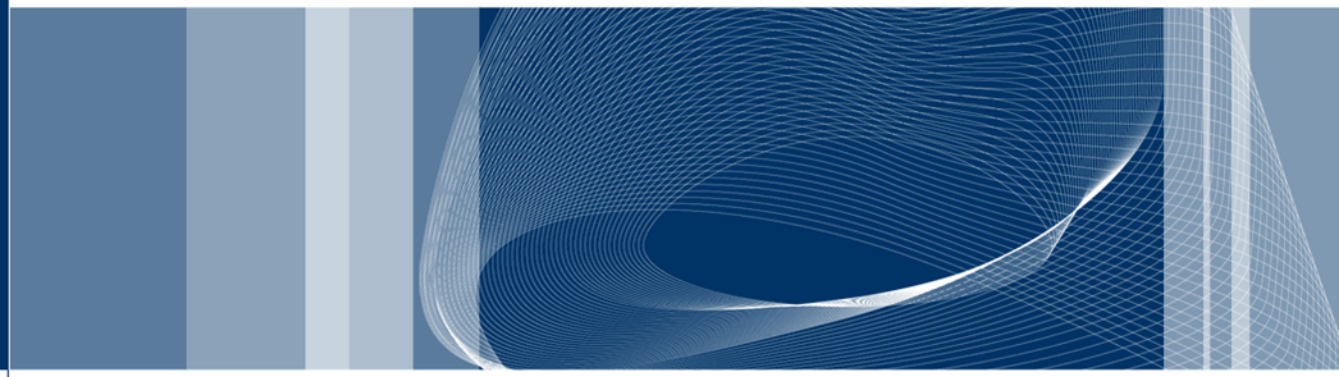
POLITECNICO DI MILANO

Scuola di Ingegneria Industriale e dell'Informazione

Corso di Laurea Magistrale in Ingegneria Informatica

Anno Accademico 2013 - 2014



Avoiding CRUD operations lock-in in NoSQL databases: extension of the CPIM library

Candidato: Fabio Arcidiacono (799001)

Relatore: Prof.ssa Elisabetta Di Nitto

Correlatore: Ing. Marco Scavuzzo

Data management systems

RDBMS

Well structured data

Relational model

Vertical scaling

ACID transactions

SQL

NoSQL

Non-structured data

Various data models

Horizontal scaling

BASE properties

Proprietary API

NoSQL Common language approaches

Meta-model

- Apache MetaModel
- SOS platform

SQLification

- Apache Phoenix
- UnQL
- Native support

ORM

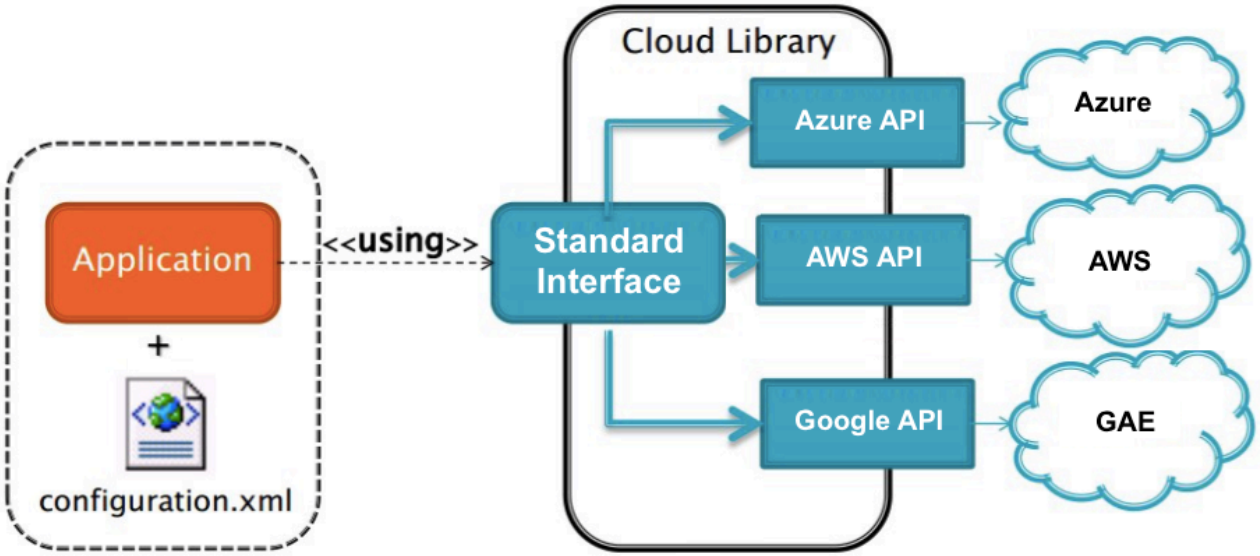
- Kundera
- PlayORM
- Spring-data
- Apache Gora

Cloud Platform Independent Model

Abstract application logic from the specific PaaS Provider to overcome the vendor lock-in

Many supported services:

