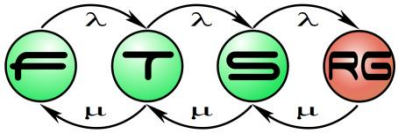
Developing and Visualizing Live Model Queries

Zoltán Ujhelyi, Tamás Szabó
István Ráth, **Dániel Varró**
Ábel Hegedüs (demonstrator)



Budapest University of Technology and Economics
Department of Measurement and Information Systems

MŰEGYETEM 1782

What is a Model Query?

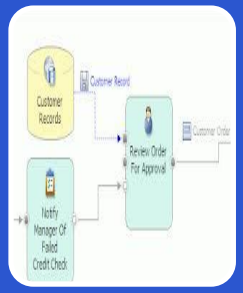
- ModelQuery(A,B)← Cond(A,B)
  - Retrieve tuples of model elements ***A,B***
  - Satisfying the query condition ***Cond***
  - Enumerate one / all instances
  - With ***A,B*** as input or output parameters

AUTOSAR

- *Select pairs of SystemSignal and its signal group which are not in the same IPDU*

IMA

- *Find instances of critical functions not running on different processors of different chassis*

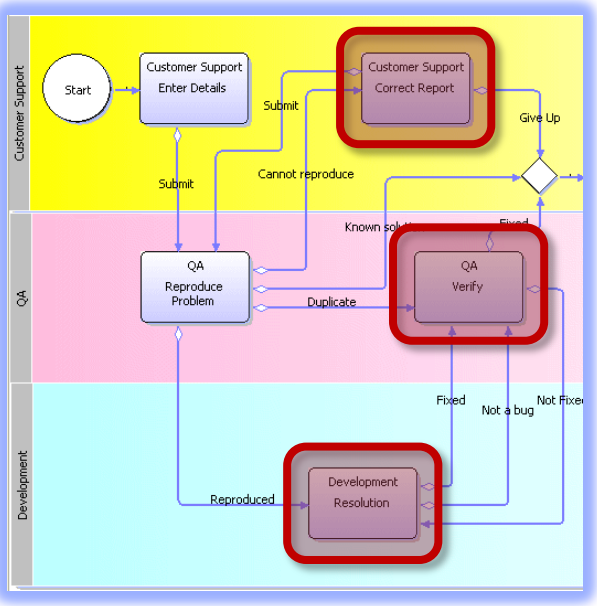


BPMN

- *Show sink activities with no outgoing flow*

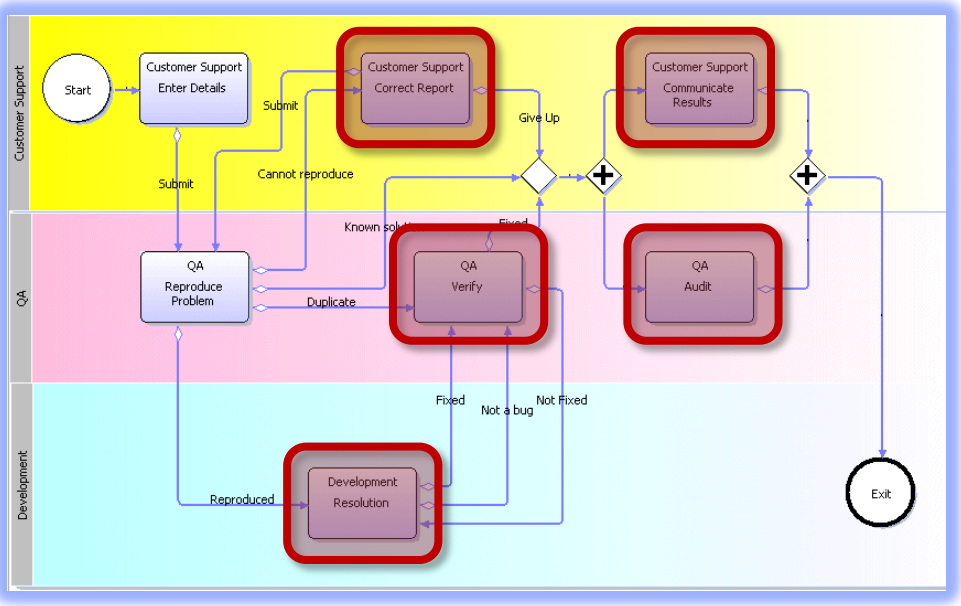
Live Model Queries

- Live Query Evaluation:
  Incremental cache of matches
  - Maintain a cheap cache
    (only memory overhead)
  - Notify about relevant changes
  - Enable reactions to complex
    structural events



Live Model Queries

- Live Query Evaluation: Incremental cache of matches
  - Maintain a cheap cache (only memory overhead)
  - Notify about relevant changes
  - Enable reactions to complex structural events



Live Model Queries for Model Transformations

- Find all contexts a MT rule is applicable for
- Live MT rules: triggered automatically upon change

