Model Driven Architecture Implementation using Linked Data and Digital Archives

Evgeny Cherkashin

Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences, Irkutsk, Russia
eugeneai@icc.ru

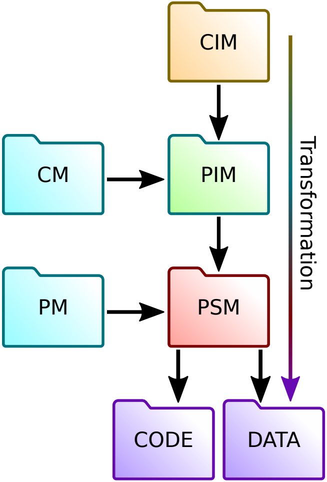
2019, December
China

Research objectives

Main objective of the research is to construct a MDA technology based on nowadays system modeling visual languages (SysML, BPMN, CMMN) and existing Semantic Web **vocabularies** and **technologies**. The following techniques and software are under development:

1. CIM representation with SysML, BPMN, CMMN, and results of source code processing,
2. CIM, PIM, PSM representation in RDF with existing vocabularies,
3. transformation implementation with logical language Logtalk,
4. usage of LOD sources in transformations for obtaining additional semantic data,
5. generation of documents and user interfaces with LOD markup.

Model–Driven Architecture



CIM	Computationally Independent Model;
CM	Model of Computations;
PIM	Platform Independent Model;
PM	Platform Model;
PSM	Platform–Specific Model;
CODE	Source code of software;
DATA	Initial database state.

Logtalk as transformation definition language

We have chosen Logtalk as it

- inherits widely known Prolog language syntax and runtime;
- implemented as macro package, performance penalties are about 1.5%;
- has flexible semantics: we can define transformations and constraints within the same syntax;
- implement object-oriented knowledge (rules) structuring, encapsulation and replacement;
- compositional way of transformation implementation;
- powerful engine to post constraints on object-to-object messages (events);
- has implementation for many Prolog engines.

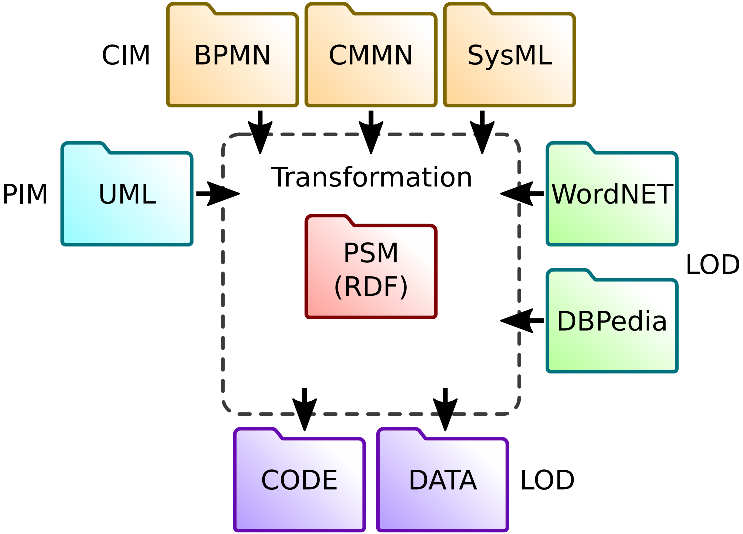
The «regular» language allow us to use its libraries not directly related to MDA transformations.

Linked Open Data, LOD

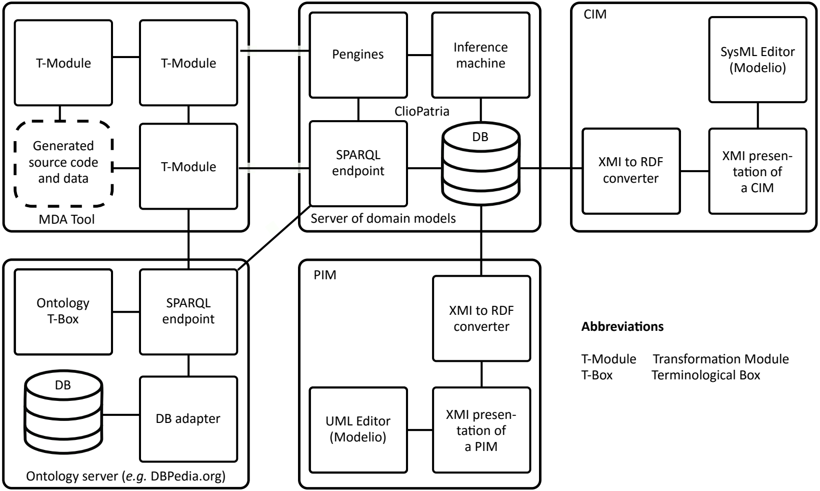
1. Information is published in Internet with open access license;
2. It is represented in a machine-readable form, e.g., Excel table instead of a bitmap picture;
3. An open format used, e.g., CSV instead of Excel;
4. The format is based on W3C recommended standards, allowing RDF and SPARQL reference;
5. Published data refer to objects, forming context.

Thus, applications publish data as relations of objects (entities).

