UNIVERSIDAD DE MURCIA

Collecting and processing 500K models

(and what to do with them)

Jesús Sánchez Cuadrado (jesusc@um.es)

22/06/2021 – Data4MDE

Context

Two years ago...

- Follow the hype: Apply ML to modelling

Context

Two years ago...

- Follow the hype: Apply ML to modelling

Our process:

Context

Two years ago...

- Follow the hype: Apply ML to modelling

Our process:

- Stage 1: We don't have enough models. We are going to fail.

Context

Two years ago...

- Follow the hype: Apply ML to modelling

Our process:

- Stage 1: We don't have enough models. We are going to fail.
- Stage 2: We have a bunch a models. Why don't we create a search engine?

Context

Two years ago...

- Follow the hype: Apply ML to modelling

Our process:

- Stage 1: We don't have enough models. We are going to fail.
- Stage 2: We have a bunch a models. Why don't we create a search engine?
- Stage 3: We have too many models. What do we do now?

This talk

Hypothesis

Anyone with interest on the application of data-driven techniques to modelling will have the need to collect and analyse models.

This talk

- A search engine for models: MAR
- A labelled dataset of models: MODELSET
- Applications

MAR: A search engine for models

Part I: Collecting and processing models

Motivation

- Models are the primary artifacts in MDE
- Model repositories make models available for reuse and learning
- In practice, models are not typically reused.
- Why? Maybe because it is not easy to find models
  - Limited or no search mechanisms
  - Many models are stored in source code repositories
  - Which are the relevant places to find models?

Motivation

Example. Searching for Ecore meta-models about state machines

- Models available in diverse repositories like GenMyModel, GitHub, AtlanMod Zoo, etc.
- What can we do to find interesting models?

Motivation

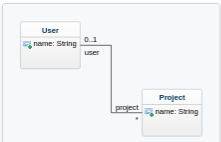
Explore

Explore public projects from the GenMyModel community

GenMyModel Public Repository

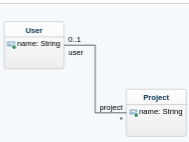
Search...

All types



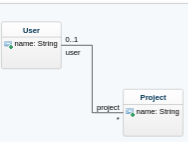
sprint1
UML a few seconds ago

Alaa%20Alajmy



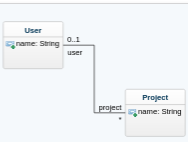
TehnoloskiFestival
UML 2 minutes ago

zanrajsek0



passport automation
UML an hour ago

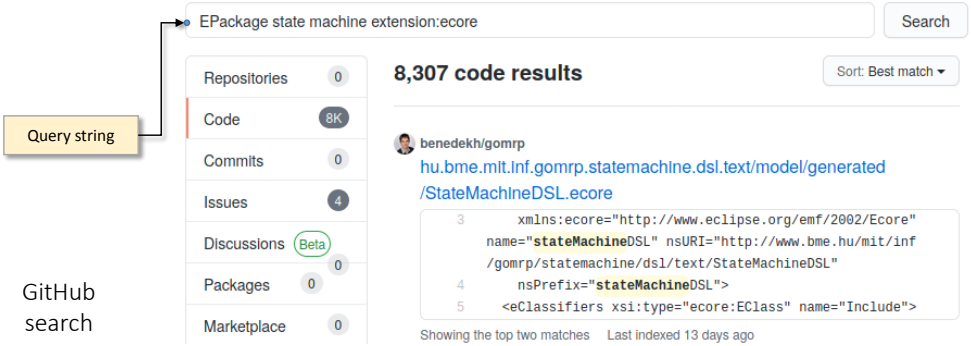
18101034



practica1
UML an hour ago

raulvallverdu

Motivation

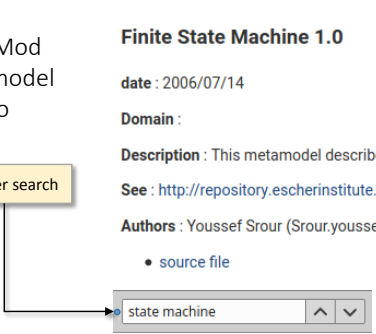


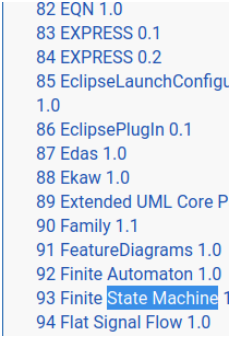
GitHub search

Motivation

AtlanMod Meta-model Zoo

Browser search





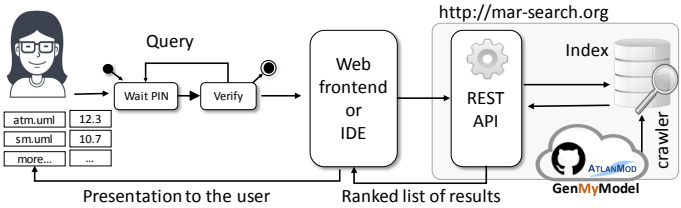
Motivation

Problem

Finding interesting models is a time consuming activity.

- Need to find out where are the models.
- Need to search in several places.
- Limited search facilities.
- Results are not ranked
- Inspecting results is complicated
- No guarantee that the obtained models are valid.

