# RDF as a Data Structure for Software Engineering

Axel Rauschmayer
Institut für Informatik
LMU München

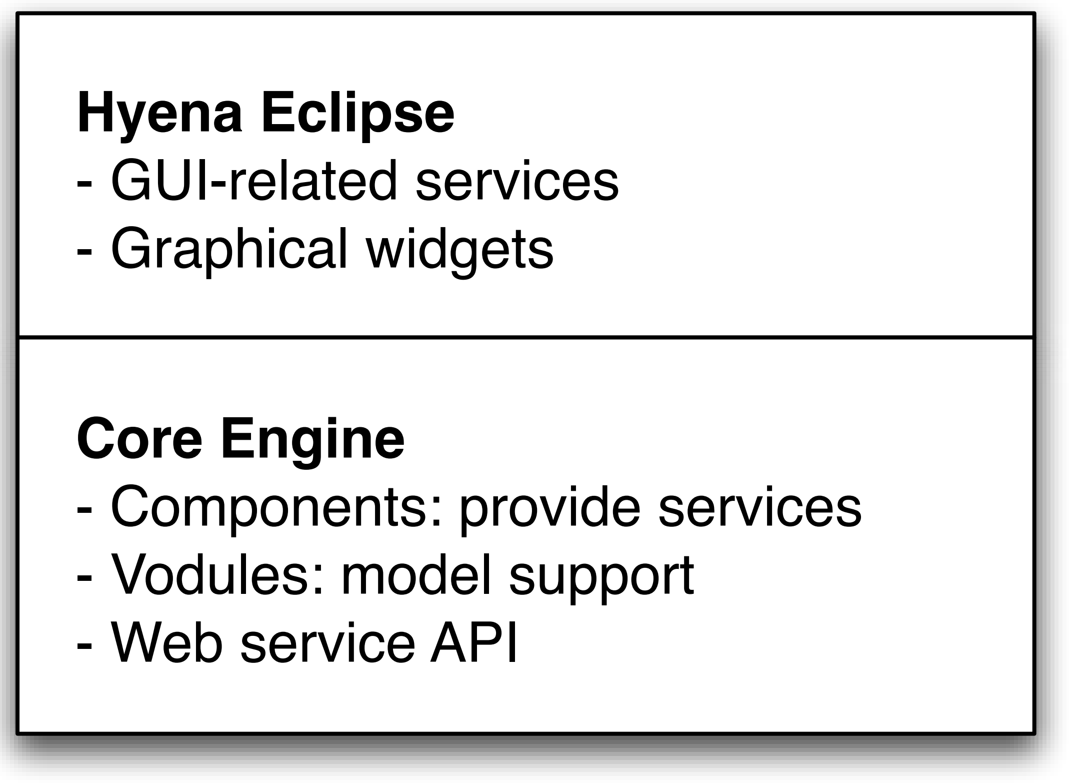
ISWC 2005

Galway, Ireland, Nov 6, 2005

# Hyena

- A platform for RDF-based software engineering.
- Integrates different kinds of SW engineering-related models within the same RDF graph.
- Manages a tree of RDF graphs.
- Editing happens via the web or via a GUI.

