456 in 00:13:56 = 34.0/s Avg: 865 Min: 159 Max: 30202 Err: 26 (0.09%) Active: 30 Started: 30 Finished: 0
summary + 1343 in 00:14:00 = 44.8/s Avg: 668 Min: 344 Max: 1765 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 29799 in 00:14:26 = 34.4/s Avg: 856 Min: 159 Max: 30202 Err: 26 (0.09%) Active: 30 Started: 30 Finished: 0
summary + 1400 in 00:14:30 = 46.6/s Avg: 644 Min: 351 Max: 1428 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 31199 in 00:14:56 = 34.8/s Avg: 846 Min: 159 Max: 30202 Err: 26 (0.08%) Active: 30 Started: 30 Finished: 0
summary + 1404 in 00:15:00 = 46.8/s Avg: 639 Min: 351 Max: 1485 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 32603 in 00:15:26 = 35.2/s Avg: 837 Min: 159 Max: 30202 Err: 26 (0.08%) Active: 30 Started: 30 Finished: 0
summary + 1359 in 00:15:30 = 45.3/s Avg: 660 Min: 345 Max: 1344 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 33962 in 00:15:56 = 35.5/s Avg: 830 Min: 159 Max: 30202 Err: 26 (0.08%) Active: 30 Started: 30 Finished: 0
summary + 1327 in 00:16:00 = 44.2/s Avg: 678 Min: 344 Max: 1549 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 35289 in 00:16:26 = 35.8/s Avg: 825 Min: 159 Max: 30202 Err: 26 (0.07%) Active: 30 Started: 30 Finished: 0
summary + 1296 in 00:16:30 = 43.2/s Avg: 695 Min: 343 Max: 1458 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 36585 in 00:16:56 = 36.0/s Avg: 820 Min: 159 Max: 30202 Err: 26 (0.07%) Active: 30 Started: 30 Finished: 0
summary + 1331 in 00:17:00 = 44.4/s Avg: 676 Min: 341 Max: 1663 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 37916 in 00:17:26 = 36.3/s Avg: 815 Min: 159 Max: 30202 Err: 26 (0.07%) Active: 30 Started: 30 Finished: 0
summary + 1414 in 00:17:30 = 47.1/s Avg: 635 Min: 345 Max: 1600 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 39330 in 00:17:56 = 36.6/s Avg: 809 Min: 159 Max: 30202 Err: 26 (0.07%) Active: 30 Started: 30 Finished: 0
summary + 1387 in 00:18:00 = 46.2/s Avg: 650 Min: 344 Max: 1614 Err: 0 (0.00%) Active: 30 Started: 30 Finished: 0
summary = 40717 in 00:18:26 = 36.8/s Avg: 803 Min: 159 Max: 30202 Err: 26 (0.06%) Active: 30 Started: 30 Finished: 0
andrewyang@cloudshell:~/demo/load-test (wiki-gke-e2e-test-1)

```

5. Press Ctrl-C to terminate load test process

6. Load test process are record in log files which can be viewed by tail command:

■ `$ tail jmeter_YYYY-MM-DD.log`

summary +	61	in 00:00:27 =	2.2/s Avg:	5007 Min:	5003 Max:	5072 Err:	61 (100.00%)	Active: 28	Started: 28	Finished: 0
summary +	177	in 00:00:30 =	5.9/s Avg:	5005 Min:	5001 Max:	5015 Err:	177 (100.00%)	Active: 30	Started: 30	Finished: 0
summary =	238	in 00:00:57 =	4.2/s Avg:	5006 Min:	5001 Max:	5072 Err:	238 (100.00%)			
summary +	197	in 00:00:30 =	6.6/s Avg:	2845 Min:	489 Max:	24643 Err:	28 (14.21%)	Active: 30	Started: 30	Finished: 0
summary =	435	in 00:01:27 =	5.0/s Avg:	4027 Min:	489 Max:	24643 Err:	266 (61.15%)			
summary +	466	in 00:00:30 =	15.6/s Avg:	2710 Min:	484 Max:	33262 Err:	13 (2.79%)	Active: 30	Started: 30	Finished: 0
summary =	901	in 00:01:57 =	7.7/s Avg:	3346 Min:	484 Max:	33262 Err:	279 (30.97%)			

5. View Metrics on DataDog

Navigate to DataDog console to view the Load Test performance metrics

Frequently used metrics:

GCE:

- Instance CPU Utilization: `gcp.gce.instance.cpu.utilization`
- HTTP request count: `gcp.loadbalancing.https.request_count`

MySQL:

- CloudSQL CPU utilization: `gcp.cloudsql.database.cpu.utilization`
- CloudSQL DB connections: `gcp.cloudsql.database.network.connections`

These metrics can also be filtered using `project_id` for more fine-grained information.