- Blob
- NoSQL
- Memcache
- Queue
- Mail
- SQL



Work objectives

Integrate Kundera in the CPIM library

Contribute to the open source project Kundera

Integrate the migration and synchronization system Hegira

Evaluation

Work objectives

Integrate Kundera in the CPIM library

Contribute to the open source project Kundera

Integrate the migration and synchronization system Hegira

Evaluation

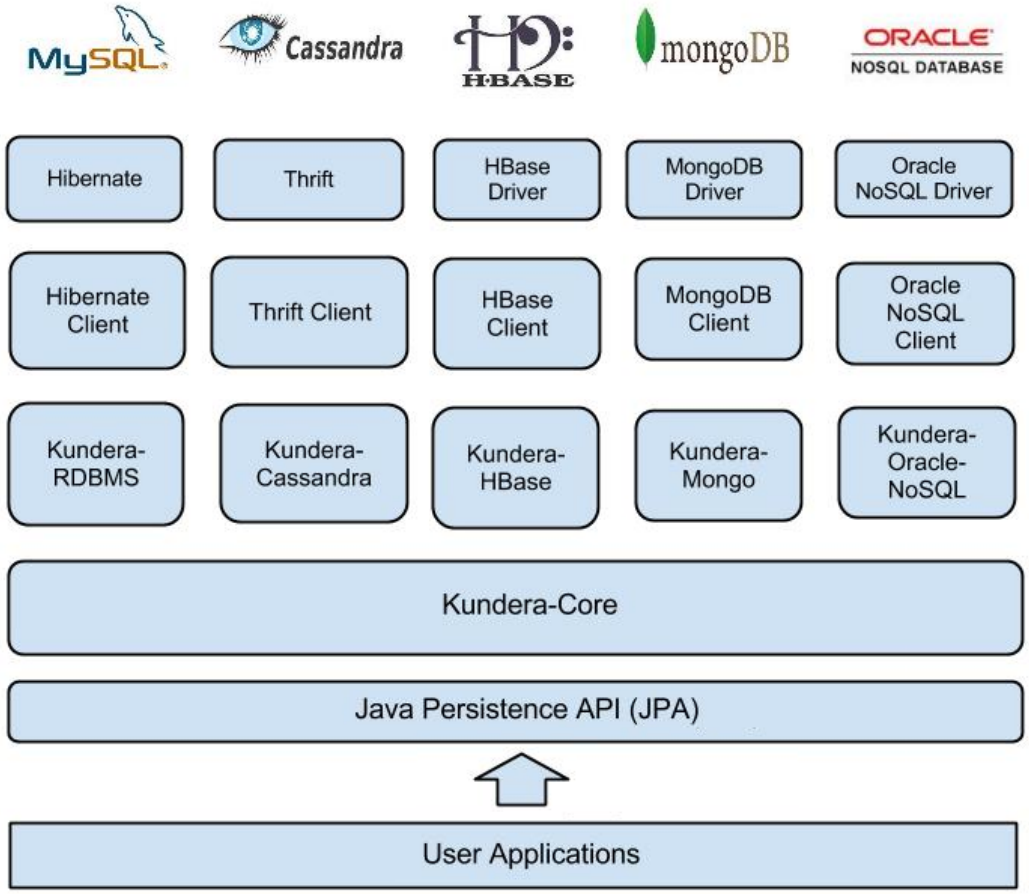
Kundera

A JPA 2.1 ORM Library for NoSQL databases

- ORM operation (through *EntityManager* interface)
- JPQL queries (DELETE and UPDATE)

On-premises databases:

- Cassandra
- HBase
- MongoDB
- Oracle NoSQL
- Redis
- Neo4j
- Couchdb
- Elastic Search
- MySQL

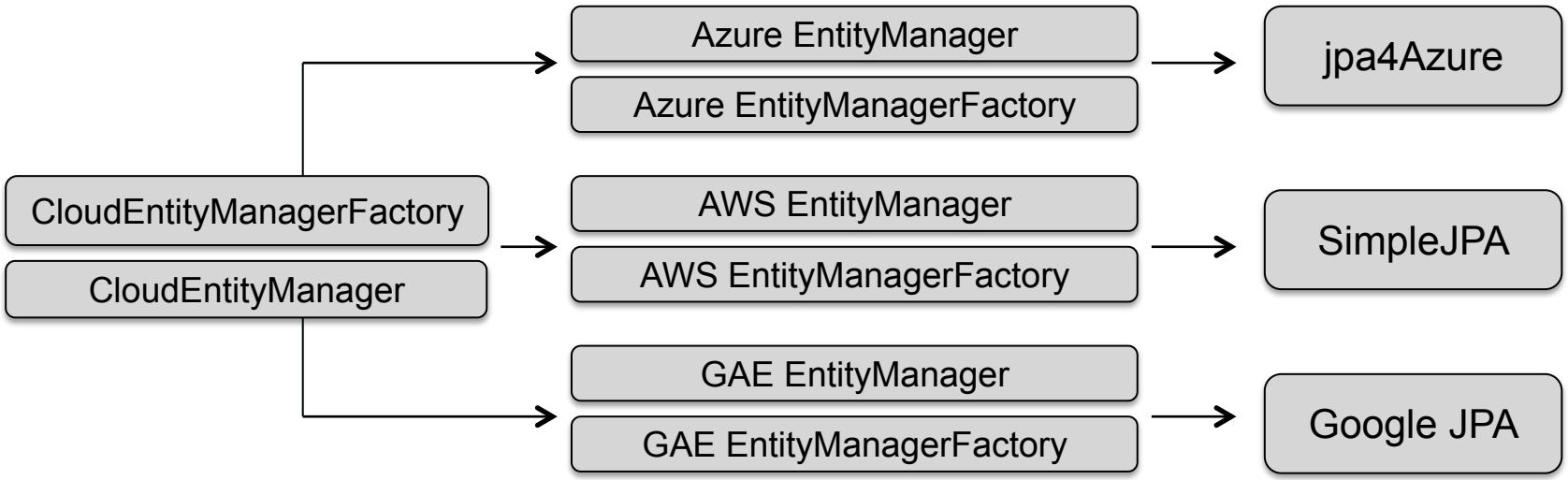


Why Kundera

- Open source
- Developed with extensibility as primary goal
- Support to many different NoSQL databases
- Polyglot persistency
- In the field since 2010 with an active community
- Already used in production