Solution



- Query by example and keyword-based queries
- Faceted search and filtering
- REST API + Web
- Inverted index
- Scoring algorithm
- Generic search
- Crawler for GitHub, GenMyModel and AtlanMod Zoo

Demo

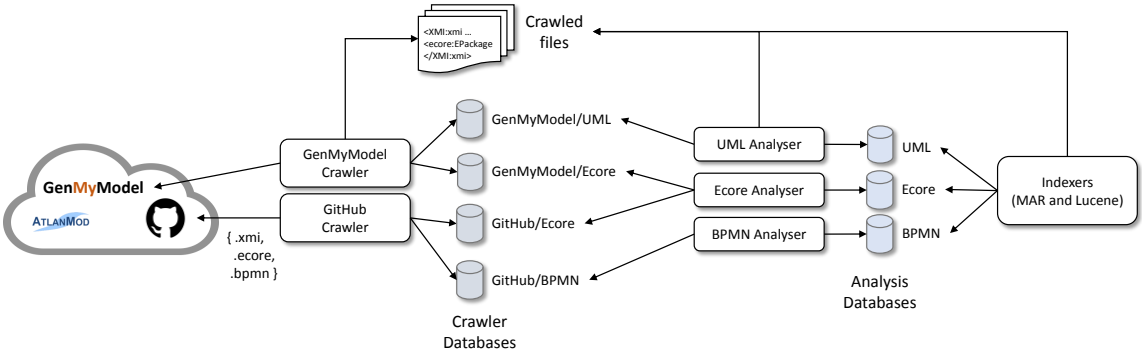
http://mar-search.org

Architecture

Main components

- Crawlers – Discover and collect models
- Analysers – Check validity and compute stats and quality metrics
- Model pre-processing pipeline
- Index
- Query processor
- Scoring algorithm

Crawling and analysis



Crawlers

- Which are sources of models?
- How to extract models from them?
  - GitHub - Rate limit issues
  - GenMyModel - Now public, before Selenium
  - AtlanMod - Webscrapping
- Which metadata is available?
  - Popularity (stars, forks)
  - Creation and update dates
  - Author
  - Topics

Analysers

Phases

1. Open the file (it may blow up the heap, crash, etc)
2. Validate (is it structurally correct?)
3. Analyse quality (e.g., detect smells)
4. Compute statistics (e.g., number of elements)

More difficult than it seems!

- Create an analysis server
- Launch it on demand and communicate via RPC
- If it crashes, the model is invalid

Available models

	Source	Crawled	Duplicates	Failed	Indexed	Observations
Ecore	GitHub	67,322	46,199	341	20,782	
	GenMyModel	3,987	3	27	3,957	
	AtlanMod	304	1	4	299	
UML	GitHub	53,082	7,282	1,699	44,101	Eclipse UML meta-model.
	GenMyModel	352,216	143	23,836	328,237	
BPMN	GenMyModel	21,285	0	200	21,085	EMF BPMN2 meta-model[1].
Archimate	GitHub	496	77	106	313	Archi meta-model[2].
PNML	GitHub	3,291	1,576	1,044	671	PNML framework[3].
Sculptor	GitHub	188	88	0	88	
RDS	GenMyModel	91,411	108	515	90,788	Entity/relationship diagrams.
Simulink	Dataset	200	0	0	200	Massif meta-model
Total	-	593,582	55,477	27,972	510,321	-

[1] https://www.omg.org/spec/BPMN/2.0/
[2] https://github.com/archi-contribs/eclipse-update-site
[3] https://pnml.lip6.fr/