Model Driven Architecture and Linked Open Data



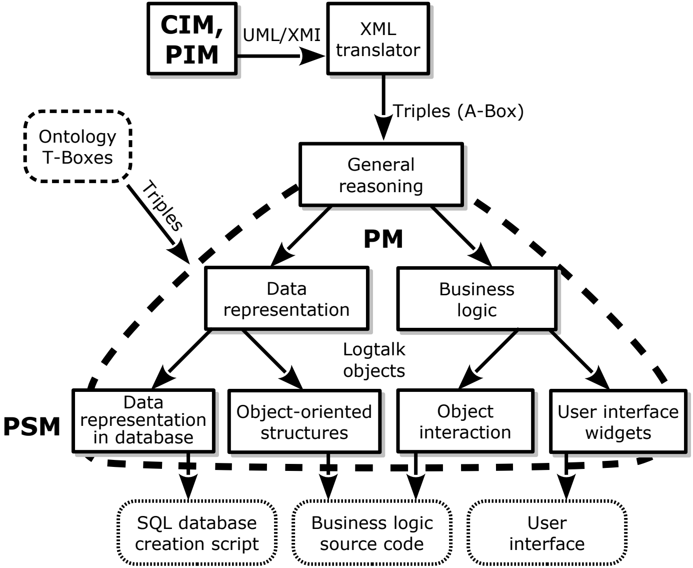
MDA infrastructure



Abbreviations

T-Module	Transformation Module
T-Box	Terminological Box

Architecture of transformation modules

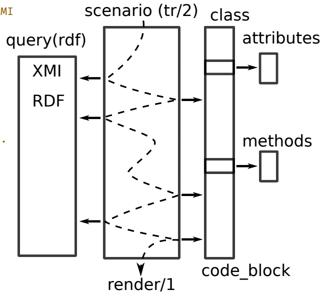


PSM: Scenario of a Class synthesis

```
:- object(direct(_Package,_LocalProf,_CodeProf)).    % Transformation driver object
:- public([tr/4,tr/3]).                              % Public interface of a class synthesis scenario
% . . . . . . . . . .
tr(class, Class, ClassID):- ::package(Package),      % Synthesize a class
    query(Package)::class(Name, ClassID),            % Query package structure in XMI
    create_object(Class,      % . . . . .            % Create a «Class» object
    create_object(Attributes, % . . . . .            % Create «Attributes» object
    create_object(Methods,    % . . . . .            % ...«Methods».
    Class::name(Name),                               % Name the class.
    % Generate attributes of the class,
    % organizing them in a local database.
    % ...methods...
    Class::attributes(Attributes),                   % Set the attributes for class.
    Class::methods(Methods).                         % ...methods.

tr(attribute, Attribute, ClassID, AttributeID):-     % Attribute transformations
    ::package(Package),
    query(Package)::attribute(Name,ClassID,AttrID),
    create_object(Attribute,  % . . . . .
    Attribute::name(Name).                           % Name the attribute.

tr(method, Method, ClassID, MethodID):-              % Transformation of methods
    ::package(Package),
    query(Package)::method(Name,ClassID,MethodID),
    create_object(Method,     % . . . . .
    Method::name(Name).                              % Name of the method
:- end_object.
```

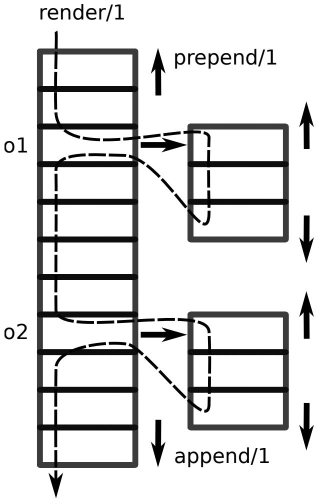


Implementation of Query object

```
:- object(query(_XMI)).
:- protected(xmi/1).
:- public([class/2, attribute/3, method/3]).
xmi(XMI) :- parameter(1, XMI).
class(Name, ID):-                            % Recognition of Class in RDF
    ::xmi(XMI),
    XMI::rdf(ID,rdf:type,uml:'Class'),
    XMI::rdf(ID,rdfs:label, literal(Name)).
attribute(Name, ClassID, ID):-               % ...attribute...
    ::xmi(XMI),
    XMI::rdf(ClassID, xmi:ownedAttribute, ID),
    XMI::rdf(ID, rdfs:label, literal(Name)).
method(Name, ClassID, ID):-                  % ...method...
    ::xmi(XMI),
    XMI::rdf(ClassID, xmi:ownedOperation, ID),
    XMI::rdf(ID, rdfs:label, literal(Name)).
% . . . . . . . . . . .
:- end_object.
```

Code Block (idea is taken from `llvmlite`*)

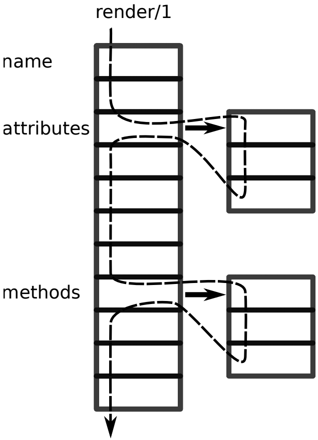
```
:- object(code_block, specializes(root)).
% Public interface of the object
:- public([append/1, prepend/1, clear/0,
   render/1, render_to/1, remove/1,
   item/1, items/1]).
% Code block items
:- dynamic([item_/1]).
:- private([item_/1]).
% Methods specialized during inheritance
:- protected([renderitem/2, render_to/2]).
% . . . . . . . . . . . . .
% Delegate rendering to object itself
renderitem(Object, String):-
    current_object(Object), !,
    Object::render(String).
% Convert a literal to its string
% representation
renderitem(literal(Item), String):-!,
    atom_string(Item, String).
% Just print the item (debugging).
renderitem(Item, String):-
    root::iswritef(String, '%q', [Item]).
:- end_object
```



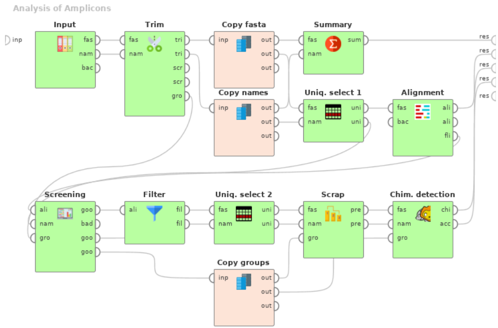
*) https://github.com/numba/llvmlite

PSM of a Python Class as a specialization of Code Block

```
:- object(class, specializes(code_block),
   imports([named])). % Category of named entities
:- public([classlist/1, methods/1, attributes/1]).
% . . . . . . . . . . . . . .
renderitem(Item, Result):-      % proceed with default
    ^^renderitem(Item, Result). % rendering
render(Result):-         % Source generator
    ^^render(Name),      % implemented in a category
    ( ::item(classlist(List)) ->
     % . . . . . . . . . . .
        [Name]) ),
    ( ::item(attributes(Attributes))->
     % . . . . . . . . . . .
        [DefAttrList]),
      Attributes::items(InstanceAttrs),
      findall(S, ( % initialize attributes
         % . . . . . . . . .
         ), AttrAssigns),
        root::unindent,
        AttrList=[ConstructorDef|AttrAssigns];
         % . . . . . . . . .
        AttrList=[ConstructorDef, Pass] ),
    ( ::item(methods(Methods))-> % If any ...
      Methods::render(MethodList);
      MethodList=[] ),
    lists::append(AttrList,MethodList,StringList),
    root::unindent, Result=[Signature|StringList].
```



Applications: Dataflow representation of NGS analysis of amplicons



Term	Description
NGS	New Generation Sequencing
Amplicon	A DNA or RNA part copied many times
Mothur	A software toolset for NGS research
Rapidminer	A visual tool for data mining modeling and execution

Green blocks are Mothur modules.
Others are Rapidminer modules.

Discussion

Interesting positive impressions obtained:

- Logtalk and RDF are flexible, sufficiently universal and convenient implementation infrastructures for MDA;
- The best implemenation means is Prolog predicate wrapping and Logtalk object encapsulation of rules;
- Not all Logtalk properties are investigated: there might be more sophisticated programming techniques developed, *e.g.*, on the base of message watchers.

Technical problems making the approach somewhat problematic:

- Very simple tasks take too much efforts, *e.g.*, text processing: convert an identifier into the CamelCase;
- It takes too long to surf Internet in order to find a vocabulary for a domain, but it is more productive than development;
- Prolog is not a popular language in MDA, neither Logtalk.

Document authoring and storage

In most cases documents are created as a result of

- creative activity of a person with a text processors (authoring);
- printing a digital copy or a data record in a database;
- aggregation operation over database records (report).

