Aprameyah
Technologies Pvt. Ltd.

CMAD Adv I

Aprameyah
Technologies Pvt. Ltd.

Polyglot Databases

- Understand Consistency Models - ACID/BASE
- Whats driving us to this
- Benefits
- Challenges
- Deciding Criteria

Aprameyah
Technologies Pvt. Ltd.

ACID/BASE

- Limits of instantaneous consistency
  - The balance between write performance and availability in instantaneous consistent systems
- What does Eventual Consistency get us
- CAP Theorem

Aprameyah
Technologies Pvt. Ltd.

Deciding Criteria

- Write performance
- Read performance
- Availability
- Consistency
- Partition Tolerance
- Query patterns
  - Text search, graph search

Aprameyah
Technologies Pvt. Ltd.

Types of DBs

- Mongo
  - Tuneable consistency, Availability, Partition Tolerance.
  - Indexed Document Store
  - Master slave model with writes only to master - Easy scalability for reads
- Neo4j
  - Graph database tuned for graph questions
  - Consistent and Available. Billions of nodes and relationships

Aprameyah
Technologies Pvt. Ltd.

Types of DB

- Cassandra
  - Column store with multi master ability. Closest to SQL
  - Available and Partition Tolerant
  - Easy scalability for writes
- Redis in-memory key-value store

Aprameyah
Technologies Pvt. Ltd.

Question of Durability

- What happens to backups
- How facebook handles redundancy and durability

Aprameyah
Technologies Pvt. Ltd.

Threading and Locking

- Understand thread racing
- Solving thread racing with locks
- Using atomic variables

Aprameyah
Technologies Pvt. Ltd.

Shared Collections

- Using shared collections
  - Write and read operations. Try multiple thread read and write into a synchronized map
  - Compare it with concurrent map implementation
- Fail fast and snapshot iterators
- CopyOnWriteArrayLists

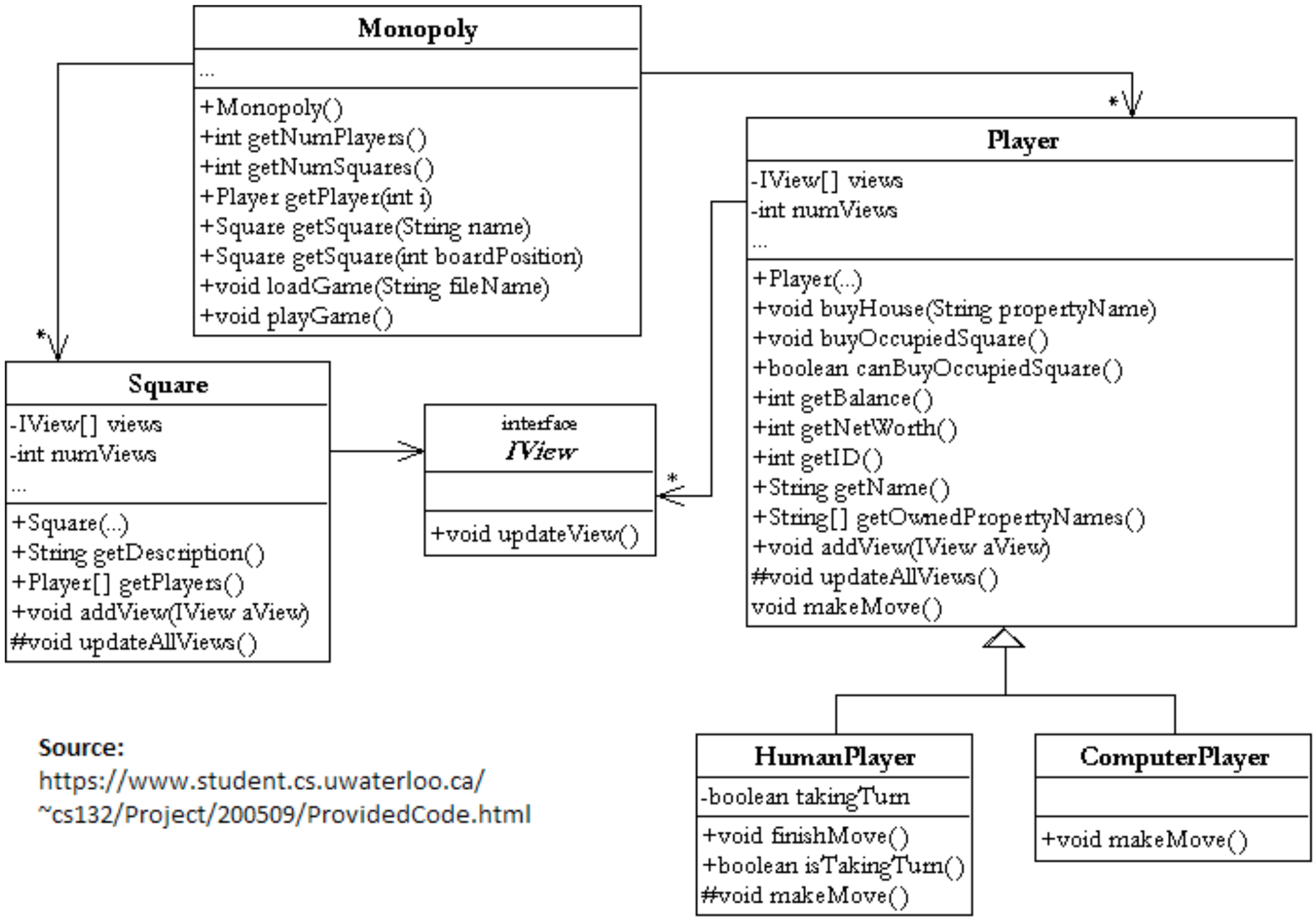
Aprameyah
Technologies Pvt. Ltd.