How to query

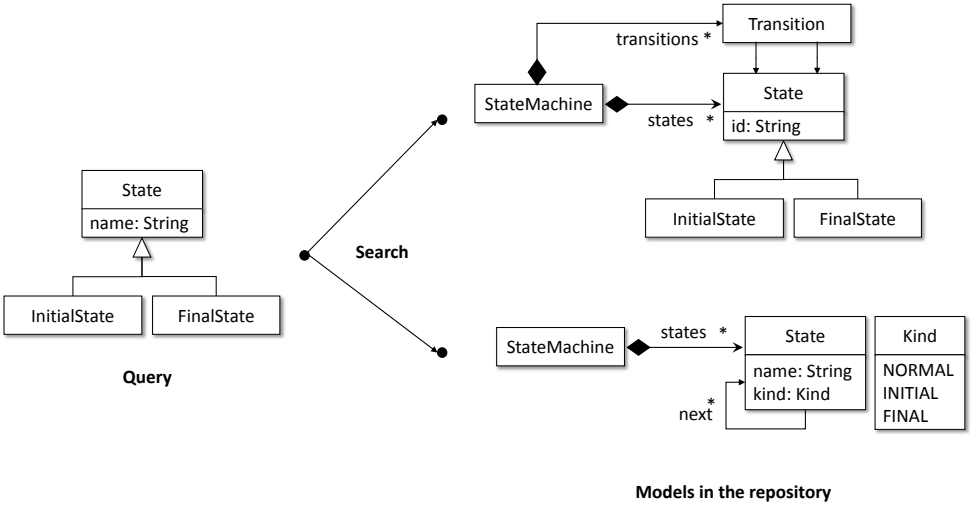
Keywords

1. Type a few keywords
2. e.g., state machine
3. Simple to implement, less precise
4. Simple to use

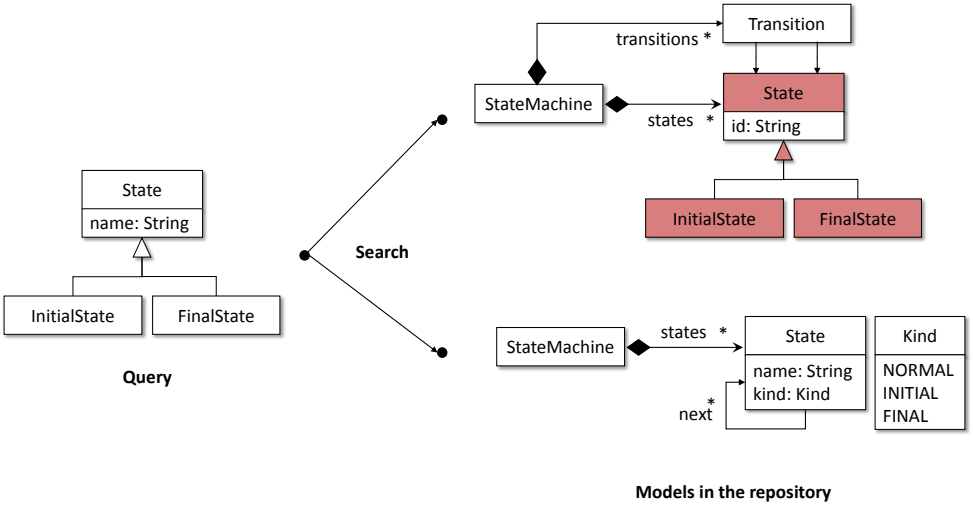
Query-by-example

- Provide an example or model fragment (or a complete model!)
- Find models which have partial matches
- More difficult to implement, more precise

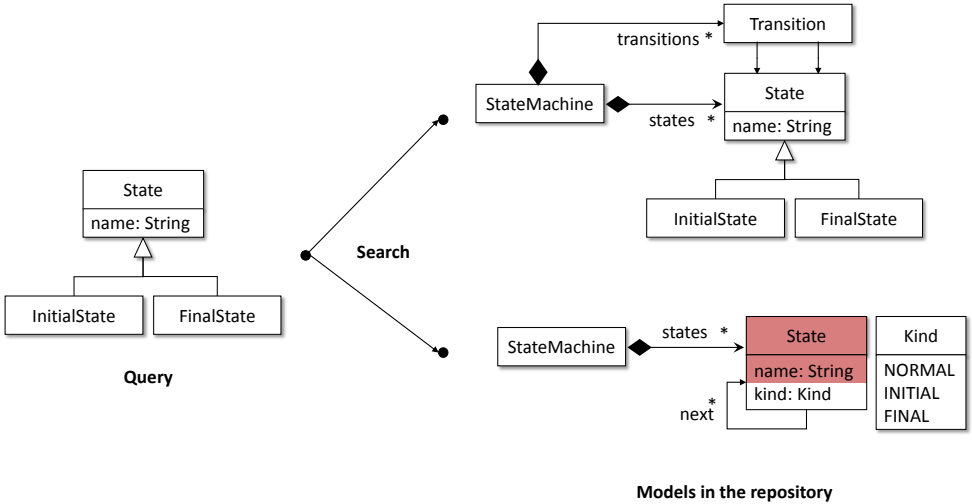
Query-by-example – An example



Query-by-example – An example



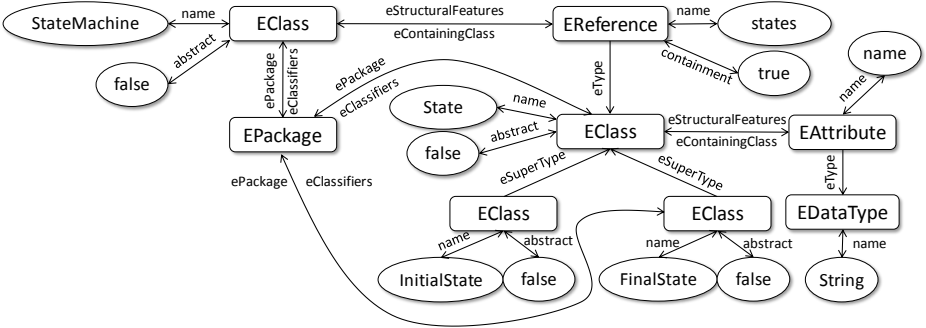
Query-by-example – An example

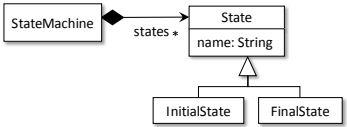


Query-by-example – Approach

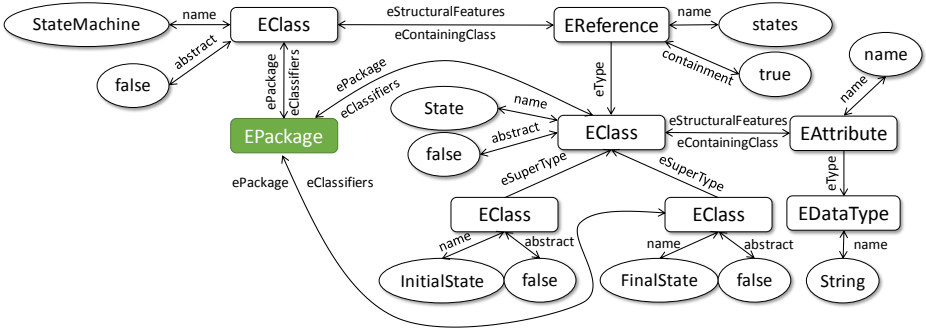
1. Encode the model
   - Structure of the model
   - Attributes represent the semantic domain
   - Notion of *Bag of paths*
   - Paths between attributes
2. Organize the paths into a index for fast retrieval
3. Apply a scoring function
   - Adapt BM25 for paths

