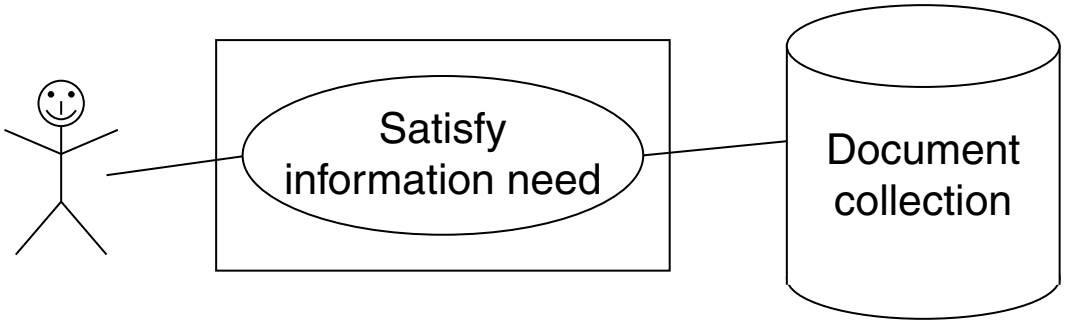
TIRA: An MDA Approach to Implement Personal IR Tools

Sven Meyer zu Eissen and Benno Stein

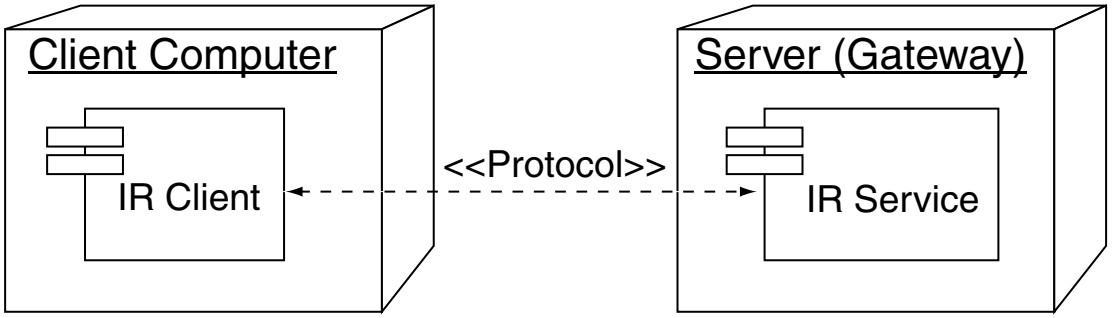
Bauhaus University Weimar
Web-Technology and Information Systems

IR in Distributed Environments

IR use case:



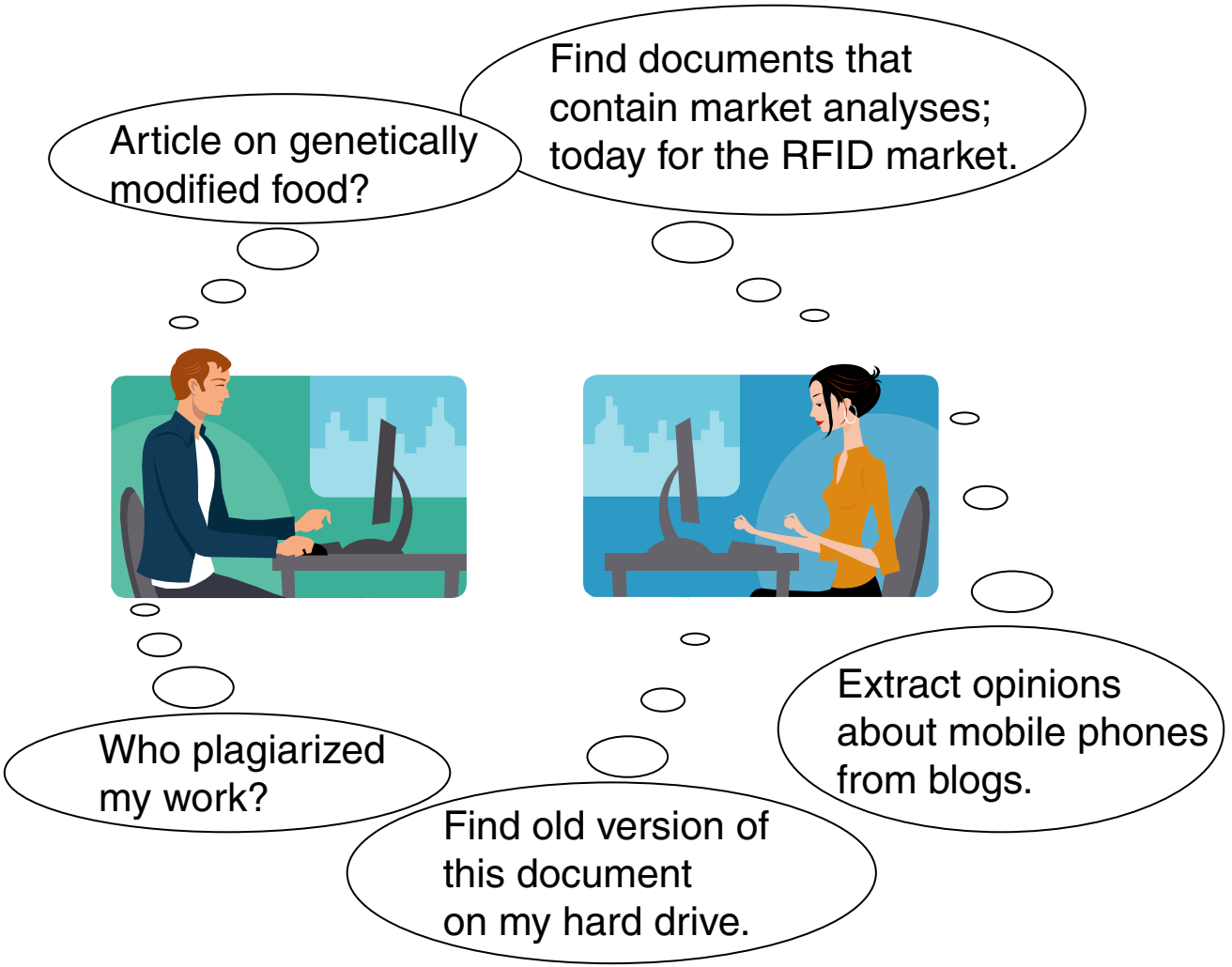
Realization of multi-user IR systems:



