swiftmind

Daten natürlich modelieren und verarbeiten mit Neo4j

SBB IT Developer Day 2012 | BEA Bern Expo | 11.12.2012

Agenda

- Speaker Profil
- Einführung
- Usecases
- Neo4j
- Querying
- Import / Export
- Tools und APIs
- Polyglot Persistence
- Highlights & Challenges beim Einsatz
- Q&A

Speaker Profile

Patrick Baumgartner

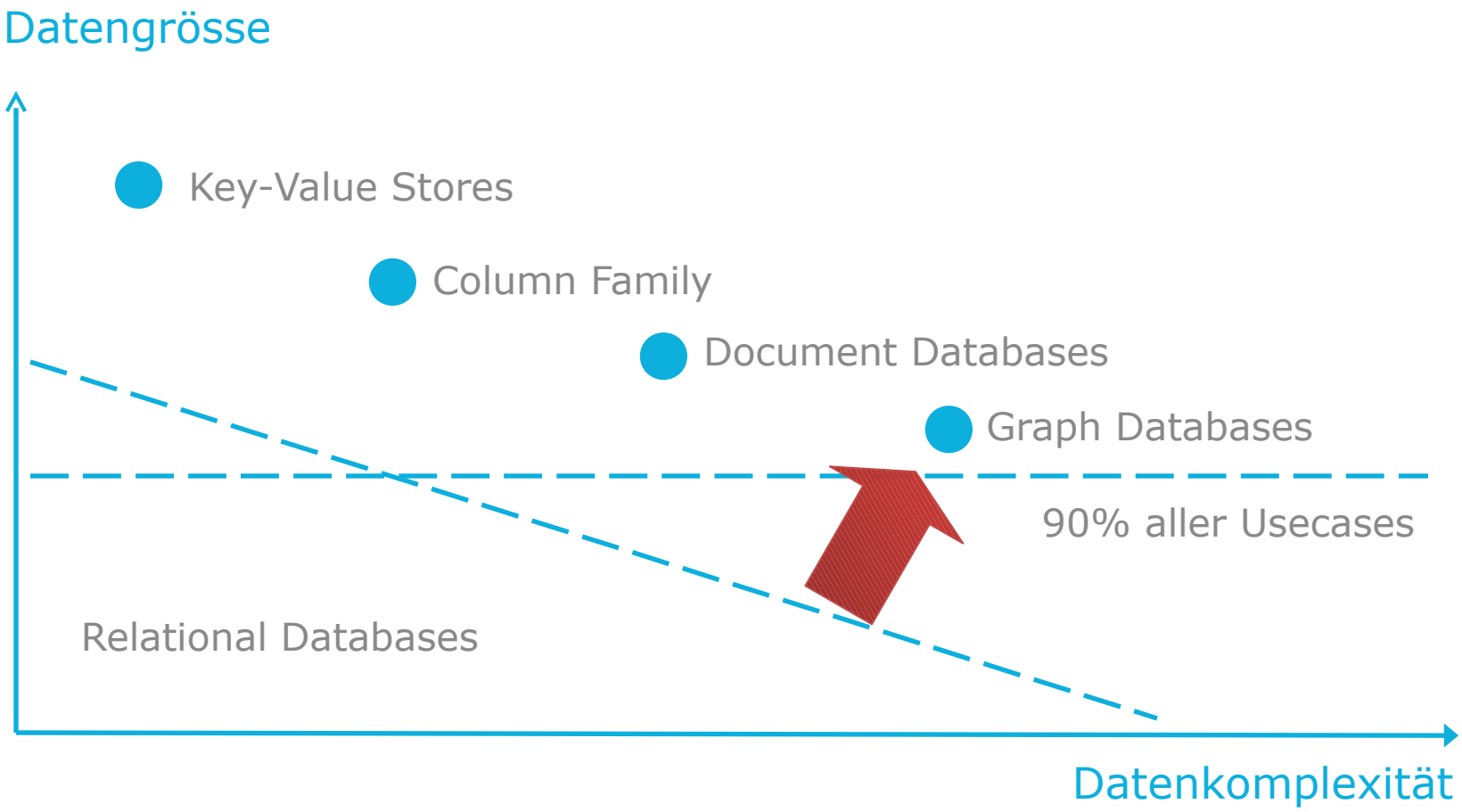
- Senior Software Consultant | Partner
- VMware/SpringSource Certified Instructor (Spring Trainer)
- Spring Framework, OSGi & Agile Engineering Practices
- Co-Autor von „OSGi für Praktiker“ (Hanser)

Swiftmind GmbH http://www.swiftmind.com

- Enterprise Java, Spring & OSGi Consulting
- Spring & OSGi Workshops & Trainings
- Agile Engineering Practices Coaching

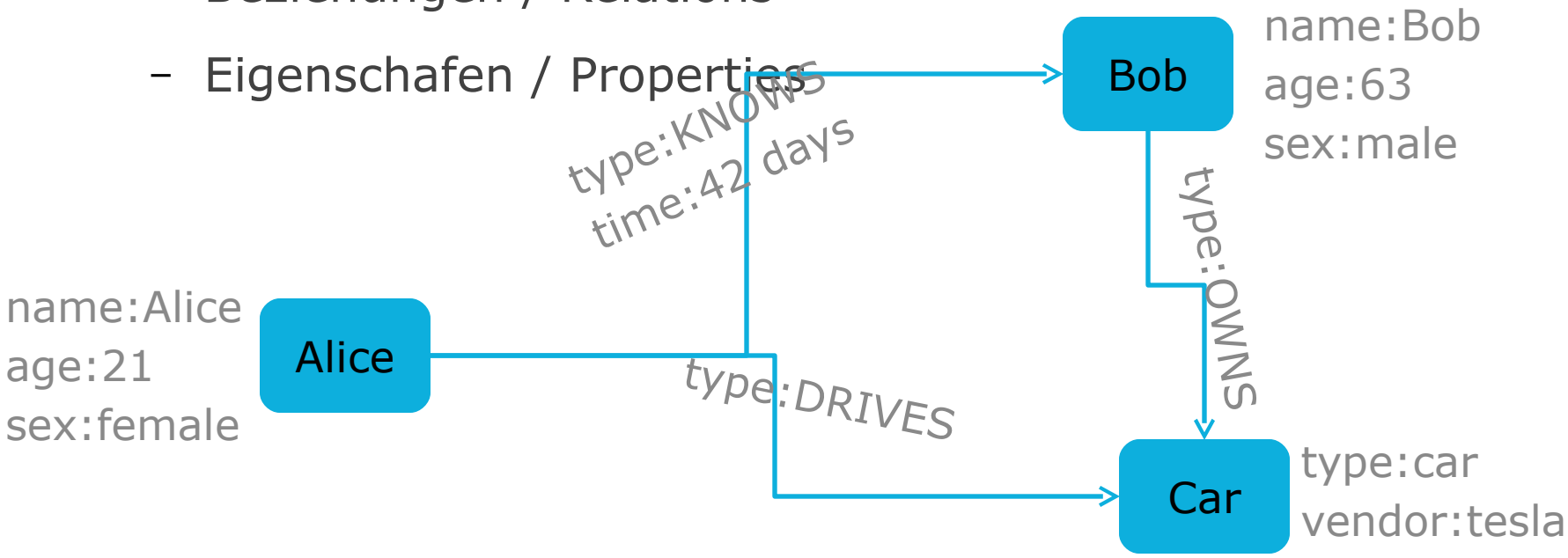
New Types of Data Stores

NoSQL


Datengrösse
Key-Value Stores
Column Family
Document Databases
Graph Databases
90% aller Usecases
Relational Databases
Datenkomplexität

