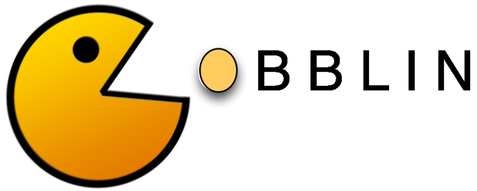
Rest.li - Microservice in LinkedIn

Jia Wei
LinkedIn

About

https://www.linkedin.com/in/jamesweipek/

About



http://linkedin.github.io/

https://engineering.linkedin.com/open-source

http://rest.li/

Microservice

https://en.wikipedia.org/wiki/List_of_buzzwords#Science_and_technology

Microservice

Buzzword?

Microservice

- Concern
  - Computing Resource
  - DevOps and Tool-chain
  - Process and Programming model

Microservice

What did companies and communities do?

History

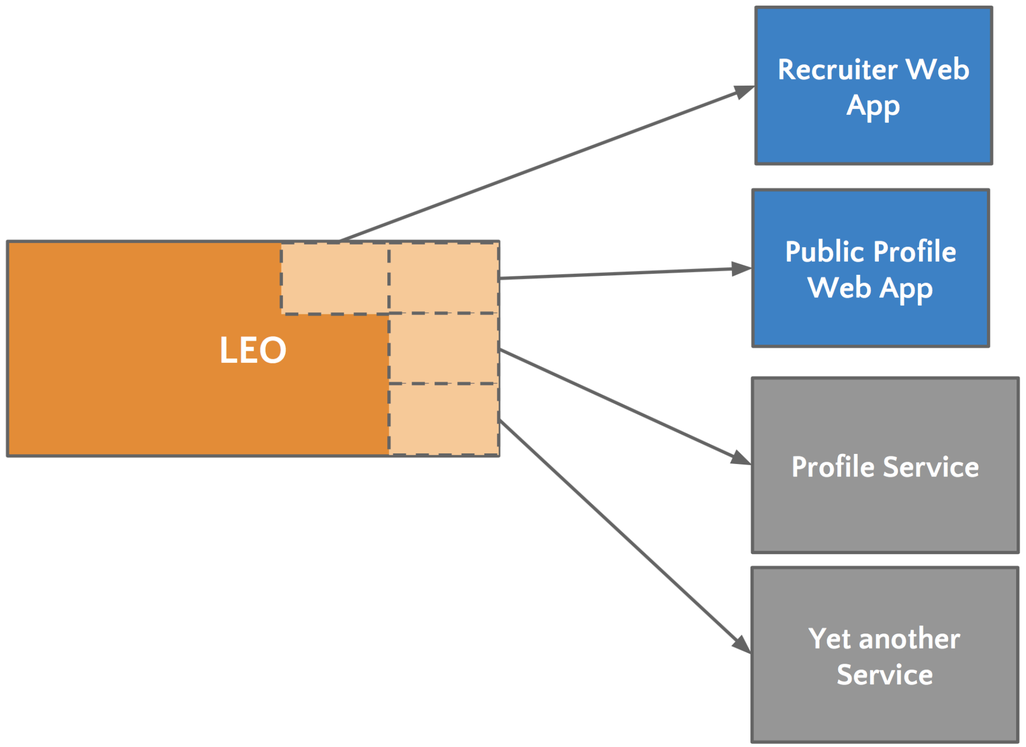
What have we done and why?

https://engineering.linkedin.com/architecture/brief-history-scaling-linkedin

History

Monolithic

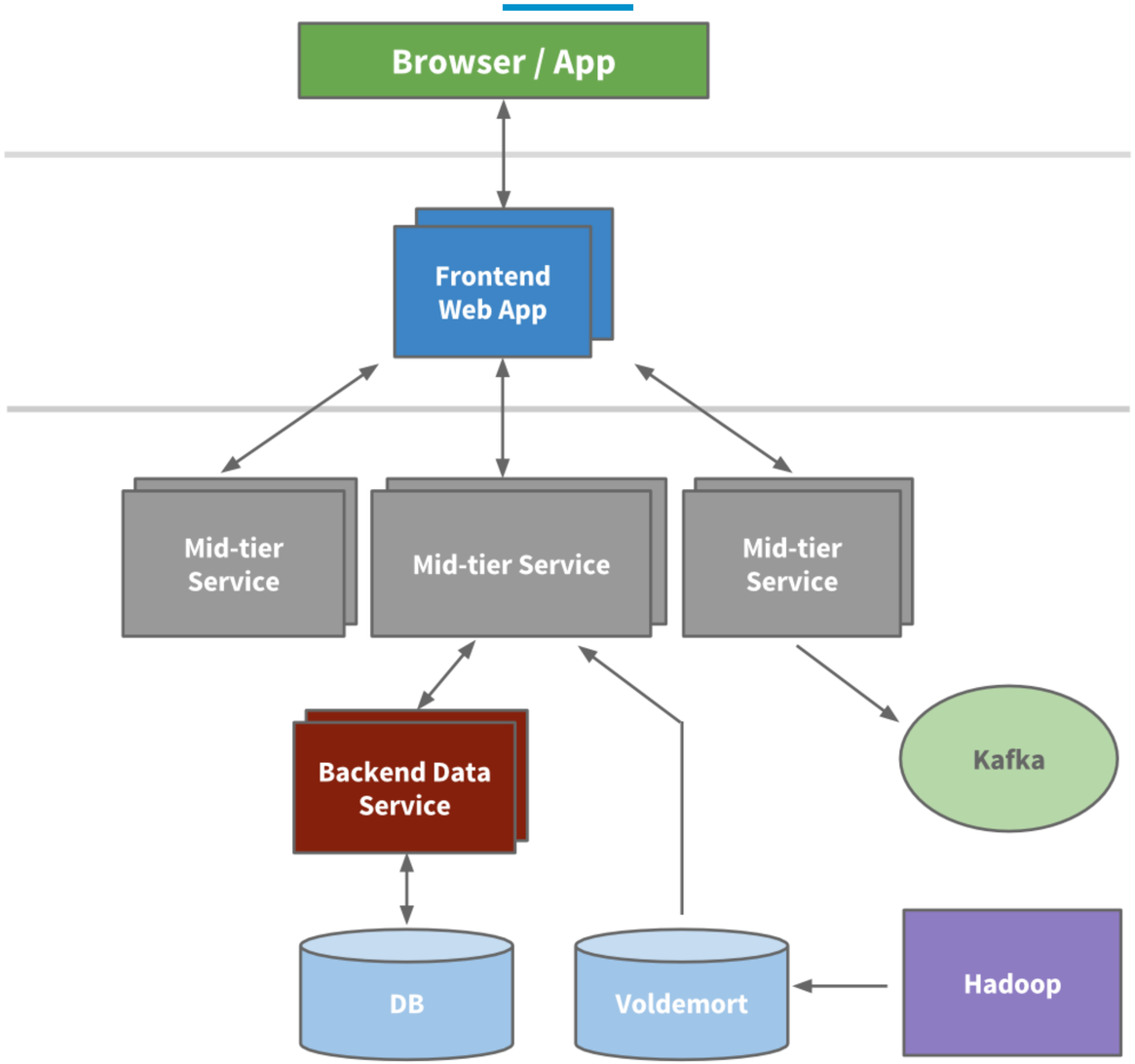
History


LEO
Recruiter Web App
Public Profile Web App
Profile Service
Yet another Service

History

Service-oriented architecture

History


Browser / App
Frontend Web App
Mid-tier Service
Mid-tier Service
Mid-tier Service
Backend Data Service
Kafka
DB
Voldemort
Hadoop

History

microservice-like

Rest.li

Rest.li is an open source REST framework for building robust, scalable RESTful architecture, using type-safe bindings, async and non-blocking I/O, and has an uniform design and end-to-end development workflow.