Model encoding

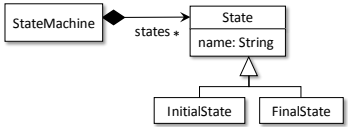




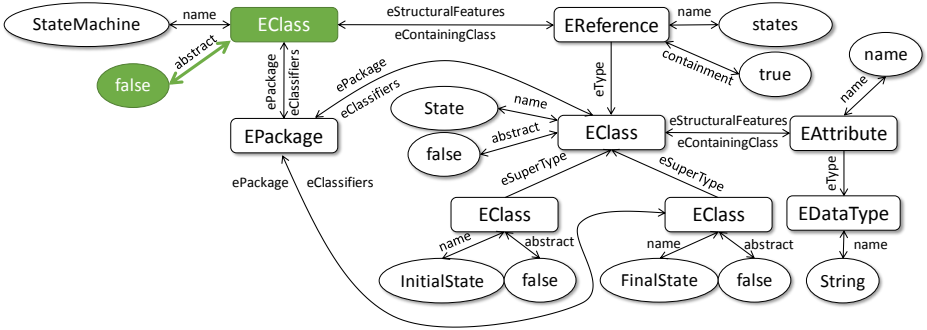
Model encoding



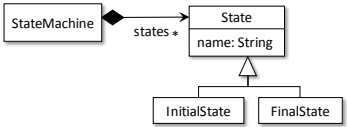
- Singleton paths



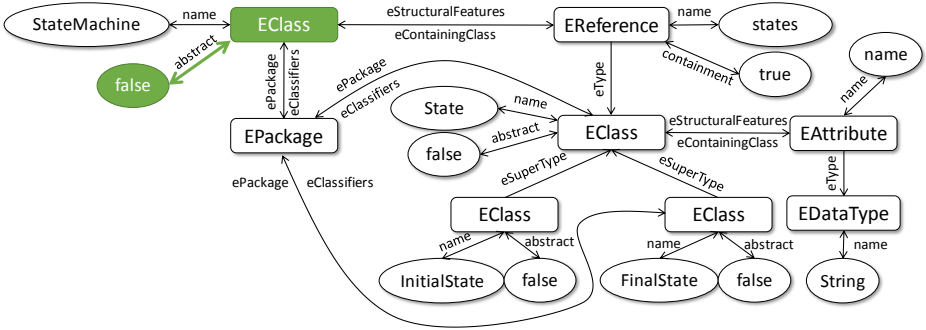
Model encoding



- Singleton paths
- Unit length paths

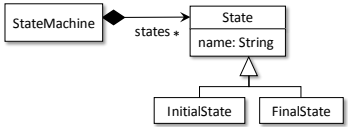


Model encoding

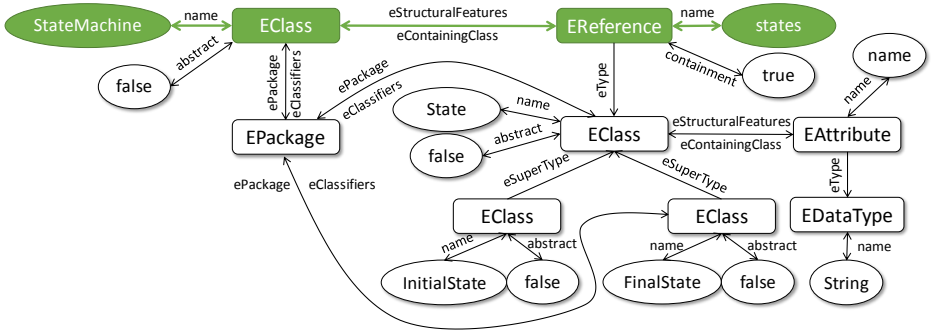


- Singleton paths
- Unit length paths

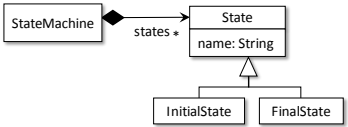
Encode the object data



Model encoding



- Singleton paths
- Unit length paths
- Paths between atributes of lengh < k } Encode the model structure



Model indexing

- Fill an *inverted index*
- Each path points to the list of models that contains them
- Scoring is done by comparing paths on the query with paths in the index

Path	Models
⟨StateMachine, $\overrightarrow{\text{name}}$, EClass, $\overrightarrow{\text{eStructuralFeatures}}$, EReference, $\overrightarrow{\text{name}}$, states⟩	sm-sub.ecore, sm-enum.ecore
⟨State, $\overrightarrow{\text{name}}$, EClass, $\overrightarrow{\text{abstract}}$, false⟩	sm-enum.ecore
⟨State, $\overrightarrow{\text{name}}$, EClass, $\overrightarrow{\text{abstract}}$, true⟩	sm-sub.ecore
⟨InitialState, $\overrightarrow{\text{name}}$, EClass, $\overrightarrow{\text{eSupertypes}}$, EClass, $\overrightarrow{\text{name}}$, State⟩	sm-sub.ecore

Model indexing

- Fill an *inverted index*
- Each path points to the list of models that contains them
- Scoring is done by comparing paths on the query with paths in the index
- In practice, this is not scalable
  - A repository of 10,000 models has millions of different paths
- We optimise this basic schema by identifying common sub-paths

Implementation

- Crawling: Python (PyGitHub) and Ruby
- Analysis: Java, EMF, Xtext, PNMLFramework
- Backend: Java, Apache Spark
- Inverted index: HBase
- Web interface: Svelte
- Source code: http://github.com/mar-platform/mar

Self-assesment

- A search engine for models
- Indexed about 500,000 models
- Web-based user interface

Self-assesment

- A search engine for models
- Indexed about 500,000 models
- Web-based user interface

Q: But... Is this useful?