I/O Performance

- Java IO library read operation
- Asynchronous Read using NIO
- Measure performance in multi threaded environment

Aprameyah
Technologies Pvt. Ltd.

Traditional OO Design



Source:
https://www.student.cs.uwaterloo.ca/~cs132/Project/200509/ProvidedCode.html

Aprameyah
Technologies Pvt. Ltd.

Stateless OO Design Classes

- Model Classes - No patterns, just DTOs
- Business Classes - Design patterns apply here
- Technology Classes
- Utility Classes

Aprameyah
Technologies Pvt. Ltd.

Vert.x

- We can build reactive, non blocking, event driven apps
- Polyglot - javascript, java, groovy and ruby
- Not an app server, modular
- Good fit for microservices

Aprameyah
Technologies Pvt. Ltd.

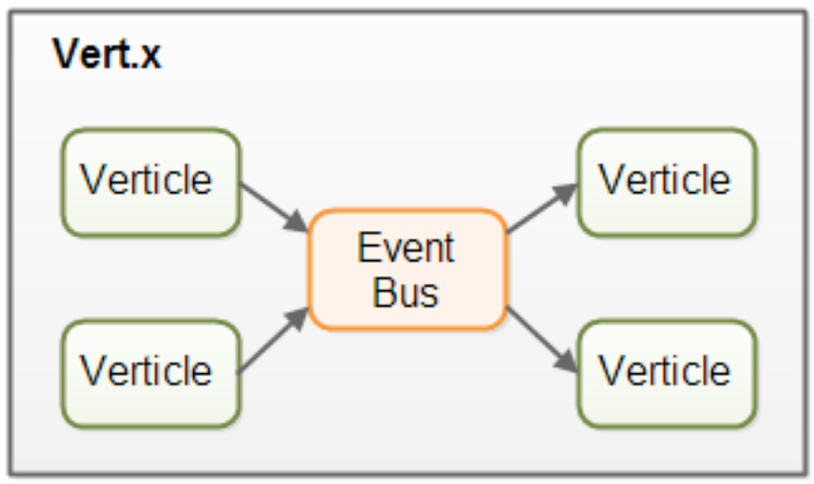
Vert.x Performance

Best plaintext responses per second, i7-2600K hardware (55 tests)