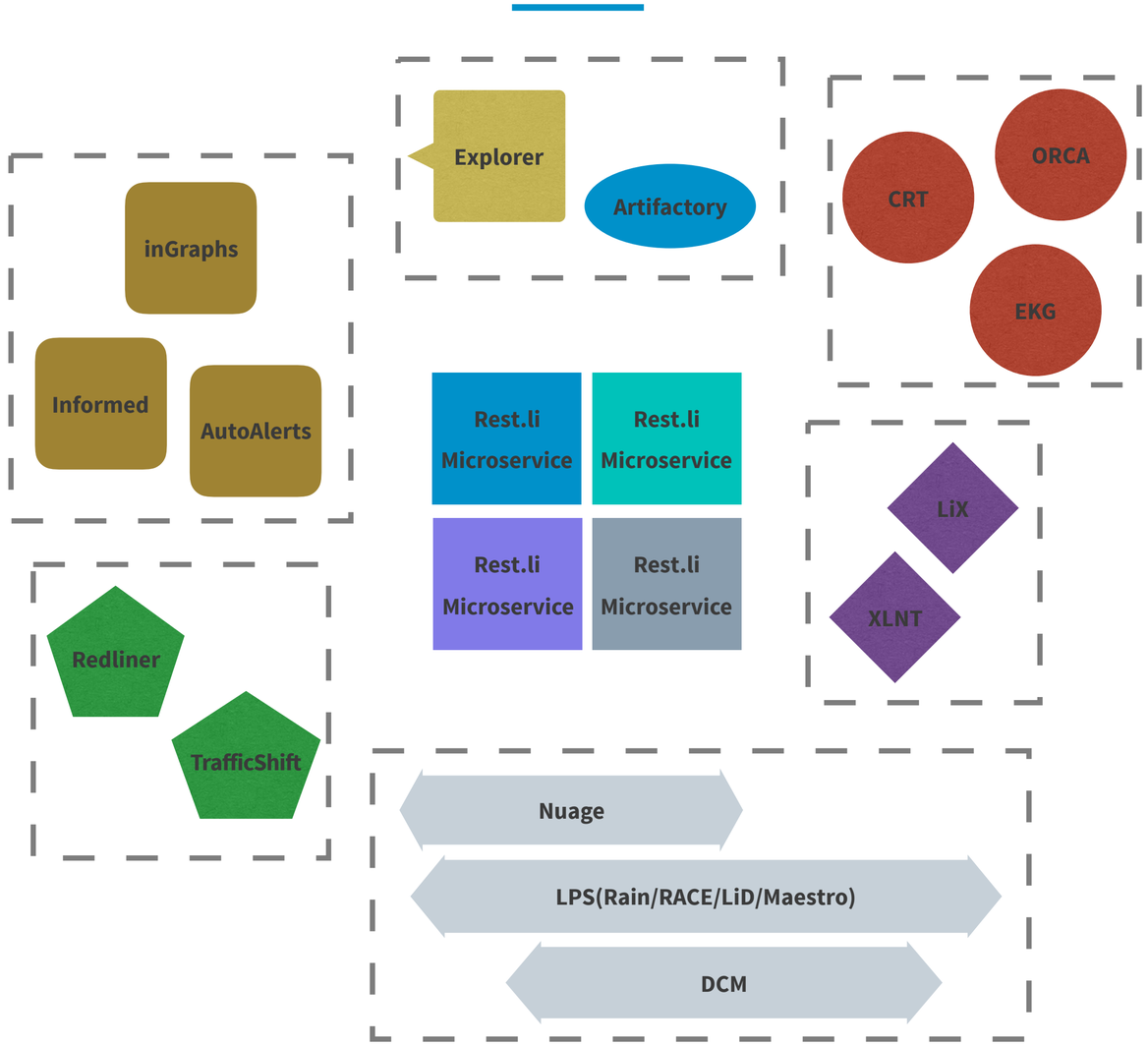
Rest.li

- Why?
  - polyglot
  - consistent & uniform

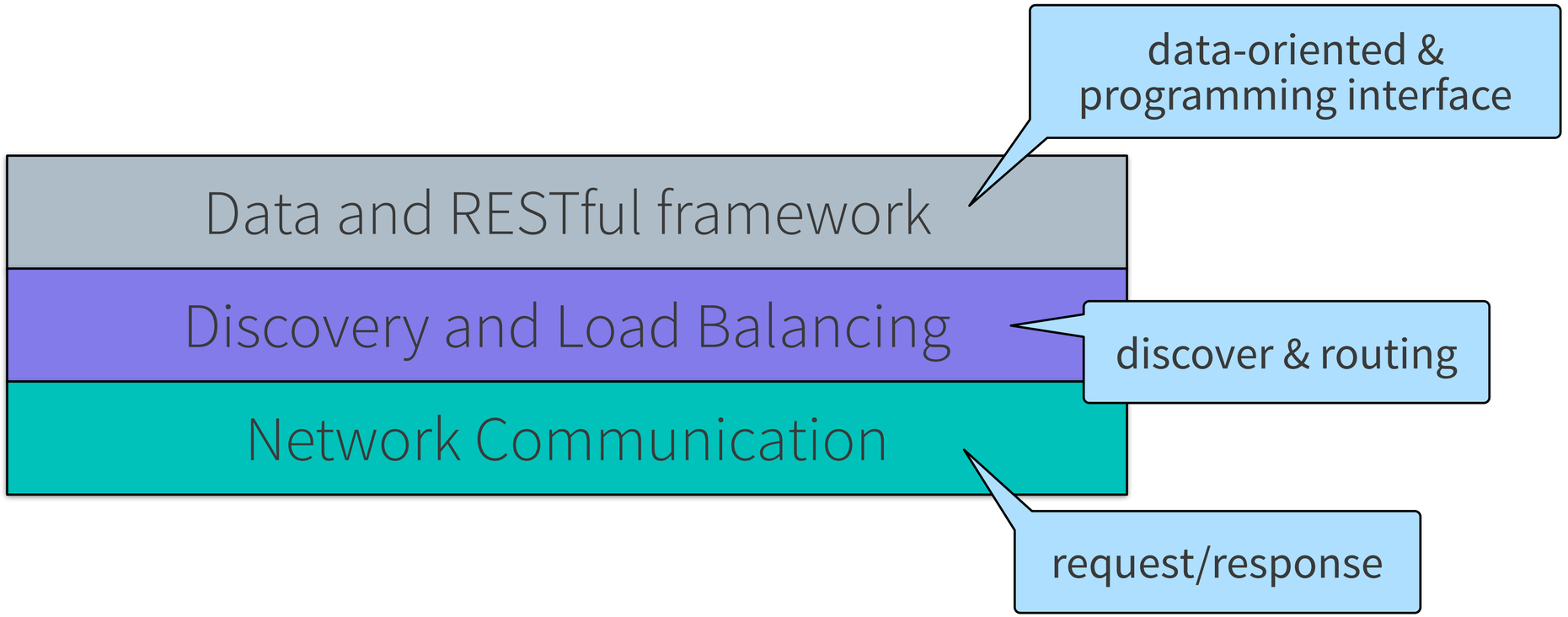
How do we use

- Widely and heavily
  - ~ 1200 Rest.li resources
  - 100+ billion Rest.li calls/day across multi-colo

Ecosystem of Rest.li

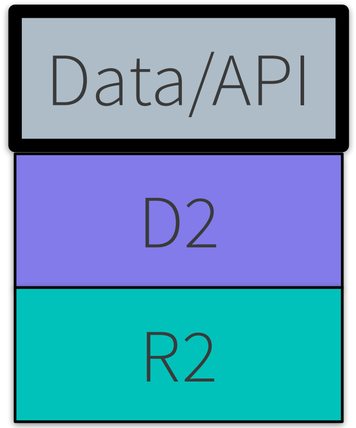


Rest.li stack and components



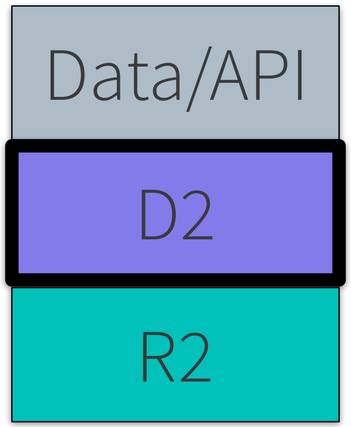
Rest.li stack and components

- `.pdsc` define data schema
- `RecordTemplate` manipulate data model
- `RequestBuilder` assemble request
- RESTful API define resource and build service
- `@Finder` and `@Action` enhance standard HTTP method



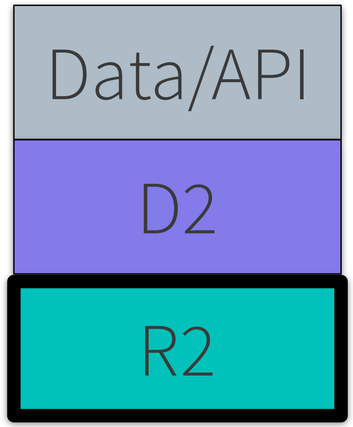
Rest.li stack and components

- Dynamic service discovery & registry
- Client-side load balancing

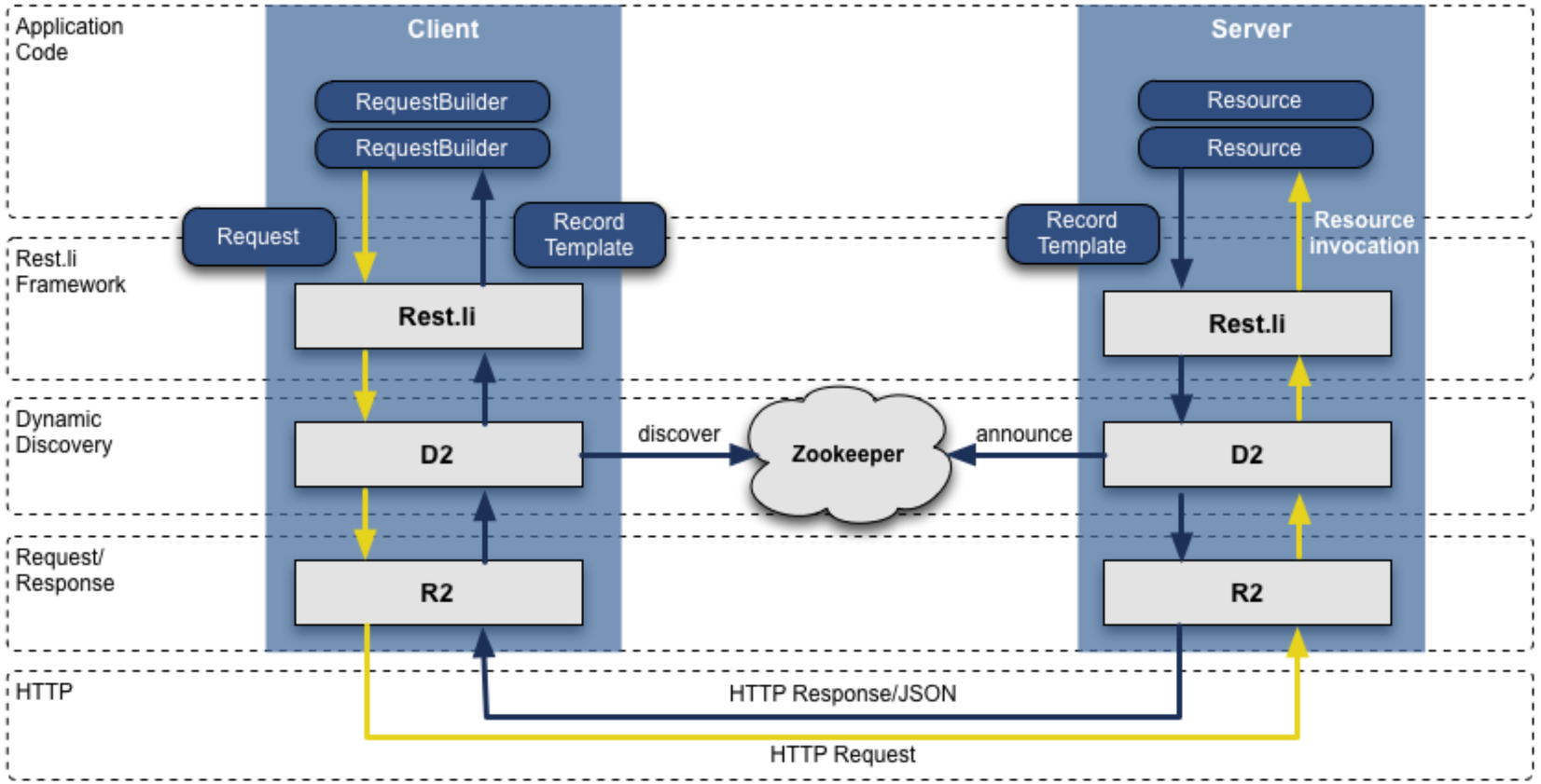


Rest.li stack and components

- Transportation abstraction
- Fully async and non-blocking
- Protocol implementation



Rest.li stack and components



Data schema & template

- Schema definition
  - Avro-like enhanced syntax and type system
  - JVM classloader-like resolver
- Serialization
- Representation
- Type-safe Java bindings

Data schema & template

- Schema def
  - Avro-like er
  - JVM classlo
- Serializati
- Represer
- Type-safe

```
{
  "type" : "record",
  "name" : "Optional",
  "namespace" : "com.linkedin.pegasus.generator.examples",
  "fields" :
  [
    {
      "name"     : "foo",
      "type"     : "string",
      "optional" : true
    }
  ]
}
```

```
{
  "foo" : "abcd"
}
```

```
{
}
```

```
{
  "type" : "record",
  "name" : "OptionalWithUnion",
  "namespace" : "com.linkedin.pegasus.generator.examples",
  "fields" :
  [
    {
      "name"    : "foo",
      "type"    : ["string", "null"]
    }
  ]
}
```

```
{
  "foo" : { "string" : "abcd" }
}
```

```
{
  "foo" : null
}
```

Data schema & template

- Schema definition
  - Avro-like enhanced syntax and type system
  - JVM classloader-like resolver
- Serialization
- Representation
- Type-safe Java bindings

Data schema & template

- Schema definition
  - Avro-like enhanced syntax and type system
  - JVM classloader-like resolver
- Serialization
- Representation
- Type-safe Java bindings

```
{
  "type" : "typeref",
  "name" : "time",
  "ref"  : "long",
  "doc"  : "Time in milliseconds since Jan 1, 1970 UTC"
}
```

Data schema & template

- Schema definition
  - Avro-like enhanced sy
  - JVM classloader-like re
- Serialization
- Representation
- Type-safe Java bindir

```
{
  "doc"  : "Bar includes fields of Foo, Bar will have fields f1 from itself and b1 from Bar"
  "type" : "record",
  "name" : "Bar",
  "include" : [ "Foo" ],
  "fields" : [
    {
      "name"     : "b1",
      "type"     : "string",
    }
  ]
}

{
  "type" : "record",
  "name" : "Foo",
  "fields" : [
    {
      "name"     : "f1",
      "type"     : "string",
    }
  ]
}
```

