An Online Business Process Model-driven Generator of the Conceptual Database Model

D. Brdjanin, D. Banjac, G. Banjac and S. Maric
University of Banja Luka, Bosnia & Herzegovina
About the paper

• Different surveys show that only a **small number of papers present implemented automatic model-driven generator of data models**
• This paper presents **the first online two-phase business process model-driven generator of the conceptual database model (CDM)**
• The implemented online generator enables automatic generation of the target data model based on business process models (BPM) represented by two concrete notations: **BPMN** and **UML activity diagram**
• Target model is represented by a **UML class diagram**
Related work

- POM-based approaches to data model synthesis

The majority of the proposed approaches enables (semi)automated generation of the target model with modest precision and completeness
Direct synthesis of the target model

- The **direct synthesis of the target model** based on business process models represented by concrete notations (e.g. BPMN or UML activity diagram) introduces dependency of the generation process from the source notation.

- Recently, we proposed an indirect **two-phase approach**, which is based on introduction of a simple **domain specific language (DSL)**, named **Business Model Representation Language (BMRL)**, as an intermediate layer between source and target notations.
Two-phase synthesis of the target model

Phase I

Phase II

BPMN extractor

UML AD extractor

BMRL to UML CD generator

UML Class Diagram
Two-phase synthesis of the target model

- **Two-phase synthesis** of the conceptual database model simplifies the process of the target model generation based on different source notations:
  - Complex set of rules is **implemented in only one generator**
  - If some modifications of the generation rules are necessary, then **only this generator is to be modified**
  - This generator is **independent of different source BPM notations**
  - Dependency on different concrete source notations is reduced (metamodel changes and/or vendor specific implementations)
  - It is **easier to extract simple BMRL concepts** from the source BPM then to directly generate target model
BPM-driven CDM synthesis

Our recent experiment conducted with database professionals confirmed that identified semantic potential and implemented generator enable automatic generation of very high percentage of the target model with very high precision:

- **Average effectiveness:**
  - ~ 78% (for automatic generation of classes)
  - ~ 85% (for automatic generation of associations)

- **Average recall of the generated model is >80%**

- **Average precision:**
  - ~ 75% (for automatically generated classes)
  - ~ 90% (for automatically generated associations)
• After the **identification of the BPM concepts having semantic potential** for the automated CDM synthesis we implemented a DSL named **BMRL**

• **BMRL grammar:**

```plaintext
grammar org.unibl.etf.BMRL with
org.eclipse.xtext.common.Terminals
generate
bMRL"http://www.etf.unibl.org/bmrl2cd/BMRL"
Model:
  (elements+=AbstractElement)*;
PackageDeclaration:
  'package' name=QualifiedName '{'
  (elements+=AbstractElement)*
  '}'
AbstractElement:
  PackageDeclaration | Import | GeneralizedParticipant | Object | ObjectReference | Task;
QualifiedName:
  ID ('.' ID)*;
Import:
  'import'
importedNamespace=QualifiedNameWithWildcard;
QualifiedNameWithWildcard:
  QualifiedName '.*'?
GeneralizedParticipant: Participant | Role;
Participant: 'participant' name=ID;
Role:
  'role' name=ID ('('
    superRole=[Role|QualifiedName] ')') | 'of'
  participant=[Participant|QualifiedName]);
Object:
  'object' name=ID;
ObjectReference:
  'objectReference' name=ID 'references'
  object=[Object|QualifiedName] ('[ ' state=ID ']')?
  (existing?='existing')?;
IOObjectReference:
  reference=[ObjectReference|QualifiedName]
  'multiplicity' multiplicity=Multiplicity;
Task:
  'task' name=ID '{'
    'actor' ':'
  actor=[GeneralizedParticipant|QualifiedName]
    ('input' '{'
      (inputObjects+=IOObjectReference)* '}'
    )?
    ('output' '{'
      (outputObjects+=IOObjectReference)* '}'
    )?
  }'
  INT | '-1';
```
**BMRL**

- **Xtext framework** is used for DSL specification:
  - Parser-based approach
  - Relies on **Eclipse Modeling Framework** models for internal AST representation
- **BMRL metamodel:**

```
+ name: EString
+ state: EString
+ existing: EBoolean = false
```