e.g. Browser

e.g. Yahoo! search,
HyperWave server,
Google desktop search

Personal Information Needs (The Client Side)



Operationalization of IR Tasks

List of wishes: An IR system should

- adapt to personal data
- adapt to personal preferences (e.g. result presentation)
- adapt to personal skills (e.g. query formulation)
- adapt to personal knowledge (e.g. about collection)
- *adapt to personal IR tasks*

Operationalization of IR Tasks

List of wishes: An IR system should

- adapt to personal data
- adapt to personal preferences (e.g. result presentation)
- adapt to personal skills (e.g. query formulation)
- adapt to personal knowledge (e.g. about collection)
- *adapt to personal IR tasks*

Today:

A query is almost always formulated in the form of keywords.

The IR process is hard-wired *at the server side*.

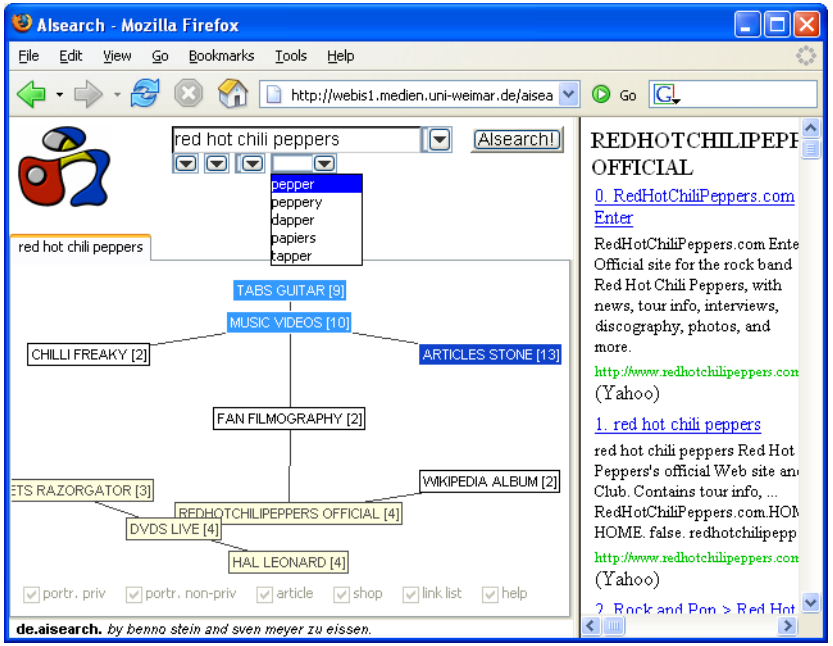
Tomorrow (with TIRA):

A query can be an IR process specification

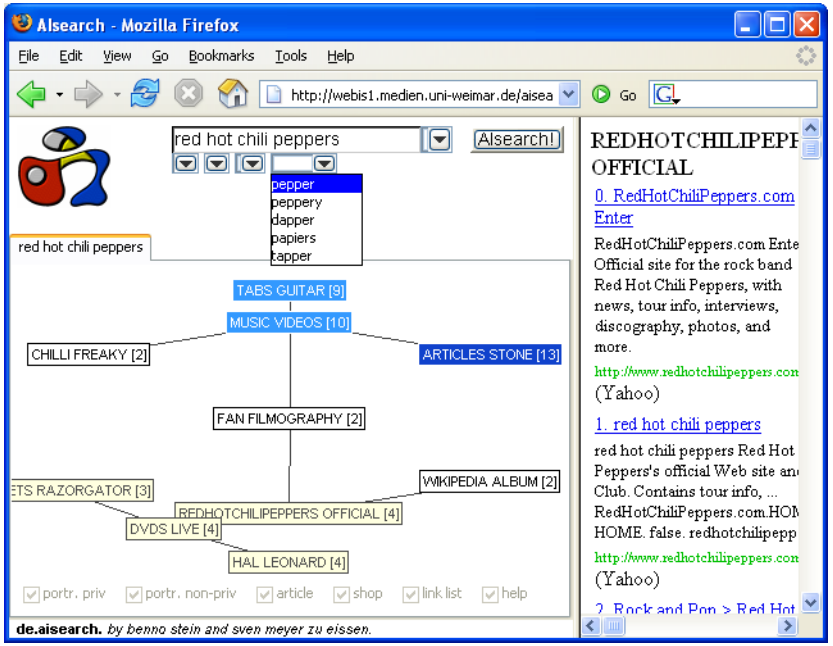
(soft-coded *at the client side*).