Data schema & template

- Schema definition
- Serialization
  - json, reference implementation of Avro 1.4.x specification
  - pson, compact binary format like MsgPack
- Representation
- Java bindings

Data schema & template

- Schema definition
- Serialization
- Representation
  - data layer
  - schema layer
  - template layer
- Java bindings

Data schema & template

- Schema definition
- Serialization
- Representation
  - data layer - totally generic and schema-aware-less
  - schema layer
  - template layer
- Java bindings

Data schema & template

- Schema definition
- Serialization
- Representation
  - data layer
  - schema layer - in-memory representation of data schema
  - template layer
- Java bindings

Data schema & template

- Schema defir
- Serialization
- Representatic
  - data layer
  - schema laye
  - template lay
- Java binding

Schema Type	Data Schema class	Relevant Specific Attributes
int	`IntegerDataSchema`	
long	`LongDataSchema`	
float	`FloatDataSchema`	
double	`DoubleDataSchema`	
boolean	`BooleanDataSchema`	
string	`StringDataSchema`	
bytes	`BytesDataSchema`	

enum	`EnumDataSchema`	`List<String> getSymbols()`
		`int index(String symbol)`
		`boolean contains(String symbol)`

array	`ArrayDataSchema`	`DataSchema getItems()`
map	`MapDataSchema`	`DataSchema getValues()`
fixed	`FixedDataSchema`	`int getSize()`

record, error	`RecordDataSchema`	`RecordType recordType()` (record or error)
		`boolean isErrorRecord()`
		`List<Field> getFields()`
		`int index(String fieldName)`
		`boolean contains(String fieldName)`
		`Field getField(String fieldName)`

union	`UnionDataSchema`	`List<Member> getMembers()`
		`boolean contains(String memberKey)`
		`DataSchema getTypeByMemberKey(String memberKey)`
		`boolean areMembersAliased()`

null	`NullDataSchema`	

ma

Data schema & template

- Schema definition
- Serialization
- Representation
  - data layer
  - schema layer
  - template layer - type-safe accessor
- Java bindings

Data schema & template

- Schema defini
- Serialization
- Representatio
  - data layer
  - schema layer
  - template layer
- Java bindings

Schema Type	Data Layer	Data Template Layer
int	`java.lang.Integer`	Coerced to `java.lang.Integer` or `int` (2)
long	`java.lang.Integer` or `java.lang.Long` (1)	Coerced to `java.lang.Long` or `long` (2)
float	`java.lang.Integer`, `java.lang.Long`, `java.lang.Float` or `java.lang.Double` (1)	Coerced to `java.lang.Float` or `float` (2)
double	`java.lang.Integer`, `java.lang.Long`, `java.lang.Float` or `java.lang.Double` (1)	Coerced to `java.lang.Double` or `double` (2)
boolean	`java.lang.Boolean`	Coerced to `java.lang.Boolean` or `boolean` (2)
string	`java.lang.String`	`java.lang.String`
bytes	`java.lang.String` or `com.linkedin.data.ByteString` (3)	`com.linkedin.data.ByteString`
enum	`java.lang.String`	Generated enum class.
array	`com.linkedin.data.DataList`	Generated or built-in array class.
map	`com.linkedin.data.DataMap`	Generated or built-in map class.
fixed	`java.lang.String` or `com.linkedin.data.ByteString`	Generated class that derives from `FixedTemplate`
record	`com.linkedin.data.DataMap`	Generated class that derives from `RecordTemplate`
error	`com.linkedin.data.DataMap`	Generated class that derives from `ExceptionTemplate`
union	`com.linkedin.data.DataMap`	Generated class that derives from `UnionTemplate`

Data schema & template

- Schema definition
- Serialization
- Representation
- Java bindings

D2

- What's D2?
  - a DNS-like layer
  - client-side load balancer

d2://<foobarService> *http://something.else:port/contextPath*

D2

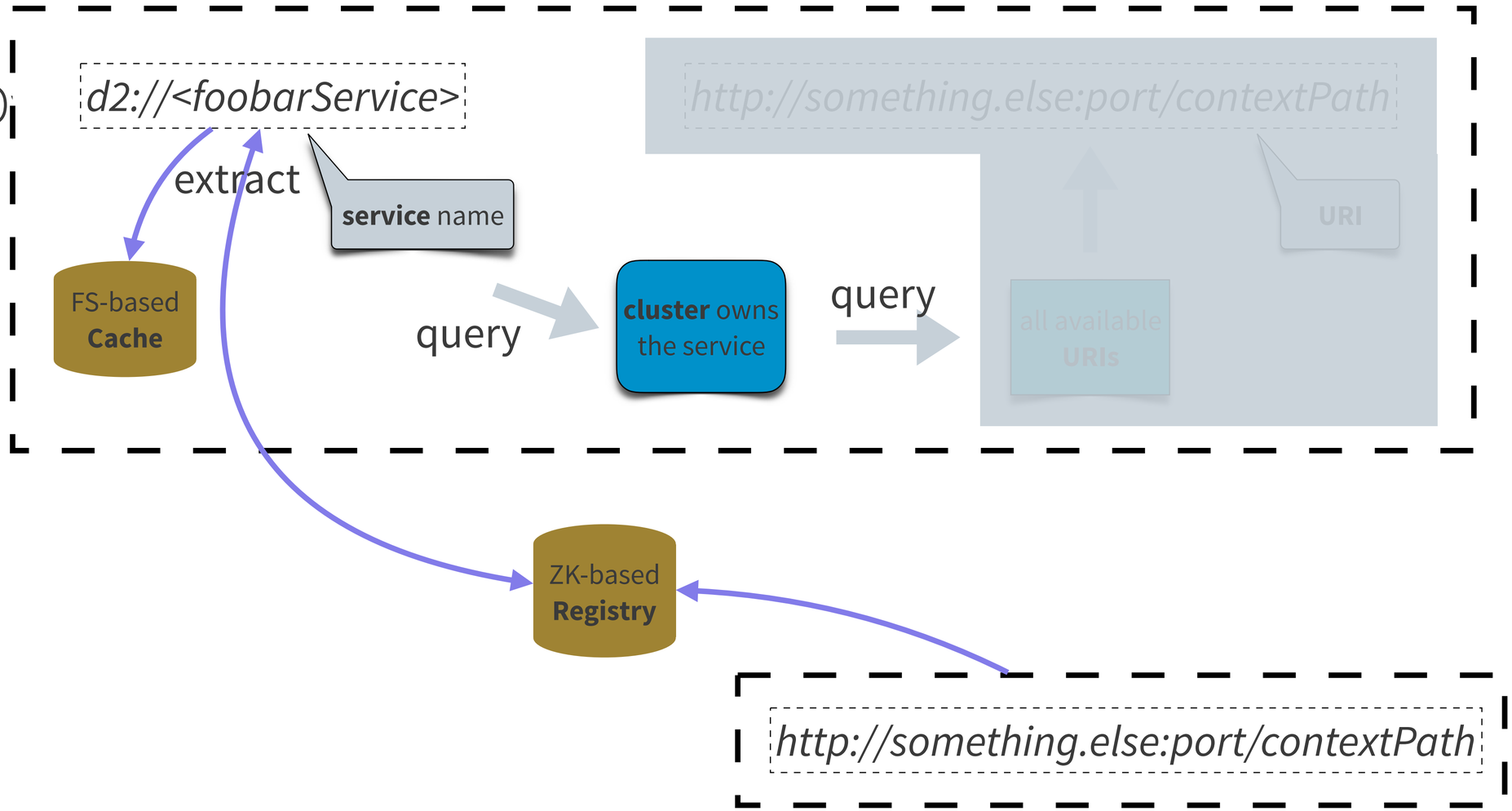
- Service, Cluster, and URI?

D2

- Dynamic discovery

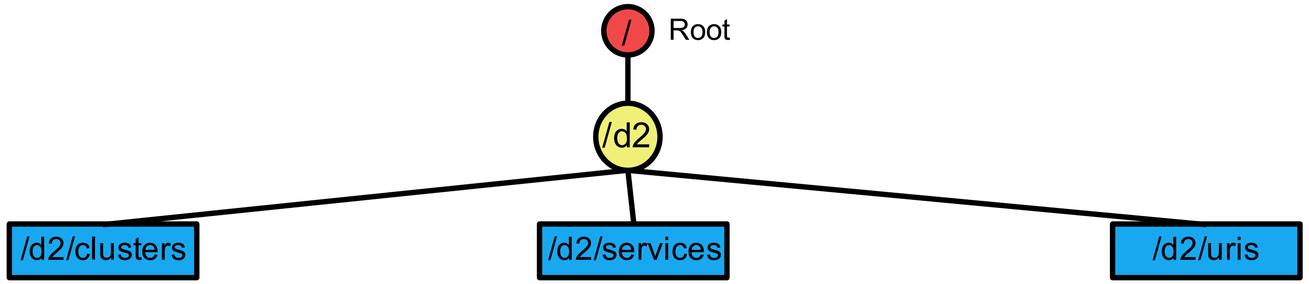
D2

- D


d2://<foobarService>
extract
service name
FS-based
Cache
query
cluster owns
the service
query
http://something.else:port/contextPath
URI
all available
URIs
ZK-based
Registry
http://something.else:port/contextPath