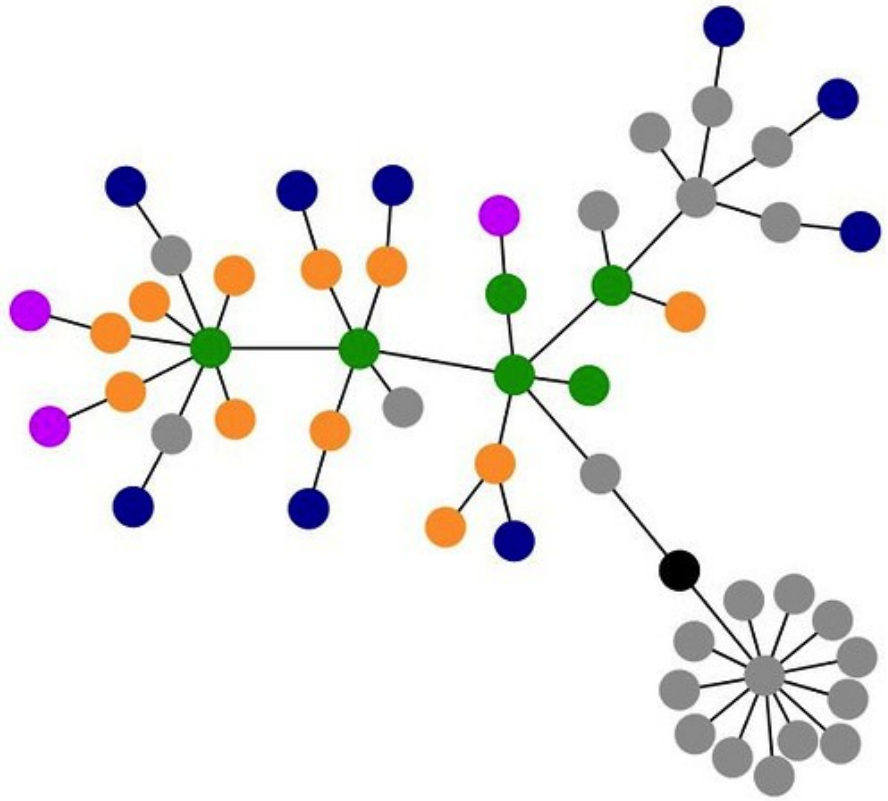
Property Graph Modell

- Graphen bestehen aus
  - Knoten / Nodes
  - Beziehungen / Relations
  - Eigenschafen / Properties

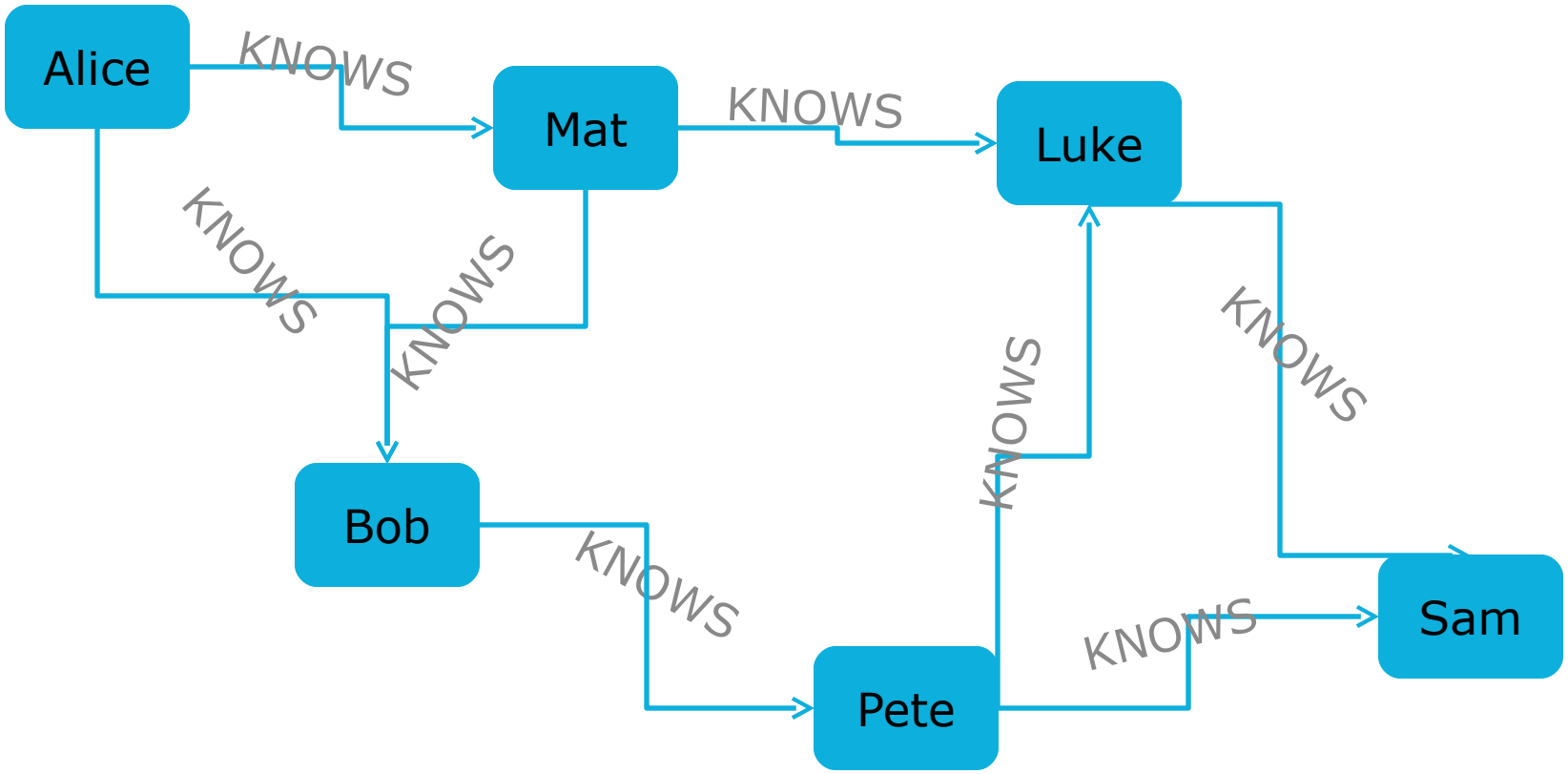


- Usecases
- Wo finde ich meinen Graphen?