Self-assesment

- A search engine for models
- Indexed about 500,000 models
- Web-based user interface

Q: But... Is this useful?
A: No, I don't think so. The current user interface is probably not good enough.

Self-assesment

How to make MAR useful

- Make raw models available
- Build datasets
- Expose search and other internal tools as services
- Identify potential applications

REST APIs

REST API

- /search/
  - By keywords
  - By example
- /metadata/
  - Metadata for a given stored model
- /analysis/
  - Smells
  - Metrics
- /ml/
  - Classification

REST APIs – Search with keywords

Example

```
curl -X POST -d "petrinet␣place␣color" http://155.54.205.39/v1/search/keyword?max=2
```

REST APIs – Search by example

Example

```
curl -X POST -d "@tournament.ecore" http://155.54.205.39/v1/search/example?type=ecore&
    max=100

[
  {
   "id":"github:ecore:/data/Gullskatten/sirius-soccer/no.ntnu.soccer.model/model/soccer.
       ecore"
   "name":"soccer.ecore",
   "modelType":"ecore",
   "url":"https://raw.githubusercontent.com/Gullskatten/sirius-soccer/00
       f8e390fa72a1a85e4d7dd5846852ac41c1c158/no.ntnu.soccer.model/model/soccer.ecore",
   "score":211.54585423692313,
   "metadata":{"smells":{"OverLoadedClassSmell":1},
     "topics":["sirius","␣intellij","␣kaggle","␣soccer"],
     "numElements":115,
     "explicitName":null,"description":null,"category":null},
  }
...
]
```

REST APIs – Smells

- Ecore smells
- http://mar-search.org/v1/analysis/smells

Example

```
$ curl -X GET -d "@relational.ecore" http://155.54.205.39/v1/analysis/smells

{
  "IrrelevantClassSmell" : ["//NamedElement"]
}
```

ModelSet

Motivation

Apply Machine Learning to Modelling

- We need datasets.
- For some types of problems, datasets need to be labelled.

- In practice: few datasets
- Labelled datasets: small (e.g., 555 models[4])
- Non-labelled datasets: can be large, but not curated (e.g., Lindholmen[5])

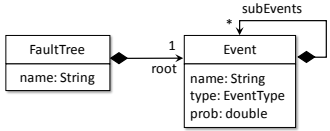
[4] https://zenodo.org/record/2585456#.YM5ziSbtb0o
[5] http://models-db.com/oss/

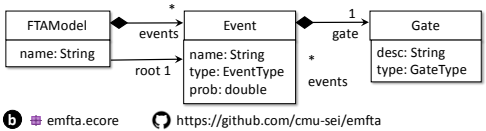
Challenge

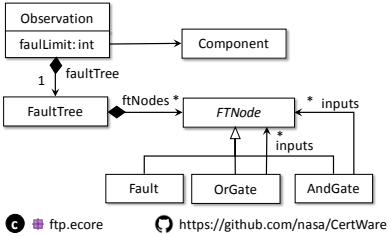
- Inspecting and labelling models is hard.
- Requires modelling expertise.
- We need to annotate these models, one by one.
- Which labels to use? Category? Tags?

Challenge – Example

- What is FTA? (model b)
- Perhaps you will find out better if you see FaultTree
- But what is a Fault Tree?

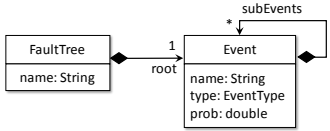




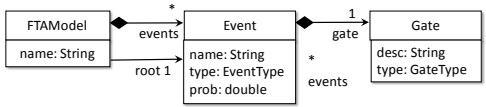


Challenge – Example

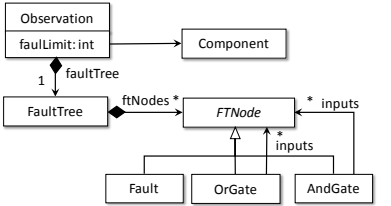
- What is FTA? (model b)
- Perhaps you will find out better if you see FaultTree
- But what is a Fault Tree?
- We need context (similar meta-models, GitHub links).
- We want to copy-paste once we understand.



a FaultTree.ecore https://github.com/osate/osate2



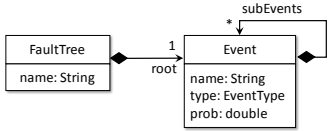
b emfta.ecore https://github.com/cmu-sei/emfta



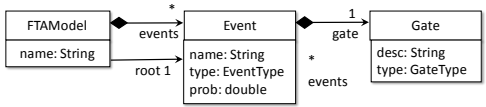
c ftp.ecore https://github.com/nasa/CertWare