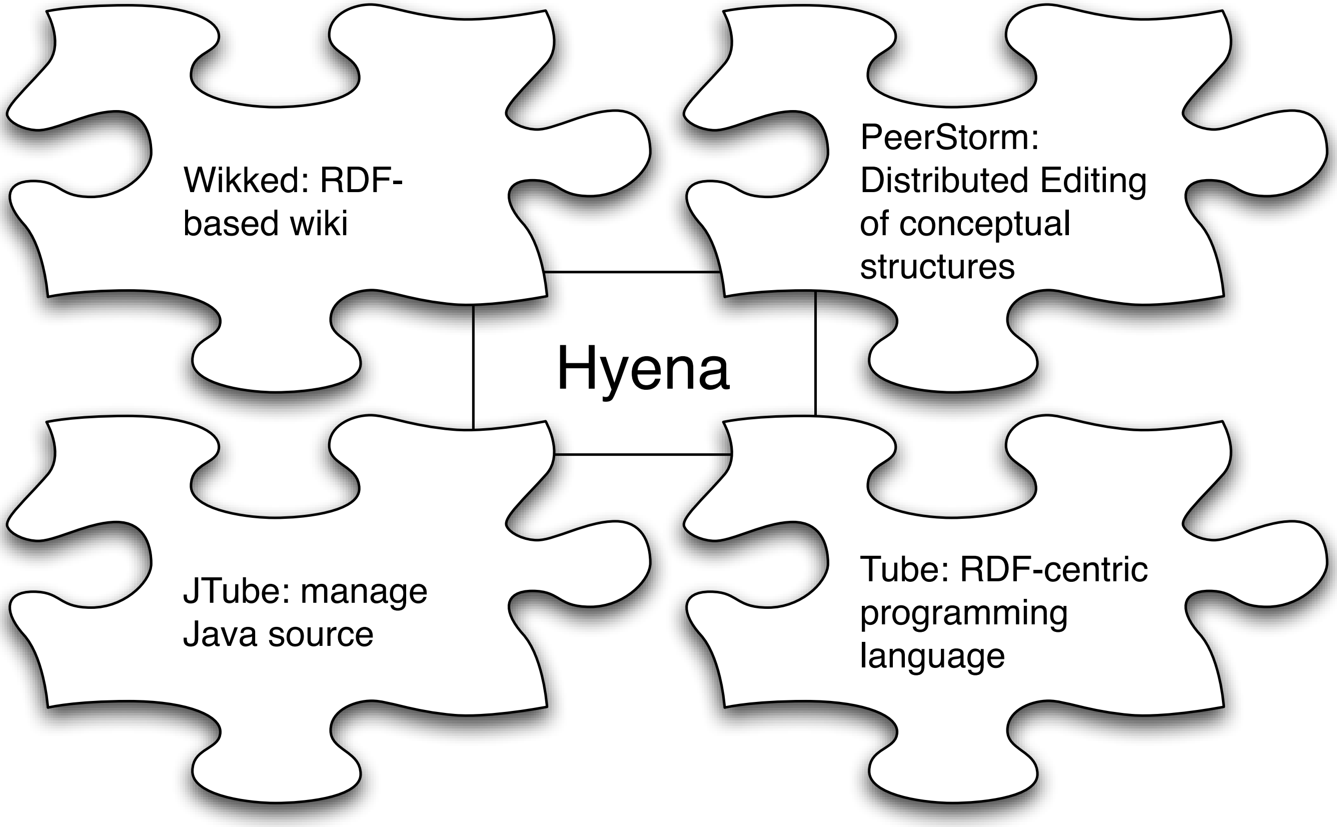
# Overview: Hyena Platform



**Hyena Eclipse**
- GUI-related services
- Graphical widgets

**Core Engine**
- Components: provide services
- Vodules: model support
- Web service API

# Overview: Vodules



Wikked: RDF-based wiki

PeerStorm: Distributed Editing of conceptual structures

Hyena

JTube: manage Java source

Tube: RDF-centric programming language

# Core Engine

Basic infrastructure.

- Components: provide services
- Vodules (plug-ins):
  - support for specific models
  - can be switched on and off
  - register contributions with components
  - contribute (declarative) RDF data, API functionality, GUI widgets (in conjunction with Hyena Eclipse)
- Web service API based on ReST (representative state transfer)

# Hyena Eclipse

Components:

- Registry for node visualization widgets
- Shelf (bookmarks): nodes, triples, node sets, triple sets
- Filtering: vodules can plug-in model-specific filter operations
- etc.

Views:

- (Filtering) Nodes, Triples
- Predicates, Shelf

# Commonalities with other Participants

- RDF-based wiki problems: e.g. syntax, integration of external data
- Ontology-based guided customization/extension of frameworks
- RDF-object mapping

# Wish List

- Specialized RDF DB, preferably pure Java
- Named graphs: simulated in Hyena for preferences etc.
- Reification: cannot really reference single triples, topic maps are better here
- Query pipelines: build queries programmatically and incrementally, pipes and filter metaphor

# RDF Experiences: Modeling

- Hide URIs: we try to avoid namespaces and use `rdfs:label` a lot. Compare: topic maps.
- Snippets (PeerStorm vodule) are simulated topic maps
  - inline-properties vs. associations (typed via subPredicate relation. Common solution?)

# RDF Experiences: Programming

- Java: *traits* for RDF/OO mapping
- Java: state-less nodes; constants and dynamically created values have same structure—as opposed to a `Resource` in Jena
  (consequence: store mapped objects externally).
- Constant management:
  - There are multiple dimensions: kind of entity, contributor etc. We are experimenting with tables (where every column is one dimension of categorization).
  - Consistency problems between source code and RDF data.

# Vodule: Wikked

- Wiki pages reside in an RDF graph

- A wiki page can publish RDF data

- Benefit: easy meta-data via RDF

- Wiki markup: layered syntax:
  - Core syntax with a clear grammar.
  - Actual (line-based) wiki markup is translated to core syntax.
  - Concise core markup language makes Wikked very extensible.

# Vodule: PeerStorm

- Distributed Editing (via Jabber protocol). . .
- . . . of conceptual structures
- Encoded as *snippets* (similar to topics) and relations.
- Use case: edit/convert/publish record-based data (BibTeX, bookmarks)
- Use case: design sketches, outlines

# Vodule: JTube

- Tag Java source code locations with RDF nodes and jump back and forth between code and RDF

- JTube automatically tracks changes in the source

- Also works with whole files (of arbitrary type)

# Vodule: Tube

- A completely new programming language
- RDF integration at the meta level: use RDF to structure the code, method IDs are RDF nodes.
- RDF integration at the object level: multiple dispatch on RDF types
- Interactive creation of RDF data and source code
- Work in progress