Drag picture to



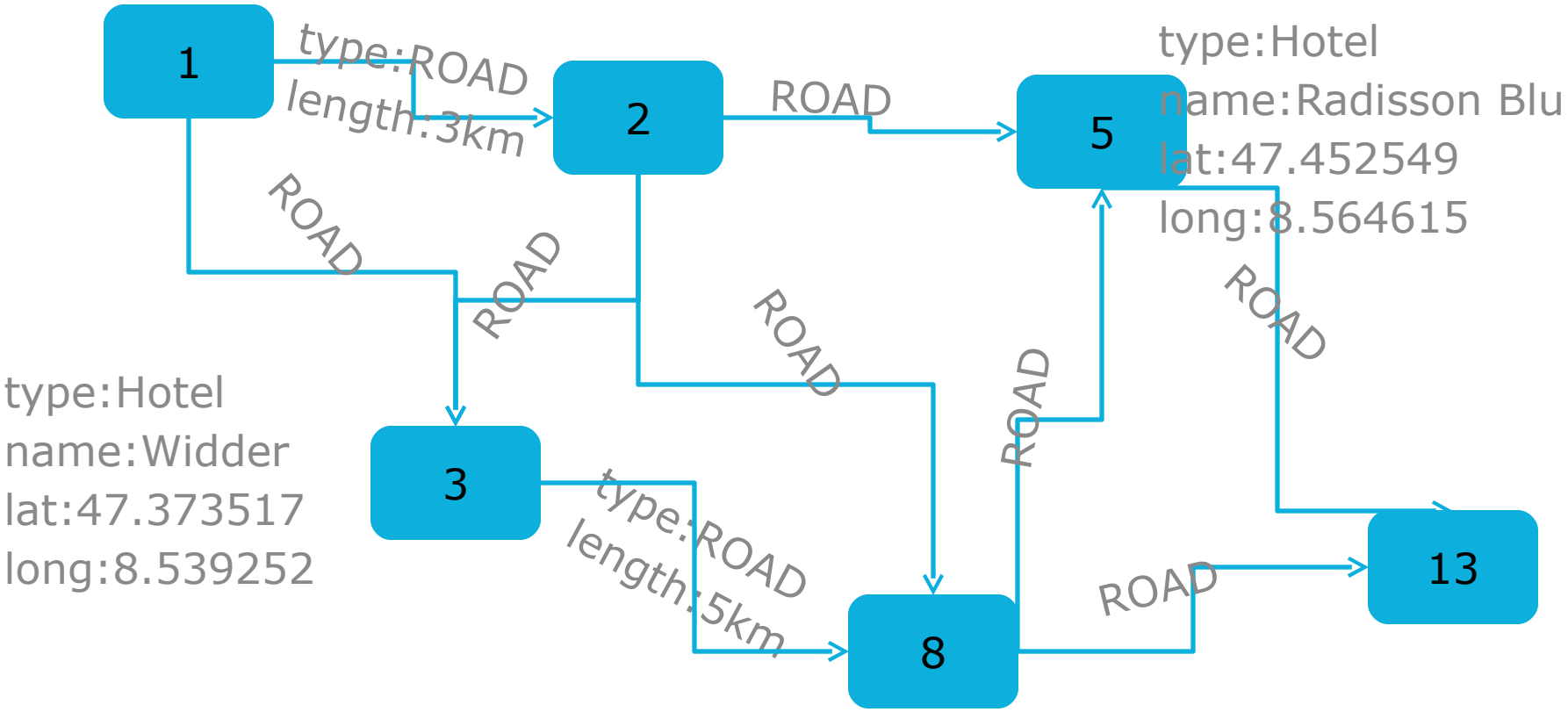
Social Data


Alice
KNOWS
Mat
KNOWS
Luke
KNOWS
KNOWS
KNOWS
KNOWS
KNOWS
Bob
KNOWS
Pete
KNOWS
Sam

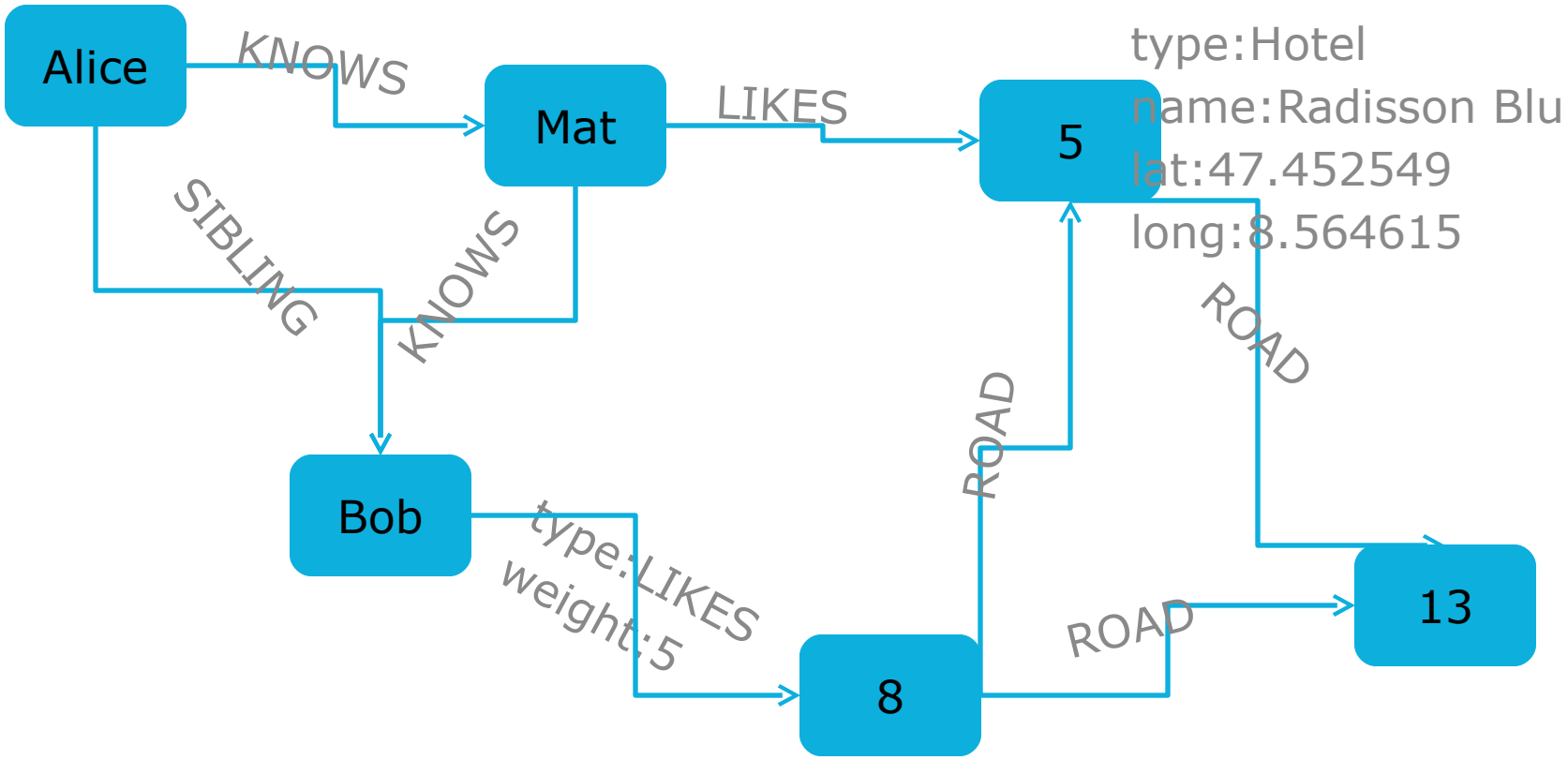
- Social Graph
- Ein einziger Anwendungsfall?