Live Model Queries for Early Validation

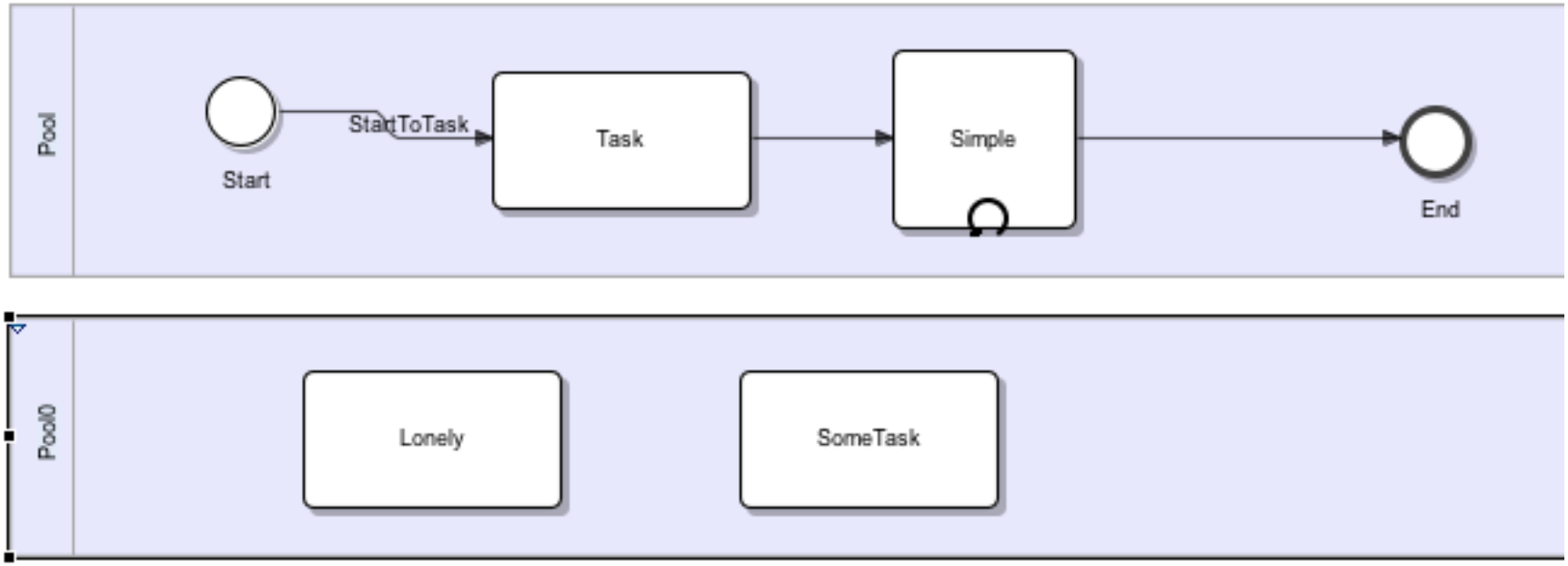
- Find all violations of a well-formedness constraint
- Immediate re-validation upon model change

Live Model Queries for Traceability Management

- See talk tomorrow at 11:30

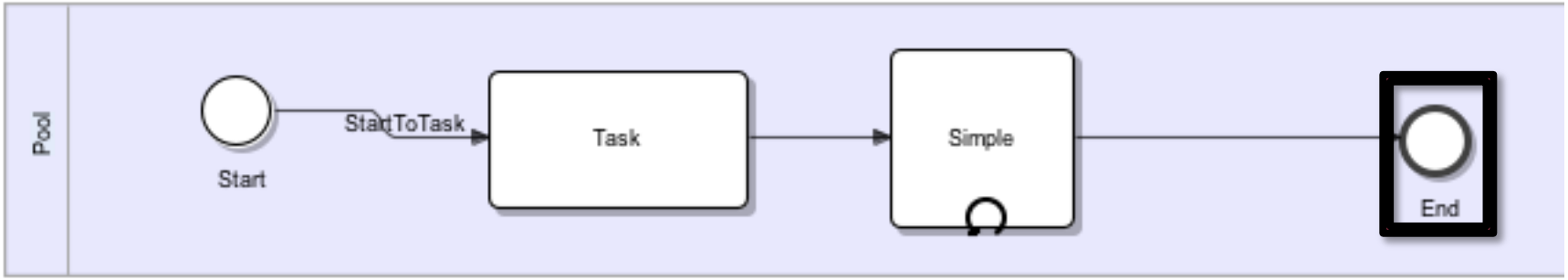
Example: Validation of BPMN Models

- BPMN models
  - Describing business processes
  - Models control and dataflow
    - Flowchart-like notation



Example: Validation of BPMN Models

- Well-formedness validation during editing
  - Sink Activity: activity without outgoing edge