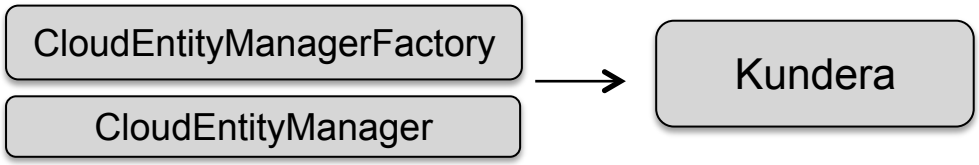
Original CPIM NoSQL service implementation

- Many JPA providers
- Duplicated code
- No complete code portability
- Choice of the NoSQL database strictly bounded to the cloud provider (e.g. App Engine → Datastore)
- Limited NoSQL databases support



Kundera integration

- Single persistence provider
- Complete code portability
- NoSQL support inherited by Kundera
- Easier Configuration through standard persistence.xml



Work objectives

Integrate Kundera in the CPIM library

Contribute to the open source project Kundera

Integrate the migration and synchronization system Hegira

Evaluation

Contributions to Kundera

Paradigm shift

- Off-premises databases → DaaS solutions
- Merged Bug fix Kundera deploy on PaaS

Open two newly developed clients

- Azure Tables[1]
- GAE Datastore[2]

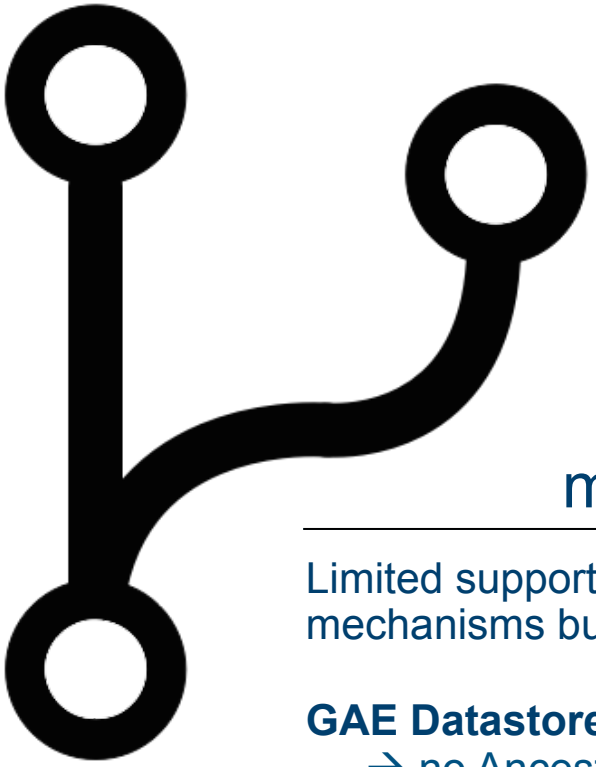
1: https://github.com/deib-polimi/kundera-azure-table
2: https://github.com/deib-polimi/kundera-gae-datastore

Developed clients

master

Exploit consistency mechanisms as much as possible

GAE Datastore
→ no Ancestor Path support
Azure Tables
→ manage partition key and row key



migration

Limited support to consistency mechanisms but achieve interoperability

GAE Datastore
→ no Ancestor Path support
Azure Tables
→ fix partition key to table name