Drag picture to placeholder or click icon to add

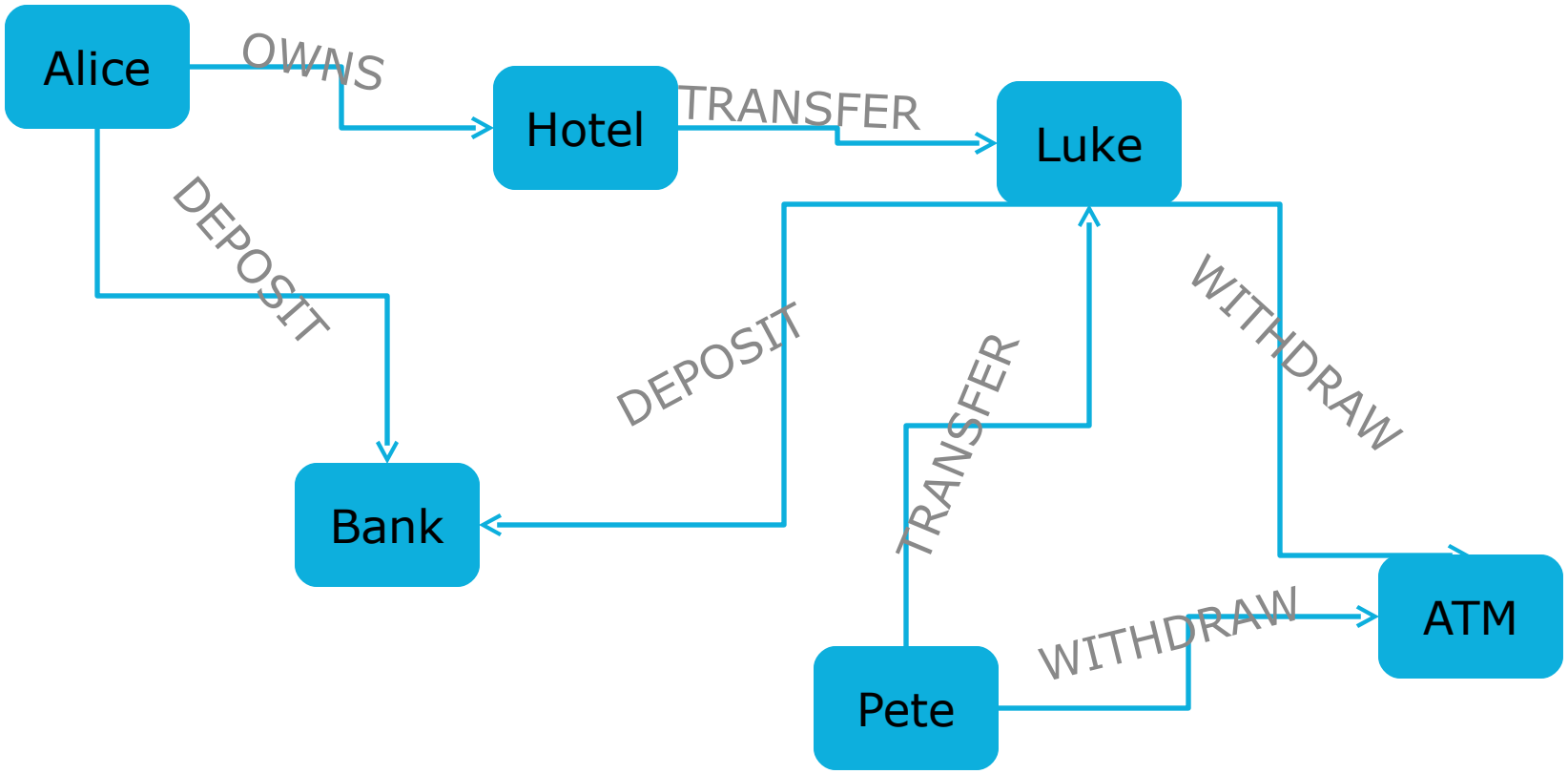
Spatial Data



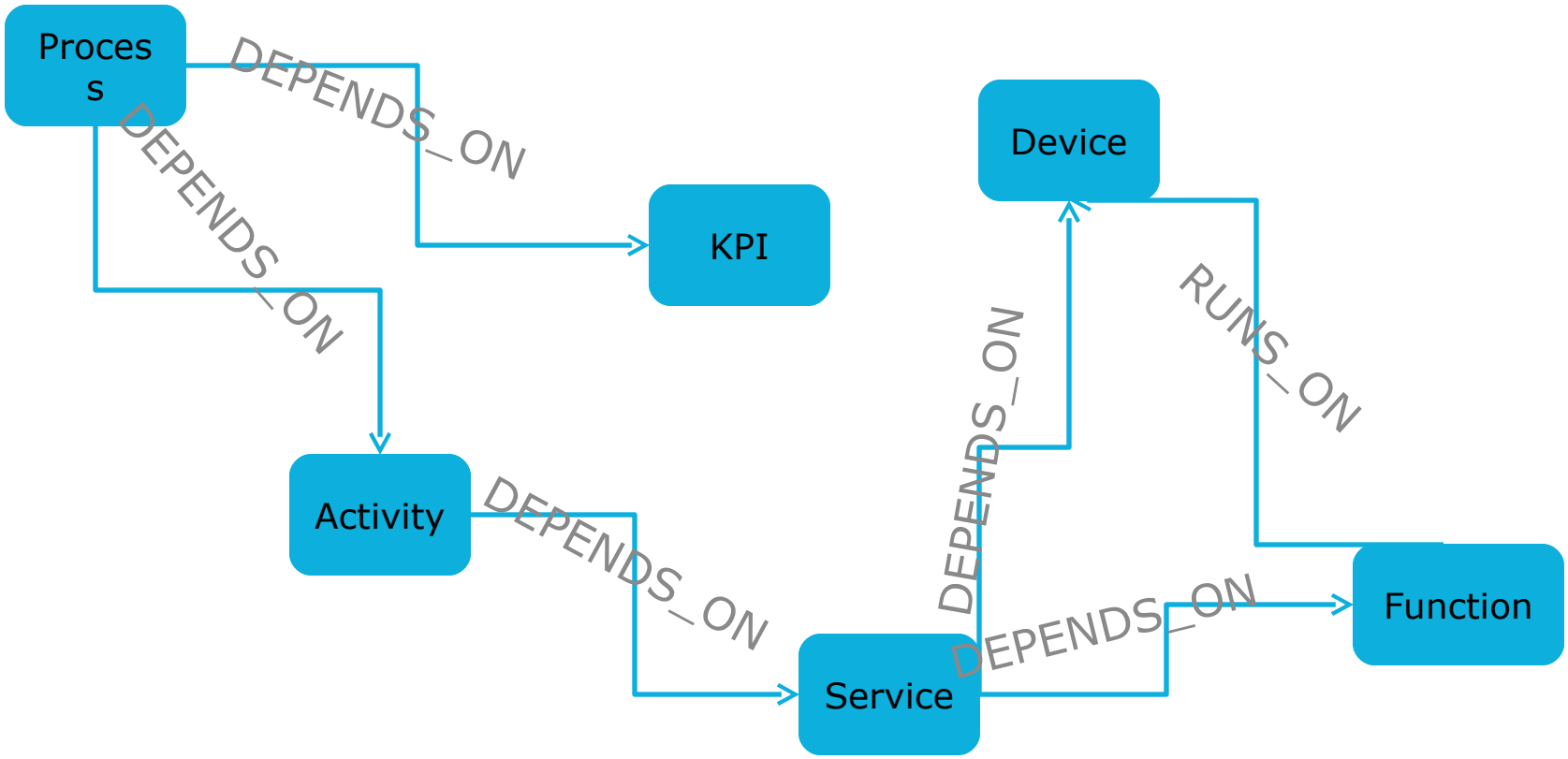
Social & Spatial Data



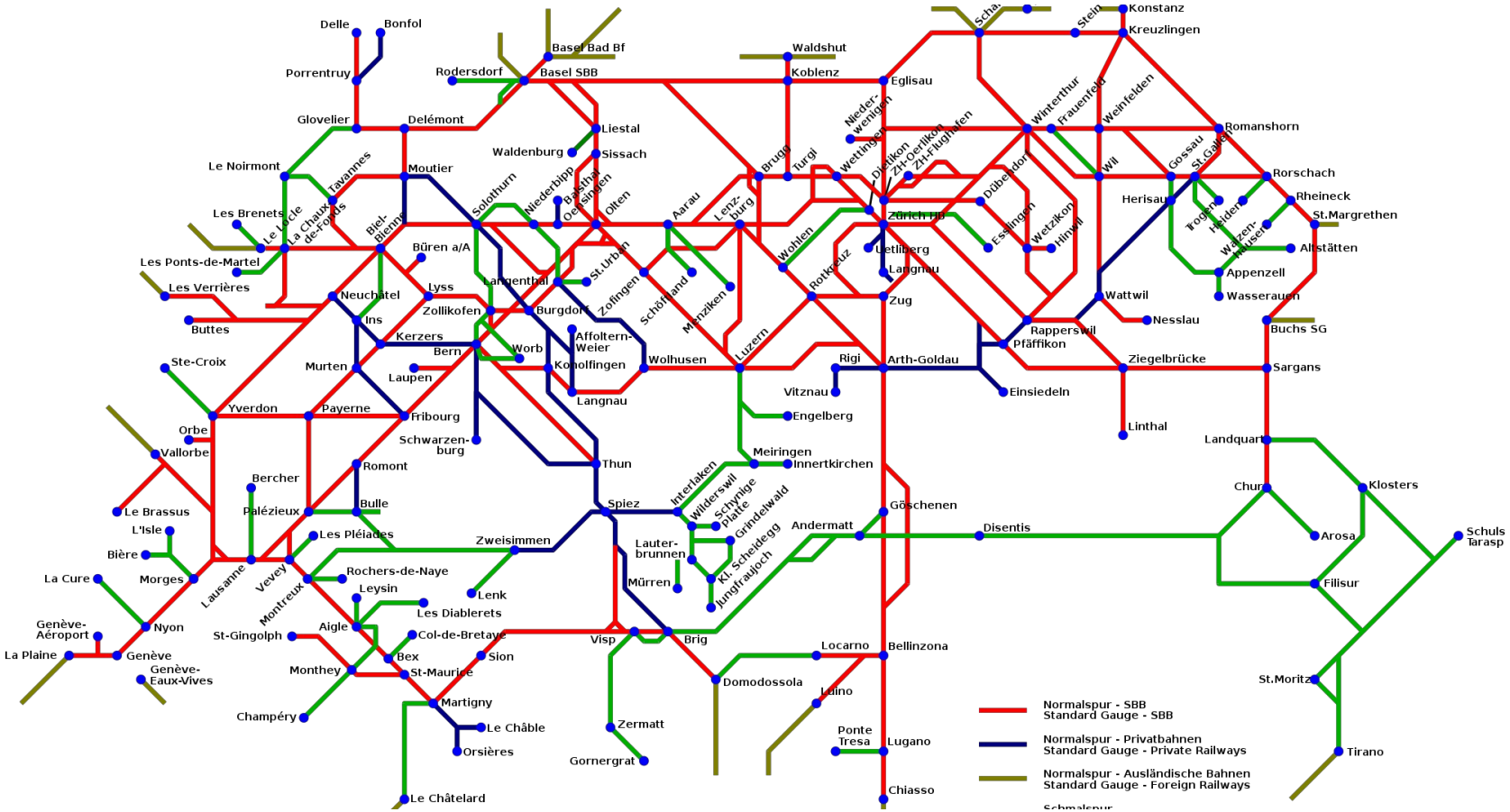
Financial Data



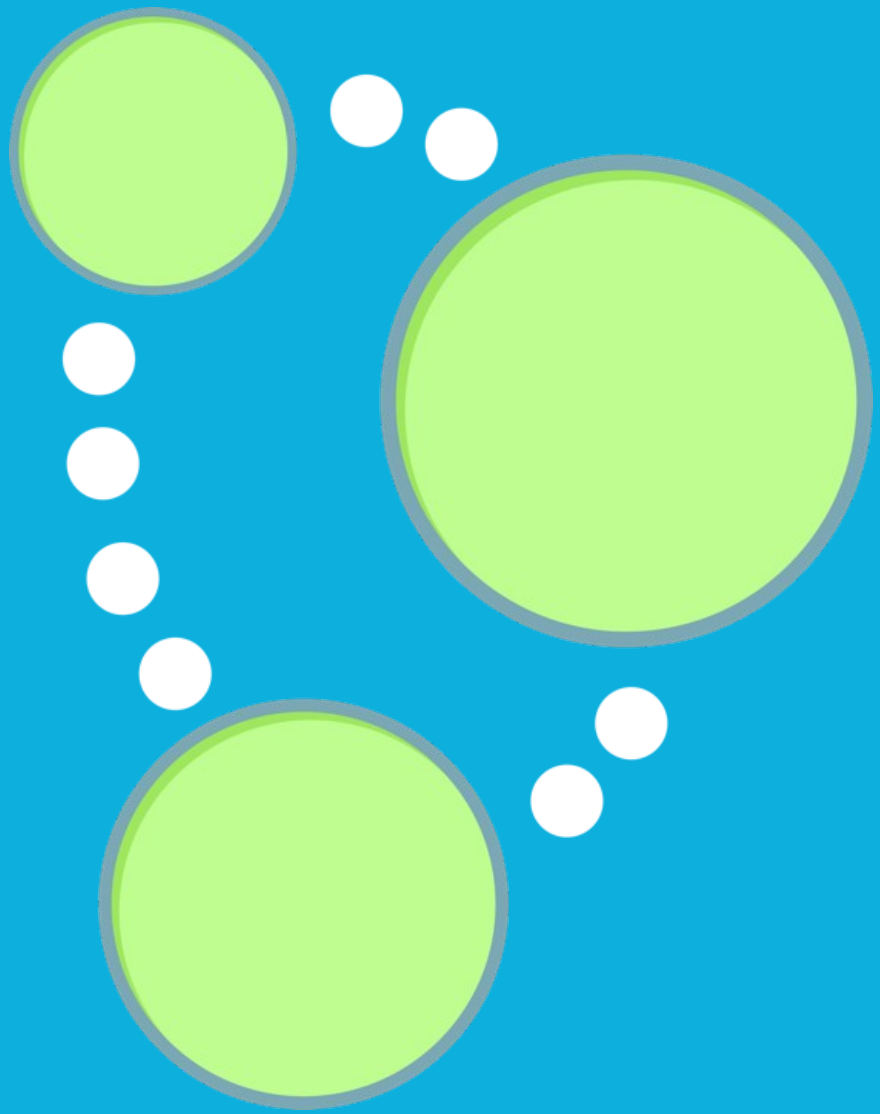
Eine beliebige Business Domäne



Schienennetz der Schweiz



Neo4j
The Big Picture



Neo4j – Big Picture

- Die verbreiteste Graphdatenbank der Welt
  - Robust: Im Betrieb 24/7 seit 2003
  - Reifegrad: Vielfach in Produktion ausgerollt
  - Community: Ecosystem mit Tools, Bindings, Frameworks

- Lizenzen
  - AGPLv3 Community Edition – OpenSource
  - Advanced/Enterprise – Für kommerzielle Anwendungen

Eigenschaften

- Objekt-orientiert, flexible Netzwerkstruktur
- Support für ACID Transaktionen
- Horizontal skalierbar
- Java API
  - Anbindung mittels Java, Groovy, Scala, JRuby
- Runtime
  - Standalone Server
  - Embedded Database
- Diskbasierter Storage Manager
- Support für Erweiterungen und Testing

- Querying
- Wie komm ich an meine Daten?