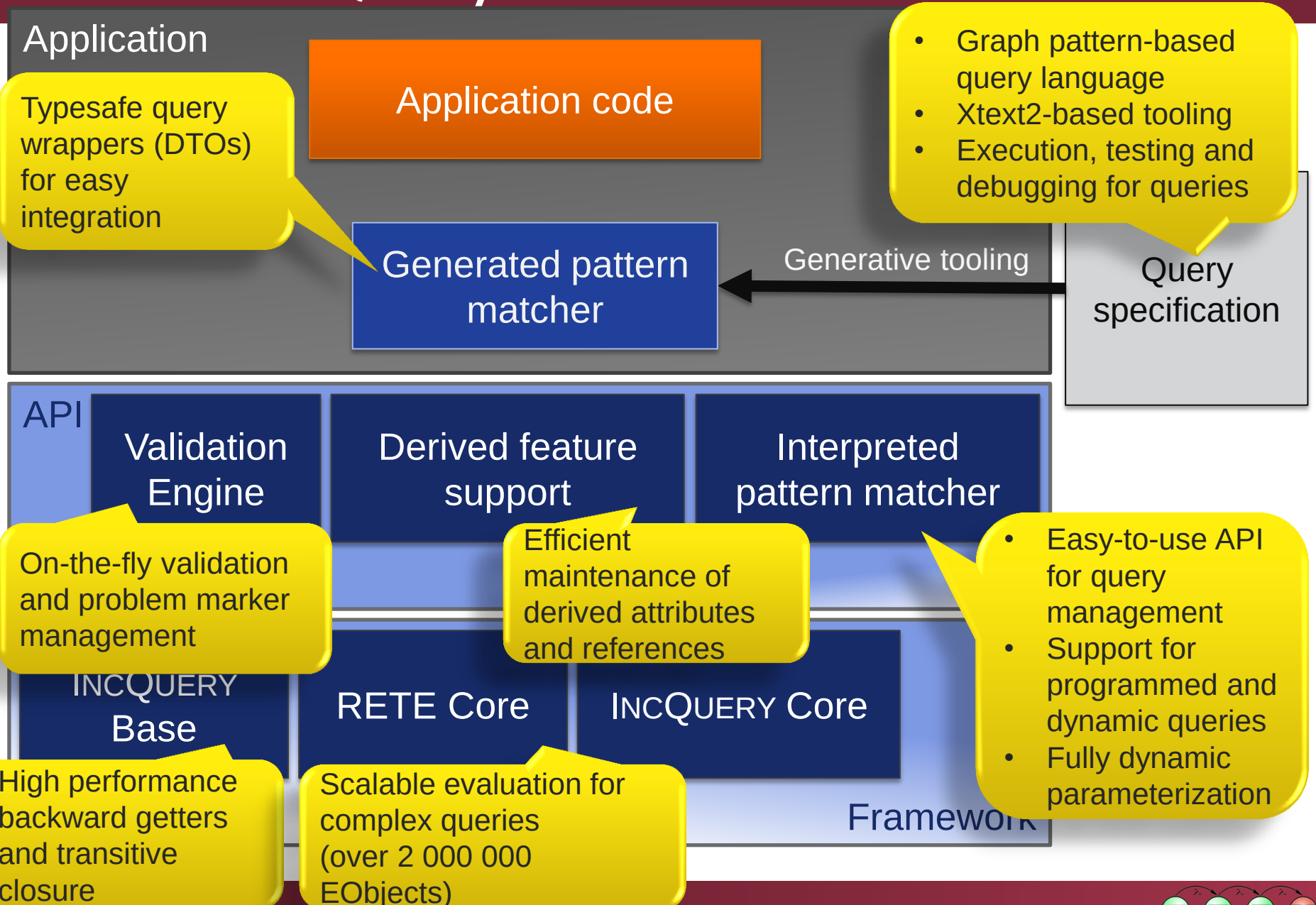
Properties | Problems | Tasks | Reviews

7 errors, 0 warnings, 0 infos

Description	Resource	Path	Location
Errors (7 items)			
Either a time date or a time cycle should	OMG E-Mail ...	eClarus BPMN Example	Unknown
Either a time date or a time cycle should	OMG E-Mail ...	eClarus BPMN Example	Unknown
Either a time date or a time cycle should	OMG E-Mail ...	eClarus BPMN Example	Unknown
Either a time date or a time cycle should	OMG E-Mail ...	eClarus BPMN Example	Unknown

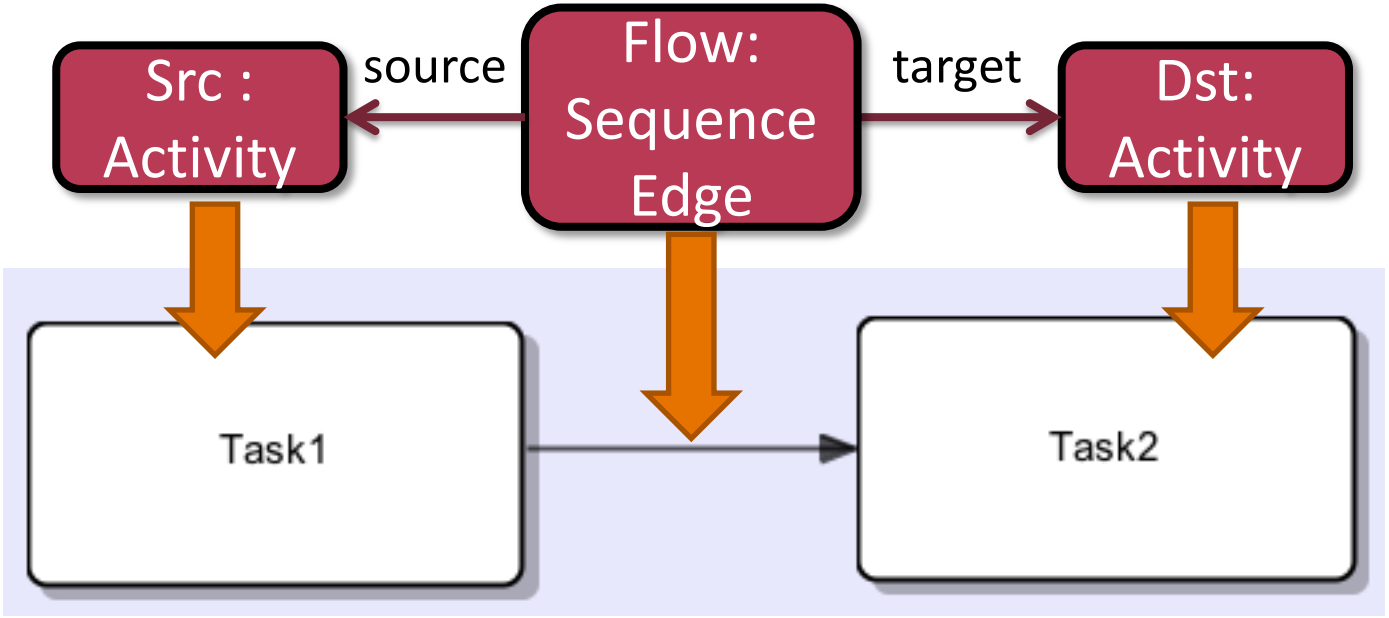
EMF-IncQuery Framework

EMF-IncQuery: Architecture Overview



Simple Graph Pattern in EMF-IncQuery

```
pattern sequenceFlowEdge
      (Flow:SequenceEdge,
       Src:Activity, Dst:Activity)= {
      SequenceEdge.source(Flow, Src);
      SequenceEdge.target(Flow, Dst);
}
```



