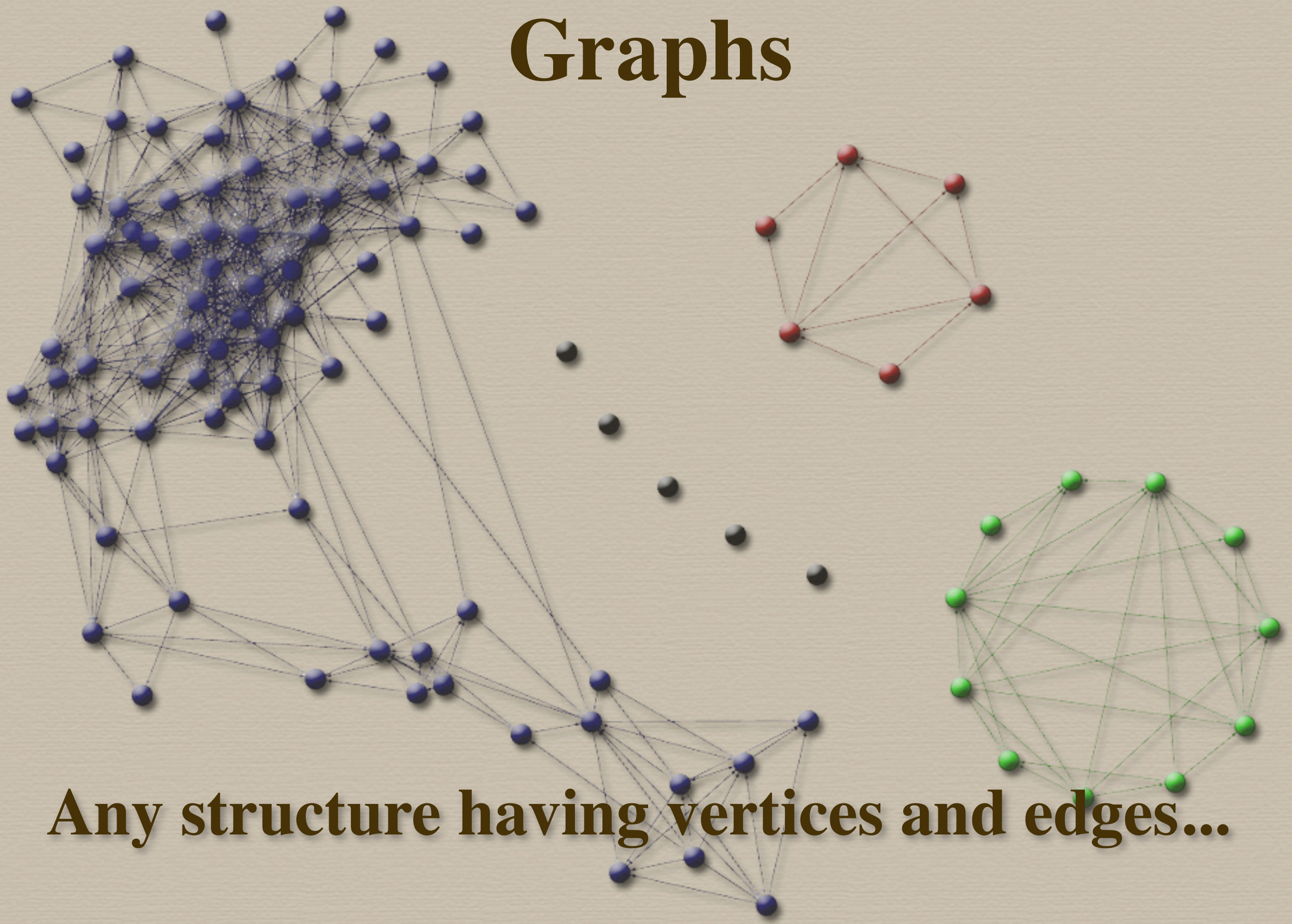
#madewithgephi

A Common Graph Database Access Layer for .NET and Mono

Achim Friedland <achim@graph-database.org>, Aperis GmbH

#madewithgephi

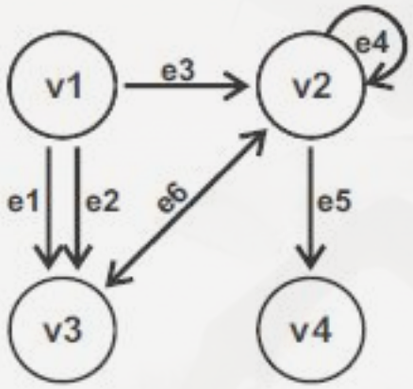
Graphs



Any structure having vertices and edges...

Photo: bombeador, flickr

...but different representations possible.



	v1	v2	v3	v4
v1	0	0	0	0
v2	1	1	1	0
v3	2	1	0	0
v4	0	1	0	0

	e1	e2	e3	e4	e5	e6
v1	1	1	1	0	0	0
v2	0	0	-1	2	1	1
v3	-1	-1	0	0	0	0
v4	0	0	0	0	-1	1

Adjacency matrix vs. Incidence matrix vs. Adjacency list vs. Edge list vs. Classes, Index-based vs. Index-free Adjacency, Dense vs. Sparse graphs, On-disc vs. In-memory graphs...

Photo: bombeador, flickr

The problem...

- Different graph representations...
  - have different levels of expressivity
  - can be very application specific
  - hard to optimize a single one for every use-case
- Multiple APIs from different vendors

#madewithgephi

So, where to find a common API?

Graph Commons

- **Edges are first-class citizens**
- **Information gathering by traversing**
  Indices are less important than in relational DBs
- **Index-free adjacency, so the cost of traversing an edge is in-depended from the size of the graph**
  (...at least when you ignore the GC ;)

The Property-Graph Model

The most common graph model within
the NoSQL GraphDB space

- **directed**: Each edge has a source and destination vertex
- **attributed**: Vertices and edges carry key/value pairs
- **edge-labeled**: The label denotes the type of relationship