Drag picture to placeholder or click icon to add

Graph Querying

- Wie erhalte ich die richtigen Nodes aus dem Graphen?
  - Gib mir alle Freunde und Freunde von Freunden welche den Film XY mit mehr als 3 Sternen bewertet haben.
  - ...
- Queryingmethoden
  - Traversing mit Neo4j API
  - Traversal Descriptions
  - Graphalgorithmen
  - Index Queries

Neo4j API

```
Node bob = ...;


for ((Relationship rel:
bob.getRelationships(RelTypes.KNOWS)) {
  Node friend = rel.getEndNode();

System.out.println(friend.getProperty("name"));
}
```

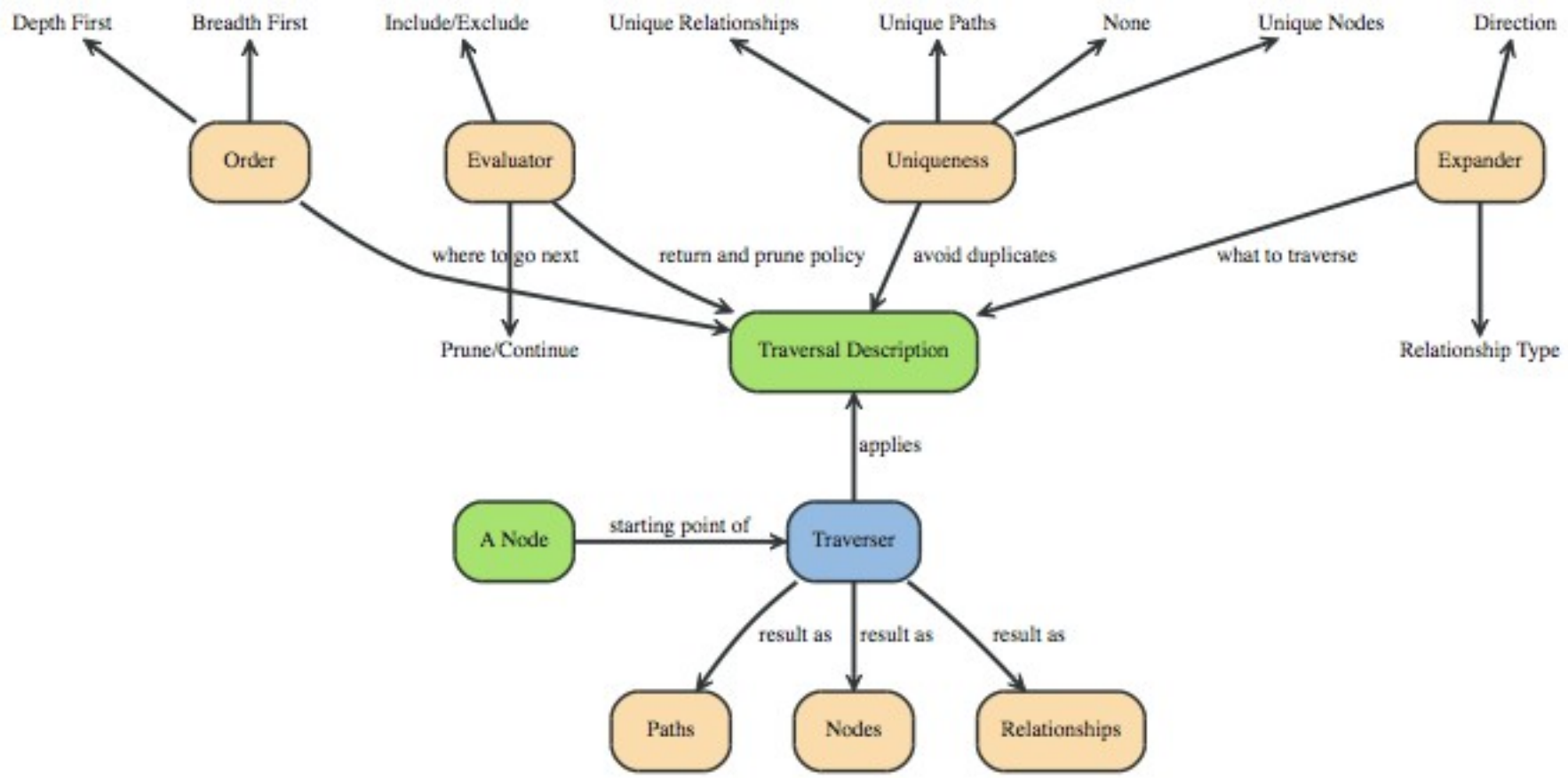
- Einfache, aber sehr „low level“ API
- Zu „verbose“ für komplexe Traversals

Traversal API

- Beschreibt Bewegungen durch den Graphen
- Callback basiert
- Abfrage API für Graphen
- Fluent API
- Konstrukte

- Neo4j implementiert folgende Algorithmen:
  - A*
  - Dijkstra
  - pathsWithLength

Traversal API – Konstrukte



Beispiel: Traversal API

```
TraversalDescription directFriends = Traversal.description()
        .breadthFirst()
        .relationships(RelTypes.KNOWS)
        .evaluator(Evaluators.toDepth(1))
        .evaluator(Evaluators.excludeStartPosition());
```

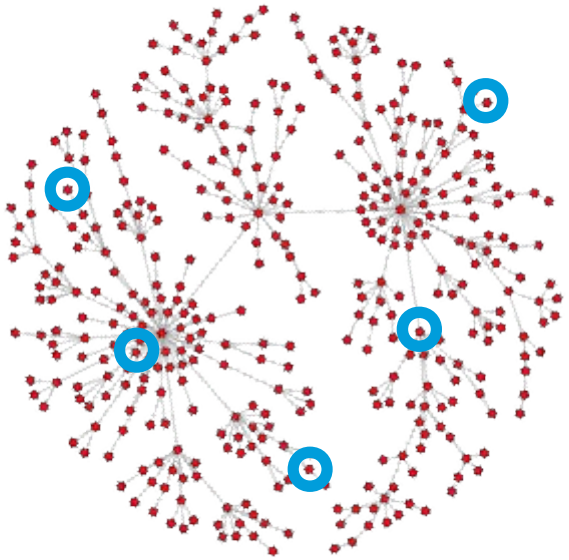
```
for(Path p : directFriends.traverse(hans)) {
    System.out.println(p.endNode().getProperty(„firstname“));
}
```

Index Queries

- Unterstützung von Node und Relationship Indices

- Lucene als Standard-implementierung

node id	property	value
15	name	Alice
15	yearOfBirth	1972
16	name	Bob
46	name	Carol
46	yearOfBirth	1983



Cypher Query Language

- Graph Query Language ähnlich SQL
- Deklarative Sprache
- Definiert das „Was“ – nicht das „Wie“
- Nutzt Pattern Matching
- Support für
  - Filterung
  - Paginierung
- Update-Funktionalität

Cypher Query Language – Elemente

Element	Bedeutung
START	Startelement(e) (Angabe von Index- o. Id-Lookup)
MATCH	Pattern für zu findende Knoten, beschrieben durch Pfade auf Identifiern (a) –[knows]-> (b)
WHERE	Resultatsfilterung (boolean-Ausdruck)
SKIP LIMIT	Paginierung
RETURN	Definiert Rückgabe-Elemente
ORDER BY	Sortierung nach Properties
PARAMETERS	Parameter-Map die im Cypher mittels Key oder Position verwendet werden kann
CREATE	Erzeugt einen Knoten

Beispiel: Cypher Query

```
String query = "START student=node(12) “
    + “MATCH student-[:KNOWS]-friend “
    + “RETURN friend“;
```

```
ExecutionEngine engine = new
ExecutionEngine(graphDb);

ExecutionResult result = engine.execute(query);
```

```
Iterator<Object> column = result.columnAs("friend");
while(column.hasNext()) {
```

Beispiel: Cypher Create Query

```
CREATE (carol {name:"Carol"}) return carol;
```

```
+-----------------------+
| carol                 |
+-----------------------+
| Node[1]{name:"carol"} |
+-----------------------+
```

```
START carol=node(1) CREATE (bob {name:"Bob"}),
    (bob)-[:knows]->(carol) return carol;
```

- Import / Export
- Wie lese und schreibe ich Daten?