What are the building blocks of an IR process?

Example IR Task: Categorizing Search



Example IR Task: Categorizing Search



Required are modules for

- importing various formats (HTML, PDF,...),
- language detection,
- stemming, stopword identification,
- clustering (k-means, MajorClust,...), cluster labeling,
- classification (discriminant analysis, SVMs,...),
- ...

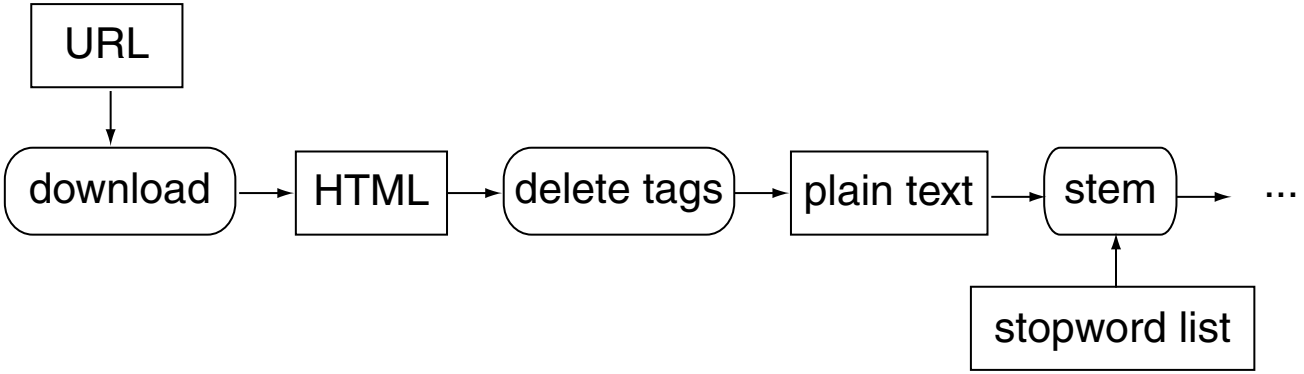
Example IR Task (Simplified)

For the sake of simplicity we regard the following IR process:

1. Download an HTML document from a URL.
2. Build document representation according to topic.
3. Build document representation according to genre.
4. Classify according to topic and genre.

Key question: How can such an IR process be specified?

The Modular Nature of IR Processes



Some characteristics:

- An IR process is a sequence of transformations.
- IR processes: composed of autonomous building blocks.
- IR theory: different solutions for the same task (e.g. stemming, categorization; cf. Strategy Pattern).
- One base algorithm for similar tasks (e.g. stemming; cf. Factory Pattern).
- IR processes are subject to frequent change (optimization, new ideas, changing information needs).
- IR process subtasks may be executed in parallel.
- Set of useful standard modules for any application.

Specification of IR Processes

Standard solution (as we call it): the “Library Approach”.

- ➜ Design generic interfaces.
- ➜ Build software libraries.
- ➜ Build special-purpose application.

Specification of example task as code:

Input: URL `u`, dictionary `dict`, stopword list `stl`.
Output: genre and topic class for the document at URL `u`.

```
Text ht=download(u);
Text plain=removeHTMLTags(ht);
Text filtered=removeStopwords(plain, stl);
Features topicModel=
           buildTopicModel(filtered, dict);

Language lang=detectLanguage(plain);
Features presentF=buildPresentationF(ht);
Features posF=buildPOSF(plain, language);
Features genreModel=union(presentF, posF);

int topicClass=classifyTopic(topicModel);
int genreClass=classifyGenre(genreModel);

return(topicClass, genreClass);
```

Specification of IR Processes

Drawbacks of the library approach:

- Needs expert knowledge in specification language / libraries
- Changing the process is tedious and error-prone.

More flexible, more abstract, more expressive:

Diagrammatic language that specifies the data / control flows.

Specification of IR Processes

Drawbacks of the library approach:

- Needs expert knowledge in specification language / libraries
- Changing the process is tedious and error-prone.