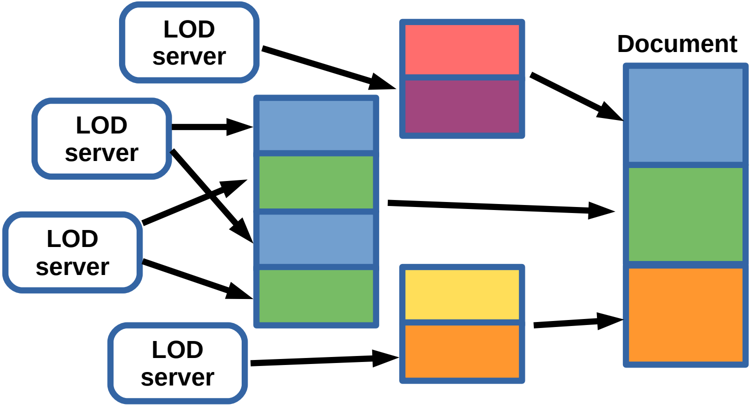
Then it is stored either as a physical paper and/or a digital document (PDF, DOCX, HTML).

Since 2000-th, Semantic Web and Linked Open Data (LOD) is being developed, allowing

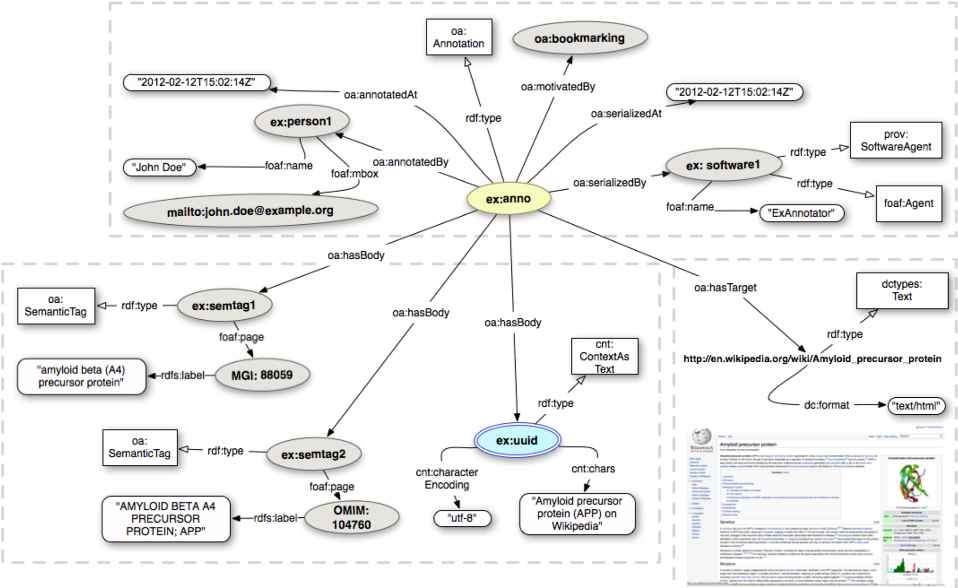
- structural storage of data within published documents;
- processing stored data computationally;
- integration of data structures and data objects globally.

The **aim of this research** is to develop technologies, software and services allowing construction of digital archives supporting document data inclusion and inference from existing documents.

Structure of a document



Open Annotaiton (oa)

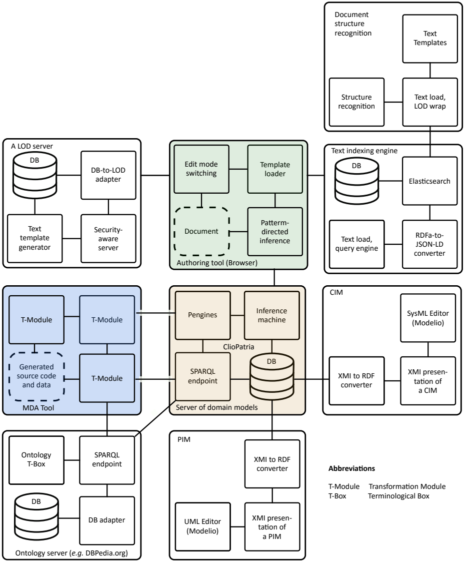


Representation

```
<html lang=”ru” xmlns=http://www.w3.org/1999/xhtml
 xmlns:taa =http://irnok.net/engine/rdfa-manipulation
xml:lang=”ru” metal:define-macro=”page”>
<head> . . . . </head>
<body prefix=”rdf: http://www.w3.org/1999/...-ns# foaf: http://xmlns.com/foaf/...
imei: imei.html# course: https://irnok.net/college/plan/01..16-...\
%D0\%BA_PB-SM.plm.xml.xlsx-....2.3.1.html#”  resource=”#post”
typeof=”schema:CreativeWork sioc:Post prov:Entity”>
<!– The application control panel –>

<main lang=”ru” resource=”#annotation” typeof=”oa:Annotation” id=”main-doc-cnt”>
<div property=”oa:hasTarget” resource=”#course-work-prog”></div>
<article property=”oa:hasBody” typeof=”foaf:Document curr:WorkingProgram”
         resource=”#course-work-program” id=”main-document”>
  <div  taa:content =”imei:title-page”></div>
  <div  taa:content =”imei:neg-UMK”></div>
  <section id=”TOC” class=”break-after”> <h2>Table of Contents</h2>
    <div id=”tableOfContents”></div>
  </section>
  <section id=”course-description” resource=”#description”
           property=”schema:hasPart” typeof=”schema:CreativeWork”>
    <div property=”schema:hasPart” resource=”#purpose”
         typeof=”dc:Text cnt:ContentAsText” >
      <div property=”cnt:chars” datatype=”xsd:string”>
        <h2 property=”dc:title” datatype=”xsd:string”>
           Aims and objectives of the discipline (module)</h2>
        <p>The aim of teaching the discipline ...</p>
      </div>
   </div>
  . . . . . . . .
```