Framework	Best performance (higher is better)	Cls	Lng	Plt	FE	Aos	IA	Errors
vertx	656,119 100.0%	Plt	Jav	vtx	Non	Lin	Rea	0
netty	632,596 96.4%	Plt	Jav	Nty	Non	Lin	Rea	10
undertow	614,547 93.7%	Plt	Jav	Utw	Non	Lin	Rea	70
plain-windows	611,095 93.1%	Ful	Sca	Pla	Non	Win	Rea	52
plain	543,670 82.9%	Ful	Sca	Pla	Non	Lin	Rea	21,473
cpoll_cppsp	522,423 79.6%	Mcr	C++	Cpl	Non	Lin	Rea	0
plain-servlet-linux	450,462 68.7%	Ful	Sca	Pla	Non	Lin	Rea	3,406
jetty-servlet	448,947 68.4%	Plt	Jav	Jty	Jty	Lin	Rea	351
grizzly	425,172 64.8%	Mcr	Jav	Svt	Grz	Lin	Rea	149
gemini	424,586 64.7%	Ful	Jav	Svt	Non	Lin	Rea	2,071
spray	422,431 64.4%	Mcr	Sca	Spr	Non	Lin	Rea	1,003
servlet	417,619 63.6%	Plt	Jav	Svt	Res	Lin	Rea	1,214
go	367,274 56.0%	Plt	Go	Go	Non	Lin	Rea	230
openresty	330,829 50.4%	Plt	Lua	OpR	ngx	Lin	Rea	1,517
spark	226,365 34.5%	Mcr	Jav	Svt	Res	Lin	Rea	0
revel	221,603 33.8%	Ful	Go	Go	Non	Lin	Rea	521
falcore	172,827 26.3%	Mcr	Go	Go	Non	Lin	Rea	0
compojure	127,671 19.5%	Mcr	Clj	Svt	Res	Lin	Rea	0
http-listener	123,434 18.8%	Plt	C#	Net	hts	Win	Rea	0
falcon-pypy	112,550 17.2%	Mcr	Py	Gun	Non	Lin	Rea	0
wsgi-nginx-uwsgi	110,522 16.8%	Plt	Py	uWS	ngx	Lin	Rea	459,212
evhttp-sharp	107,494 16.4%	Mcr	C#	Mon	Non	Lin	Rea	0
bottle-pypy	88,328 13.5%	Mcr	Py	Tor	Non	Lin	Rea	52
nodejs	80,363 12.2%	Plt	JS	njs	Non	Lin	Rea	0

Verticles

- Vert.x can deploy and execute components called Verticles.
- You can think of verticles as being similar to servlets or message driven EJBs driven by an event bus



Aprameyah
Technologies Pvt. Ltd.

Verticle Messaging

- Messages can be simple objects (e.g. Java objects), strings, CSV, JSON, binary data or whatever else you need

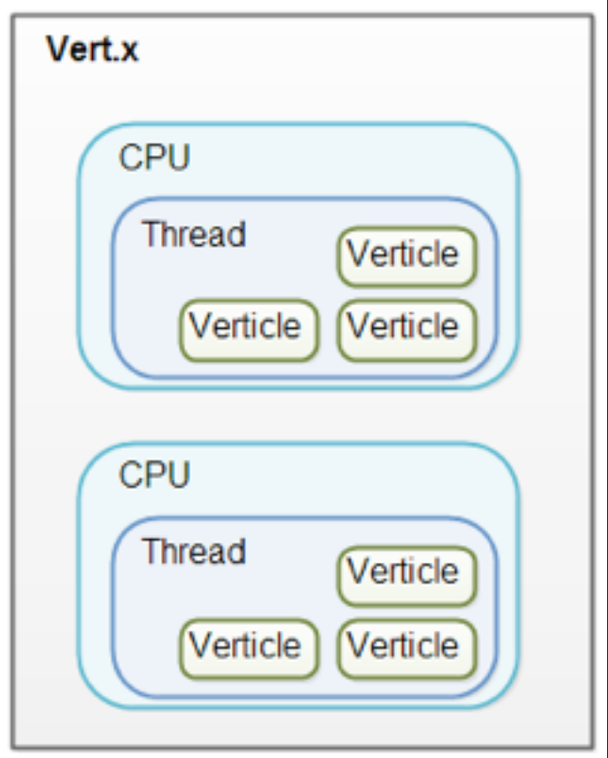
- Verticles can send and listen to addresses. An address is like a named channel.