- **multi-graph**: Multiple edges between any two vertices allowed

#madewithgephi

A Property Graph Model Interface for .NET and Mono

```
// Use a class-based in-memory graph
var graph = new InMemoryGraph();

var v1 = graph.AddVertex(new VertexId(1));
var v2 = graph.AddVertex(new VertexId(2));
v1.SetProperty("name", "Alice");
v1.SetProperty("age" ,  21);
v2.SetProperty("name", "Bob");
v2.SetProperty("age" ,  23);

var e1 = graph.AddEdge(v1, v2, new EdgeId(1), "Friends");
e1.SetProperty(“since”, ”2009/09/21”);
```

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin G=(V,E) Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Or, if you prefer the Dynamic Language Runtime...

```
// Use a class-based in-memory graph
var graph = new InMemoryGraph();

var v1    =  graph.AddVertex().AsDynamic();
var v2    =  graph.AddVertex().AsDynamic();
v1.name   = "Alice";
v1.age    =  21;
v2.name   = "Bob";
v2.age    =  23;

var e1 = graph.AddEdge(v1, v2, "Friends").AsDynamic();
e1.since = ”2009/09/21”;
```

Photo: jonathanpercy, flickr

Current status...

- **About 70% of the current JAVA original**
- **e.g. Transactions, Auto-Indexing still missing**
- **Expect first release within the next 4-6 weeks**

Photo: jonathanpercy, flickr

github
SOCIAL CODING

The future…

- **Different implementations for different graph representations**
- **Implementations for remote graphs**
  (Rexster and Neo4J REST in separate projects)
- **Extensions, e.g. for semantic graphs**

#madewithgephi

Ok, but how to query a property-graph?