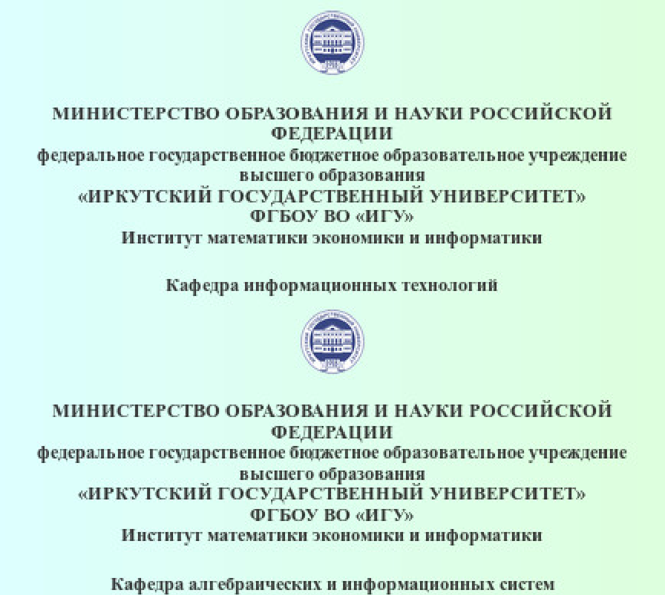
Architecture



Abbreviations

T-Module	Transformation Module
T-Box	Terminological Box

Generated list of title page preambles



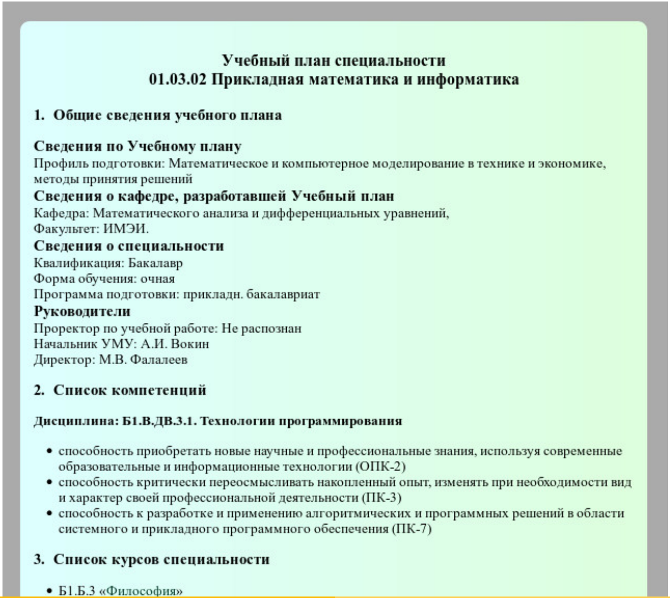
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ФГБОУ ВО «ИГУ»
Институт математики экономики и информатики

Кафедра информационных технологий

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ФГБОУ ВО «ИГУ»
Институт математики экономики и информатики

Кафедра алгебраических и информационных систем

Generated part of a study program



Учебный план специальности
01.03.02 Прикладная математика и информатика

1. Общие сведения учебного плана

Сведения по Учебному плану
Профиль подготовки: Математическое и компьютерное моделирование в технике и экономике, методы принятия решений

Сведения о кафедре, разработавшей Учебный план
Кафедра: Математического анализа и дифференциальных уравнений,
Факультет: ИМЭИ.

Сведения о специальности
Квалификация: Бакалавр
Форма обучения: очная
Программа подготовки: прикладн. бакалавриат

Руководители
Проректор по учебной работе: Не распознан
Начальник УМУ: А.И. Вокин
Директор: М.В. Фалалеев

2. Список компетенций

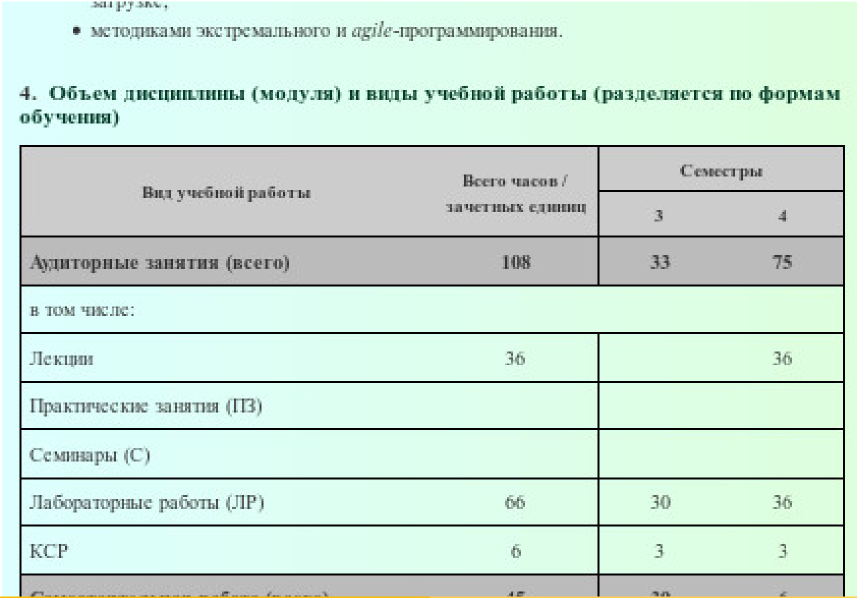
Дисциплина: Б1.В.ДВ.3.1. Технологии программирования

- способность приобретать новые научные и профессиональные знания, используя современные образовательные и информационные технологии (ОПК-2)
- способность критически переосмысливать накопленный опыт, изменять при необходимости вид и характер своей профессиональной деятельности (ПК-3)
- способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения (ПК-7)

3. Список курсов специальности

- Б1.Б.3 «Философия»

Imported time distribution for lecture, seminary, ...



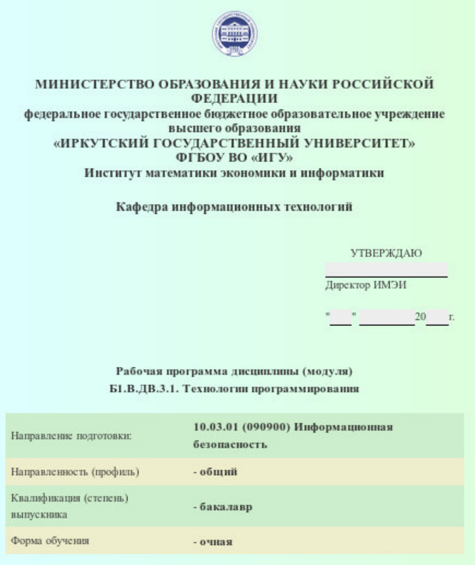
загрузке;

- методиками экстремального и *agile*-программирования.

4. Объем дисциплины (модуля) и виды учебной работы (разделяется по формам обучения)

Вид учебной работы	Всего часов / зачетных единиц	Семестры	
		3	4
Аудиторные занятия (всего)	**108**	**33**	**75**
в том числе:			
Лекции	36		36
Практические занятия (ПЗ)			
Семинары (С)			
Лабораторные работы (ЛР)	66	30	36
КСР	6	3	3

Complete document



МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ФГБОУ ВО «ИГУ»
Институт математики экономики и информатики

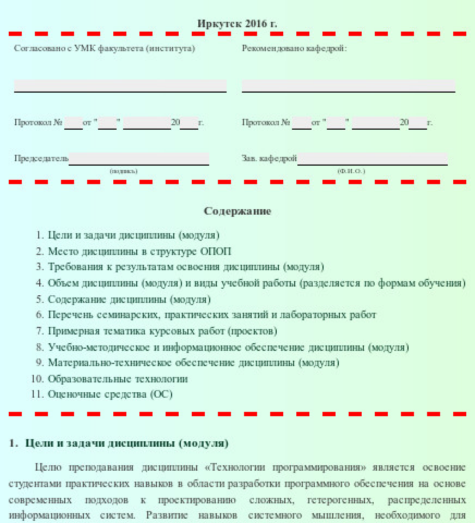
Кафедра информационных технологий

УТВЕРЖДАЮ
____
Директор ИМЭИ

"___" ________ 20___ г.

Рабочая программа дисциплины (модуля)
Б1.В.ДВ.3.1. Технологии программирования

Направление подготовки:	**10.03.01 (090900) Информационная безопасность**
Направленность (профиль)	**- общий**
Квалификация (степень) выпускника	**- бакалавр**
Форма обучения	**- очная**



Иркутск 2016 г.