Challenge – Example

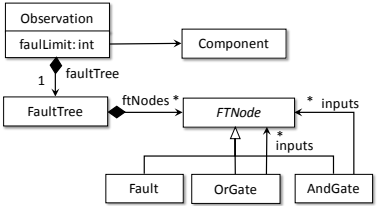
- What is FTA? (model b)
- Perhaps you will find out better if you see FaultTree
- But what is a Fault Tree?
- We need context (similar meta-models, GitHub links).
- We want to copy-paste once we understand.
- category: fault-tree
- tags: safety, hazard



a FaultTree.ecore https://github.com/osate/osate2



b emfta.ecore https://github.com/cmu-sei/emfta

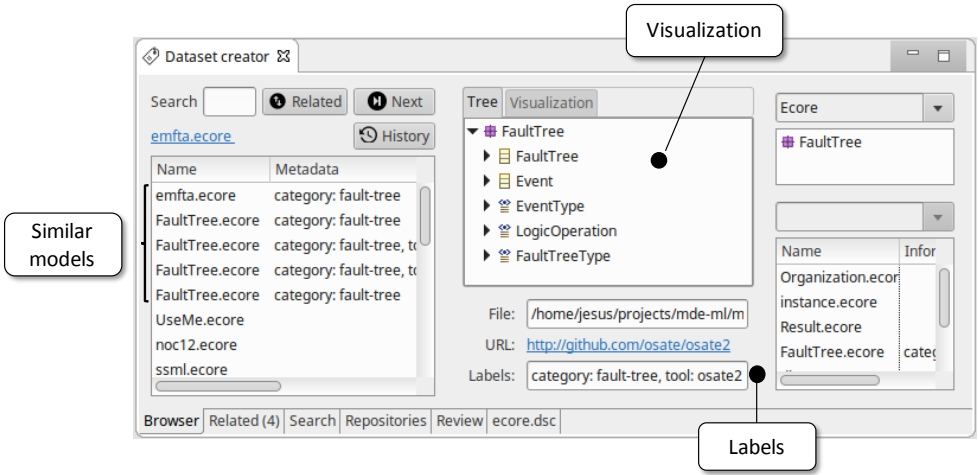


c ftp.ecore https://github.com/nasa/CertWare

Labelling method

- Interactive labelling algorithm
- Dynamic clustering (kind of interactive DB-SCAN)
- Steps:
  1. Pick an unlabelled model *m*
  2. Use MAR to search for similar models
  3. Inspect and label these models together
     - In the background, search models similar to the ones just labelled
  4. Keep labelling the same “streak” of models or go to step 1

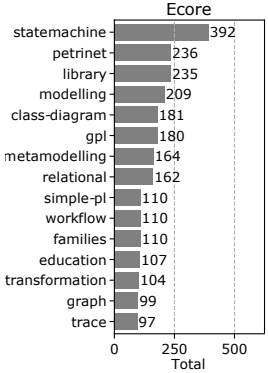
Dataset creator

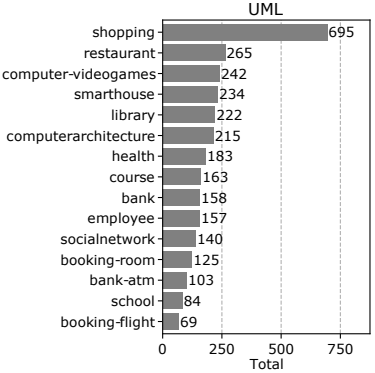


ModelSet

- 5,466 Ecore models – from GitHub
- 5,120 UML models – from GenMyModel
- See http://modelset.github.io.

- 28,719 labels
- Category, tags, purpose, notation, tool





Applications

Part II: What to do with the models

Applications

Available resources

- Raw models (about 500,000)
- Labelled models (about 10,000)
- Services

Now what?

Applications

- Using services
  - Enhancing modelling tools
  - Avoid re-inventing the wheel
- Using the dataset
  - Category, tags inference (classification)
  - Detecting dummy models (classification)
  - Build embeddings
  - Stratified k-fold
- Using the raw models
  - Recommendation
  - Model analytics

Example – Enhancing modelling tools

Scenario: Reuse

A developer is creating a DSL in Xtext. It would be desirable not to start from scratch. How one could find similar DSLs?

Example – Enhancing modelling tools

Scenario: Reuse

A developer is creating a DSL in Xtext. It would be desirable not to start from scratch. How one could find similar DSLs?

- See the abstract syntax of the DSL as its interface
- Index Xtext grammars using its abstract syntax
- Search by example

Example – Enhancing modelling tools

- Easily integrated in an Eclipse plug-in

```
Resource r = /* get an EMF resource somehow */
ByteArrayOutputStream bos = new ByteArrayOutputStream();
r.save(bos, null);

HttpResponse<JsonNode> jsonResponse = Unirest.post(getURL("v1/search/example?type=" +
    searchType + "&max="+max))
  .multiPartContent()
  .accept("application/json")
  .field("uploaded_file", bos.toString().getBytes(), "model.ecore")
  .asJson();

// [{name: 'relational.ecore', url: 'http://github...', score: 1523.3}, ...]
```

Example – Experiments

Scenario

A researcher is investigating about automatic fixing of meta-models.

- Everything typically starts from scratch
- Manually implement a catalogue of smells
- Need models for doing experiments

Example – Experiments

Scenario

A researcher is investigating about automatic fixing of meta-models.