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Pipes.NET

A data flow framework for property graph models

: IEnumerator<E>, IEnumerable<E>

S

Source Elements

AbstractPipe<S, E>

E

Emitted Elements

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Pipes.NET

A simple pipe yielding all objects without modification

```
public class IdentityPipe<S> : AbstractPipe<S, S> {

    public override Boolean MoveNext() {
        if (_InternalEnumerator == null)
            return false;

        if (_InternalEnumerator.MoveNext()) {
            _CurrentElement = _InternalEnumerator.Current;
            return true;
        }

        return false;

    }

}
```

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Pipes.NET

A simple pipe yielding all strings turned to uppercase

```
public class ToUpperPipe<String> : AbstractPipe<String, String> {

    public override Boolean MoveNext() {

        if (_InternalEnumerator == null)
            return false;

        if (_InternalEnumerator.MoveNext()) {
            _CurrentElement = _InternalEnumerator.Current.ToUpper();
            return true;
        }

        return false;

    }

}
```

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Pipes.NET

A simple pipe yielding the vertex property “name”

```
public class GetNameProperty : AbstractPipe<IVertex, String> {

    public override Boolean MoveNext() {

        if (_InternalEnumerator == null)
            return false;

        if (_InternalEnumerator.MoveNext()) {
            _CurrentElement = _IVertex.GetProperty(“name”);
            return true;
        }

        return false;

    }

}
```

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

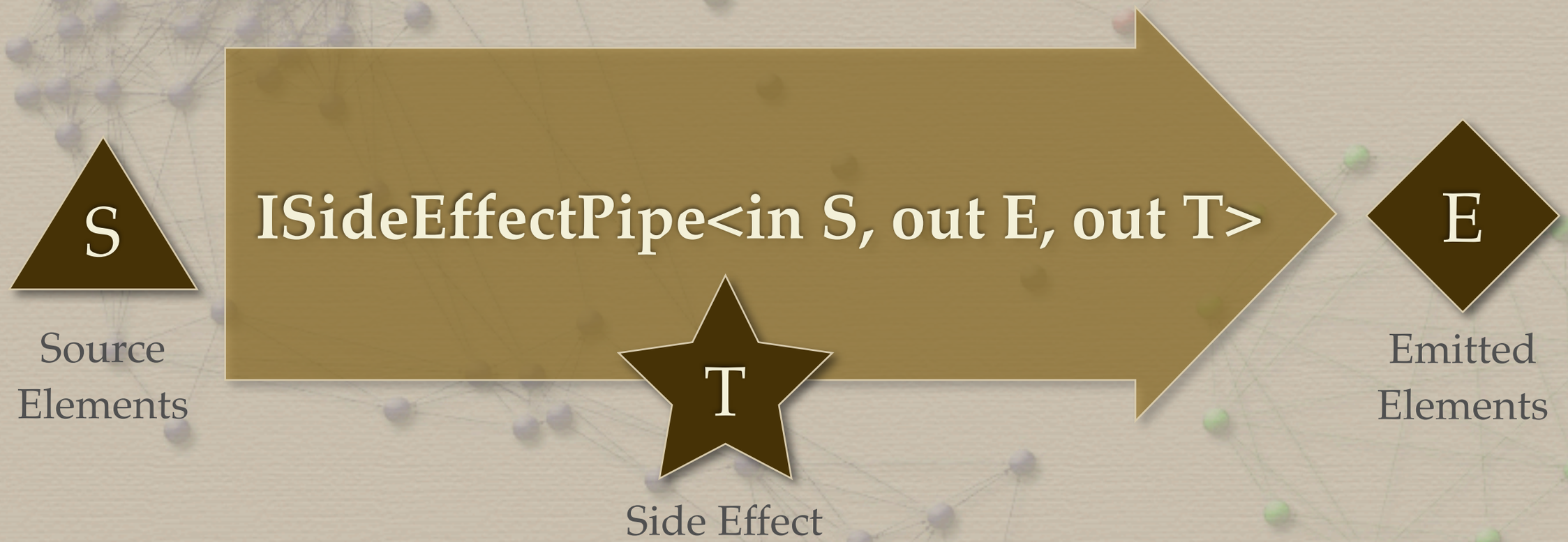
Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Pipes.NET