![Diagram of BMRL metamodel](image-url)
BPM extractors

- Two different BPM extractors:
  - UML activity diagram
  - BPMN
- **Acceleo** is used for implementation of extractors
- **Simple set of rules** enables the extraction of:
  - Participants and their roles
  - Generated and existing objects
  - Actions/tasks having input and/or output objects
CDM generator

• The target CDM is to be generated based on the BMRL-based representation of the extracted concepts

• Xtend-based generator is used for implementation:
  – Implements complex set of rules that enables automated synthesis of the conceptual database model based on extracted BMRL concepts
  – Dependent on simple and unique BMRL concepts
  – Independent of different source BPM notations
Illustrative example

participant Librarian
participant Member
object Book
object Request
object Catalog
objectReference ExistingBook_ references Book existing
objectReference ExistingCatalog_ references Catalog existing
objectReference Book_ references Book
objectReference Request_ references Request
objectReference Book_Loaned references Book[Loaned]
task Reception {
    actor: Member
    input { Book_ multiplicity 1 }
    output { } }
task Issuing {
    actor: Librarian
    input { Book_Loaned multiplicity 1 }
    output { Book_ multiplicity 1 } }
task Requesting {
    actor: Member
    input { ExistingCatalog_ multiplicity 1 }
    output { Request_ multiplicity 1 } }
task Loaning {
    actor: Librarian
    input { ExistingBook_ multiplicity 1
        Request_ multiplicity 1 }
    output { Book_Loaned multiplicity 1 } }

Brdjanin, Banjac, Banjac and Maric: An Online Business Process Model-driven Generator of the Conceptual Database Model
Online CDM generator

• During the last several years of research, we identified the semantic capacity of the BPMs, specified the formal transformation rules and implemented a set of tools for the automatic CDM synthesis.

• This set of tools, like all existing tool-supported MDSDM approaches, is platform-dependent – all tools are implemented as Eclipse plug-ins.

• In order to obtain a platform independent and publicly available tool for the BPM-driven CDM synthesis, we performed the migration of these tools into a SOA application.
Architecture of the online generator

![Diagram showing the architecture of the online generator](image_url)

- **Client** connects to the **REST** service through **HTTP**.
- **input.uml** and **input.bpmn** are sent to the **REST** service.
- The **REST** service sends **cdm.uml** back to the client.
- The **REST** service interacts with the **Apache Tomcat 8.5** server.
- The **Web application** contains **extractors**:
  - **ad2bmrl.jar**
  - **bpmn2bmrl.jar**
- The **generator** is responsible for processing the **input.bmrl** and generates **bmrl2cd.jar**.
Usage of the online generator

http://m-lab.etf.unibl.org:8080/generator/services/generate/cdm

```java
FileDataBodyPart filePart = new FileDataBodyPart("input", new File("path_to_source_model.uml"));
FormDataMultiPart multipart = new FormDataMultiPart();
multipart.field("source_model_type", "AD");
    bodyPart(filePart);

ClientConfig clientConfig = new ClientConfig().register(MultiPartFeature.class);
Client client = ClientBuilder.newClient(clientConfig);
WebTarget target = client.target("http://m-lab.etf.unibl.org:8080/generator/services/").path("generate").path("cdm");
Response response = target.request().post(Entity.entity(multipart, multipart.getMediaType()));
if (response.getStatus() == 200) {
    InputStream is = response.readEntity(InputStream.class);
    File f = new File("path_to_target_model.uml");
    FileUtils.copyToFile(is, f);
    is.close();
} filePart.cleanup();
multipart.close();
client.close();
response.close();
```
Conclusion and future work

• First online BPM-driven CDM generator implemented as a web-based, platform-independent tool, is presented
• Its usage can be twofold:
  – Database designers are able to use it through the implemented client application
  – Developers are able to invoke the exposed web service from their own applications
• The two-phase synthesis enables the automatic generation of the target CDM represented by UML class diagram, based on BPMs represented by two concrete notations: BPMN and UML activity diagram
• In the future we plan to:
  – Further identify the semantic capacity of BPMs for automated CDM design
  – Further improve BMRL
  – Further improve implemented tools
  – Provide visualization and editing functionalities of the generated models in the Web browser
D. Brdjanin, D. Banjac, G. Banjac and S. Maric
University of Banja Luka, Bosnia & Herzegovina

An Online Business Process Model-driven Generator of the Conceptual Database Model

Thank You!