Согласовано с УМК факультета (института)

____

Протокол № ___ от "___" ________ 20___ г.

Председатель ______________
(подпись)

Рекомендовано кафедрой:

____

Протокол № ___ от "___" ________ 20___ г.

Зав. кафедрой ______________
(Ф.И.О.)

Содержание

1. Цели и задачи дисциплины (модуля)
2. Место дисциплины в структуре ОПОП
3. Требования к результатам освоения дисциплины (модуля)
4. Объем дисциплины (модуля) и виды учебной работы (разделяется по формам обучения)
5. Содержание дисциплины (модуля)
6. Перечень семинарских, практических занятий и лабораторных работ
7. Примерная тематика курсовых работ (проектов)
8. Учебно-методическое и информационное обеспечение дисциплины (модуля)
9. Материально-техническое обеспечение дисциплины (модуля)
10. Образовательные технологии
11. Оценочные средства (ОС)

1. Цели и задачи дисциплины (модуля)

Целю преподавания дисциплины «Технологии программирования» является освоение студентами практических навыков в области разработки программного обеспечения на основе современных подходов к проектированию сложных, гетерогенных, распределенных информационных систем. Развитие навыков системного мышления, необходимого для

Used ontologies

- Friend-of-a-friend (**foaf**) - agent information: individuals, legal entities, program agents.
- Provenance (**prov**) - references between documents.
- Dublin Core (**dc**) - edited annotation mark up.
- DBPedia resource (**dbr**) – references to instant objects and classes.
- Schema.org (**schema**) - Google, Yandex, Yahoo, *etc.* searchable objects, structural elements.
- The Bibliographic Ontology (**bibo**) - literature reference mark up.

Conclusion

A tools (components) for digital archive implementation, which allows to device information systems and document processing services with the following features:

- load LOD marked up document, extract, store in a graph and index RDF data;
- retrieve RDF data as triples or as a result of full-text search query;
- combine existing LOD data and its content in new documents dynamically with browser based context inference machine;
- use server-site inference machine (Prolog) to process RDF data upon request from browser's part of the system;
- convert created RDFa marked up HTML5 documents into Excel and Word formats.

Applications

- Document authoring automation;
- Context-depended editing;
- Self-organizing global document flows;
- Documents as data sources for information systems.

TabbyXL

Software Platform for Rule-Based Spreadsheet Data Extraction and Transformation

Alexey Shigarov, Vasiliy Khristyuk, et al

shigarov@icc.ru

Motivation

- About arbitrary spreadsheet tables
  - A large volume of valuable data for science and business applications
  - A big variety of layout, style, and content features
  - Human-centeredness (incorrect structure and messy content)
  - No explicit semantics for interpretation by computers
- Challenges
  - How to extract tables from worksheets
  - How to recognize and correct cell structure anomalies
  - How to recover semantics needed for the automatic interpretation
  - How to conceptualize extracted data by using external vocabularies

Background

Table understanding includes the following tasks

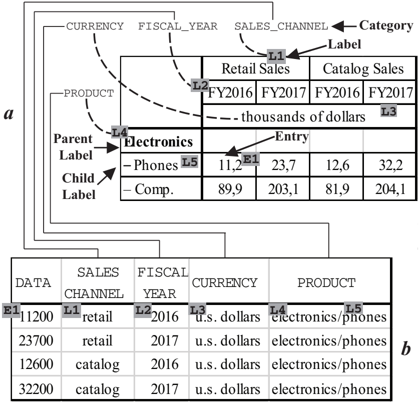
1. **Extraction** — detecting a table and recognizing the physical structure of its cells
2. **Role analysis** — extracting functional data items from cell content
3. **Structural analysis** — recovering internal relationships between extracted functional data items
4. **Interpretation** — linking extracted functional data items with external vocabularies (general-purpose or domain-specific ontologies)

Contribution

TabbyXL is a software platform aiming at the development and execution of rule-based programs for spreadsheet data extraction and transformation from arbitrary (*a*) to relational tables (*b*)

Novelty

- Table object model assigning roles to data items, not cell
- CRL, domain-specific language to express user-defined rules for table analysis and interpretation
- CRL-to-Java translator to synthesize executable programs for spreadsheet data transformation



	Retail Sales		Catalog Sales	
	FY2016	FY2017	FY2016	FY2017
	thousands of dollars			
Electronics				
– Phones	11,2	23,7	12,6	32,2
– Comp.	89,9	203,1	81,9	204,1

DATA	SALES CHANNEL	FISCAL YEAR	CURRENCY	PRODUCT
11200	retail	2016	u.s. dollars	electronics/phones
23700	retail	2017	u.s. dollars	electronics/phones
12600	catalog	2016	u.s. dollars	electronics/phones
32200	catalog	2017	u.s. dollars	electronics/phones

User-Defined Rules

- The user-defined rules map the physical structure into the logical structure of a table
  - **WHEN-part** queries facts about the structure by using constraints
  - **THEN-part** modifies available facts and asserted new ones
- The facts are represented by items of the *table object model*
- The rules can be expressed in a rule-based language (e.g. Drools[1], Jess[2], or CRL[3])

[1] https://www.drools.org

[2] https://jessrules.com

[3] https://github.com/tabbydoc/tabbyxl/wiki/crl-language

Table Object Model

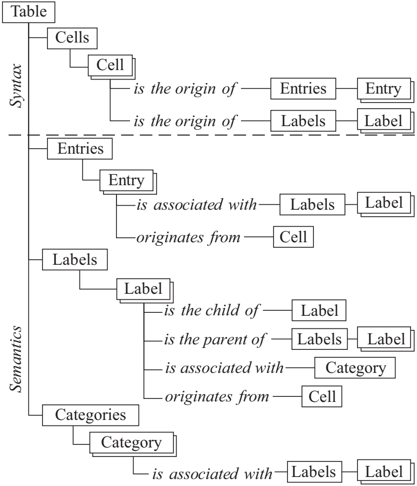
Physical Layer

Cells characterized by layout, style, and content features

Logical Layer

Functional data items and their relationships:

- ❑ entries (values)
- ❑ labels (keys)
- ❑ categories (concepts)
- ❑ entry-label pairs
- ❑ label-label pairs
- ❑ label-category pairs



CRL Grammar

```
rule        = 'rule' <a Java integer literal> 'when' condition
              'then' action 'end' <EOL> {rule} <EOF>
condition   = query identifier [':' constraint {',' constraint}
              [',' assignment {',' assignment}]] <EOL> {condition}
constraint  = <a Java boolean expr>
assignment  = identifier ':' <a valid Java expr>
query       = 'cell' | 'entry' | 'label' | 'category' | 'no cells' |
              'no entries' | 'no labels' | 'no categories'
action      = merge | split | set text | set indent | set mark |
              new entry | new label | add label | set parent |
              set category | group <EOL> {action}
merge       = 'merge' identifier 'with' identifier
split       = 'split' identifier
set text    = 'set text' <a Java string expr> 'to' identifier
set indent  = 'set indent' <a Java integer expr> 'to' identifier
set mark    = 'set mark' <a Java string expr> 'to' identifier
new entry   = 'new entry' identifier ['as' <a Java string expr>]
new label   = 'new label' identifier ['as' <a Java string expr>]
add label   = 'add label' identifier | (<a Java string expr>
              'of' identifier | <a Java string expr>)
              'to' identifier
set parent    = 'set parent' identifier 'to' identifier
set category = 'set category' identifier | <a Java string expr>
                'to' identifier
group         = 'group' identifier 'with' identifier
identifier    = <a Java identifier>
```