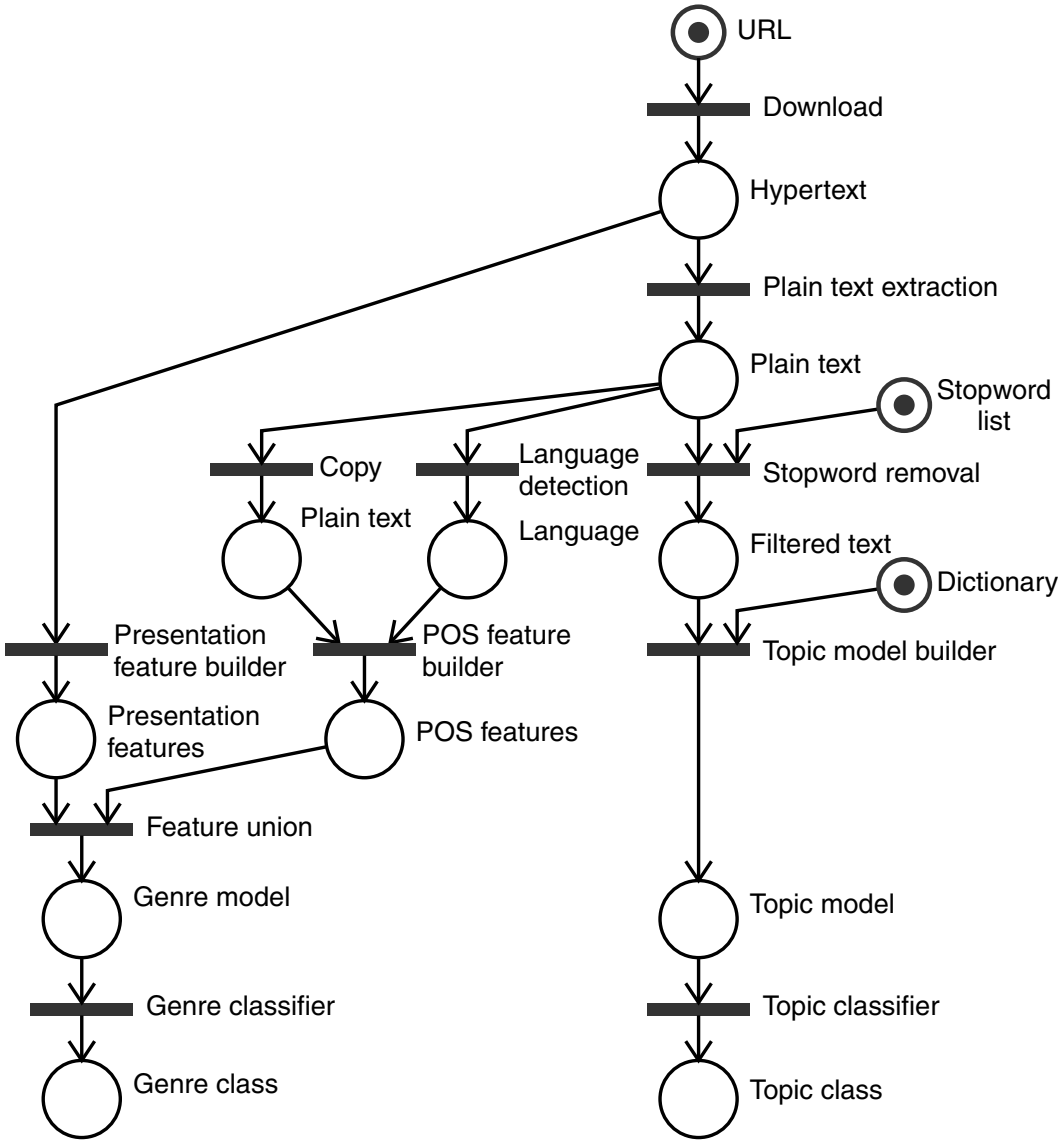
More flexible, more abstract, more expressive:

Diagrammatic language that specifies the data / control flows.

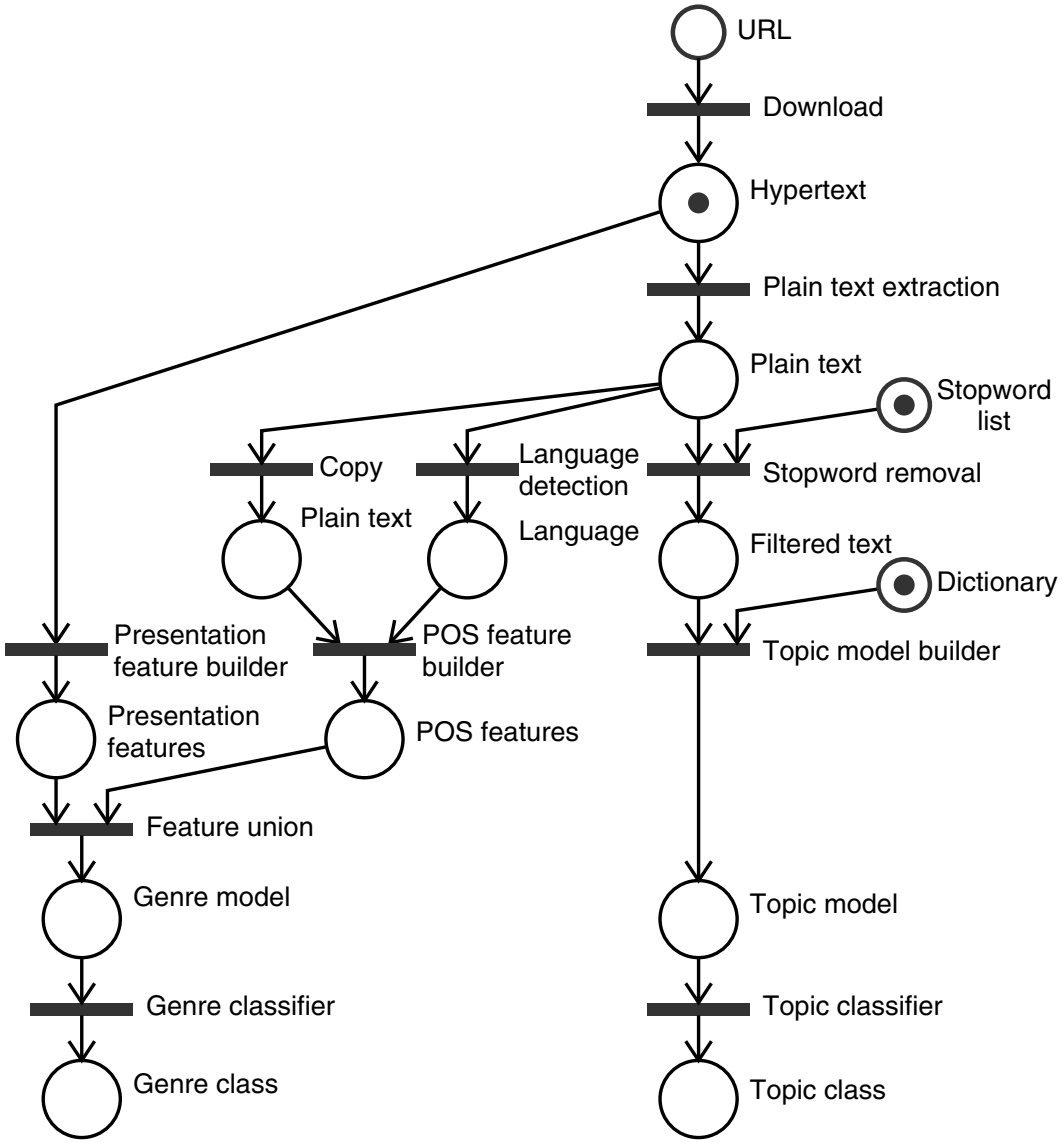
Classification of diagrammatic modeling tools: [Teich 1997]