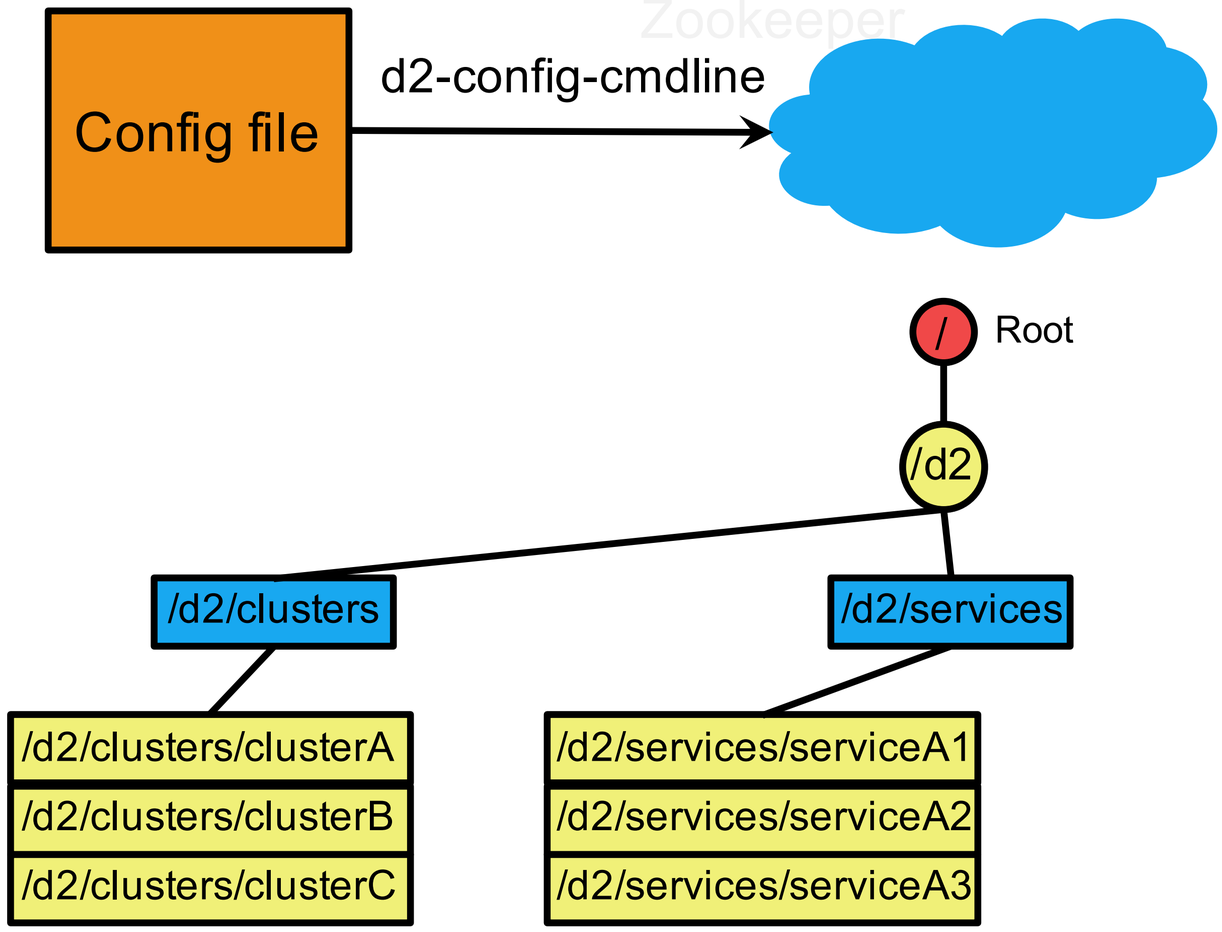
D2

- Store & representation



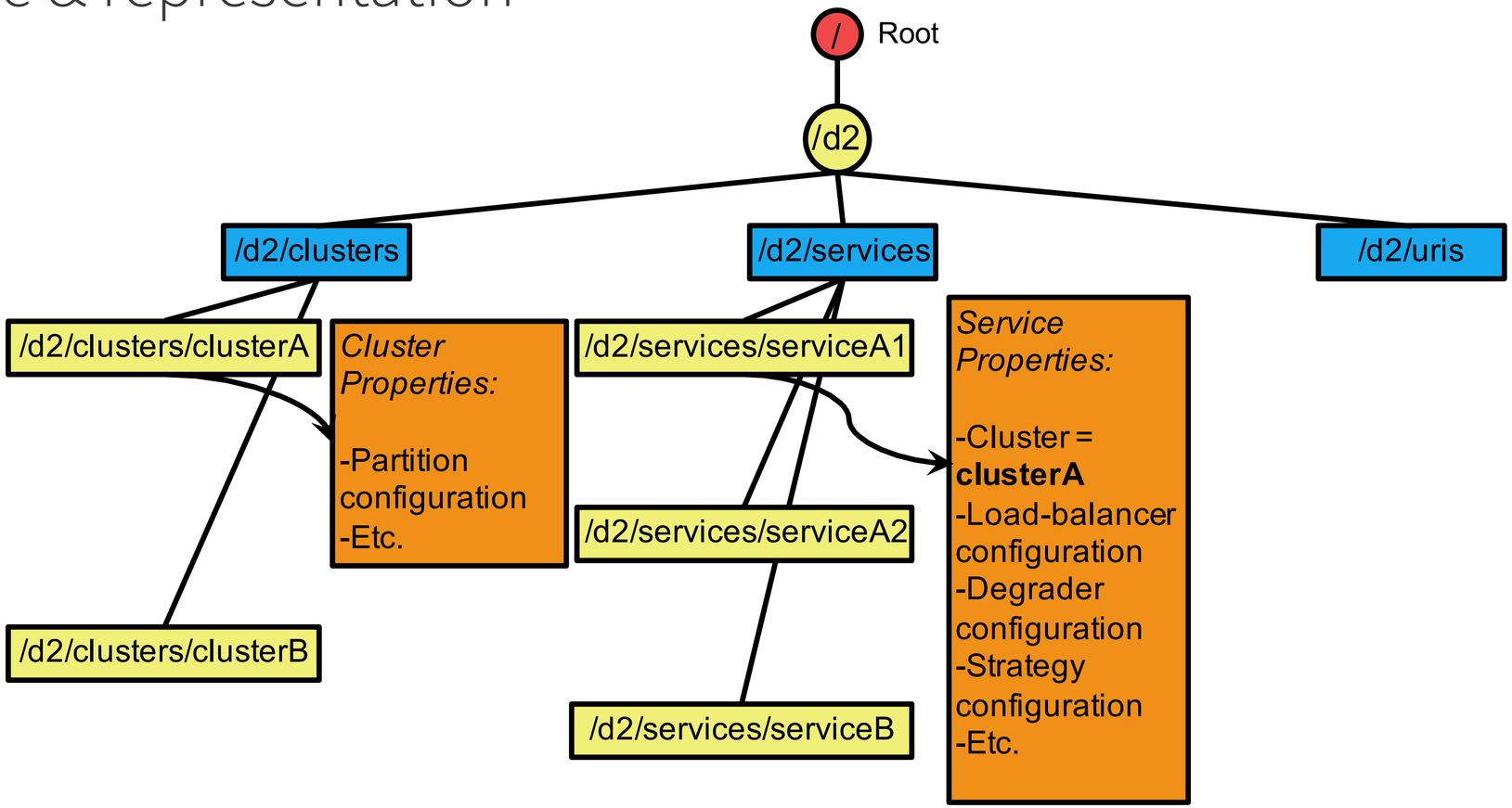
D2

- Store & representation



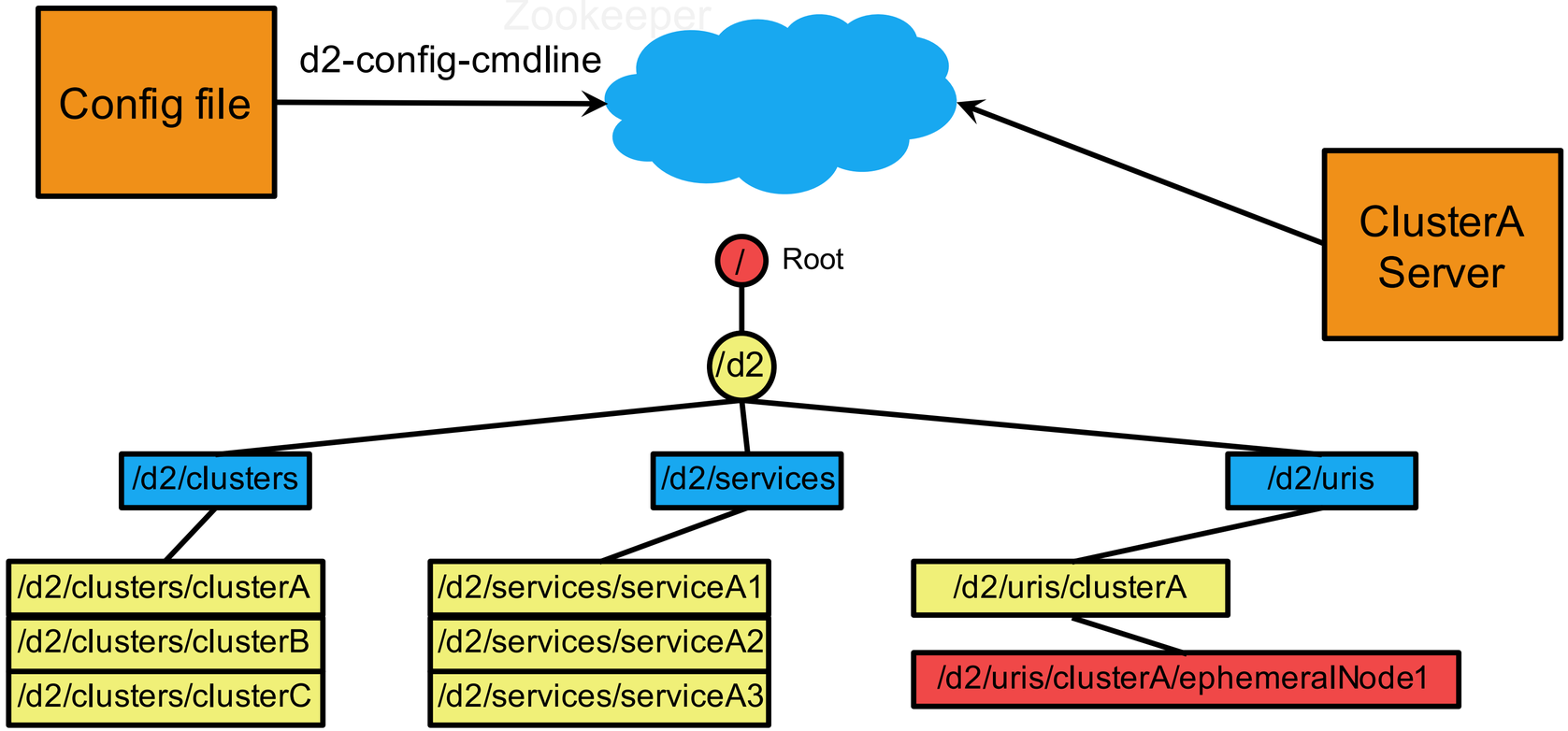
D2

- Store & representation



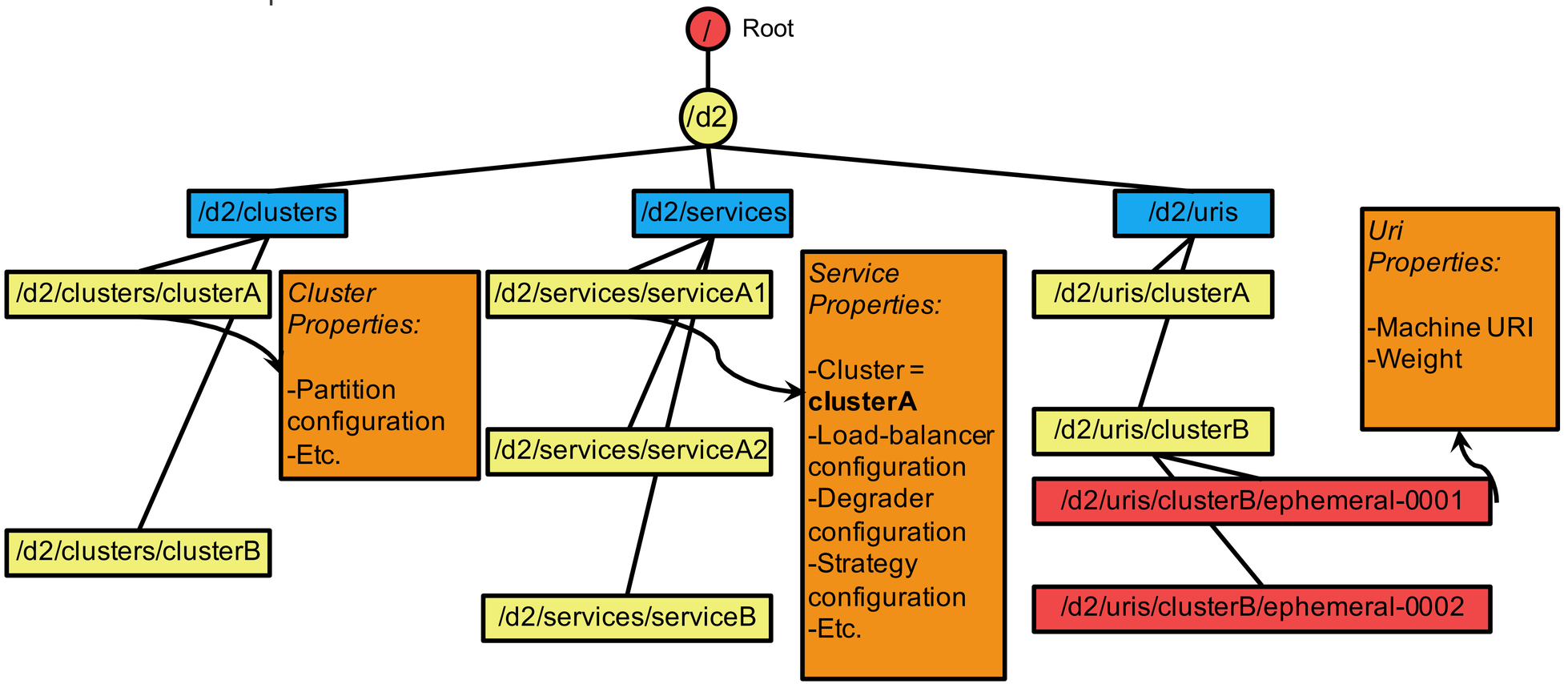
D2

- Store & representation

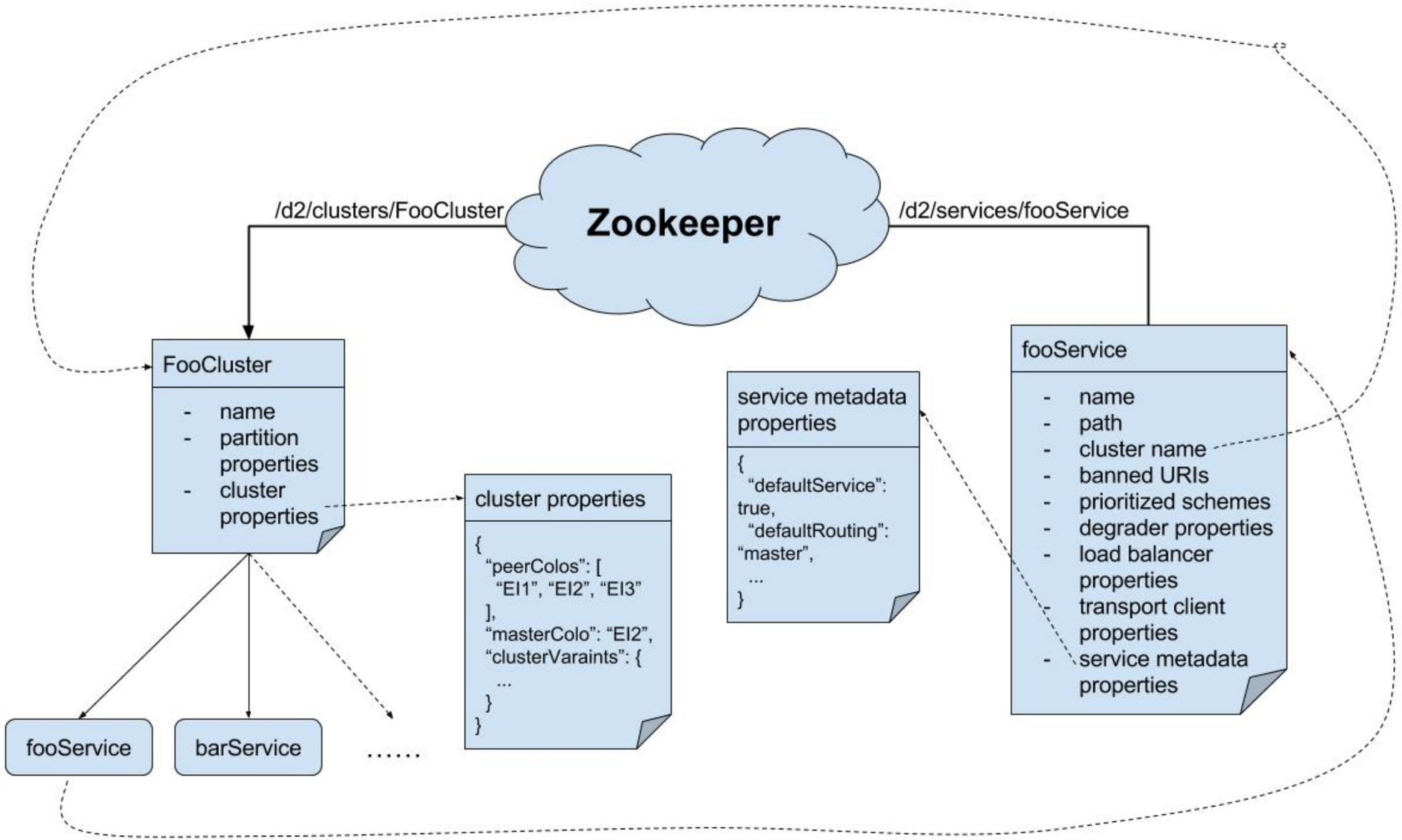


D2

- Store & representation



D2

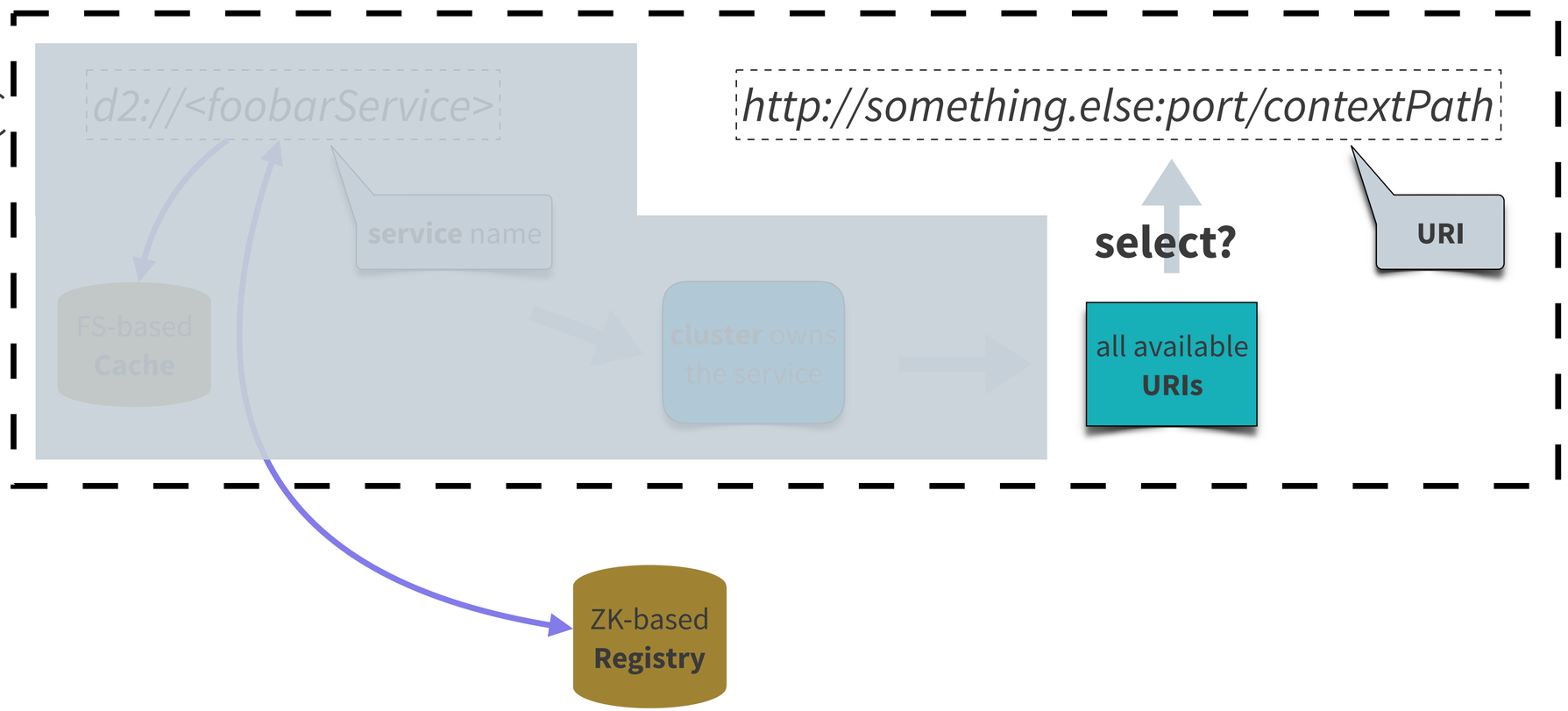

/d2/clusters/FooCluster
Zookeeper
/d2/services/fooService
FooCluster
- name
- partition properties
- cluster properties
cluster properties
{
"peerColos": [
"EI1", "EI2", "EI3"
],
"masterColo": "EI2",
"clusterVaraints": {
...
}
}
service metadata properties
{
"defaultService": true,
"defaultRouting": "master",
...
}
fooService
- name
- path
- cluster name
- banned URIs
- prioritized schemes
- degrader properties
- load balancer properties
- transport client properties
- service metadata properties
fooService
barService
.....

D2

- Client-side load balancing

D2

- C


d2://<foobarService>
http://something.else:port/contextPath
service name
FS-based
Cache
cluster owns
the service
select?
URI
all available
URIs
ZK-based
Registry