Drag picture to placeholder or click icon to add

Erzeugung von Nodes mit Java

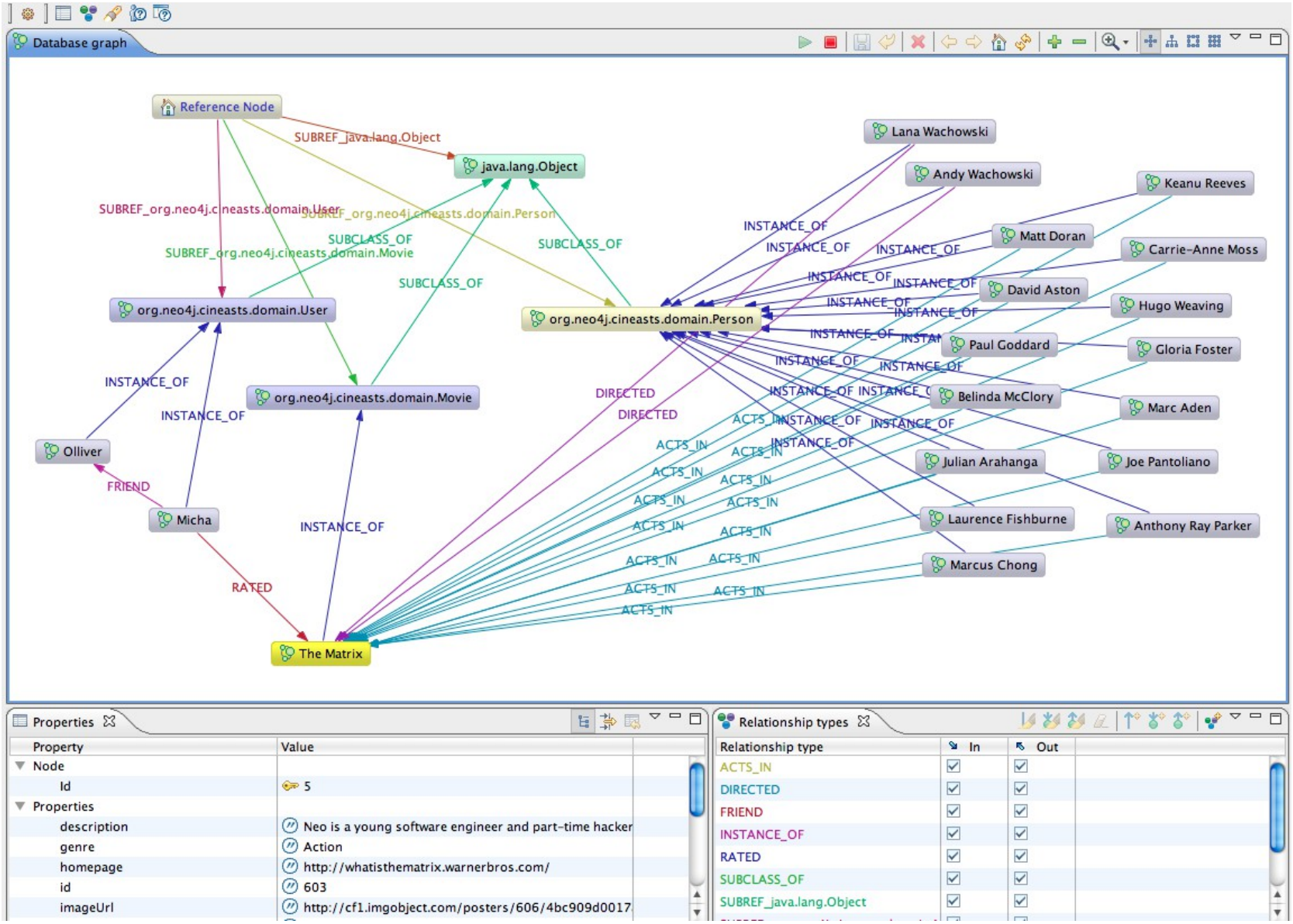
```
org.neo4j.graphdb.GraphDatabaseService graphDb =                    new
GraphDatabaseFactory().newEmbeddedDatabase(DB_PATH);

Transaction tx = graphDb.beginTx();


try {
    Node peter = createNode(graphDb, "Peter");
    Node hans = createNode(graphDb, "Hans");
    peter.createRelationshipTo(hans, RelTypes.KNOWS);
    ..
    tx.success();
} finally {tx.finish();}
```



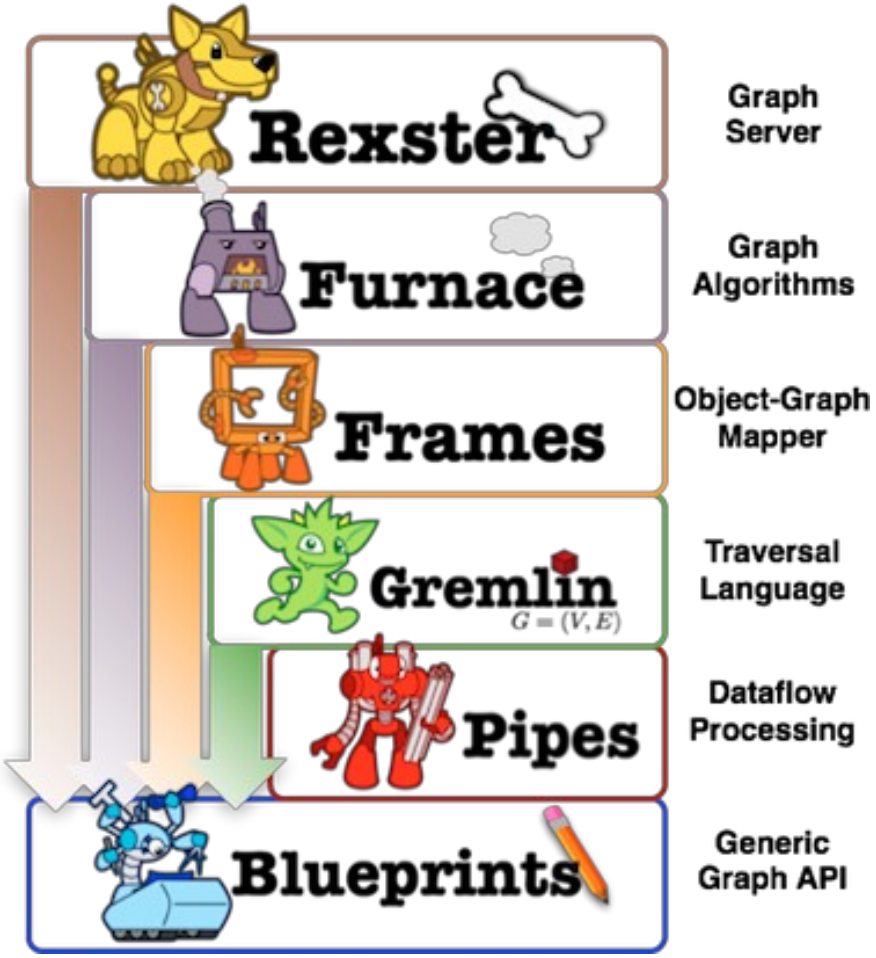
```
public Node createNode(GraphDatabaseService graphDb, String firstname) {
```