Cell Cleansing

The actions correct an inaccurate layout and content of a hand-coded table

- <merge> combines two adjacent cells when they share one border
- <split> divides a merged cell that spans n-tiles (row-column intersections) into n-cells
- <set text> modifies a textual content of a cell
- <set indent> modifies a text indentation of a cell

Example

```
when
  cell corner: cl == 1, rt == 1, blank
  cell c: cl > corner.cr, rt > corner.rb
then
  split c
```

Role Analysis

The actions recover entries and labels as functional data items presented in a table

- <set mark> annotates a cell with a user-defined tag that can be used in subsequent table analysis
- <new entry> (<new label>) creates an entry (label) from a cell content with the use of an optional string processing

Example

```
when
  cell corner: cl == 1, rt == 1, blank
  cell c: cl > corner.cr, rt > corner.rb
then
  new entry c
```

Structural Analysis

The actions recover pairs of two kinds: entry-label and label-label

- <add label> associates an entry with a label
- <set parent> binds two labels as a parent and its child

Example

```
when
  cell c1: cl == 1
  cell c2: cl == 1, rt > c1.rt, indent == c1.indent + 2
  no cells: cl == 1, rt > $c1.rt, rt < $c2.rt, indent == $c1.indent
then
  set parent c1.label to c2.label
```

Interpretation

The actions serve to recover label-category pairs

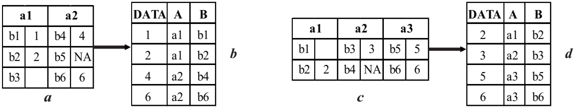
- <set category> associates a label with a category
- <group> places two labels to one group that can be considered as an undefined category

Example

```
when
  label l1: cell.mark == ”stub”
  label l2: cell.mark == ”stub”, cell.rt == l1.cell.rt
then
  group l1 with l2
```

Illustrative Example

The transformation of arbitrary tables with the same layout features (*a* and *c*) to their canonicalized versions (*b* and *d*)



a

a1		a2	
b1	1	b4	4
b2	2	b5	NA
b3		b6	6

b

DATA	A	B
1	a1	b1
2	a1	b2
4	a2	b4
6	a2	b6

c

a1		a2		a3	
b1		b3	3	b5	5
b2	2	b4	NA	b6	6

d

DATA	A	B
2	a1	b2
3	a2	b3
5	a3	b5
6	a3	b6

The ruleset for the cell cleansing (*a*), role analysis (*b*, *c*), structural analysis (*d*, *e*), and interpretation (*f*, *g*)

a

```
when cell c: c.text.matches("NA")
then set text "" to c
```

b

```
when cell c: (cl % 2) == 0, !blank
then new entry c
```

c

```
when cell c: (cl % 2) == 1
then new label c
```

d

```
when
   entry e
   label l: cell.cr == e.cell.cr
then add label l to e
```

e

```
when
   entry e
   label l: cell.rt == e.cell.rt, cell.cl == e.cell.cl - 1
then add label l to e
```

f

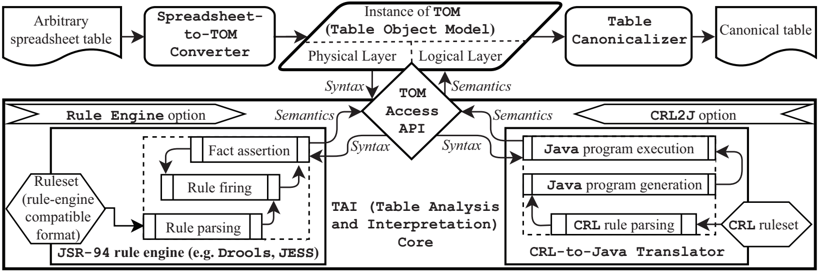
```
when label l: cell.rt == 1
then set category "A" to l
```

g

```
when label l: cell.rt > 1
then set category "B" to l
```

This example is reproducible at https://codeocean.com/capsule/5326436

Architecture



Two options are provided

Rule Engine option

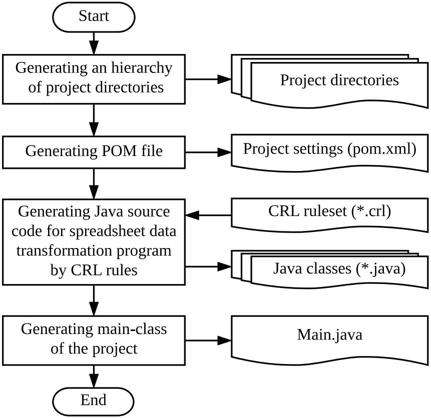
Executing a ruleset in an appropriate format with a JSR-94 compatible rule engine (e.g. Drools, Jess)

CRL2J option

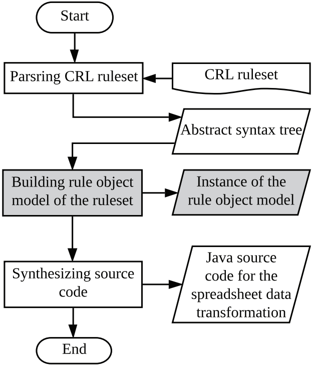
Translating a ruleset expressed in CRL to an executable Java program

CRL2J Translation

Workflow for generating a Maven-project of a spreadsheet data transformation program

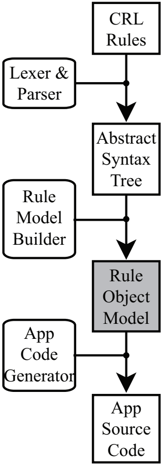

Start
Generating an hierarchy of project directories
Project directories
Generating POM file
Project settings (pom.xml)
Generating Java source code for spreadsheet data transformation program by CRL rules
CRL ruleset (*.crl)
Java classes (*.java)
Generating main-class of the project
Main.java
End

Workflow for translating a CRL ruleset to Java source code


Start
Parsring CRL ruleset
CRL ruleset
Abstract syntax tree
Building rule object model of the ruleset
Instance of the rule object model
Synthesizing source code
Java source code for the spreadsheet data transformation
End