- Everything typically starts from scratch
- Manually implement a catalogue of smells
- Need models for doing experiments

Resources

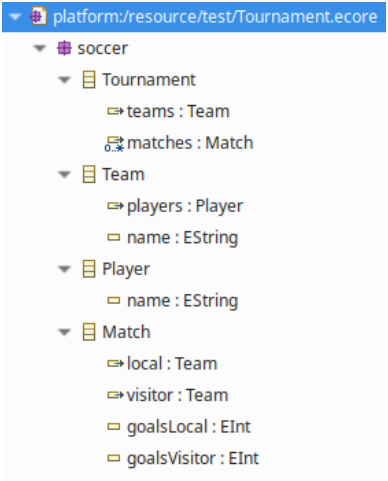
- Use smells API
- Use the models from ModelSet or MAR

Example – Categories and tags

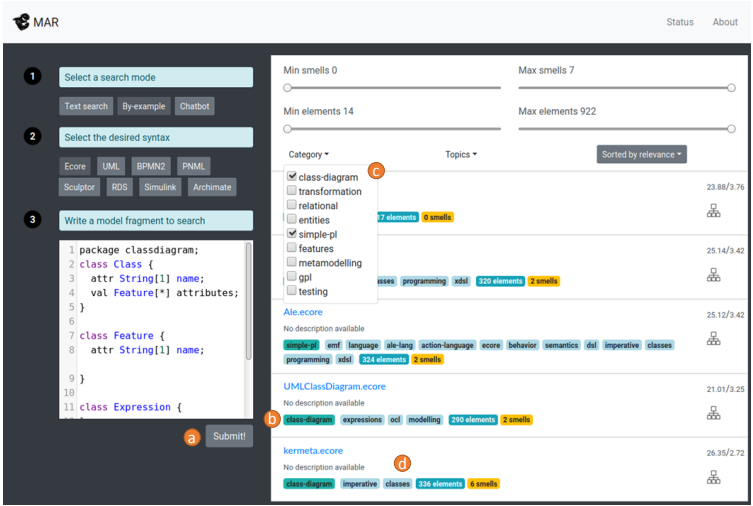
Example

```
$ curl -X POST -d "@tournament.ecore" \
    http://155.54.205.39/v1/ml/classify?type=ecore

{
  "category":"tournament",
  "tags":["domainmodel"]
}
```



Example – Categories and tags



Example – Categories

- Train models to infer the category of an unseen model
- Use standard Python libraries
- Make use of modelset.py

Example – Classifier for categories

```
import modelset as ms
import pandas as pd

dataset = ms.load('..', modeltype = 'ecore')
df = dataset.to_normalized_df(min_ocurrences_per_category = 7, languages = ['english'])
```

	category	tags	language	id
2661	visualization	graph	english	repo-ecore-all/data/MDEGroup/QMM/validation-su...
1029	statemachine	behaviour	english	repo-ecore-all/data/silverspy/DSL_TP/fr.ut2j.m...
331	library	domainmodel	english	repo-ecore-all/data/prayasb/org.eclipse.emf.te...
3666	behaviourmodelling	statemachine|activities|behaviour	english	repo-ecore-all/data/tue-mdse/ocl-dataset/datas...
4415	simple-pl	imperative|expressions|programming	english	repo-ecore-all/data/Alexandra93/DT/dt.workflow...
324	library	domainmodel	english	repo-ecore-all/data/eclipse/emf/tests/org.ecli...
156	petrinet	behaviour	english	repo-ecore-all/data/tue-mdse/ocl-dataset/datas...
4319	tournament	domainmodel	english	repo-ecore-all/data/dlitvinov/FastEMFStore.oth...
1979	modelling	biology	english	repo-ecore-all/data/rodriguez-facundo/model/ge...
570	families	university|domainmodel	english	repo-ecore-all/data/MDEGroup/QMM/validation-su...

Example – Classifier for categories

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score

all_id = list(df['id'])
all_labels = list(df['category'])

list_train, list_test, y_train, y_test = train_test_split(all_id, all_labels,
    stratify= all_labels, test_size=0.3, random_state=42)

train_corpus = [dataset.as_txt(id_) for id_ in list_train]
test_corpus = [dataset.as_txt(id_) for id_ in list_test]
```

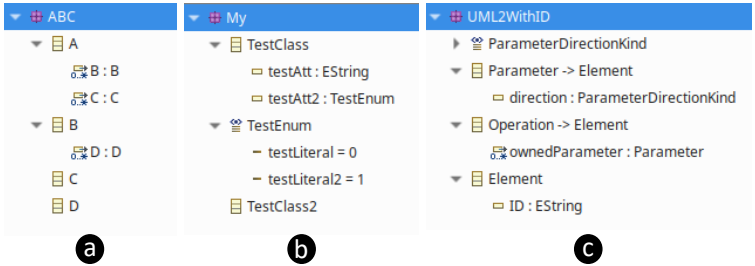
Example – Classifier for categories

```
# Encode as vectors using TF/IDF
vectorizer = TfidfVectorizer(stop_words = None,
  tokenizer = ms.simple_tokenizer, min_df = 2)
X_train = vectorizer.fit_transform(train_corpus)
X_test = vectorizer.transform(test_corpus)