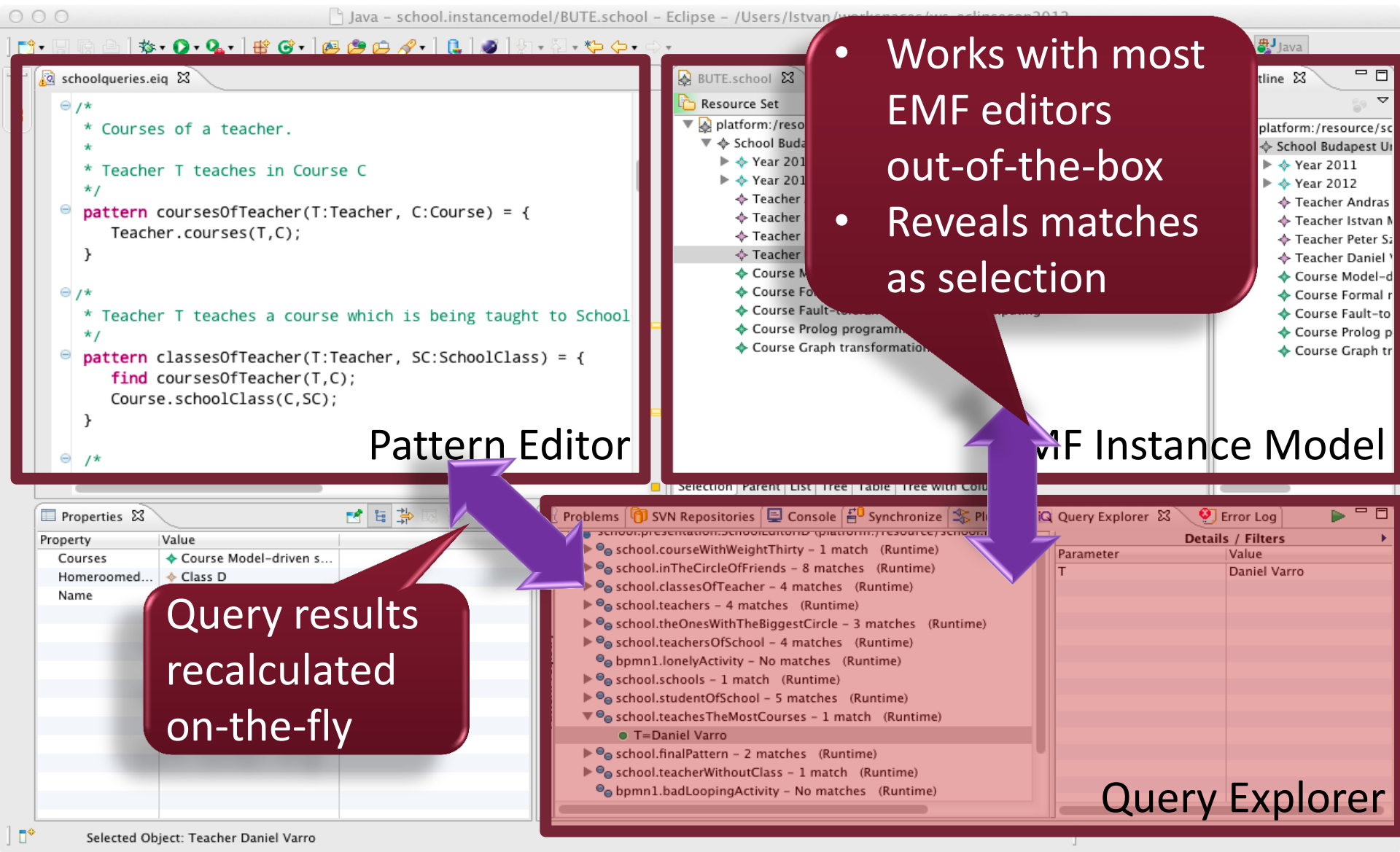
And Some More Complex Examples…

```
pattern hasOutEdge(A: Activity) {
   find sequenceFlowEdge(_Fr, A, _Other);
 } or {
   find messageFlowEdge(_Fr, A, _Other);
}
pattern sinkActivityNames(Name) {
      Activity(A);
      Activity.name(A, Name);
      neg find hasOutEdge(A);
}
```

Pattern composition (for reuse)

Negative composition (negation, quantification)