- Control flow dominant or state oriented (FSM, state charts)
- Data flow dominant or activity oriented
  (Petri nets, UML activity diagrams, marked graphs)
- Structure oriented (UML class diagrams)
- Time oriented (UML time diagrams)
- Data oriented (ER diagrams)
- hybrid

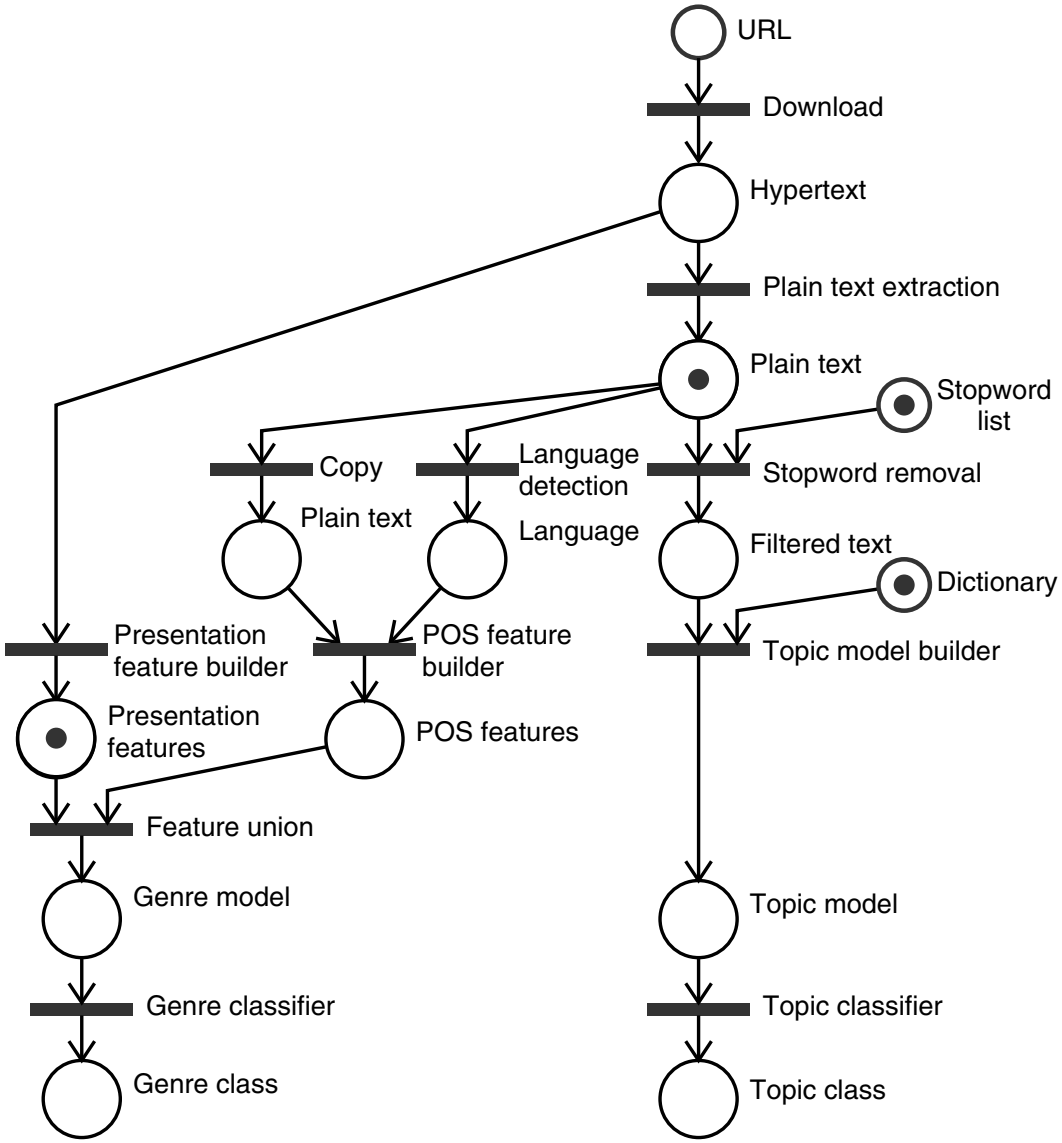
Petri Net Specification of Sample Task



Petri Net Specification of Sample Task



Petri Net Specification of Sample Task



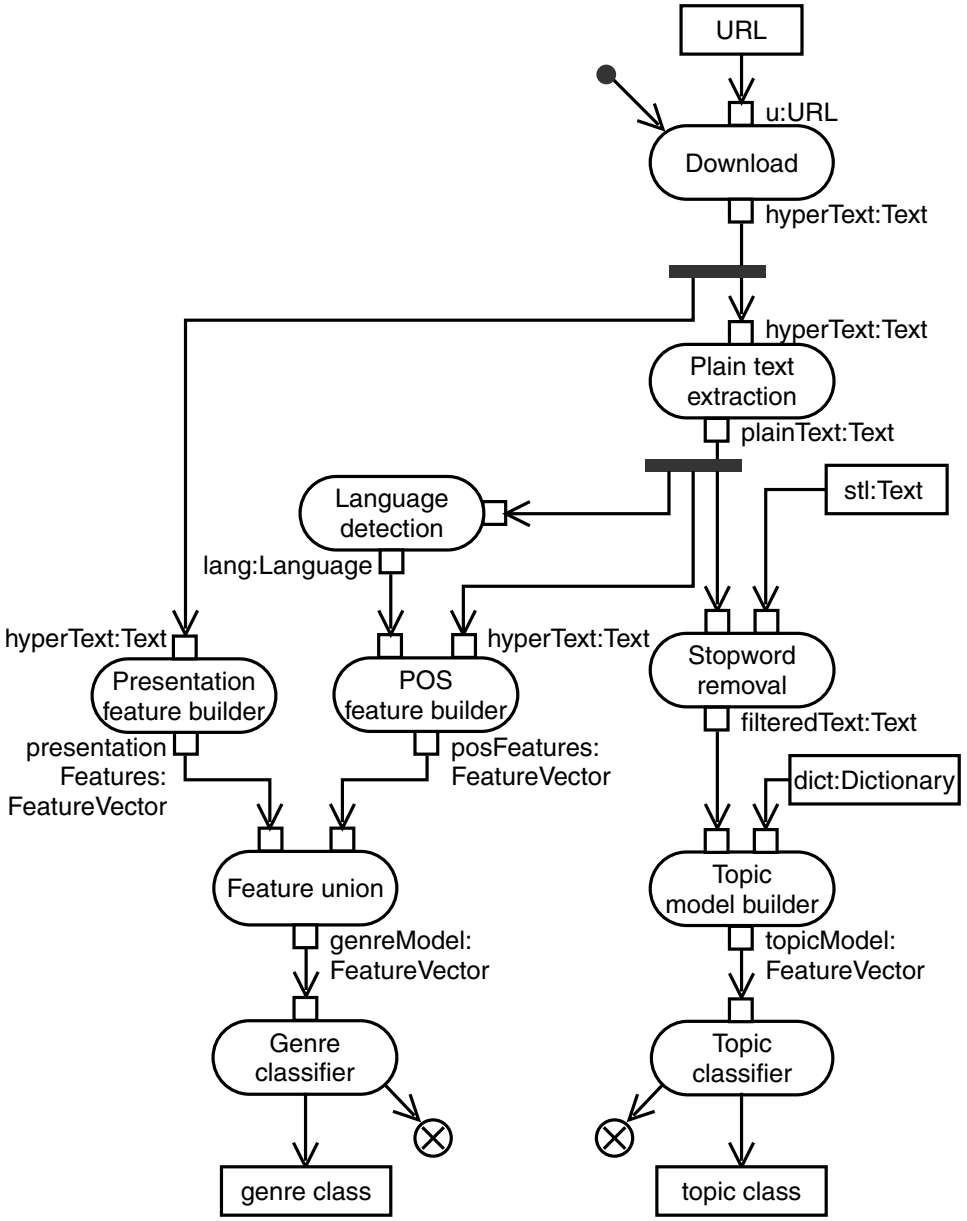
Petri Net Specification

Discussion

+ Petri nets are well researched.
+ Modeling concurrency is possible.

- Data types cannot be modeled.
- Modeling iterations is complicated.
- Iterations based on the “content” of tokens are impossible.
- Scheduling policy for the places cannot be specified.

UML Activity Diagram Specification of Sample Task



UML Activity Diagram Specification

Discussion

+ Intuitive and widely accepted.
+ Modeling of iterations, concurrency, and data types.
+ Advanced concepts like exception handling, streams,...
+ Diagrams are updated frequently.

- “Simulation” unclear.

Operationalizing IR Processes with TIRA

Activity diagrams are *independent from*

- programming languages,
- operating systems,
- middleware platforms,
- system architectures.

➜ An activity diagram is a platform independent model (PIM).