Allow the creation of state or side effects within a pipe



Source Elements

Emitted Elements

Side Effect

Count the total number of elements passed through the pipe

```
public class CountPipe<S> : AbstractPipe<S,S>, ISideEffectPipe<S,S,Int64 {
    [...]
    public override Boolean MoveNext() {

        if (_InternalEnumerator == null)
            return false;

        if (_InternalEnumerator.MoveNext()) {
            _CurrentElement = _InternalEnumerator.Current;
            Interlocked.Increment(ref _Counter);
            return true;
        }

        return false;

    }

    public Int64 SideEffect { get { return _Counter; }}

}
```

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

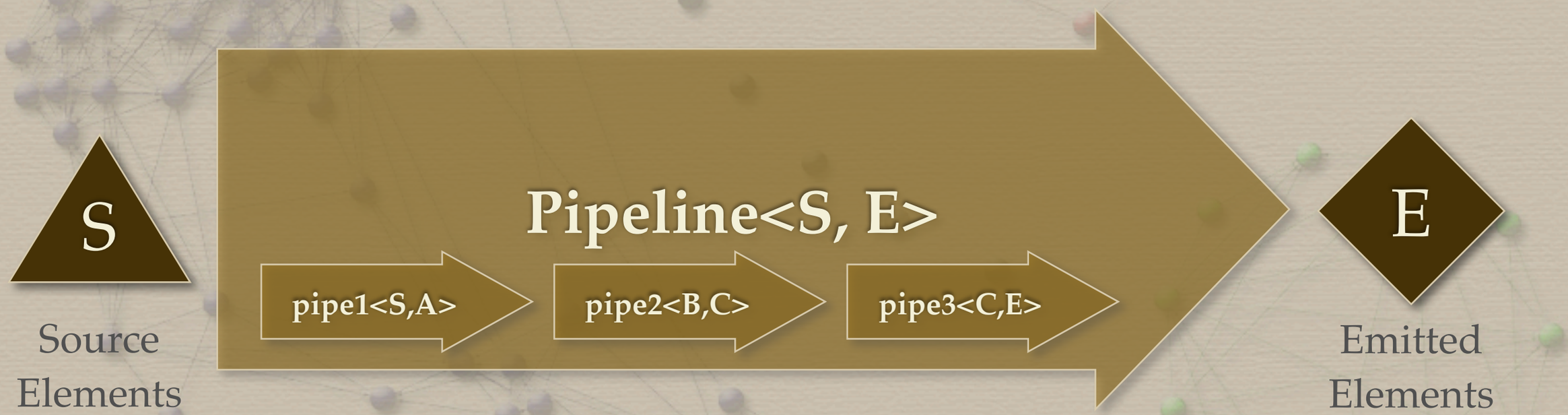
Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Pipes.NET

Create complex pipes by combining pipes to pipelines



#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Pipes.NET

A data flow framework for property graph models

```
// Friends-of-a-friend
var pipe1  = new VertexEdgePipe(VertexEdgePipe.Step.OUT_EDGES);
var pipe2  = new LabelFilterPipe("Friends", ComparisonFilter.EQUALS);
var pipe3  = new EdgeVertexPipe(EdgeVertexPipe.Step.IN_VERTEX);
var pipe4  = new VertexEdgePipe(VertexEdgePipe.Step.OUT_EDGES);
var pipe5  = new LabelFilterPipe("Friends", ComparisonFilter.EQUALS);
var pipe6  = new EdgeVertexPipe(EdgeVertexPipe.Step.IN_VERTEX);
var pipe7  = new PropertyPipe("name");

var pipeline = new Pipeline(pipe1,pipe2,pipe3,pipe4,pipe5,pipe6,pipe7);
pipeline.SetSource(new SingleEnumerator(
                   graph.GetVertex(new VertexId(1))));

foreach (var _foaf in pipeline.Take(5)) {
  Console.WriteLine(_foaf);
}
```