EMF-IncQuery Development Tools



Visualization of Live Model Queries in EMF-IncQuery

Requirements for Query Visualization

Genericity

- Multiple Model Sources
  - Model Editors
  - Current Selection

Incrementality

- Query Changes
- Model Changes

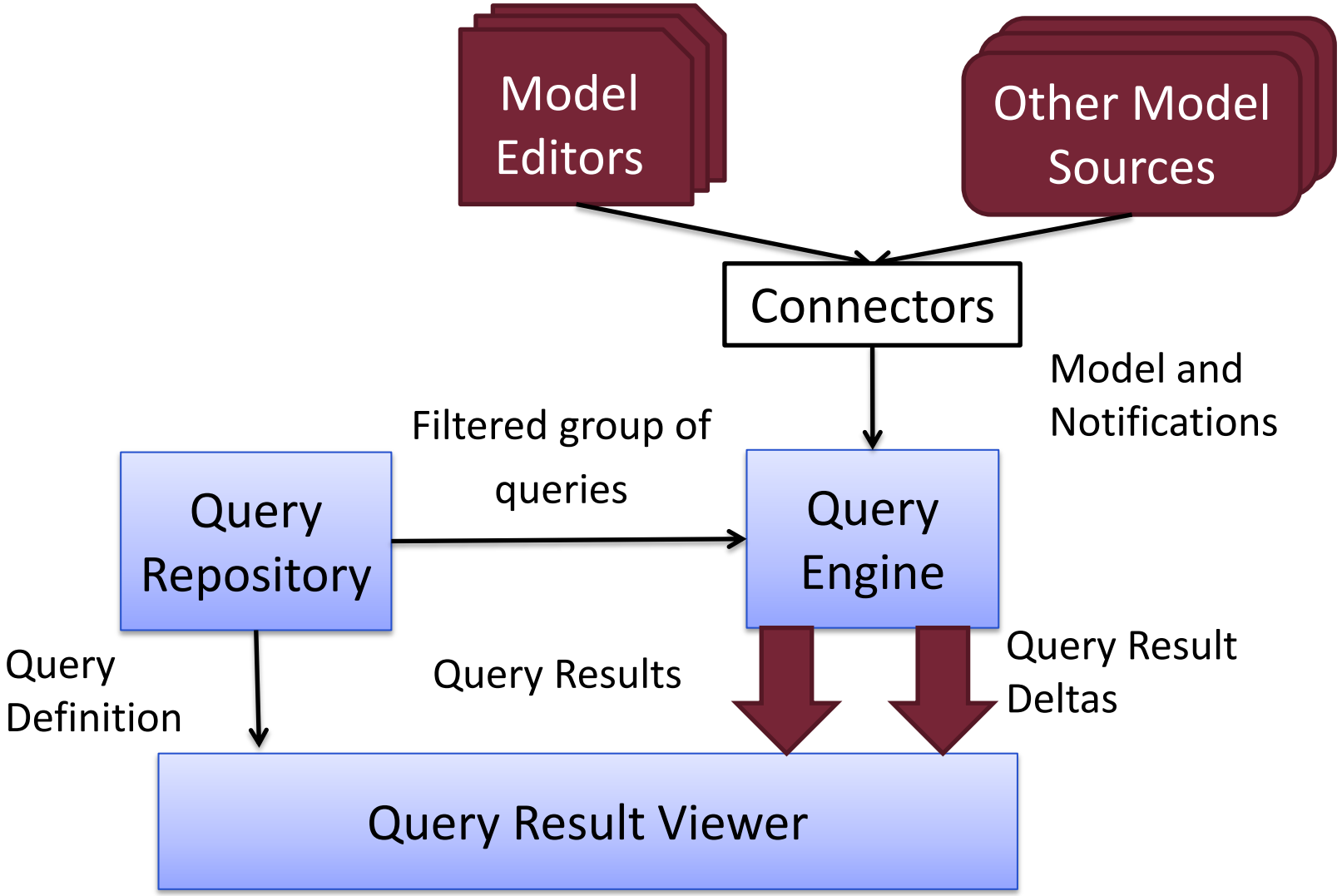
Traceability

- Query Definitions
- Input Model

Presentation

- Filtering
- Grouping

Proposed Architecture of Query Visualizer

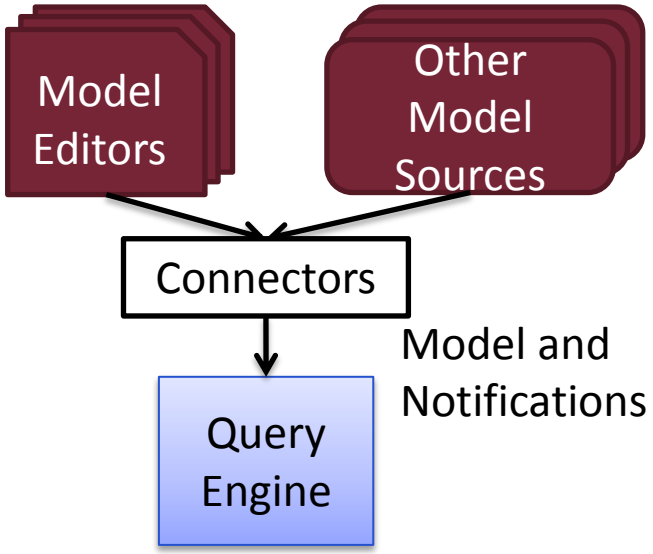


Model Source Connectors for Genericity

- Different model sources
  - Graphical (e.g BPMN, UML)
  - Textual (e.g. OCL)
  - Different implementation technologies

➔**Model source connectors**

- Operations
  - Open model
  - Send notification upon model change
  - Get current selection
  - Close editor



Genericity
- Multiple Model Sources
  - Model Editors
  - Current Selection

Incrementality
- Query Changes
- Model Changes

Traceability
- Query Definitions
- Input Model

Presentation
- Filtering
- Grouping

Query-based Indexing for Incrementality

- Update query results **incrementally** upon
  - Model updates
    - E.g. using the built-in editor
    - Engine: handles this case internally
  - Query updates
    - E.g. using the query editor
    - Engine: rebuilds internal model indexes

Genericity
- Multiple Model Sources
  - Model Editors
  - Current Selection

Incrementality
- Query Changes
- Model Changes

Traceability
- Query Definitions
- Input Model

Presentation
- Filtering
- Grouping

Query-based indexing

- Contains model elements accessed by the query
- Populated by an exhaustive model traversal (can be slow!)
- Goal: Avoid unnecessary re-traversal

Strategy 1: Generic model indexer

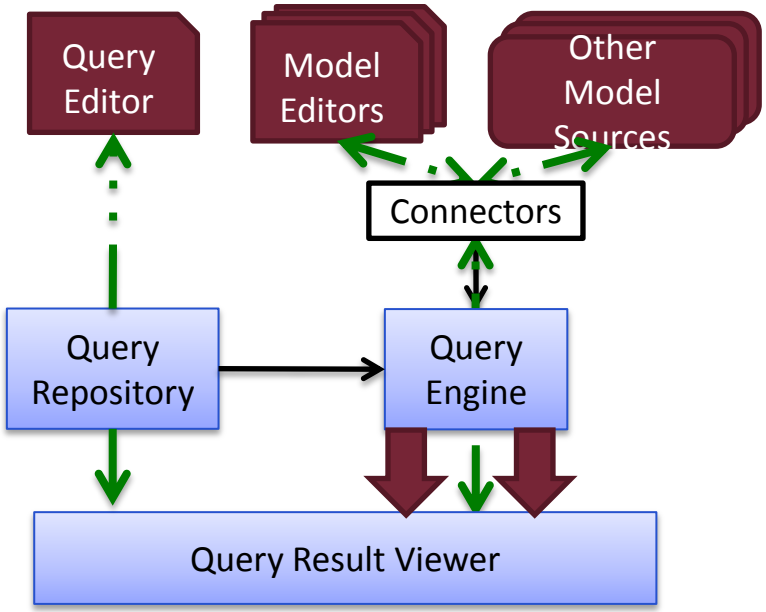
- "Wildcard mode": index every model element
- Higher memory consumption

Strategy 2: Group initialization

- Initialize several patterns together

Traceability and Navigation

- Maintaining source **models** and **queries**
- Support for navigating to/highlighting
  - Corresponding query definitions
    - Highlighting functionality in the query editor
  - Referenced model element(s)
    - Editor-dependent implementation
    - *Model source connector* handles functionality