D2

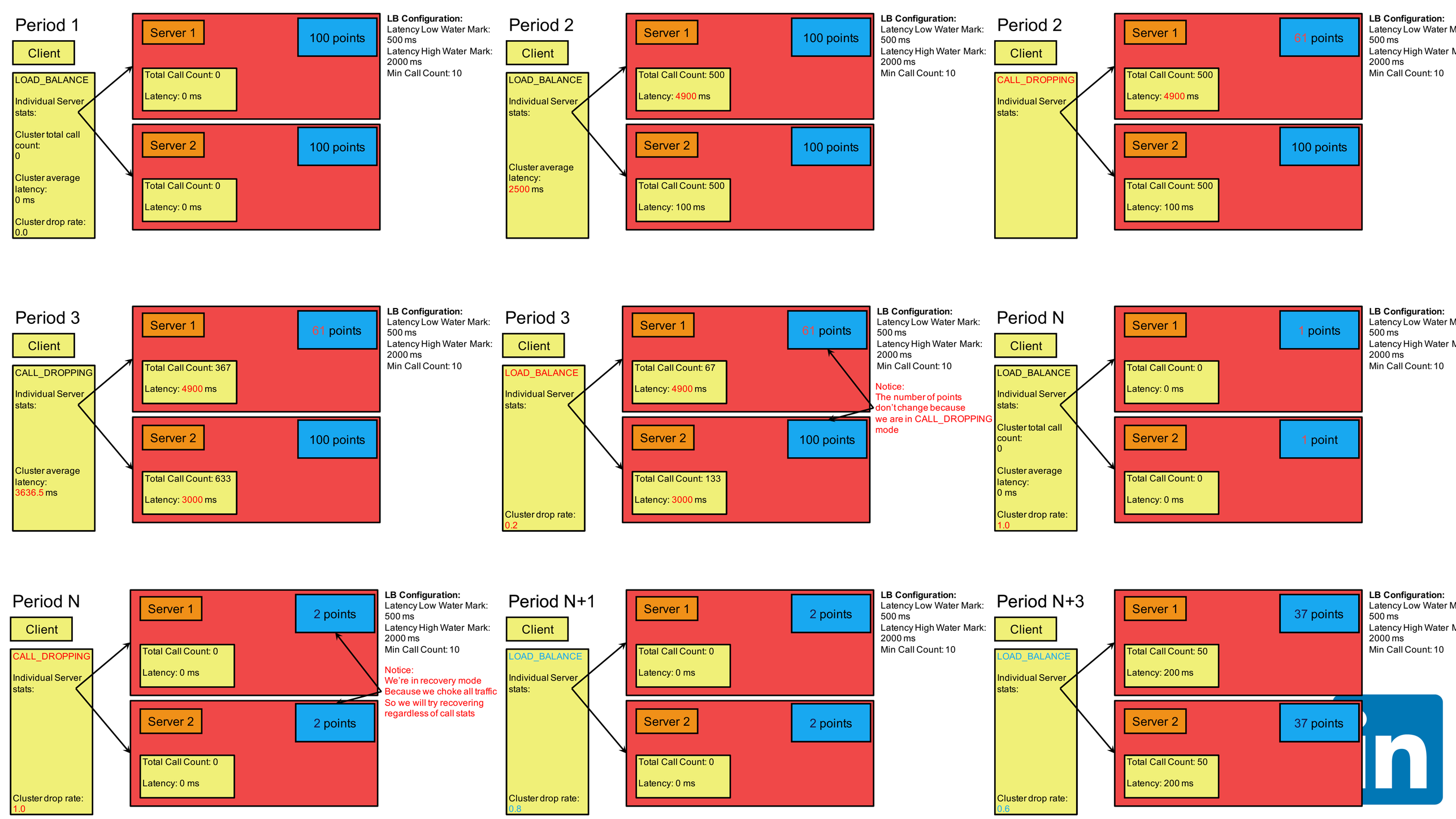
- Client-side load balancing
  - implementation
    - hierarchical properties
  - strategy

D2

- Client-side load balancing
  - implementation
  - strategy
    - random
    - degradable

D2

- Degradable load balancer
  - keep tracking call stats and health state
  - high/low water mark
  - 2 modes: load-balancing, call-dropping


Period 1
Client
LOAD_BALANCE
Individual Server stats:
Cluster total call count: 0
Cluster average latency: 0 ms
Cluster drop rate: 0.0
Server 1
100 points
Total Call Count: 0
Latency: 0 ms
Server 2
100 points
Total Call Count: 0
Latency: 0 ms
LB Configuration:
Latency Low Water Mark: 500 ms
Latency High Water Mark: 2000 ms
Min Call Count: 10
Period 2
Client
LOAD_BALANCE
Individual Server stats:
Cluster average latency: 2500 ms
Server 1
100 points
Total Call Count: 500
Latency: 4900 ms
Server 2
100 points
Total Call Count: 500
Latency: 100 ms
LB Configuration:
Latency Low Water Mark: 500 ms
Latency High Water Mark: 2000 ms
Min Call Count: 10
Period 2
Client
CALL_DROPPING
Individual Server stats:
Server 1
61 points
Total Call Count: 500
Latency: 4900 ms
Server 2
100 points
Total Call Count: 500
Latency: 100 ms
LB Configuration:
Latency Low Water M
500 ms
Latency High Water M
2000 ms
Min Call Count: 10
Period 3
Client
CALL_DROPPING
Individual Server stats:
Cluster average latency: 3636.5 ms
Server 1
61 points
Total Call Count: 367
Latency: 4900 ms
Server 2
100 points
Total Call Count: 633
Latency: 3000 ms
LB Configuration:
Latency Low Water Mark: 500 ms
Latency High Water Mark: 2000 ms
Min Call Count: 10
Period 3
Client
LOAD_BALANCE
Individual Server stats:
Cluster drop rate: 0.2
Server 1
61 points
Total Call Count: 67
Latency: 4900 ms
Server 2
100 points
Total Call Count: 133
Latency: 3000 ms
LB Configuration:
Latency Low Water Mark: 500 ms
Latency High Water Mark: 2000 ms
Min Call Count: 10
Notice:
The number of points don't change because we are in CALL_DROPPING mode
Period N
Client
LOAD_BALANCE
Individual Server stats:
Cluster total call count: 0
Cluster average latency: 0 ms
Cluster drop rate: 1.0
Server 1
1 points
Total Call Count: 0
Latency: 0 ms
Server 2
1 point
Total Call Count: 0
Latency: 0 ms
LB Configuration:
Latency Low Water M
500 ms
Latency High Water M
2000 ms
Min Call Count: 10
Period N
Client
CALL_DROPPING
Individual Server stats:
Cluster drop rate: 1.0
Server 1
2 points
Total Call Count: 0
Latency: 0 ms
Server 2
2 points
Total Call Count: 0
Latency: 0 ms
LB Configuration:
Latency Low Water Mark: 500 ms
Latency High Water Mark: 2000 ms
Min Call Count: 10
Notice:
We're in recovery mode
Because we choke all traffic
So we will try recovering regardless of call stats
Period N+1
Client
LOAD_BALANCE
Individual Server stats:
Cluster drop rate: 0.8
Server 1
2 points
Total Call Count: 0
Latency: 0 ms
Server 2
2 points
Total Call Count: 0
Latency: 0 ms
LB Configuration:
Latency Low Water Mark: 500 ms
Latency High Water Mark: 2000 ms
Min Call Count: 10
Period N+3
Client
LOAD_BALANCE
Individual Server stats:
Cluster drop rate: 0.6
Server 1
37 points
Total Call Count: 50
Latency: 200 ms
Server 2
37 points
Total Call Count: 50
Latency: 200 ms
LB Configuration:
Latency Low Water M
500 ms
Latency High Water M
2000 ms
Min Call Count: 10

D2

Call-Dropping Mode	Load Balancing Mode
entire clusters	**individual instance** in the cluster
graceful degradation	load balancing traffic
Drop Rate	Points
avg. latency, etc.	individual node latency, error rate, etc.

D2

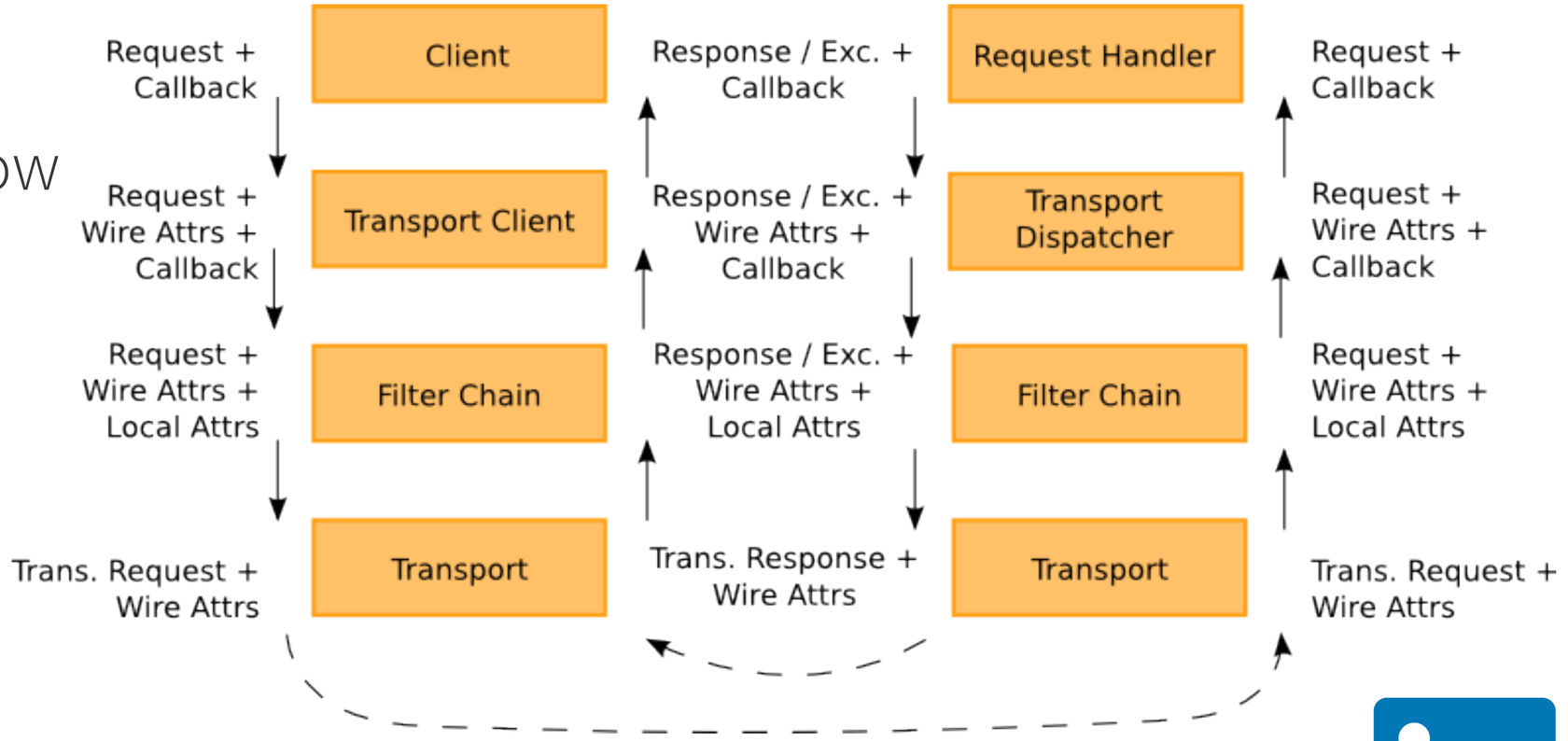
- Partitioning
  - range
  - hash
- Sticky routing
  - URI regex
  - consistent-hash