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Pipes.NET

A data flow framework for property graph models

- **Think of "LINQ for graphs"**
- **...but without the syntactic sugar.**
- **Very Powerful, but also very "noisy" :(**

Photo: jonathanpercy, flickr

Pipes.NET

github SOCIAL CODING

tinkerpop.com

Current status & the future

- **About 95% of the current JAVA original**
- **Expect first release with the next weeks**

#madewithgephi

github
SOCIAL CODING

And what about ad hoc querying for exploring a property-graph?

Photo: jonathanpercy, flickr

Gremlin

$G = (V, E)$

github
SOCIAL CODING

tinkerpop.com

Ad hoc Query Language for graphs

- **Ad Hoc querying a property graph**
- **User-friendly wrapper around pipes**
- **“perl for graphs” ;)**

Photo: jonathanpercy, flickr

GET http://{host}/{resource}

reXster.NET

Gremlin $G = (V, E)$

Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Gremlin

$G = (V, E)$

Ad hoc Query Language for graphs

```
// Friends-of-a-friend
var pipe1  = new VertexEdgePipe(VertexEdgePipe.Step.OUT_EDGES);
var pipe2  = new LabelFilterPipe("Friends", ComparisonFilter.EQUALS);
var pipe3  = new EdgeVertexPipe(EdgeVertexPipe.Step.IN_VERTEX);
var pipe4  = new VertexEdgePipe(VertexEdgePipe.Step.OUT_EDGES);
var pipe5  = new LabelFilterPipe("Friends", ComparisonFilter.EQUALS);
var pipe6  = new EdgeVertexPipe(EdgeVertexPipe.Step.IN_VERTEX);
var pipe7  = new PropertyPipe("name");

var pipeline = new Pipeline(pipe1,pipe2,pipe3,pipe4,pipe5,pipe6,pipe7);
pipeline.SetSource(new SingleEnumerator(
                    graph.GetVertex(new VertexId(1))));
```

g:id-v(1)/outE[@label='Friends']/inV/outE
[@label='Friends']/inV/@name

Photo: jonathanpercy, flickr

Current status & the future

- **Reimplementation not yet started :(**
- **Will probably be based on the DLR**
- **Expect first release with the next 2-3 months**

Photo: jonathanpercy, flickr

But...

- **Do <u>not</u> expect Gremlin as the one-and-only query language for each use-case!**
- **More application specific Query Languages could be implemented using pipes**
  e.g. sones GQL, OrientDB SQL

#madewithgephi

github
SOCIAL CODING

How to access a remote graph?

How to expose a graph via the network?

#madewithgephi

github SOCIAL CODING

A HTTP/REST interface for Blueprints Property Graph Models

- **Exposes your application-embedded Blueprints graph via HTTP/REST**
- **Vertices and edges are REST resources**
- **The rexster client library allows you to access most JAVA-based GraphDBs**

  Neo4J, OrientDB available; DEX, InfiniteGraph coming soon...

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

reXster.NET

Common CRUD Operations…

GET Operations

returns	uri	description
graphs	`http://base`	get all the graphs
graph	`http://base/graph`	get the graph
vertices	`http://base/graph/vertices`	get all vertices
vertex	`http://base/graph/vertices/1`	get vertex with id `1`
edges	`http://base/graph/edges`	get all edges
edge	`http://base/graph/edges/1`	get edge with id `1`
edges	`http://base/graph/vertices/1/outE`	get the out edges of vertex `1`
edges	`http://base/graph/vertices/1/inE`	get the in edges of vertex `1`
edges	`http://base/graph/vertices/1/bothE`	get the both in and out edges of vertex `1`
edges	`http://base/graph/vertices/1/xxxE?_label=written_by`	get the edges of vertex `1` with edge label equal to "written_by"
indices	`http://base/graph/indices`	get all the indices associated with the graph
elements	`http://base/graph/indices/index?key=k1&value=v1`	get all elements with `k1` property equal to `v1` in `index`
strings	`http://base/graph/indices/index/keys`	get the auto index keys of the automatic index named `index`