CRL2J Translation

In the Workflow



CRL Rules

Lexer & Parser

Abstract Syntax Tree

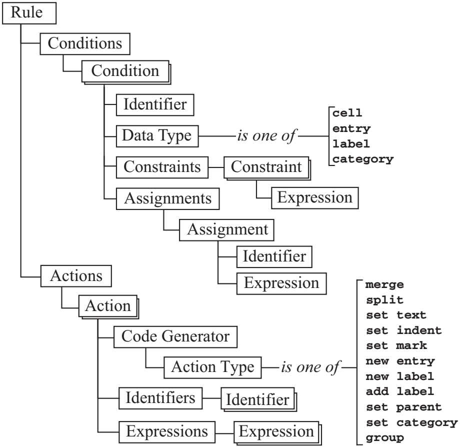
Rule Model Builder

Rule Object Model

App Code Generator

App Source Code

Rule Object Model



Rule

Conditions

Condition

Identifier

Data Type *is one of*

`cell`
`entry`
`label`
`category`

Constraints

Constraint

Expression

Assignments

Assignment

Identifier

Expression

Actions

Action

Code Generator

Action Type *is one of*

`merge`
`split`
`set text`
`set indent`
`set mark`
`new entry`
`new label`
`add label`
`set parent`
`set category`
`group`

Identifiers

Identifier

Expressions

Expression

CRL2J Translation

Example (Source Rule)

```
when
  cell corner: cl == 1, rt == 1, blank
  cell c: cl > corner.cr, rt > corner.rb, ! marked
then
  set mark ”@entry” to c
  new entry c
```

Example (Fragment of the Generated Java Code)

```
...
Iterator<CCell> iterator1 = getTable().getCells();
while (iterator1.hasNext()) {
  corner = iterator1.next();
  if ((corner.getCl() == 1) && (corner.getRt() == 1) && ...
    Iterator<CCell> iterator2 = getTable().getCells();
    while (iterator2.hasNext()) {
...
```

Performance Evaluation

The results of the transformation of 200 tables of Troy200 dataset

	Role analysis		Structural analysis	
	Type of instances			
Metrics	entries	labels	entry-label pairs	label-label pairs
Recall	0.9813 $\frac{16602}{16918}$	0.9965 $\frac{4842}{4859}$	0.9773 $\frac{34270}{35066}$	0.9389 $\frac{1951}{2078}$
Precision	0.9996 $\frac{16602}{16609}$	0.9364 $\frac{4842}{5171}$	0.9965 $\frac{34270}{34389}$	0.9784 $\frac{1951}{1994}$
F-score	0.9904	0.9655	0.9868	0.9582

Metrics

$$\text{recall} = \frac{|R \cap S|}{|S|} \quad \text{precision} = \frac{|R \cap S|}{|R|}$$

S is a set of instances in a source table, R is a set of instances in its canonical form

All data and steps to reproduce the results are available at http://dx.doi.org/10.17632/ydcr7mcrtp.5

Performance Evaluation

The comparison of the running time by using TabbyXL with three different options for transforming 200 tables of Troy200 dataset [Nagy, 2016]

Running time of	CRL2J	Drools	Jess
Ruleset preparation (t_1)	2108* ms	1711† ms	432† ms
Ruleset execution (t_2)	367** ms	1974‡ ms	4149‡ ms

* t_1 — a time of parsing and compiling the original ruleset into a Java program
** t_2 — a time of executing the generated Java program

† t_1 — a time of parsing the original ruleset and adding the result into a rule engine session
‡ t_2 — a time of asserting facts into the working memory and matching rules against the facts

For testing, we used 3.2 GHz 4-core CPU

Comparison with Others

Role Analysis

- *Contest task*: The segmentation of a table into typical functional cell regions
- *Testing dataset*: Troy200 [Nagy, 2016]
- *Contestant*: MIPS (TANGO) [Embley et al., 2016]
- *Accuracy*: MIPS (TANGO) — **0.9899** vs. TabbyXL — **0.9950**

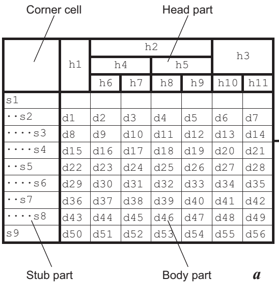
Structural Analysis

- *Contest task*: The extraction of header hierarchies from tables
- *Testing dataset*: A random subset of SAUS[a]
- *Contestant*: Senbazuru [Chen and Cafarella, 2014]
- *F-score*: Senbazuru — **0.8860** vs. TabbyXL — **0.8657**

[a] http://dbgroup.eecs.umich.edu/project/sheets/datasets.html

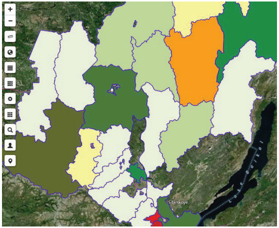
Application Experience

Populating a web-based statistical atlas of the Irkutsk region — (*b*) via extracting data from government statistical reports — (*a*)



	h1	h2				h3	
		h4		h5			
		h6	h7	h8	h9	h10	h11
s1							
··s2	d1	d2	d3	d4	d5	d6	d7
····s3	d8	d9	d10	d11	d12	d13	d14
····s4	d15	d16	d17	d18	d19	d20	d21
··s5	d22	d23	d24	d25	d26	d27	d28
····s6	d29	d30	d31	d32	d33	d34	d35
··s7	d36	d37	d38	d39	d40	d41	d42
····s8	d43	d44	d45	d46	d47	d48	d49
s9	d50	d51	d52	d53	d54	d55	d56

DATA	HEAD	STUB
d1	h1	s1|s2
d2	h2|h4|h6	s1|s2
d3	h2|h4|h7	s1|s2
d4	h2|h5|h8	s1|s2
d5	h2|h5|h9	s1|s2
d6	h3|h10	s1|s2
d7	h3|h11	s1|s2
d8	h1	s1|s2|s3
d9	h2|h4|h6	s1|s2|s3
d10	h2|h4|h7	s1|s2|s3
d11	h2|h5|h8	s1|s2|s3
d12	h2|h5|h9	s1|s2|s3
d13	h3|h10	s1|s2|s3
d14	h3|h11	s1|s2|s3
...	...	...
d56	h3|h11	s9

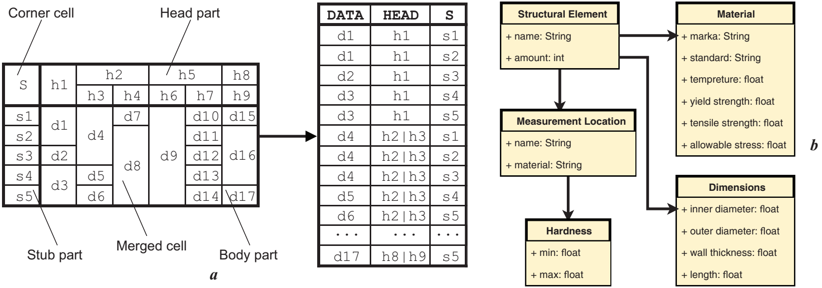


b

The more detail can be found at `https://github.com/tabbydoc/tabbyxl/wiki/statistical-atlas`

Application Experience

Generating conceptual models — (*b*) from arbitrary tables presented in industrial safety inspection reports — (*a*)