R2

- Abstraction
  - request-response
  - transportation
- Layer and Data flow

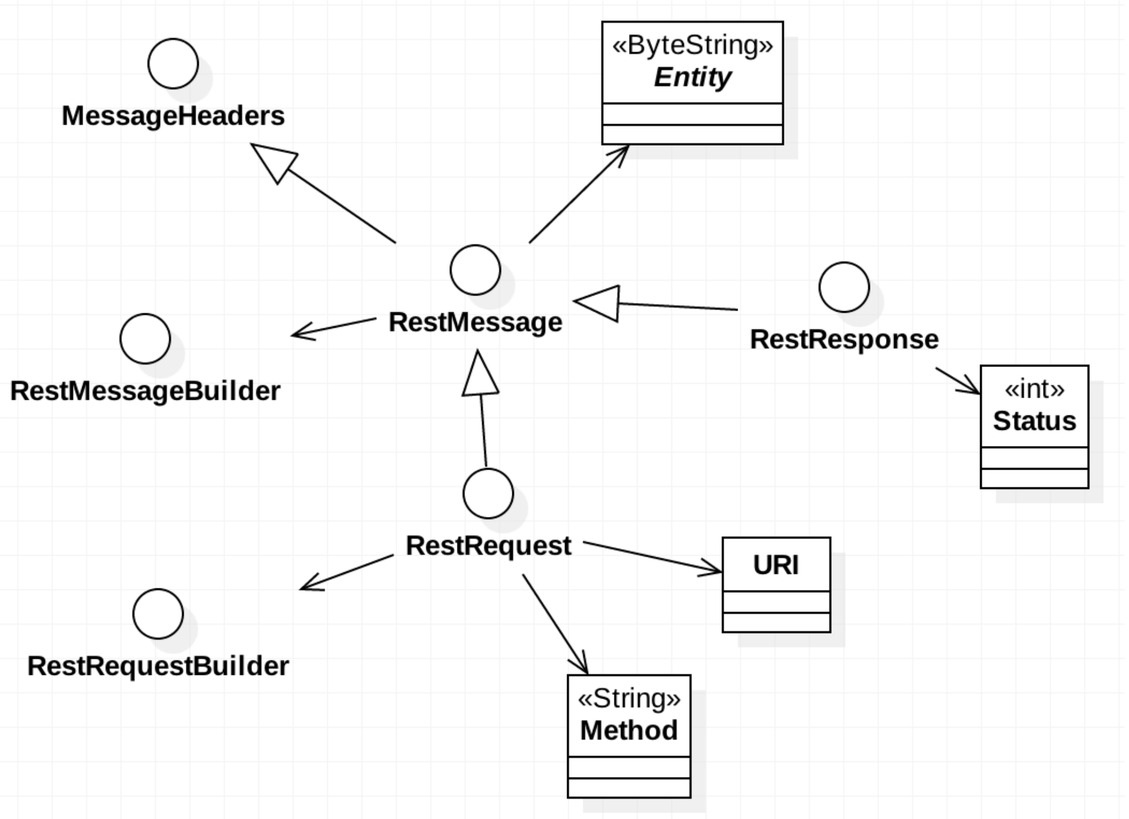
R2

- Abstraction
- Layer and Data flow



R2

- Abstraction
- Layer and Data flow
- Message in R2
  - hierarchical
  - immutable and thread-safe
  - only manipulated by builder



R2

- Abstraction
- Layer and Data flow
- Message in R2
- Filter chain
  - a set of out-of-box filters
  - ClientQueryTunnelFilter/ServerQueryTunnelFilter
- Transportation

Protocol

- HTTP based
  - Content type
  - Status code
- Representation

Protocol

- HTTP based
- Object representation
  - primitive
  - list
  - map

```
(encoded(k1):encoded(v1),encoded(k2):encoded(v2),...)
```

```
List(encoded(a1),encoded(a2),encoded(a3),...)
```

Protocol

- Request body
- Response body

```
POST /widgets
Content-Type: application/json

{"widgetName":"Lever"}
```

```
{
  "elements: [
    { "id": 1, "message": "Good morning!", "tone": "FRIENDLY" }
    // ...
  ],
  "metadata" { // only returned by finders that define a metadata schema as part of the
    // ...
  },
  "paging": {
    "count": 10,
    "links": [
      "href": "/greetings?count=10&start=10&q=search",
      "rel": "next",
      "type": "application/json"
    ],
    "start": 0
  }
}
```

Protocol

- X-Restli-Method

```
PUT /widgets?ids=1&ids=2 HTTP/1.1
Content-Type: application/json
X-RestLi-Method: BATCH_UPDATE

{
  "entities": {
    "1": {"widgetName":"Trebuchet"},
    "2": {"widgetName":"Gear"}
   }
}
```

```
POST /widgets HTTP/1.1
Content-Type: application/json
X-RestLi-Method: BATCH_CREATE

{
  "elements": [
    {"widgetName":"Ratchet"},
    {"widgetName":"Cog"},
    {"widgetName":"!@&%@$#"}
  ]
}
```

```
POST /widgets?ids=1&ids=2 HTTP/1.1
Content-Type: application/json
X-RestLi-Method: BATCH_PARTIAL_UPDATE

{
  "entities": {
    "1": {"patch": { "$set": { "name":"Sam"}}},
    "2": {"patch": { "$delete": ["name"]}}
   }
}
```

Protocol

- X-HTTP-Method-Override

```
http://localhost?ids=List(1,2,3)
```

```
POST -H "X-HTTP-Method-Override: GET" -H "Content-Type: application/x-www-form-urlencoded"
    --data $'ids=1,2,3' http://localhost
```

Protocol

- X-HTTP-Method-Override

```
curl -X POST -H "X-HTTP-Method-Override: PUT" -H "Content-Type: multipart/mixed; boundary=xyz"
       --data $'--xyz\r\nContent-Type: application/x-www-form-
urlencoded\r\n\r\nids=List(1,2,3)\r\n--xyz\r\n Content-Type:
application/json\r\n\r\n{"foo":"bar"}\r\n--xyz--' http://localhost
```

Projection

- Goal
  - ease of use
  - efficiency
- Design
- Syntax
- Implementation

Projection

- Goal
- Design
  - defined and described by JSON
- Syntax
- Implementation

Projection

- Goal
- Design
- Syntax
- Implementation

Projection

```
{
  "person": {
    "phone": 1,
    "firstname": 1,
    "lastname": 1,
    "current_position": {
      "job_title": 1
    }
  }
}
```

```
{
  "profile": {
    "phone": 0
  }
}
```

ation

```
"array_field": {
    "$start": 10,
    "$count": 15,
    "$*": {
      (...)
    }
  }
```

```
"map_field": {
      "$*": {
        /* mask for every values in the map */
      },
      "key1": {
        /* mask for value of key1 */
      },
      "key2": {
        /* mask for value of key1 */
      }
    }
  }
```

Projection

- Goal
- Design
- Syntax
- Implementation

```
{
  "person": {
    "firstname": 1,
    "lastname": 1
  }
}
```



```
:(person:(firstname,lastname))
```



```
http://host:port/context/resource/id?fields=person:(firstname,lastname)
```

Projection

- Goal
- Design
- Syntax
- Implementation

```
PathSpec pathSpec = Foo.fields().bar();
```

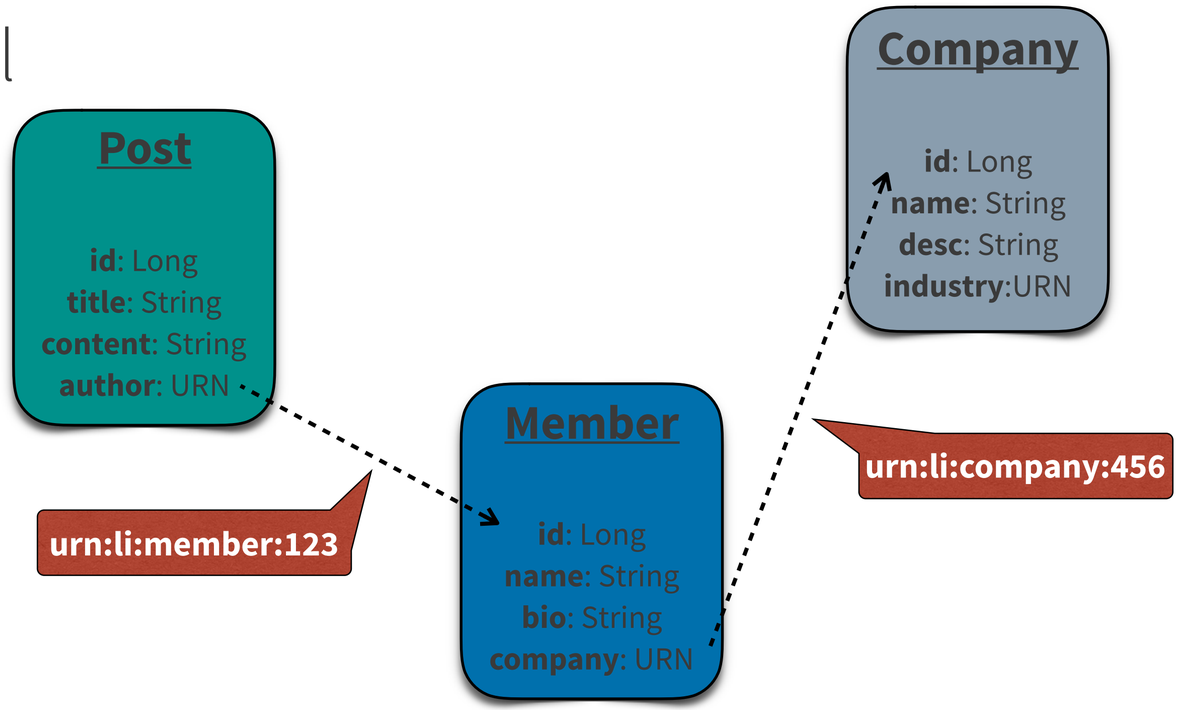
```
builder.fields(pathSpec);
```

```
builder.fields(pathSpec1, pathSpec2, pathSpec3);

builder.fields(pathSpecArray);
```

Deco(ration)