Work objectives

Integrate Kundera in the CPIM library

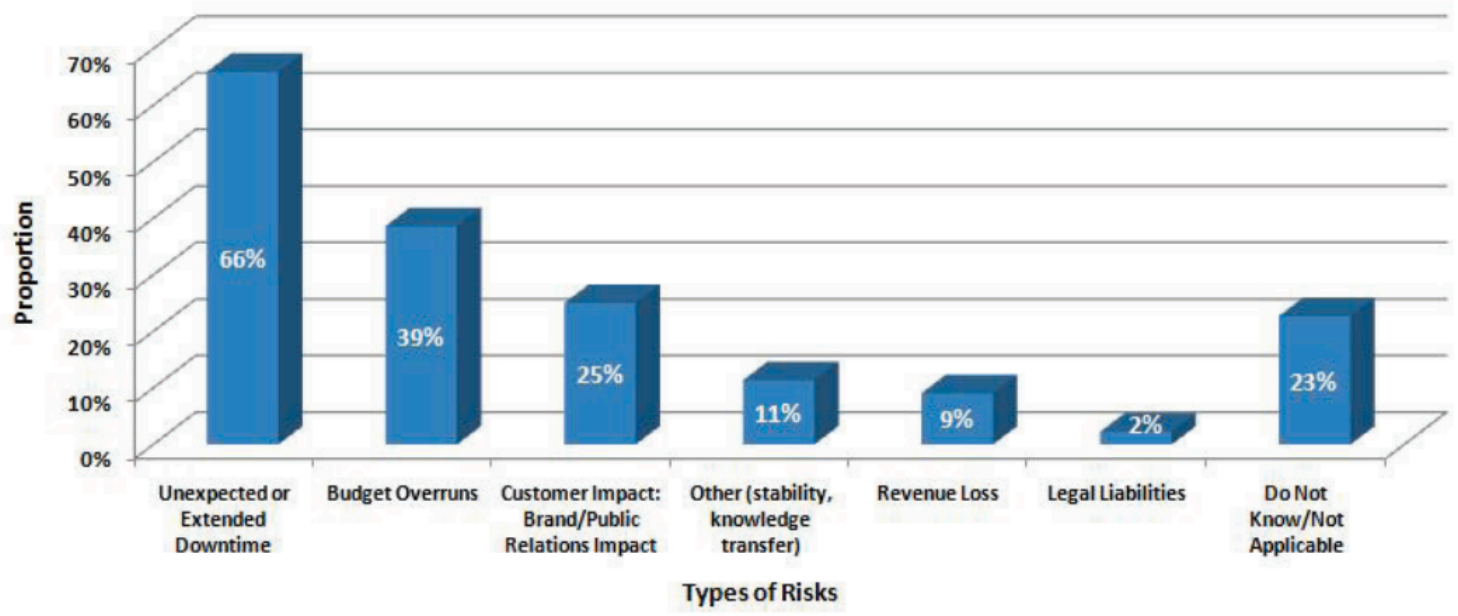
Contribute to the open source project Kundera

Integrate the migration and synchronization system Hegira

Evaluation

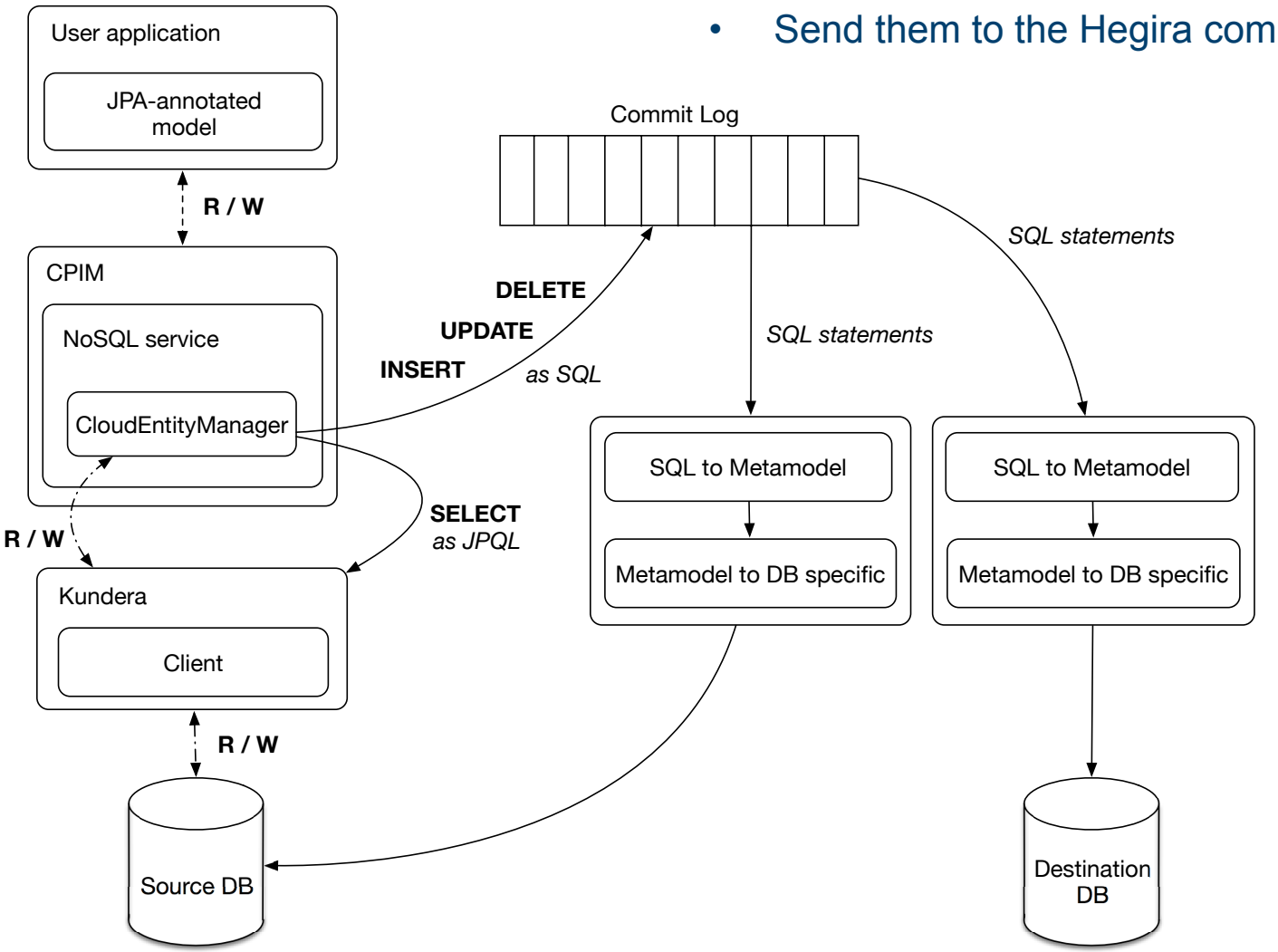
Data migration

- move application to another cloud provider
- move data to a database that better fit requirements
- load balancing, system expansion, failure recovery, costs, etc.
- modern computer systems are expected to be up continuously
- data synchronization between the two involved systems