<table>
<tr><td rowspan="2">S</td><td rowspan="2">h1</td><td colspan="2">h2</td><td colspan="2">h5</td><td>h8</td></tr>
<tr><td>h3</td><td>h4</td><td>h6</td><td>h7</td><td>h9</td></tr>
<tr><td>s1</td><td rowspan="2">d1</td><td rowspan="3">d4</td><td>d7</td><td rowspan="5">d9</td><td>d10</td><td>d15</td></tr>
<tr><td>s2</td><td rowspan="4">d8</td><td>d11</td><td rowspan="3">d16</td></tr>
<tr><td>s3</td><td>d2</td><td>d12</td></tr>
<tr><td>s4</td><td rowspan="2">d3</td><td>d5</td><td>d13</td></tr>
<tr><td>s5</td><td>d6</td><td>d14</td><td>d17</td></tr>
</table>

DATA	HEAD	S
d1	h1	s1
d1	h1	s2
d2	h1	s3
d3	h1	s4
d3	h1	s5
d4	h2|h3	s1
d4	h2|h3	s2
d4	h2|h3	s3
d5	h2|h3	s4
d6	h2|h3	s5
...	...	...
d17	h8|h9	s5

Structural Element
+ name: String
+ amount: int

Material
+ marka: String
+ standard: String
+ tempreture: float
+ yield strength: float
+ tensile strength: float
+ allowable stress: float

Measurement Location
+ name: String
+ material: String

Hardness
+ min: float
+ max: float

Dimensions
+ inner diameter: float
+ outer diameter: float
+ wall thickness: float
+ length: float

The more detail can be found at https://github.com/tabbydoc/tabbyxl/wiki/industrial-safety-inspection

Conclusions & Further Work

- Impact on software development for spreadsheet data management
  - Table object model associating functional roles with data items
  - Table analysis and interpretation driven by user-defined rules
  - Formulated actions to recover missing semantics of arbitrary tables
  - Translation of rules to executable spreadsheet transformation programs
- Limitations
  - The inaccurate cell structure prevents the table analysis
  - The very limited interpretation (without external vocabularies)
- Further work
  - Rearrangement of cell structure by using visual (human-readable) cells
  - Detecting derived data by spreadsheet formulas
  - Enriching the table analysis by named entity recognition
  - Linking extracted data items with LOD cloud

References I

Chen, Z. and Cafarella, M. (2014).

Integrating spreadsheet data via accurate and low-effort extraction.
In *Proc. 20th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pages 1126–1135.

Embley, D. W., Krishnamoorthy, M. S., Nagy, G., and Seth, S. (2016).

Converting heterogeneous statistical tables on the web to searchable databases.
Int. J. Document Analysis and Recognition, 19(2):119–138.

Nagy, G. (2016).

TANGO-DocLab web tables from international statistical sites (Troy_200), 1, ID: Troy_200_1.

Thanks

Read more about the project at
http://td.icc.ru

The project source code is available at
https://github.com/tabbydoc/tabbyxl

Thanks for Your interest to our project!

Rapidminer module

```
vector<string> AlignCommand::setParameters(){ // PART OF MODULE SOURCE
try {
  CommandParameter ptemplate("reference", "InputTypes", "", "", "none", "none", "none","",false,true,true); parameters.push_back(ptemplat
  CommandParameter pcandidate("fasta", "InputTypes", "", "", "none", "none", "none","fasta-alignreport-accnos",false,true,true); paramete
  CommandParameter psearch("search", "Multiple", "kmer-blast-suffix", "kmer", "", "", "","",false,false,true); parameters.push_back(psear
  CommandParameter pksize("ksize", "Number", "", "8", "", "", "","",false,false); parameters.push_back(pksize);
  CommandParameter pmatch("match", "Number", "", "1.0", "", "", "","",false,false); parameters.push_back(pmatch);
// . . . . . . .
package com.rapidminer.ngs.operator; // GENERATED JAVA MODULE
// imports

class MothurChimeraCcodeOperator extends MothurGeneratedOperator {
  private InputPort fastaInPort = getInputPorts().createPort("fasta");
  private InputPort referenceInPort = getInputPorts().createPort("reference");
  private OutputPort chimeraOutPort = getOutputPorts().createPort("chimera");
  private OutputPort mapinfoOutPort = getOutputPorts().createPort("mapinfo");
  private OutputPort accnosOutPort = getOutputPorts().createPort("accnos");

  public MothurChimeraCcodeOperator (OperatorDescription description) {
    super(description);
  }
  @Override
  public void doWork() throws OperatorException {
    super();
    // . . . . . . .
  }
  @Override
  public List<ParameterType> getParameterTypes() {
    super();
        // . . . . . . .
  }
  @Override
  public String getOutputPattern(String type) {
    if (type=="chimera") return "[filename],[tag],ccode.chimeras-[filename],ccode.chimeras";
    if (type=="mapinfo") return "[filename],mapinfo";
    if (type=="accnos") return "[filename],[tag],ccode.accnos-[filename],ccode.accnos";
    return super.getOutputPattern(type);
```

RDF (TTL) representation and ad its query object

```
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
ngsp:spec a ngsp:Specification ;
  ngsp:module mothur:NoCommand,
    mothur:align-check,
    mothur:align-seqs,
# . . . . .
mothur:align-check a ngsp:Module ;
  ngsp:outputPattern [ a cnt:Chars ;
      ngsp:parameterName "type" ;
      ngsp:pattern [ ngsp:patternString
          "[filename],align.check" ;
          dc:identifier "aligncheck" ] ;
      cnt:chars # . . . .
# . . . . .
mothur:align-check-idir-parameter a ngsp:Parameter ;
  ngsp:important false ;
  ngsp:multipleSelectionAllowed false ;
  ngsp:optionsDefault "" ;
  ngsp:required false ;
  ngsp:type mothur:String ;
  dc:title "inputdir" .

mothur:align-check-map-parameter a ngsp:Parameter ;
  ngsp:important true ;
  ngsp:multipleSelectionAllowed false ;
  ngsp:optionsDefault "" ;
  ngsp:required true ;
  ngsp:type mothur:InputTypes ;
  dc:title "map" .

mothur:align-check-name-parameter a ngsp:Parameter ;
  ngsp:chooseOnlyOneGroup "namecount" ;
  ngsp:important false ;
  ngsp:multipleSelectionAllowed false ;
```

```
:- object(queryparam(_RDF,_Parameter),
       extends(ngsquerybase)).

:- public(type/1).
type(Type) :-
    ::attr(type, Type).
:- public(name/1).
name(Name) :- ::attr(dc:title, literal(Name)).
:- public(options/1).
options(Value):- ::attr(options, Value).
:- public(options_default/1).
options_default(Value):-
    ::attr(optionsDefault, Value).
% . . . . . . . .
:- public(multiple_selection_allowed/0).
multiple_selection_allowed:-
    ::bool_attr(multipleSelectionAllowed).
:- public(required/0).
required:-
    ::bool_attr(required).
:- public(important/0).
important:-
    ::bool_attr(important).
:- protected(attr/2).
attr(NS:Name, Value):-
    ::ngs(RDF),
    ::second(Parameter),
    rdf_db::rdf_global_object(Value, V),
    RDF::rdf(Parameter, NS:Name, V).
attr(Name, Value):-
    \+ Name=_:_,!,
    ::ngs(RDF),
    ::second(Parameter),
    rdf_db::rdf_global_id(Value, V),
    RDF::rdf(Parameter, ngsp:Name, V).
```