- When a message is sent to a given address, all verticles that listen on that address receive the message.

- Verticles can subscribe and unsubscribe to addresses without the senders knowing.

Threading Model

- verticle is only ever executed by a single thread, and always by the same thread.
- A single thread can distribute messages to multiple verticles.
- Vert.x creates one thread per CPU
- Vert.x comes with a set of built-in services (functionality). Some of these services are:
  - HTTP server
  - JDBC connector
  - MongoDB connector



Number of Instances

- Deploy multiple verticle instances to run in different threads

- DeploymentOptions options = new DeploymentOptions().setInstances(16);

Aprameyah
Technologies Pvt. Ltd.

Creating Verticles

```
public class MyVerticle extends AbstractVerticle {

    @Override
    public void start(Future<Void> startFuture) {
        System.out.println("MyVerticle started!");
    }

    @Override
    public void stop(Future stopFuture) throws Exception {
        System.out.println("MyVerticle stopped!");
    }

}

public static void main(String[] args) {
      VertxOptions options = new VertxOptions().setWorkerPoolSize(10);
      Vertx vertx = Vertx.vertx(options);
      vertx.deployVerticle("com.mydomain.MyVerticle");
}
```

Verticle Events

- The start() method : start HTTP or TCP server, register event handlers on the event bus, deploy other verticles, or whatever else your verticle needs to do its work.

- Shutdown stuff in stop method

- The verticle will be deployed asynchronously

```
vertx.deployVerticle("com.mydomain.MyVerticle",new
Handler<AsyncResult<String>>() {
        @Override
        public void handle(AsyncResult<String> stringAsyncResult) {
            System.out.println("Verticle deployment complete");
            }
        });
```

Registering to Events

- When a verticle wants to listen for messages from the event bus, it listens on a certain address. An address is just a name (a String) which you can choose freely.

- An address is thus more like the name of a channel with multiple receivers

```
vertx.eventBus().consumer("Channel1", message -> {
        System.out.println("message.body() = "
                + message.body());
});
vertx.eventBus().publish("Channel1", "message 2");
```

Aprameyah
Technologies Pvt. Ltd.

Types Of Verticles

- Standard Verticles
    - These are the most common and useful type - they are always executed using an event loop thread.
- Worker Verticles
    - These run using a thread from the worker pool. An instance is never executed concurrently by more than one thread.

```
DeploymentOptions options = new DeploymentOptions().setWorker(true);

vertx.deployVerticle("com.mycompany.MyOrderProcessorVerticle", options);
```

- Multi-threaded worker verticles
    - These run using a thread from the worker pool. An instance can be executed concurrently by more than one thread.

Aprameyah
Technologies Pvt. Ltd.

Vert.x buffers

- Carry Binary Information in buffers. Dynamically resizable

- Can be used as message payloads

```
byte[] initialData = new byte[]{1, 2, 3};

Buffer buffer      = Buffer.buffer(initialData);
buffer.setShort ( 10, (short) 127);
buffer.appendByte  ((byte)  127);
```

Aprameyah
Technologies Pvt. Ltd.