Database graph
Reference Node
SUBREF_java.lang.Object
java.lang.Object
SUBREF_org.neo4j.cineasts.domain.User
SUBREF_org.neo4j.cineasts.domain.Person
SUBCLASS_OF
SUBREF_org.neo4j.cineasts.domain.Movie
SUBCLASS_OF
SUBCLASS_OF
org.neo4j.cineasts.domain.User
org.neo4j.cineasts.domain.Person
INSTANCE_OF
org.neo4j.cineasts.domain.Movie
INSTANCE_OF
DIRECTED
DIRECTED
Olliver
FRIEND
Micha
INSTANCE_OF
RATED
The Matrix
ACTS_IN
Lana Wachowski
Andy Wachowski
Keanu Reeves
Matt Doran
Carrie-Anne Moss
David Aston
Hugo Weaving
Paul Goddard
Gloria Foster
Belinda McClory
Marc Aden
Julian Arahanga
Joe Pantoliano
Laurence Fishburne
Anthony Ray Parker
Marcus Chong
Properties
Property
Value
Node
Id
5
Properties
description
Neo is a young software engineer and part-time hacker
genre
Action
homepage
http://whatisthematrix.warnerbros.com/
id
603
imageUrl
http://cf1.imgobject.com/posters/606/4bc909d0017
Relationship types
Relationship type
In
Out
ACTS_IN
DIRECTED
FRIEND
INSTANCE_OF
RATED
SUBCLASS_OF
SUBREF_java.lang.Object

Thinkerpop – Blueprint

Thinkerpop Blueprint Extensions

- GraphML Reader/Writer Library
- GraphSon Reader/Writer Library
- GML Reader/Writer Library
- Java Universal Network/Graph Framework



GraphML

- GraphML (http://graphml.graphdrawing.org/)
  - Standardisierte Beschreibungssprache für Graphen
  - Modellierungswerkzeuge
  - XML

```
<graph id="G" edgedefault="directed">
  <node id="n0"/>
  <node id="n1"/>
   <edge source="n0" target="n1"/>
</graph>
```

Neo4j Bordmittel

- GraphML
  - Gremlin Plugin
- CSV
  - BatchInserter
  - ParallelBatchInserter

- Tools & APIs
- Nützliche Helfer für die tägliche Arbeit

Drag picture to placeholder or click icon to add

Tools & APIs (2)

- Admin Web-Interface
  - Überblick über Daten
  - Abfrage
  - Commandline Tool
- Neo4j Shell
  - Commandline Tool
  - Send UNIX-like Commands via RMI (cd, ls, pwd)
  - Support für Cypher Queries
- REST API
  - Zugriff mittels HTTP und JSON

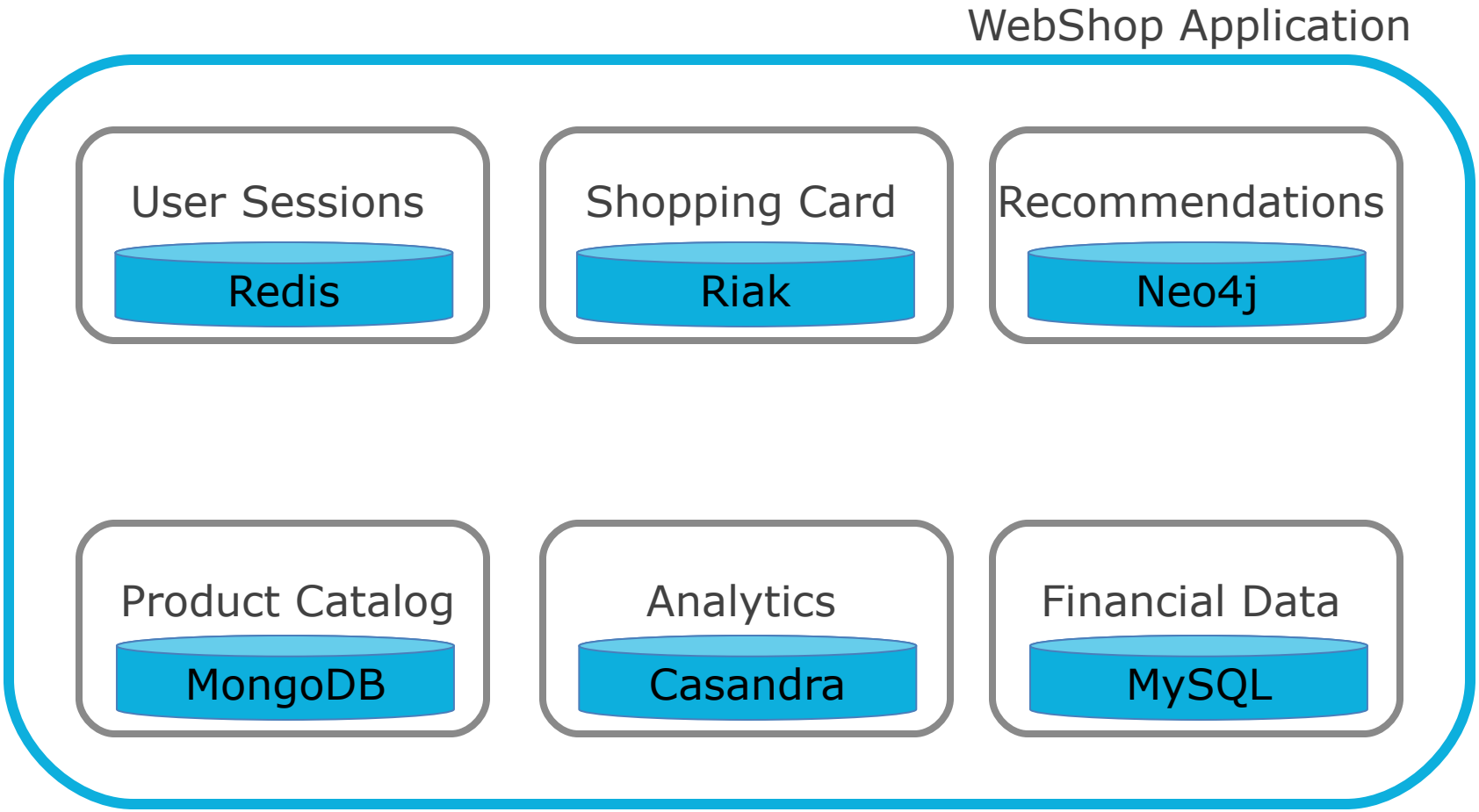
Tools & APIs

- Gremlin Plugin
  - Groovy basierte graph traversal language via REST API
  - Graph importer
- Indexing
  - Lucene
- Third Party Libraries
  - Thinkerpop
  - Spring Data Neo4j (SDN)

- Polyglot Persistence
- Gibt es nur *EINEN* zentralen Datentopf?

Drag picture to placeholder or click icon to add

Polyglot Persistence



Challenges and Highlights

Challenges

- Developer Onboarding
- Indexierung
- “Schemaänderungen” -> Migrations
- Extrem viele Daten
- Oft neue Neo4j Releases

Highlights

- Community mit guten Support
- Sourcen auf Github
- Performance
- Whiteboard friendliness
- Spring Programmier-Modell mit Spring Data Neo4j

Q&A

Herzlichen Dank!

Patrick Baumgartner, @patbaumgartner

patrick.baumgartner [at] swiftmind [dot] com