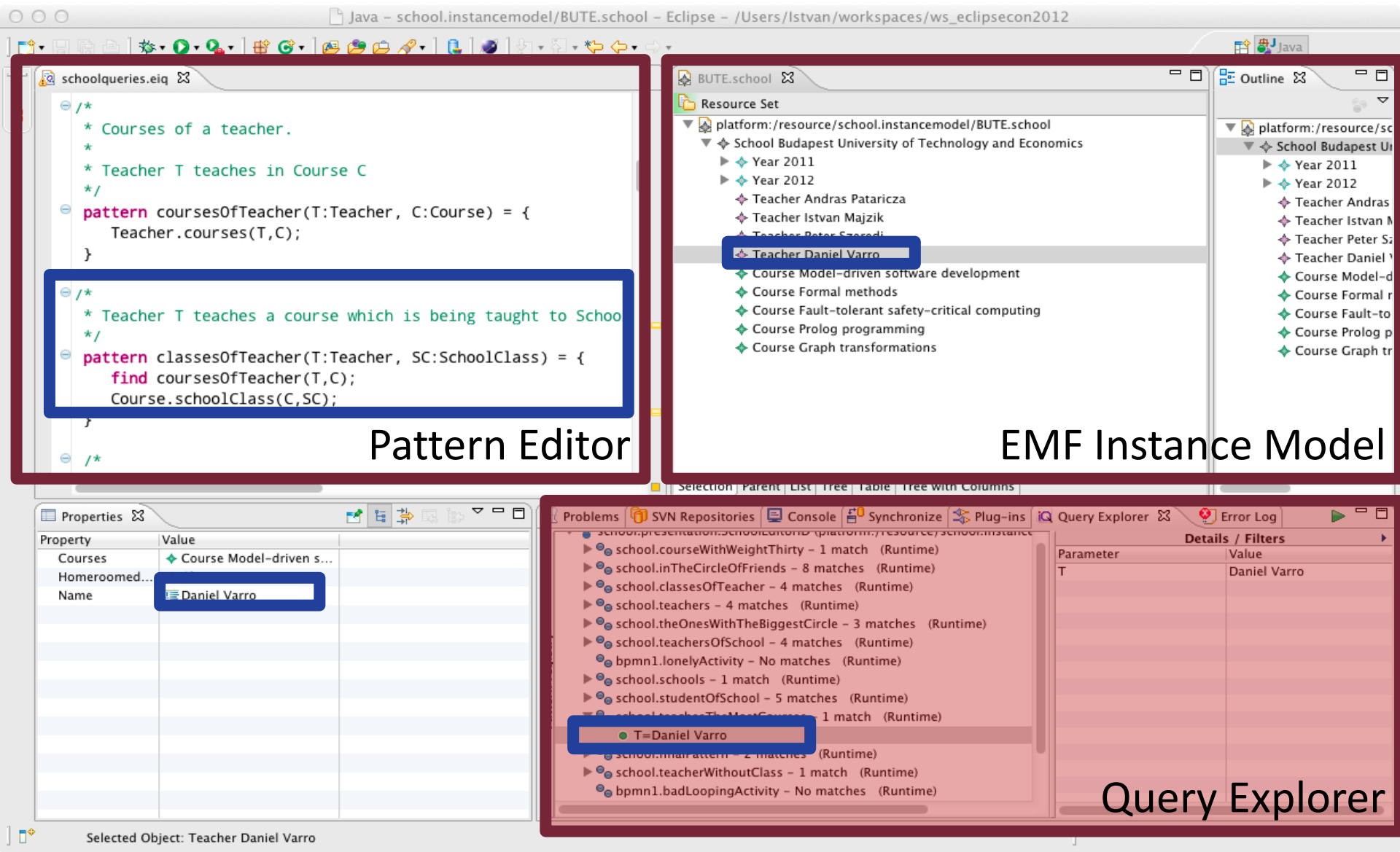
Genericity
- Multiple Model Sources
  - Model Editors
  - Current Selection