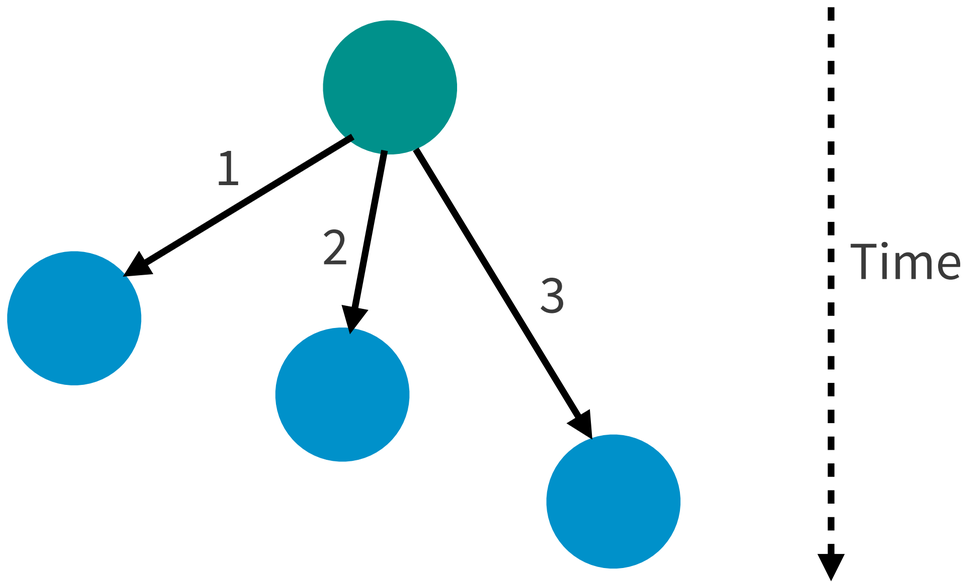
- Normalized domain model
- URN



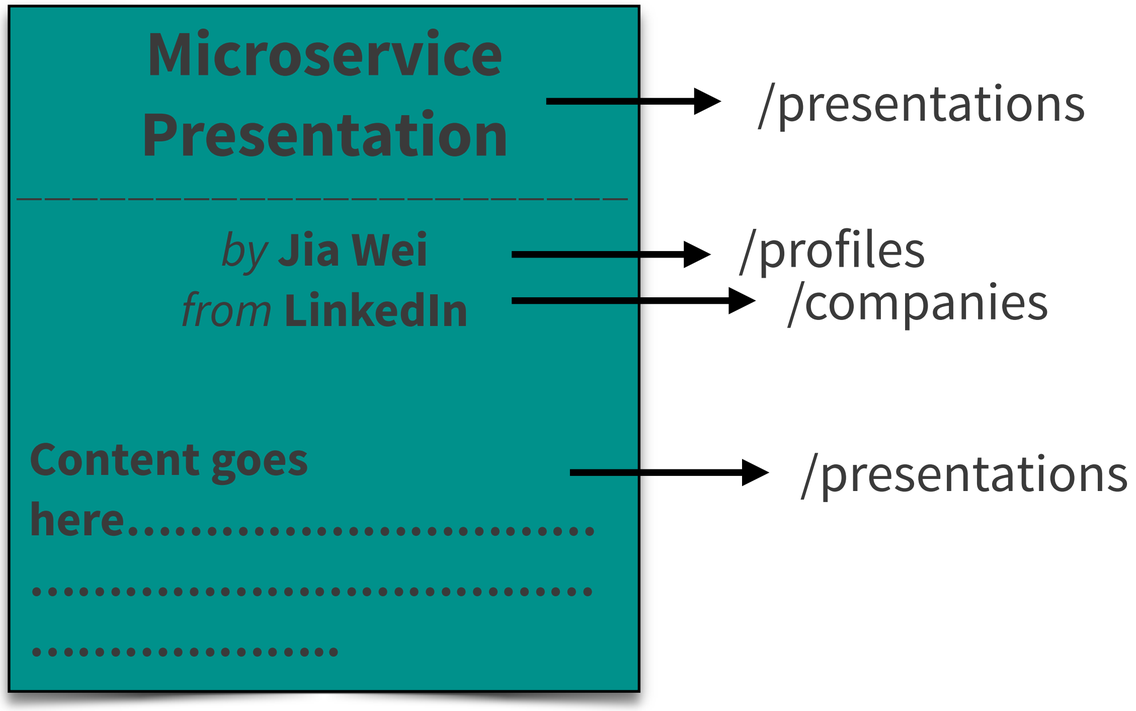
https://en.wikipedia.org/wiki/Uniform_Resource_Name

Deco(ration)

- What's deco?
  - URN resolution



Deco(ration)


Microservice
Presentation
by Jia Wei
from LinkedIn
Content goes
here.....
.....
.....
/presentations
/profiles
/companies
/presentations

Deco(ration)

deco://presentations/123?**projection**=(title, content,
speaker~(firstName, lastName, **company~**(companyName)))

Deco(ration)

- Service explosion
- Abstract away services from clients

Async

Rest.li is async and non-blocking under the hood!

Async

- How’s R2?

Async

- How’s D2?

Async

- How’s Data layer and Rest.li framework?

ParSeq

- What's ParSeq?

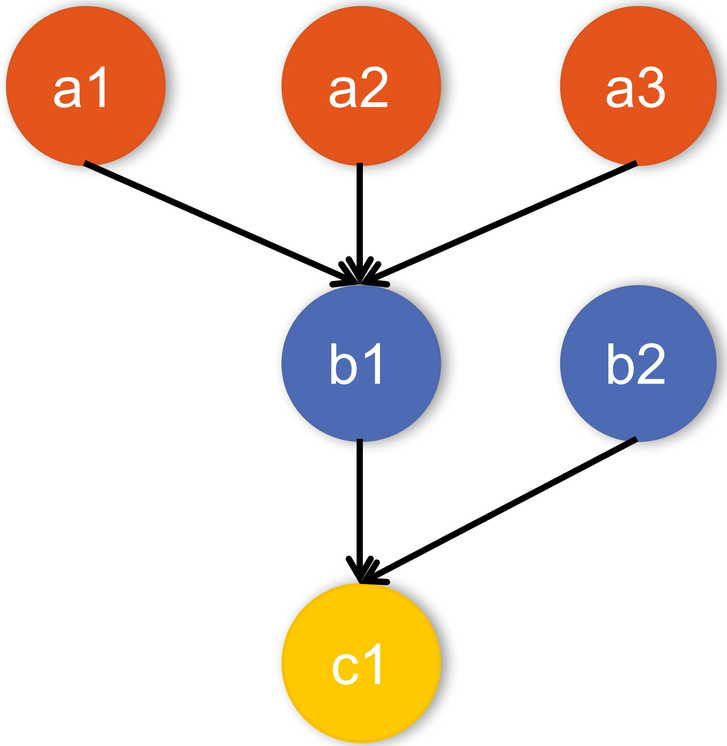
Par(allelize) Seq(uential code) = ParSeq!

https://github.com/linkedin/parseq

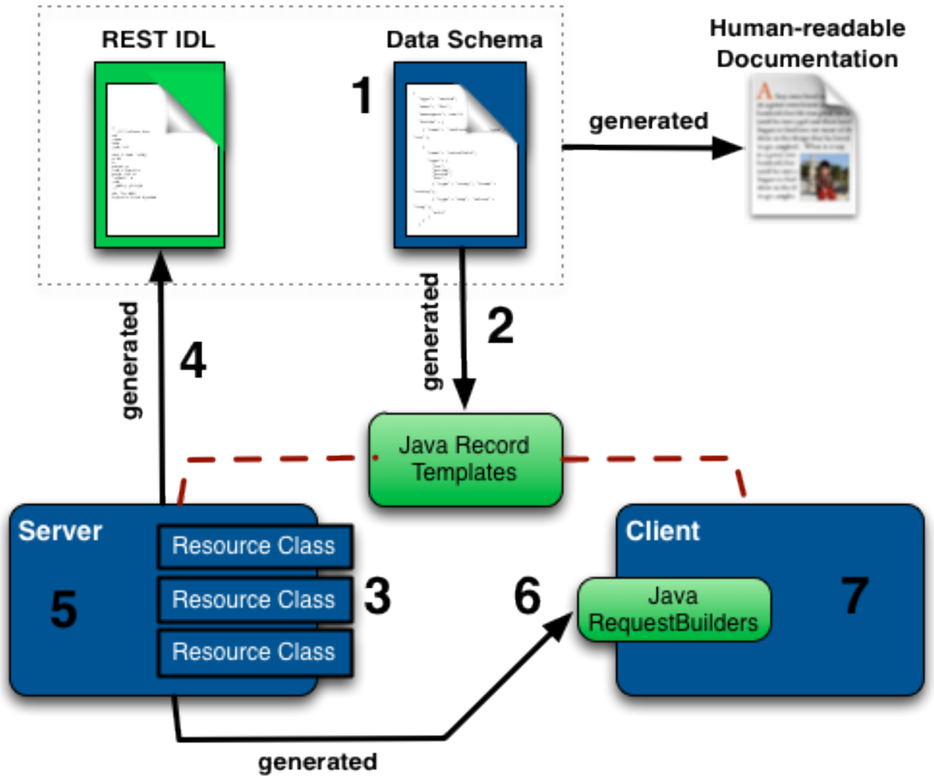
ParSeq

- Promise, Future + listener
- Task, similar to Callable, can be set result async
- Engines, Thread-pool based
- Context, task to sub-task

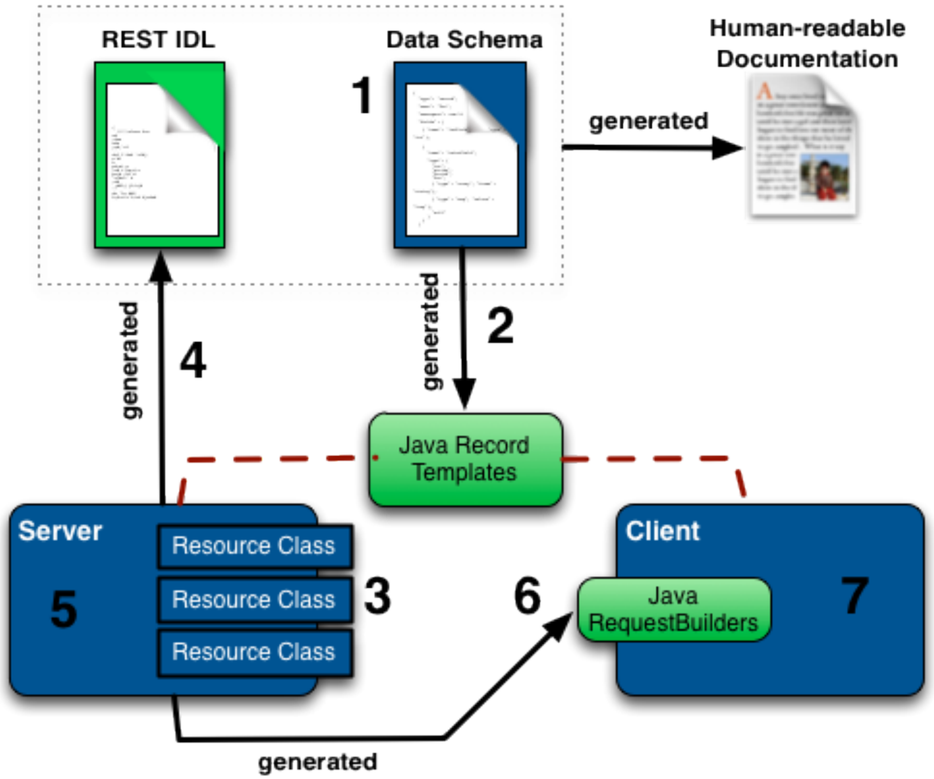
ParSeq



Workflow


REST IDL
Data Schema
1
Human-readable Documentation
generated
generated
4
generated
2
Java Record Templates
Server
Resource Class
Resource Class
Resource Class
5
3
6
Client
Java RequestBuilders
7
generated

Workflow


REST IDL
Data Schema
1
Human-readable Documentation
generated
generated
4
generated
2
Java Record Templates
Server
Resource Class
Resource Class
Resource Class
5
3
6
Client
Java RequestBuilders
7
generated

Workflow

Request

```
GET /fortunes/1 HTTP/1.1
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: ...

{
 "message": "Today's your lucky day!"
}
```

Fortune.pdsc

```
{
 "name" : "Fortune",
 "namespace" : "com.example",
 "type" : "record",
 "fields" : [
  { "name" : "message", "type" : "string" }
 ]
}
```

Generated Code

Fortune.java

```
public class Fortune extends
      RecordTemplate {
   String getMessage();
   void setMessage(String);
}
```

FortunesResource.java

```
@RestLiCollection(name = "fortunes")
class FortunesResource  implements KeyValueResource<Long, Fortune> {

   @RestMethod.GET
   public Fortune get(Long key) {
      return new Fortune()
         .setMessage("Today's your lucky day!");
   }
}
```

Workflow

Generated Code

fortunes.restspec.json

```
{
  "path": "/fortunes",
  "supports": [ "get" ],
  ...
}
```

Generated Code

FortunesBuilders.java

```
public class FortunesBuilders {
   GetRequestBuilder get();
}
```

ExampleClient.java

```
Response response = restClient.sendRequest(new FortunesBuilders.get().id(1L)).get();
Fortune fortune = response.getEntity();
System.out.println(fortune.getMessage());
```

FortuneClient.java

```
Task<Response<Fortune>> getFortune =
   parSeqClient.createTask(new FortunesBuilders.get().id(1L));

final Task<Void> printFortune =
   Tasks.action("printFortune", new Runnable() {
      @Override public void run() {
         Response<Fortune> response = getFortune.get();
         Fortune fortune = getResponseEntity();
         System.out.println(fortune.getMessage());
      }
});

engine.run(Tasks.seq(getFortune, printFortune));

printFortune.await();
```

Thanks