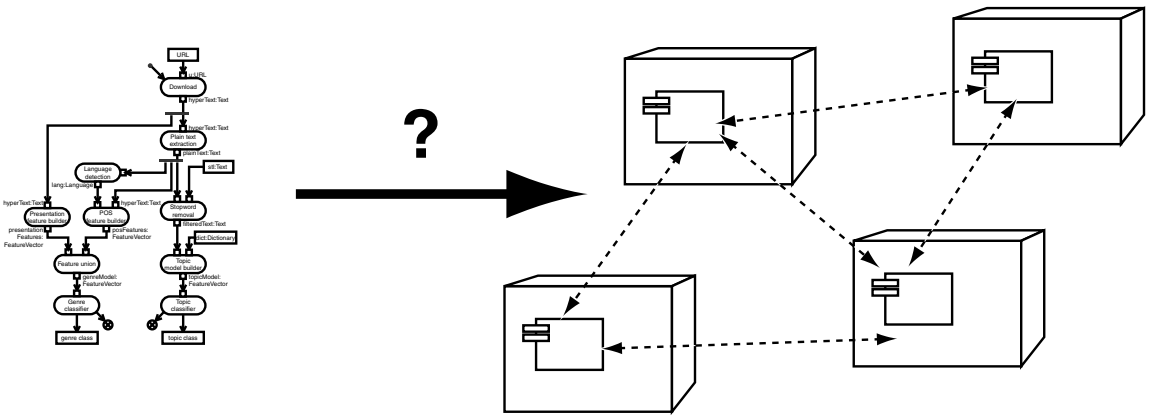
Operationalizing IR Processes with TIRA

Activity diagrams are *independent from*

- programming languages,
- operating systems,
- middleware platforms,
- system architectures.

➜ An activity diagram is a platform independent model (PIM).

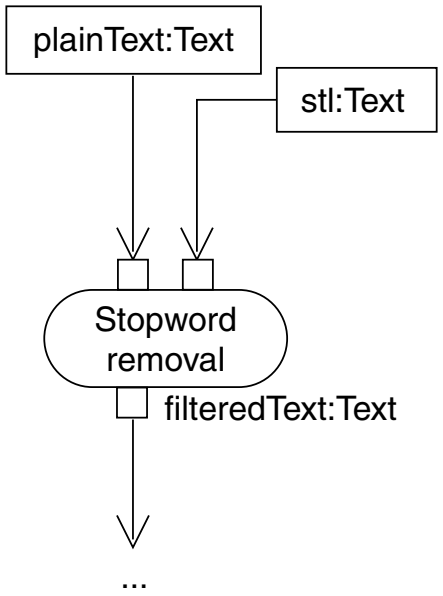
Required for execution (in terms of MDA):
A sequence of transformations to a platform specific model (PSM).



Operationalizing IR Processes with TIRA

TIRA solution:

- Encapsulate library functions as Web services.
- Specify data types with XML schema. Serialize data as XML. Visualize data with XSLT.
- For global access: data are published under a certain URL.
- Simulate the activity diagram:
  Execute Web services with the data URLs as parameters.



Client code
(Platform specific):

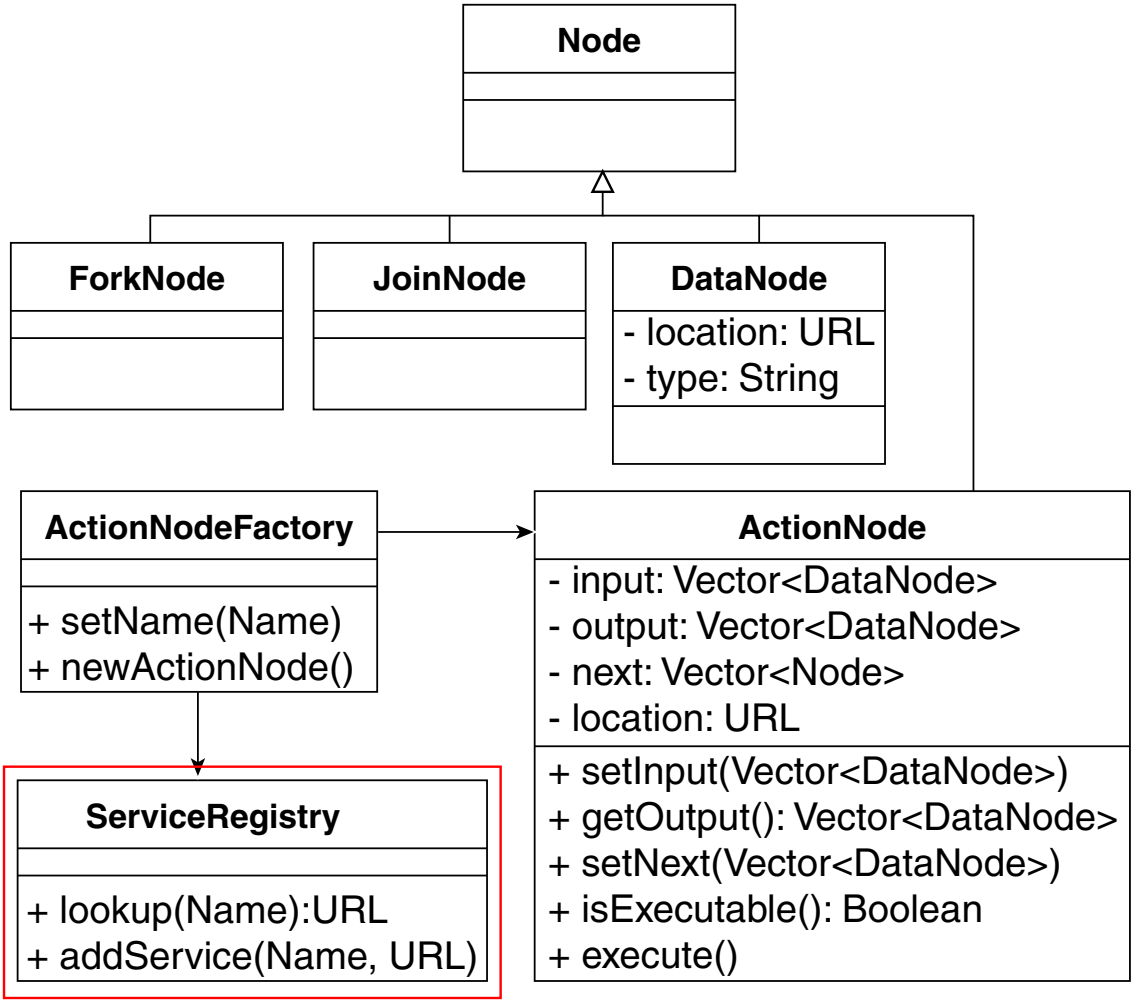
```
URL plain= http://.../data/513442.xml ;
URL stl= http://.../data/stopwords.xml ;

URL service= http://.../tira/stopwremover ;

WebService ws=new WebService();
ws.setParameter(plain, stopwords);
URL filteredText=ws.call(service);
```