#madewithgephi

github SOCIAL CODING

Common CRUD Operations…

POST Operations

returns	uri	description
vertex	`http://base/graph/vertices`	create a vertex with no specified identifier
vertex	`http://base/graph/vertices/1`	create a vertex with id `1`
vertex	`http://base/graph/vertices/1?k1=v1&k2=v2`	create a vertex with id `1` and the provided properties (or update vertex properties if vertex already exists).
edge	`http://base/graph/edges?_outV=1&_label=friend&_inV=2&k1=k2`	create an out edge with no specified identifier from vertex `1` to vertex `2` labeled "friend" with provided properties.
edge	`http://base/graph/edges/3?_outV=1&_label=friend&_inV=2&k1=k2`	create an out edge with id `3` from vertex `1` to vertex `2` labeled "friend" with provided properties.
edge	`http://base/graphname/edges/3?k1=k2`	update the properties of the edge with id `3`
index	`http://base/graph/indices/index?class=vertex&type=automatic`	create a new automatic vertex index named `index`
void	`http://base/graph/indices/index?key=k1&value=v1&class=vertex&id=1`	put vertex with `id` 1 into `index` at `k1/v1`
void	`http://base/graph/indices/index?k1&k2`	add auto index keys `k1` and `k2` to index named `index`

#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

reXster.NET

Common CRUD Operations…

DELETE Operations

returns	uri	description
void	`http://base/graph/vertices/1`	remove vertex `1`
void	`http://base/graph/vertices/1?key1&key2`	remove properties `key1` and `key2` from vertex `1`
void	`http://base/graph/edges/3`	remove the edge with id `3`
void	`http://base/graph/edges/3?key1&key2`	remove properties `key1` and `key2` from edge `3`
void	`http://base/graph`	clear the graph of all its elements and indices
void	`http://base/graph/indices/index`	drop the index named `index`
void	`http://base/graph/indices/index?key=k1&value=v1&class=vertex&id=1`	remove the vertex 1 from `index` at `k1/v1`
void	`http://base/graph/indices/index?k1&k2`	remove the auto index keys `k1` and `k2` from auto index `index`

#madewithgephi

Default resource representation: JSON

```
curl -H Accept:application/json http://localhost:8182/graph1/vertices/1
{
  "version" : "0.1",
  "results" : {
    "_type" : "vertex",
    "_id"   : "1",
    "name"  : "Alice",
    "age"   : 21
  },
  "query_time" : 0.014235
}
```

Photo: jonathanpercy, flickr

github
SOCIAL CODING

Current status

- **About 60% of the current JAVA original**
- **Expect first release with the next 3-4 weeks**
- **Rexster Shell might take some more time...**

Photo: jonathanpercy, flickr

github
SOCIAL CODING

Future Resource Representations

- **Link-aware, self-describing hypermedia (see Neo4J)**
  - **e.g. ATOM, XML + XLINK, RDFa**
- **Application specific protocols**
  - **GEXF for exporting a graph to GEPHI**
  - **GraphML/GDF/... graph formats**
  - **MediaRSS your photo graph traversal via Cooliris**