Hegira support

- Intercept transparently user operations (DMQ)
- Translate operations to SQL statements
- Send them to the Hegira commit-log



Work objectives

Integrate Kundera in the CPIM library

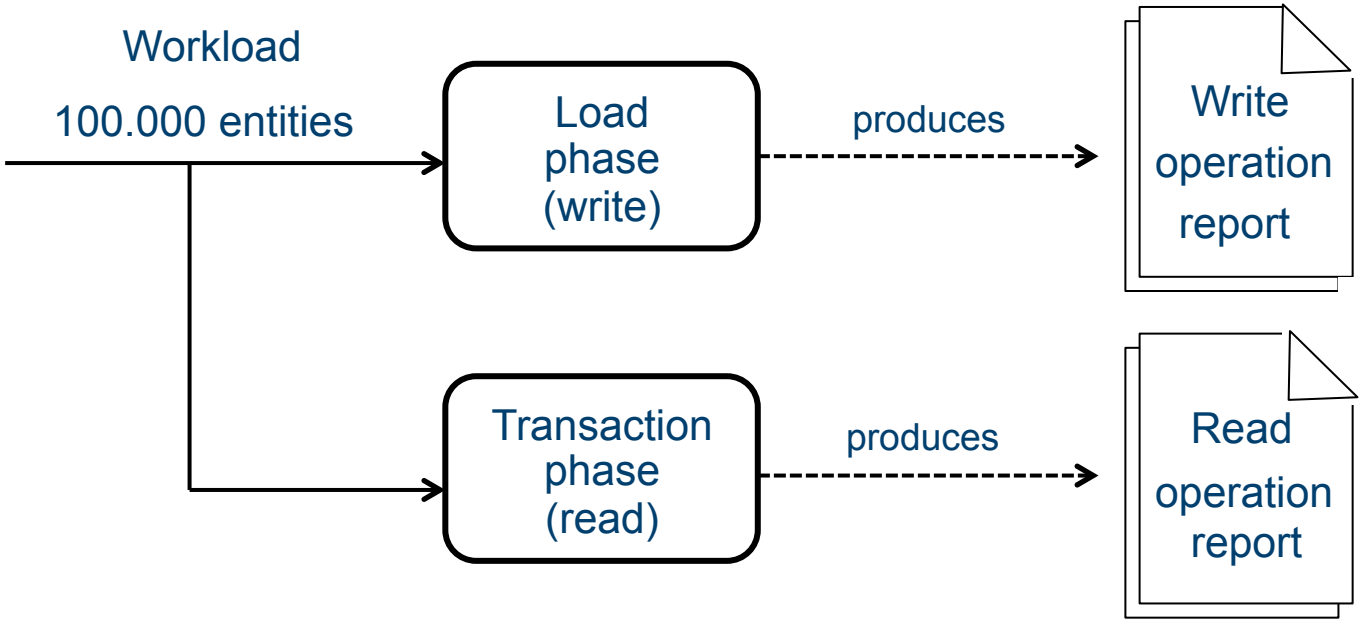
Contribute to the open source project Kundera