Operationalizing IR Processes with TIRA

Example: Operationalizing action nodes.



Conclusion / Outlook

- TIRA lets a user specify and execute personal IR processes.
- TIRA is an *open* MDA-based architecture for personal IR. Open means that everybody can contribute own services.
- TIRA is flexible, modular, scalable.

- Development of PSM-transformations to other platforms: clusters, P2P (BSP), grids,...
- Research question: Estimation of IR module execution times, scheduling, binding.

Demo

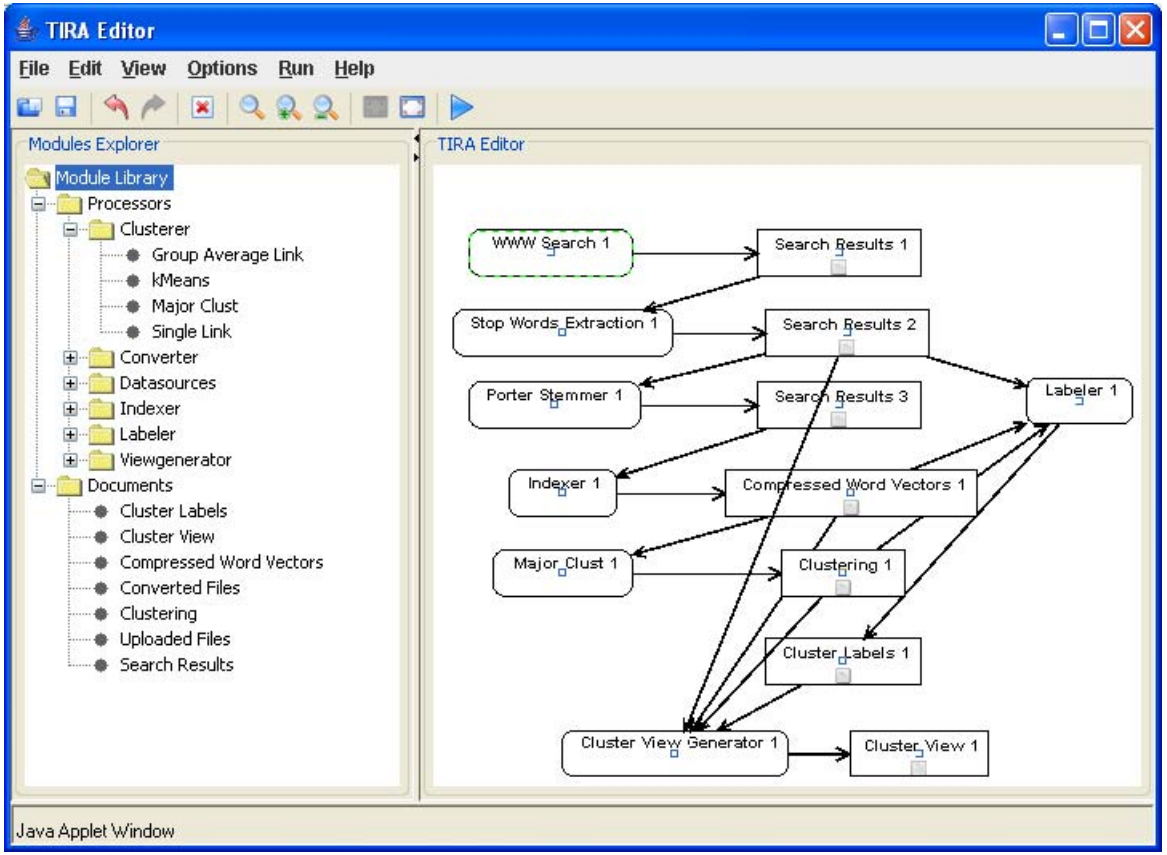
http://webis1.medien.uni-weimar.de/tira/

http://webis1.medien.uni-weimar.de/aisearch/aisearch-demo.html

Thank You!

Questions?

TIRA **Screenshot**



TIRA Architecture