#madewithgephi

GET http://{host}/{resource}

reXster.NET

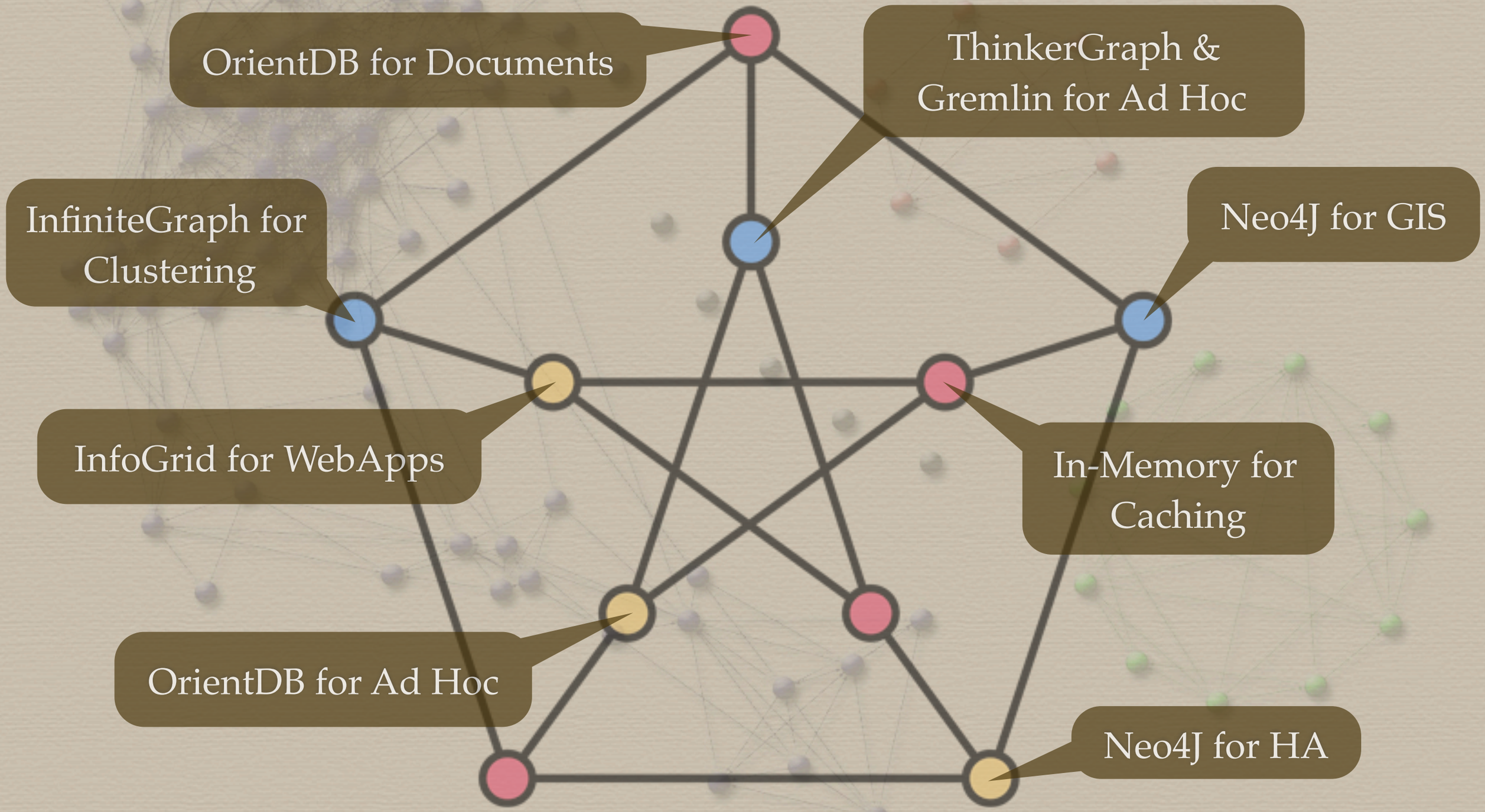
Gremlin G=(V,E) Pipes.NET

Blueprints.NET

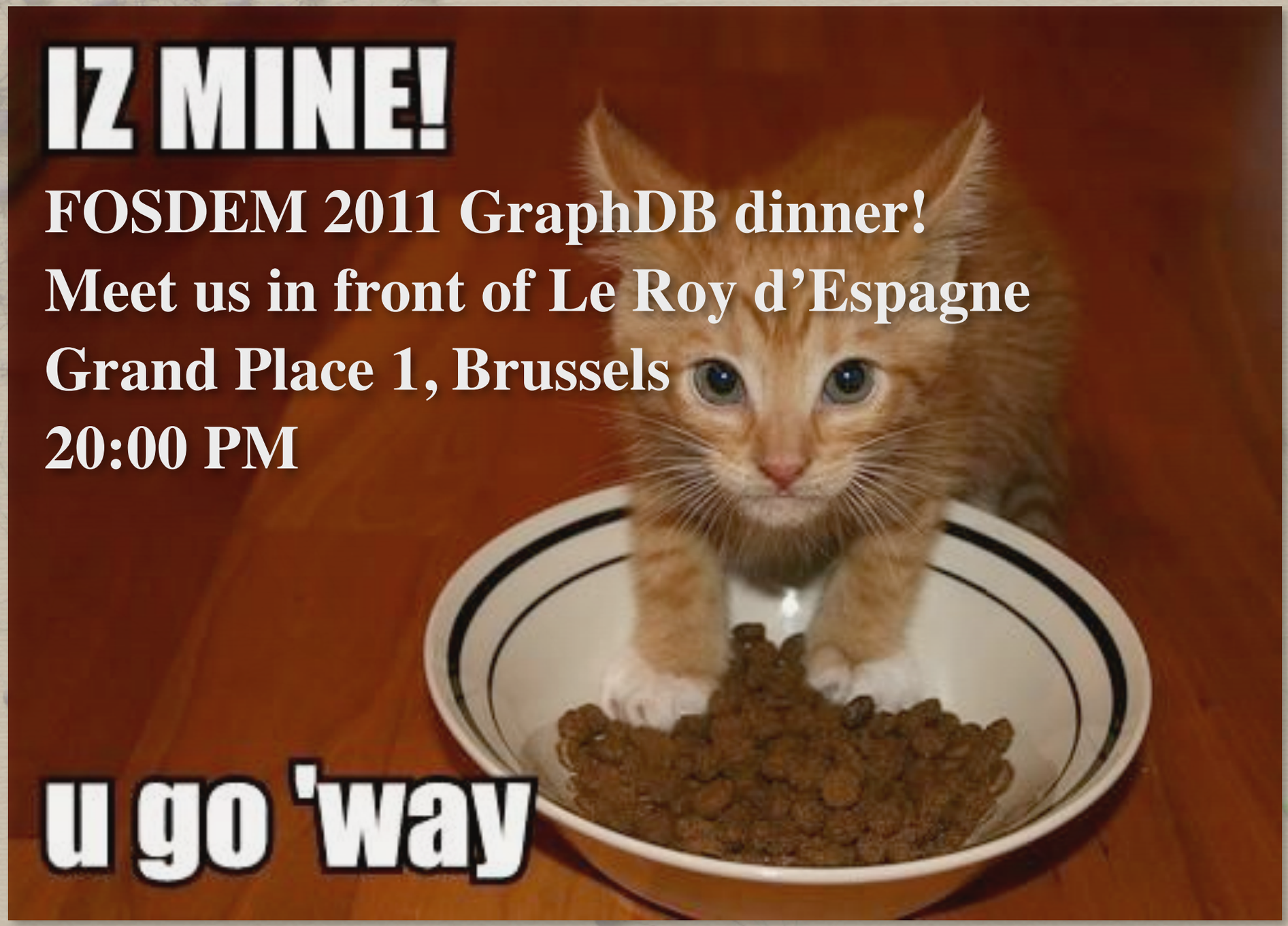
github SOCIAL CODING

tinkerpop.com

The GraphDB Graph...



Still hungry for more?



#madewithgephi

GET http://{host}/{resource}

reXster.NET

Gremlin G=(V,E)

Pipes.NET

Blueprints.NET

github SOCIAL CODING

tinkerpop.com

Questions?

http://www.graph-database.org
http://www.twitter.com/graphdbs