Integrate the migration and synchronization system Hegira

Evaluation

YAHOO! Cloud Serving Benchmark

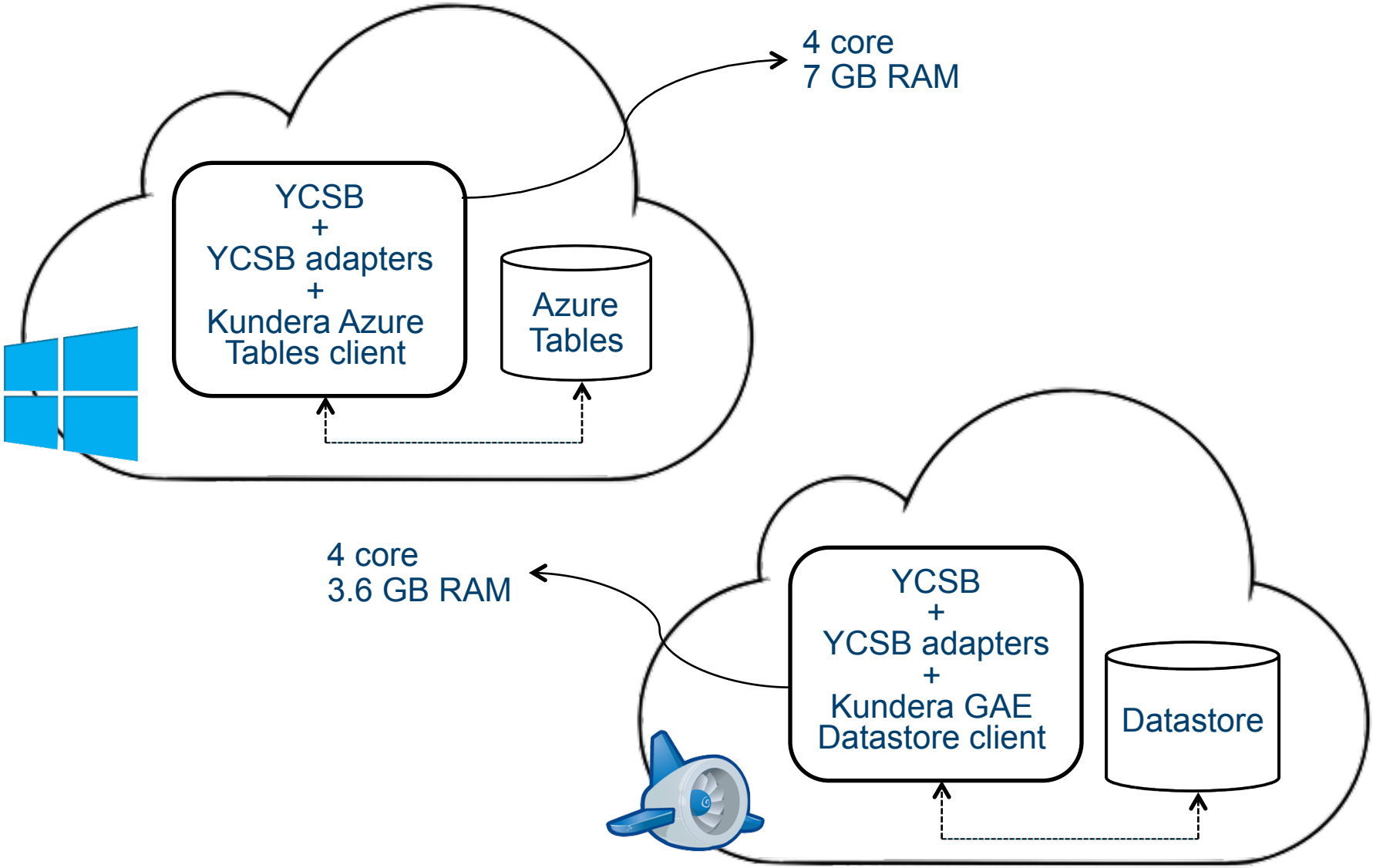
Framework for evaluating the performance of different NoSQL databases



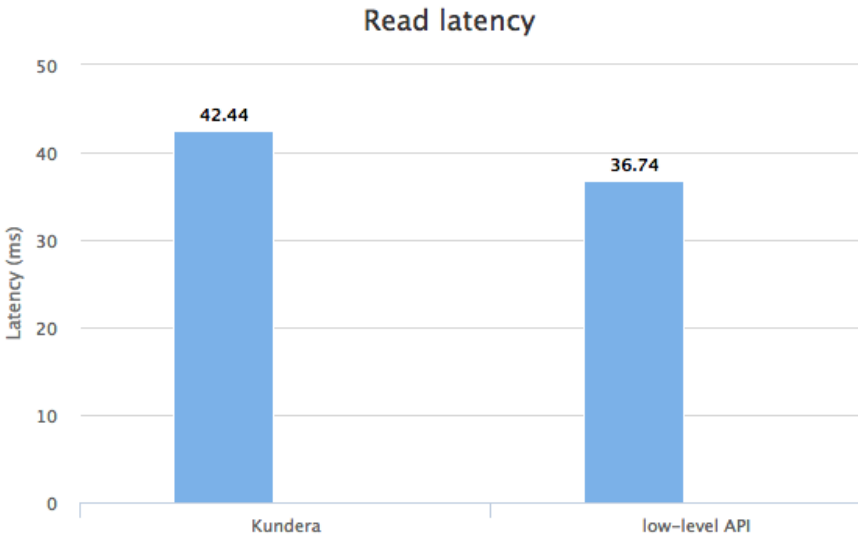
Compare Kundera client w.r.t. the use of low-level API for the same operations

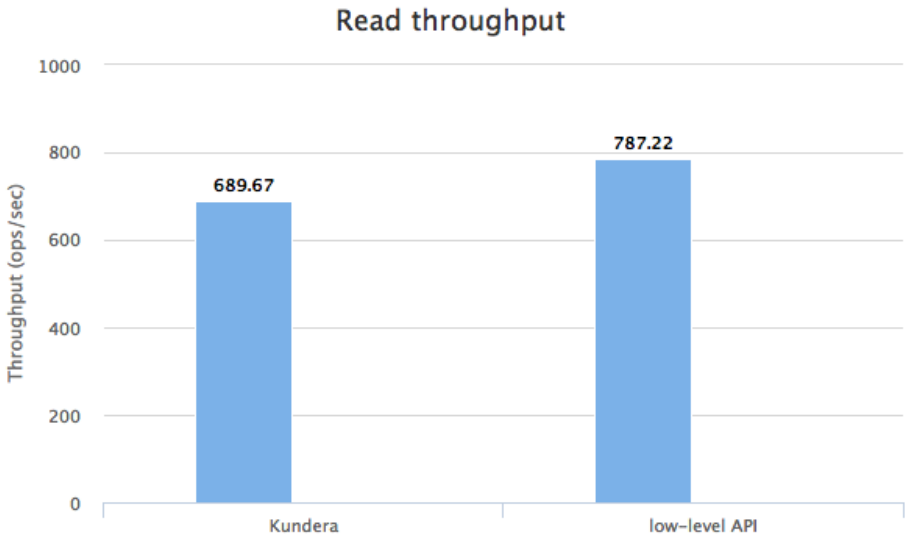
- Development of new adapter for operations through Kundera
- Development of new adapter for operations through the low-level API