# Train with 100 neurons
n = 100
clf = MLPClassifier(random_state=1,hidden_layer_sizes = (n,), max_iter=1000).fit(X_train,
     y_train)
y_pred = clf.predict(X_test)
score = accuracy_score(y_test, y_pred)
```

Example – Dummy models

- Dummy model: example or test models not meant to be complete
- Build a simple classifier (e.g., decision tree)
- Features:
  - Number of elements per type (e.g., number of EClasses)
  - Median of the number of characters in string attributes
  - Count of dummy names (e.g., ClassA)



Example – Recommendation

- Infer the next edit operation and attribute values based on the context
- On-going work
- Video https://www.youtube.com/watch?v=Lm_1PHPPZYQ

Example – Model analytics (simple)

```
$ sqlite3 repo-github-ecore/analysis.db
sqlite> . schema
CREATE TABLE models (
   id         varchar(255) PRIMARY KEY,
   relative_file text NOT NULL,
   hash       text NOT NULL,
   status     varchar(255) NOT NULL,
   metadata_document TEXT,
   duplicate_of varchar(255)
);
CREATE TABLE stats (
   id   varchar(255) NOT NULL,
   type varchar (255) NOT NULL,
   count integer NOT NULL
);
```

Example – Simple Model Analytics

```
sqlite> select status, count(*) from models group by status;
status     count(*)
---------- ----------
CRASHED    574
DUPLICATED 46199
NOT_HANDLE 37
NO_VALIDAT 7598
TIMEOUT    7
VALID      12907
```

How many valid/invalid models are?

Example – Simple Model Analytics

```
sqlite> select type, avg(count) from stats group by type;
type       avg(count)
---------- ----------------
attributes 14.9562058034626
classes    17.4278956352109
datatypes  0.80736405754694
elements   144.151377712753
enum       1.06944647646915
packages   1.394391611802
references 18.2064862228725
```

Average number of elements (by Ecore element)

Example – Simple Model Analytics

```
sqlite> select json_each.key as smell, count(json_each.value) as total from models,
    json_each(json_extract(metadata_document, '$.smells')) group by json_each.key order
    by total desc;
smell          total
-------------------- ----------
IsolatedClassSmell 7534
OverLoadedClassSme 3892
ReferredAlotClassS 3617
OnlyOneClassSuperS 3390
RefersAlotClassSme 2910
TooManyChildrenSme 2080
IrrelevantClassSme 1855
UninstantiableClas 1532
DepthHierarchySmel 1327
TooLongNamesSmell 310
```

Number of smell ocurrences

Model analytics

- Finding quasi-duplicates
  - Many models stored in GitHub and GenMyModel are quasi-duplicates
- Comparing model repositories
  - Is a repository (structurally) similar to another?
- Clustering
  - Detect patterns

Other applications

- Automatic model modularity
- Learning to generate realistic models
- Recommender system
- Learning to rank
- Model summarization
- Architecture recovery of MDE projects
- Model clone detection

Conclusions

Resources

Papers

- José Antonio Hernández, Jesús Sánchez Cuadrado.
  MAR: A structure-based search engine for models. MoDELS'20.
- José Antonio Hernández, Jesús Sánchez Cuadrado.
  Searching models in the wild with MAR. SoSyM.
- José Antonio Hernández, Javier Luis Cánovas Izquierdo, Jesús Sánchez Cuadrado.
  ModelSet: A Dataset for Machine Learning in Model-Driven Engineering. SoSyM (submitted).

URLs

- http://mar-search.org
- http://modelset.github.io

Future work

MAR

- A proper server (current server is my office computer!)
- Improve the user interface (e.g., model summarization)
- Which are other sources of models?
- Improve the crawlers. Is it possible to crawl generically?
- Which are the services that people want?
- How to make the search more robust to naming?
- **Find out what is useful for the community**

Future work

ModelSet

- More labelling
  - Probably needs to be collaborative
  - Obtain tags from external sources (e.g., DBPedia)
- Create python package for models and ML (e.g., with pyEcore)
- Make examples easily available

Applications

- Keep developing applications

Thanks for your attention! ❤️

Any questions?

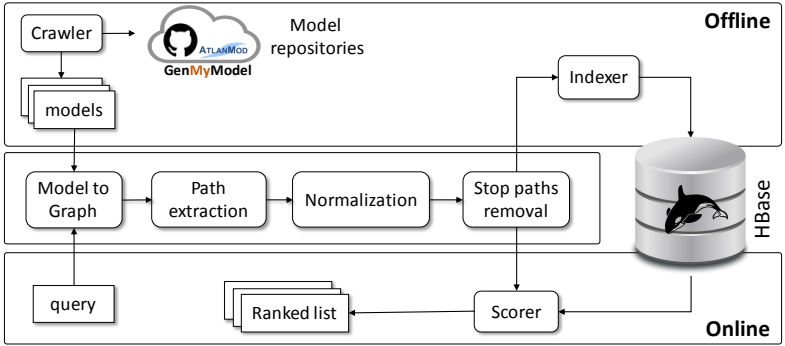
Try it!

http://mar-search.org

joseantonio.hernandez6@um.es
@antolin_hl
https://github.com/Antolin1

jesusc@um.es
@sanchezcuadrado
http://github.com/jesusc

Architecture



Performance

- Create synthetic models by mutating Ecore models
- Query the repository up to 20,000 Ecore models
- Small (#elements < 20)
  - ~ 0.3 seconds
- Medium (20 < #elements < 70)
  - ~ 0.6 seconds
- Large (#elems > 70)
  - ~ 1 second