Running Blocking Code

- When we HAVE to invoke synchronous APIs, vertx provides a way to do that:

```
vertx.executeBlocking(future -> {
      // Call some blocking API that takes a significant amount of time
to return
      String result = someAPI.blockingMethod("hello");
      future.complete(result);
   }, res -> {
     System.out.println("The result is: " + res.result());
});
```

Aprameyah
Technologies Pvt. Ltd.

Mongodb

- Mongodb is an indexed document store.
- A document is typically something like a json structure
- Can be indexed based on any of the fields
- Can be replicated, shraded, tunable consistency with single master and leader election.

- db.users.insert({name:'Hari',email:'hari@abc.com'})
- show collections
- db.users.update({age: { $gt: 18}},{$set: {status:'A'}}, {multi: true})
- db.users.remove({status: 'D'})

Aprameyah
Technologies Pvt. Ltd.

Mongodb Queries

```
db.users.find(                                  <-- collection
   { age: { $gt: 18 } },                        <-- query criteria
   { name: 1, address: 1 }                      <-- projection
).limit(5)                                      <-- cursor modifier
```

```
SELECT _id, name, address   <-- projection
FROM   users                <-- table
WHERE  age > 18             <-- select criteria
LIMIT  5                    <-- cursor modifier
```

Aprameyah
Technologies Pvt. Ltd.

Creating Index

- db.users.createIndex({ email: 1 })
- db.users.createIndex({ email: 1, name: 1 })
- Fully covered queries:
  - db.users.find { email: /.*yahoo.com/},{ name: 1, _id: 0 })

Aprameyah
Technologies Pvt. Ltd.