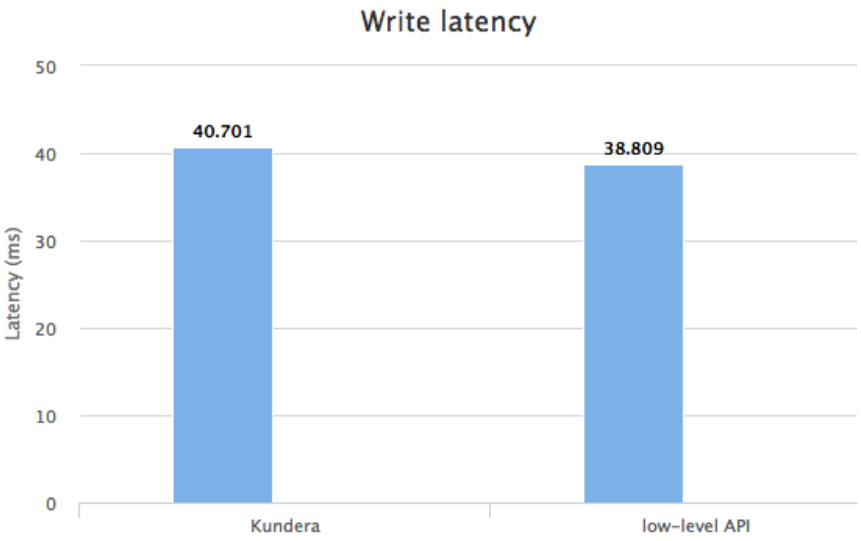
Environment setup

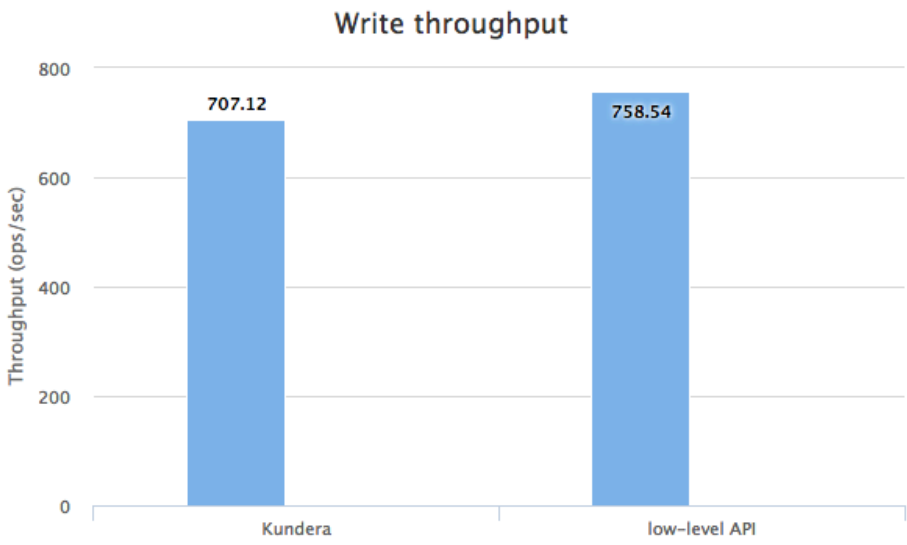


Results - Azure Tables

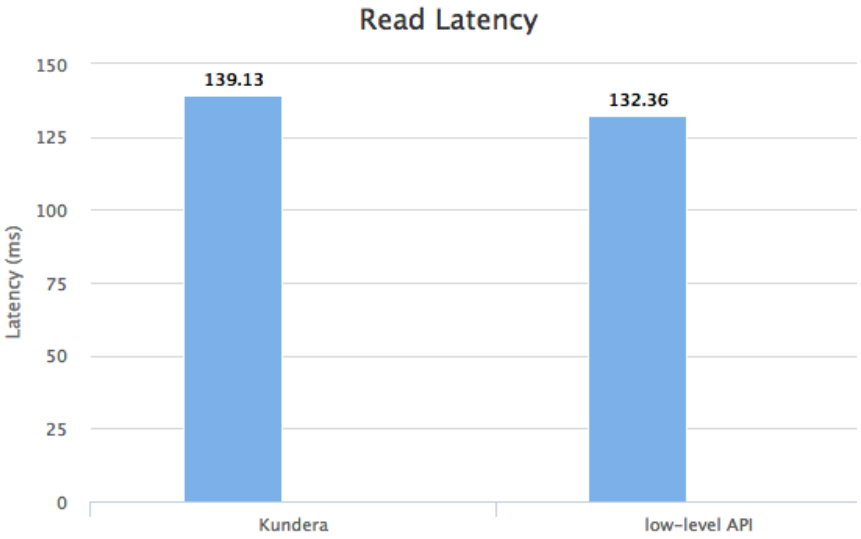


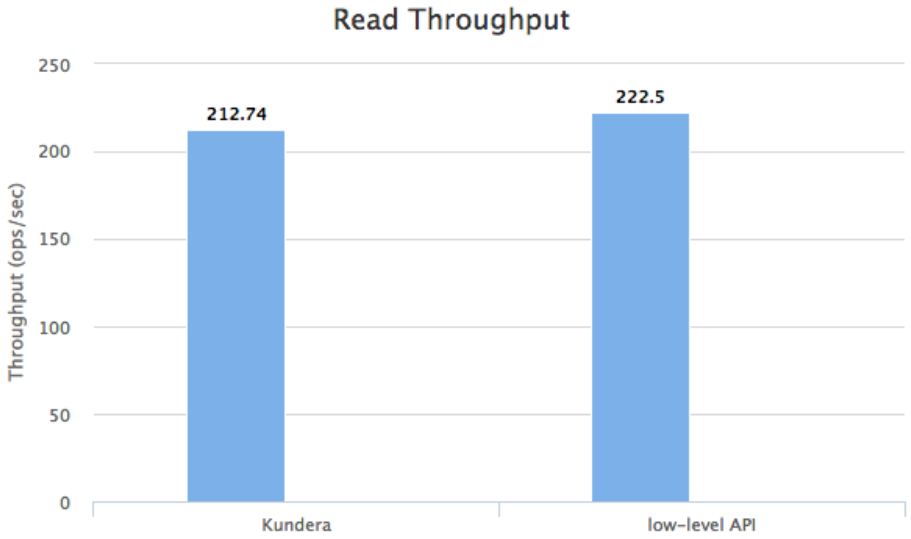


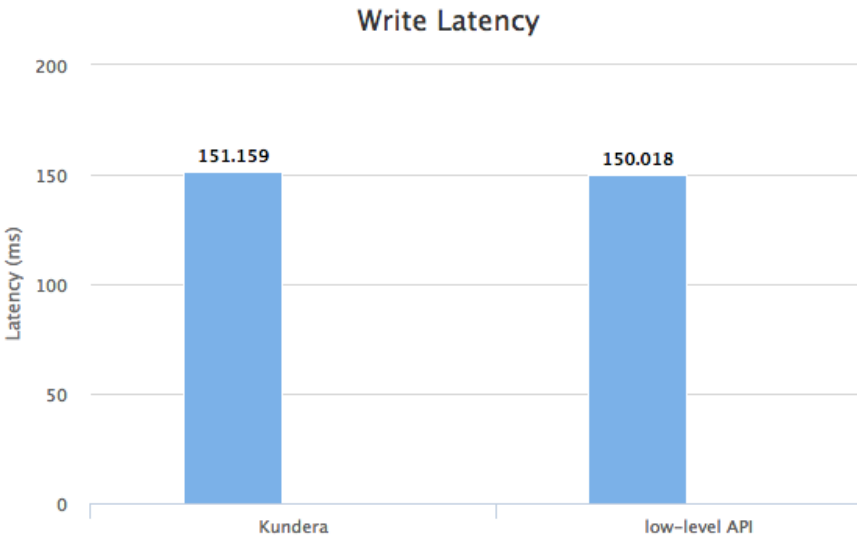


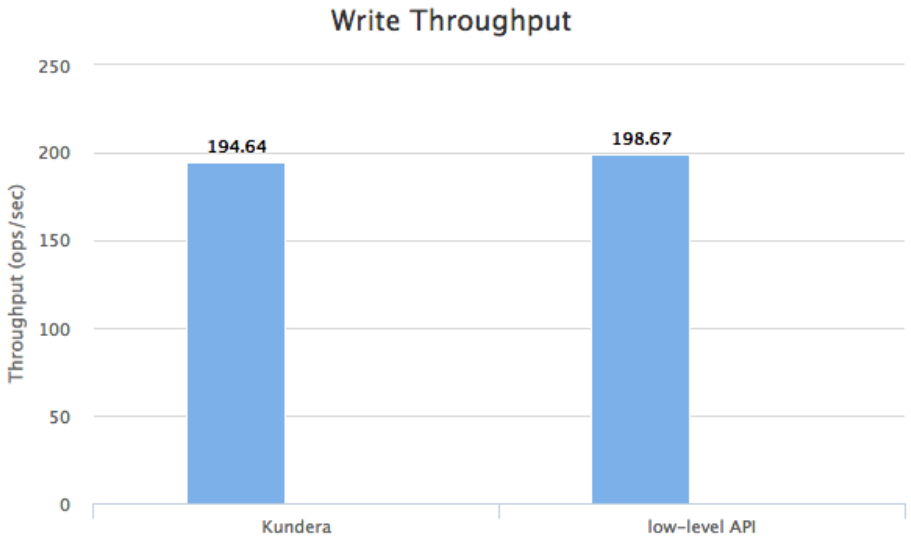


Results - GAE Datastore









Results comparison

Azure Tables

Kunedra overhead w.r.t low-level API

Read latency	Read throughput	Write latency	Write throughput
13,43 %	12,39 %	4,75 %	6,78 %

Google Datastore

Kundera overhead w.r.t low-level API

Read latency	Read throughput	Write latency	Write throughput
4,36 %	4,39 %	0,76 %	2,03 %

Conclusions

Contributions:

- Integration of Kundera in CPIM library
- New Kundera clients to support Google Datastore and Azure Tables
- Hegira integration in the CPIM library

Future work:

- Compare developed client performance with the ones of the other client developed by Kundera team

THANK YOU