Incrementality
- Query Changes
- Model Changes

Traceability
- Query Definitions
- Input Model

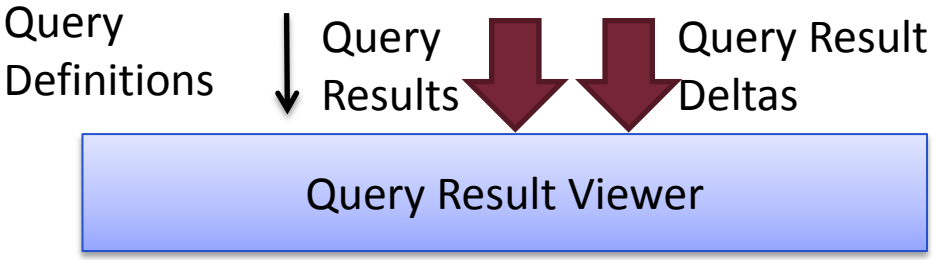
Presentation
- Filtering
- Grouping

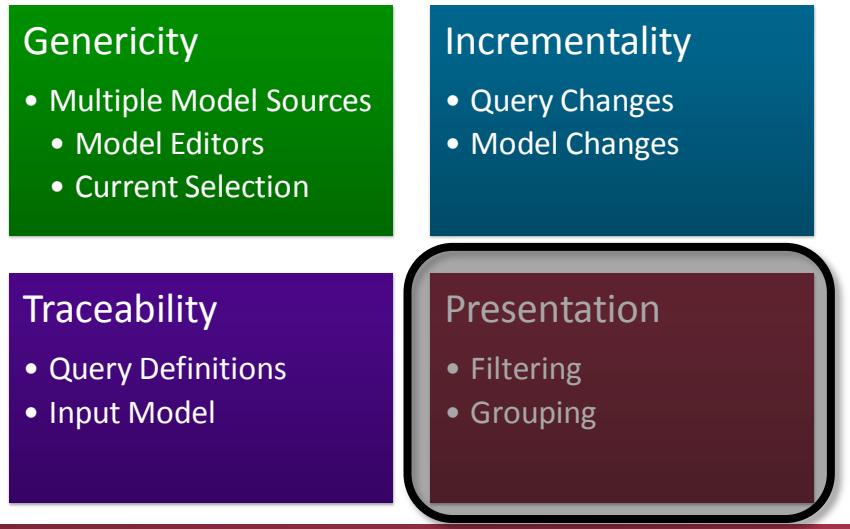
Traceability in Query Visualizer



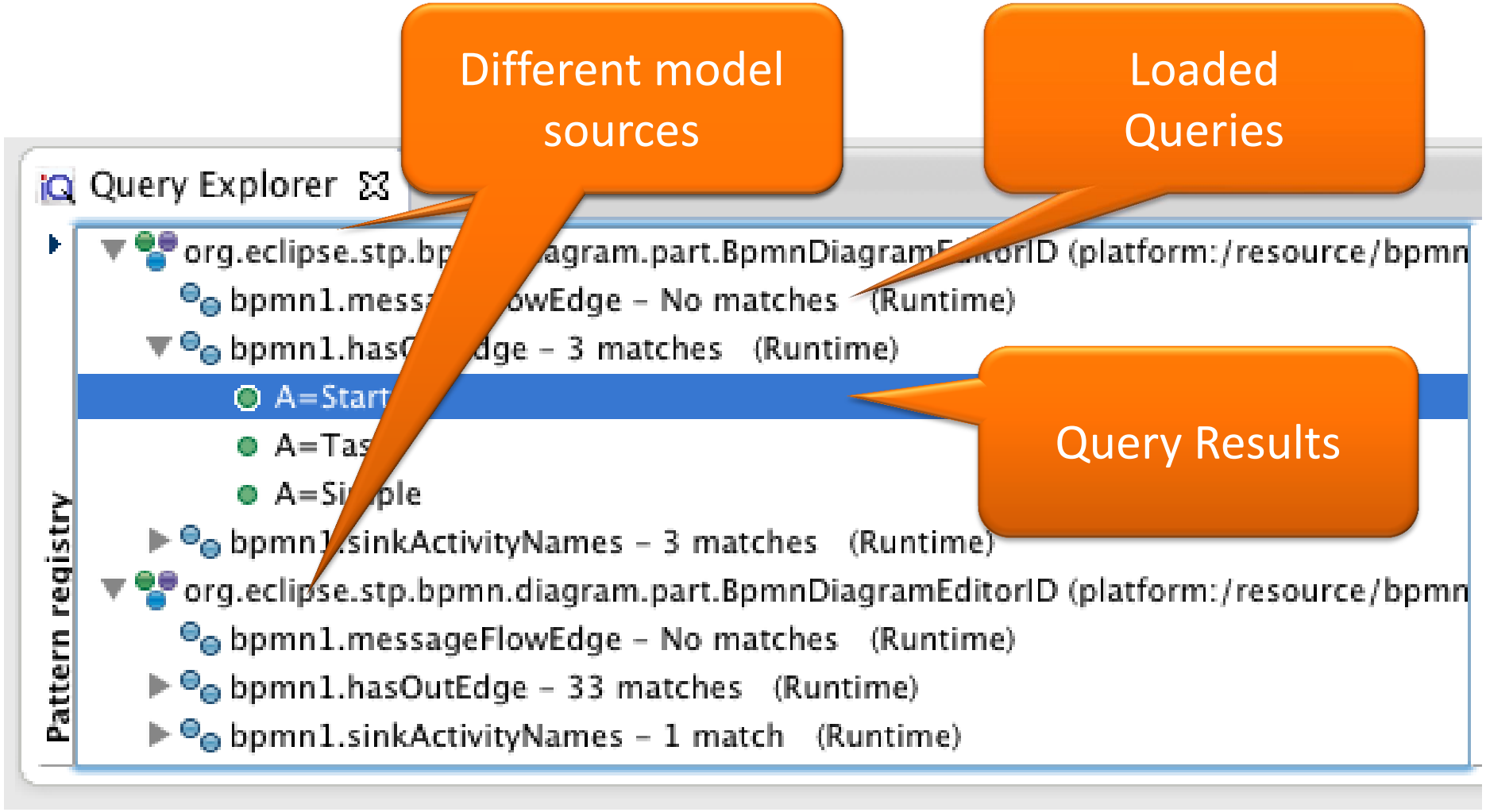
Grouping and Filtering for Presentation

- Define query groups („related” queries)
  - Static groups
    - E.g. Namespaces
  - Dynamic groups
    - E.g. Dependency based
- Filtering
  - Filter visible queries
    - Uses hints from developer
    - Manual overriding
  - Filter query results
    - By binding query parameters