Integrating With Java

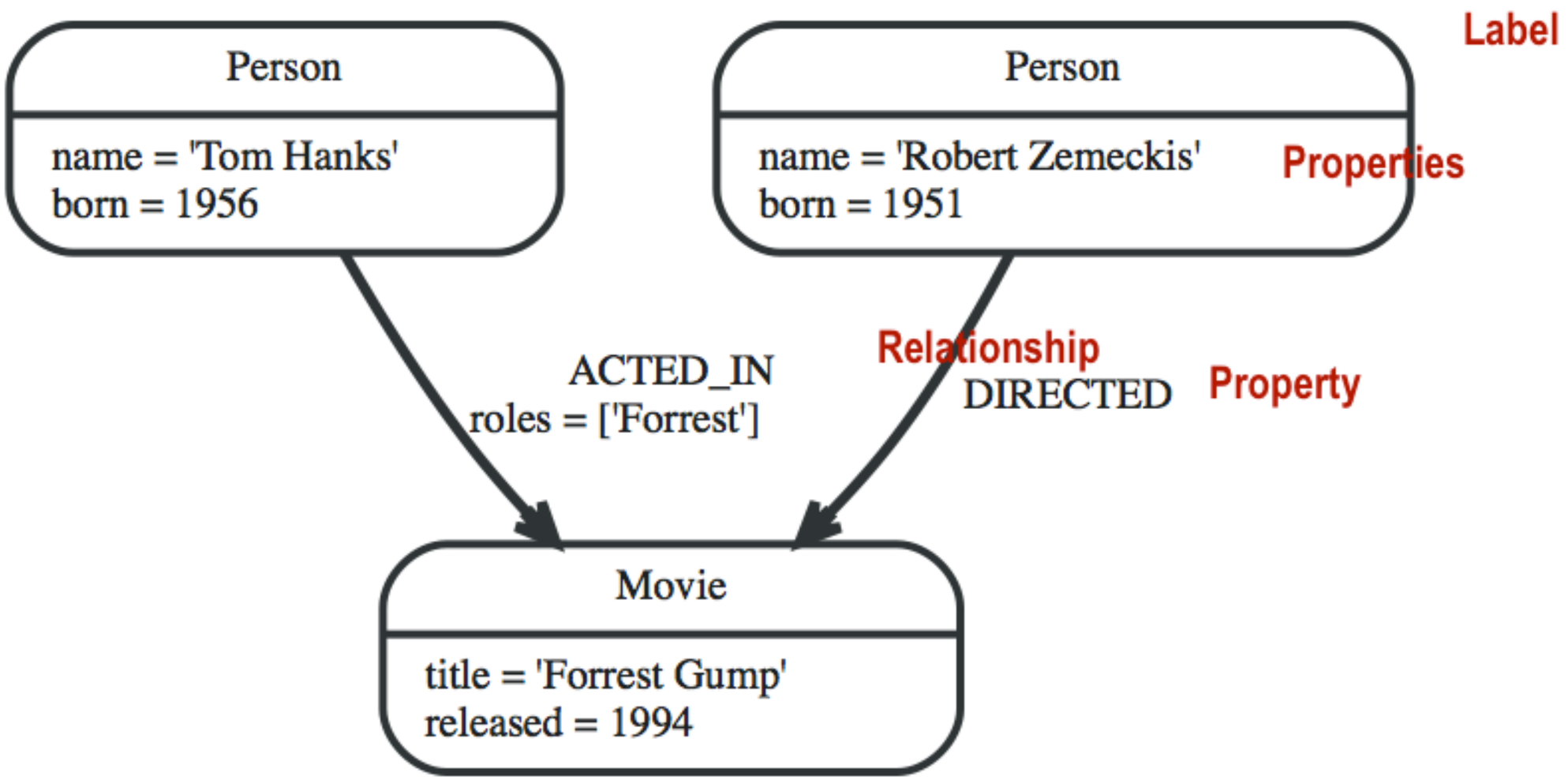
- POM dependencies

Aprameyah
Technologies Pvt. Ltd.

Neo4j

- ACID transactions,
- High availability,
- Scales to billions of nodes and relationships,
- High speed querying through traversals,
- Declarative graph query language.

Graph


Person
name = 'Tom Hanks'
born = 1956
Person
name = 'Robert Zemeckis'
born = 1951
Label
Properties
ACTED_IN
roles = ['Forrest']
Relationship
DIRECTED
Property
Movie
title = 'Forrest Gump'
released = 1994

Aprameyah
Technologies Pvt. Ltd.

Features

- SQL Like easy query language Neo4j CQL
- It supports Indexes by using Apache Lucence
- It supports UNIQUE constraints
- It contains a UI to execute CQL Commands : Neo4j Data Browser
- It supports full ACID
- It uses Native graph storage with Native GPE(Graph Processing Engine)
- It supports exporting of query data to JSON and XLS format

Aprameyah
Technologies Pvt. Ltd.

Features

- It provides REST API for the data
- It supports two kinds of Java API: Cypher API and Native Java API to develop Java applications.

Aprameyah
Technologies Pvt. Ltd.

CQL

- create (blog:Blog) - node name blog, label Blog
- create (:Blog{title:'Some blog',desc:'This is some blog'}) - Create with properties
- match(b:Blog) return b - select * from blog b
- MATCH (b:Blog),(u:User) CREATE (b)-[r:AUTHORED_BY]->(u)
    - Form relationship between EVERY blog and EVERY User

Aprameyah
Technologies Pvt. Ltd.

CQL

- MATCH (u:User{name:'Faizal'}) CREATE (blog:Blog{title:'related blog',desc:'This is some related blog'})-[r:AUTHORED_BY{when:'5th Sept'}]->(u)
  - Create an object with a relationship
- match(b:Blog)-[a:AUTHORED_BY]-(u:User{name:'Faizal'}) return a.when
  - Find all dates when Faizal authored blogs
- MATCH (b:Blog) WHERE b.title =~ '.*Some.*' RETURN b

Aprameyah
Technologies Pvt. Ltd.

CQL

- MATCH (b:Blog)-[rel]-(u:User) DELETE rel
  - Delete only the relationships
- MATCH (u:User{name:’Faizal'}) SET u.sex = ‘Male' RETURN u
  - Set a field and return the node

Java API support

- Neo4J supports two types of Java APIs
  - Native Java API
  - CQL API