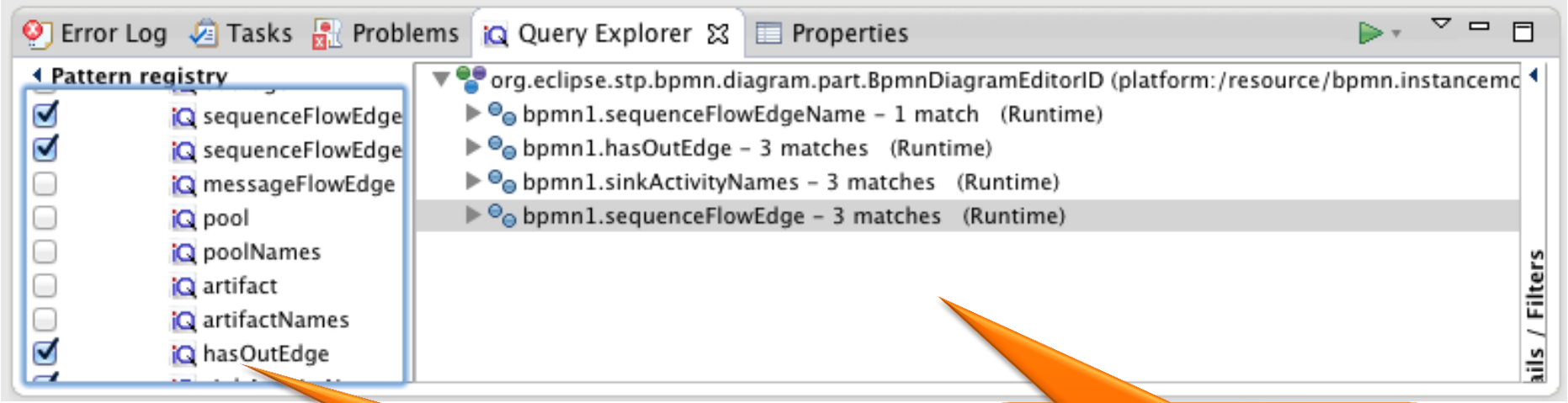




Result Viewer



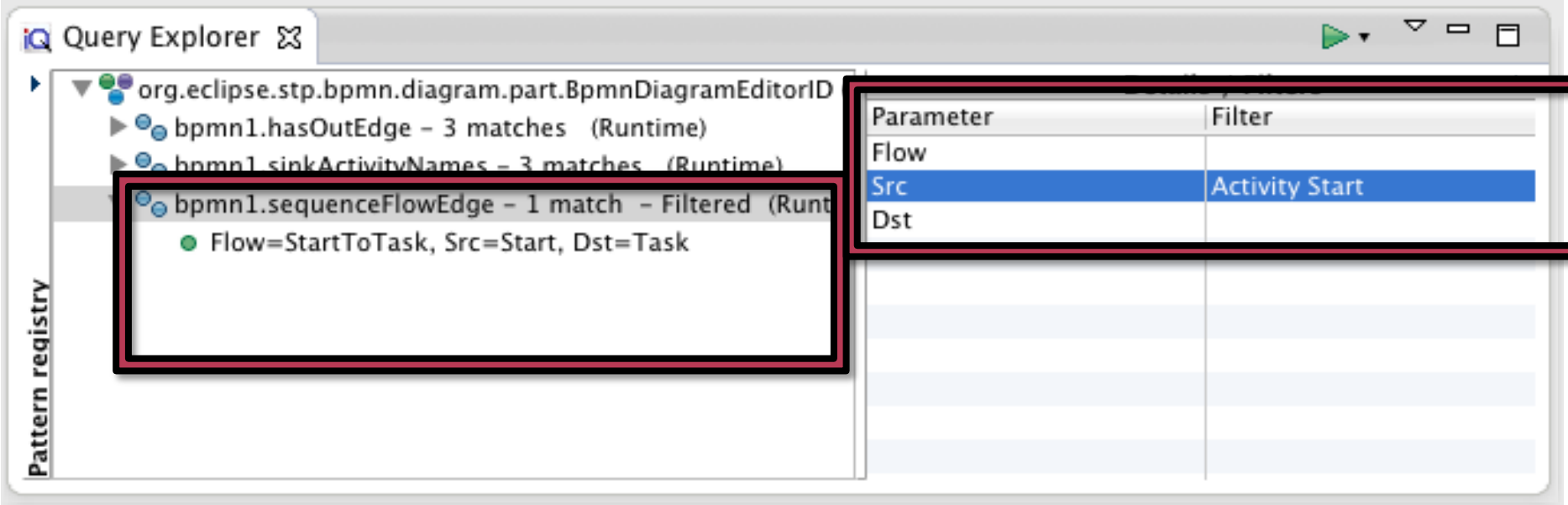
Filtering Visible Queries



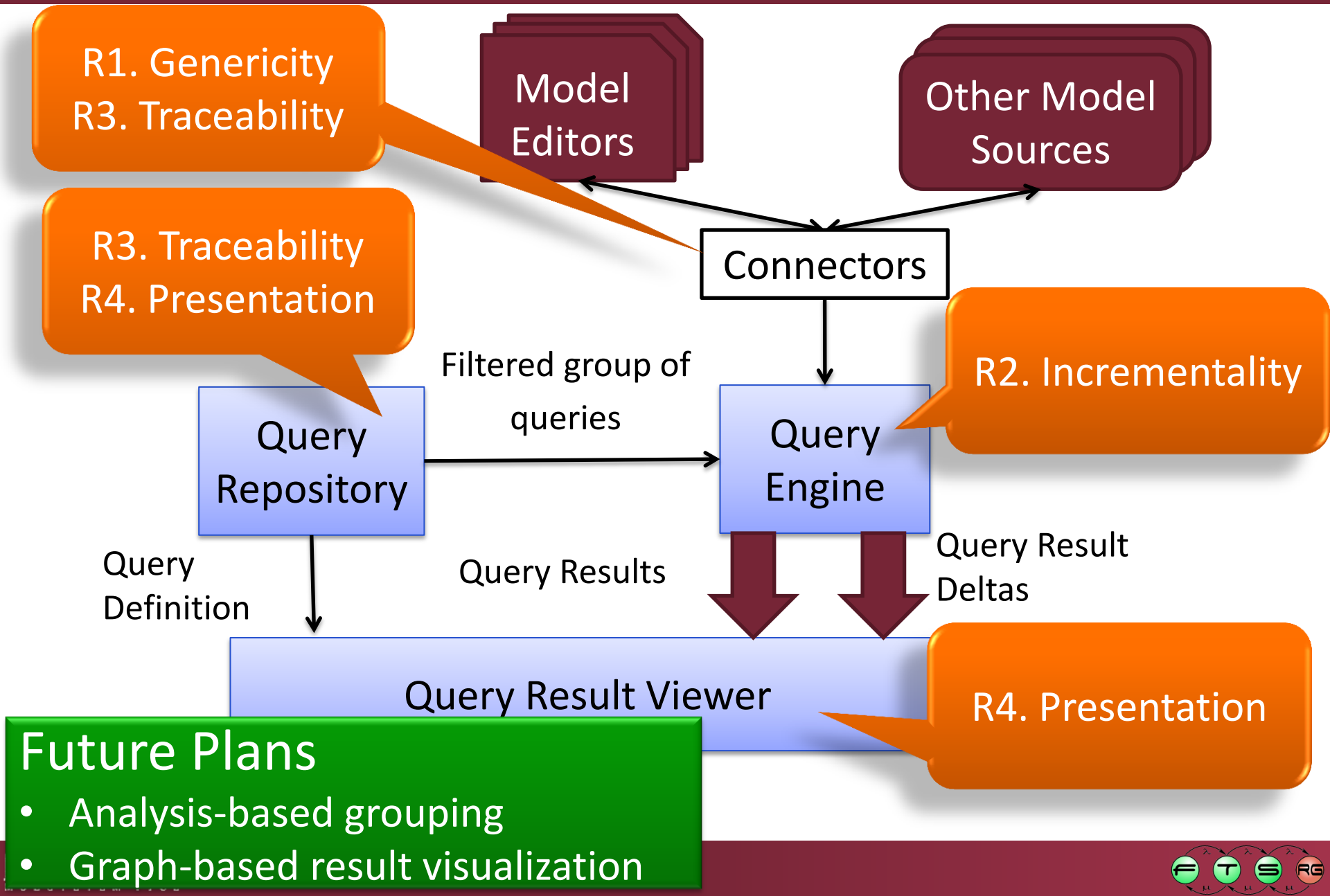
Elements in Query Registry

Queries Loaded

Query Result Filtering



Conclusions and Future Work



Future Plans

- Analysis-based grouping
- Graph-based result visualization