Virtual Integration of Heterogeneous Data and Data Model Unification
Унификация моделей данных

Сергей Ступников

ФИЦ ИУ РАН

sstupnikov@ipiran.ru

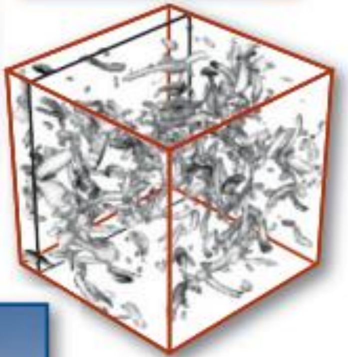
План

- Четвертая научная парадигма
- Синтез канонических моделей, унификация моделей ресурсов
- Ядро канонической модели
- Уточнение и его формализация
- Метод унификации моделей
- Графовые модели данных и их унификация
- Модель данных атрибутированных графов
- Отображение модели атрибутированных графов в каноническую информационную модель
  - Отображения языка определения данных
  - Отображение языка манипулирования данными
- Сохранение информации и семантики операций ЯМД при отображении

Science Paradigms

- Thousand years ago:
  science was **empirical**
  describing natural phenomena
- Last few hundred years:
  theoretical branch
  using models, generalizations

$$\left(\frac{\dot{a}}{a}\right)^2 = \frac{4\pi G\rho}{3} - K\frac{c^2}{a^2}$$



- Last few decades:
  a **computational** branch
  simulating complex phenomena
- Today: **data exploration** (eScience)
  unify theory, experiment, and simulation
  - Data captured by instruments
    or generated by simulator
  - Processed by software
  - Information/knowledge stored in computer
  - Scientist analyzes database/files
    using data management and statistics

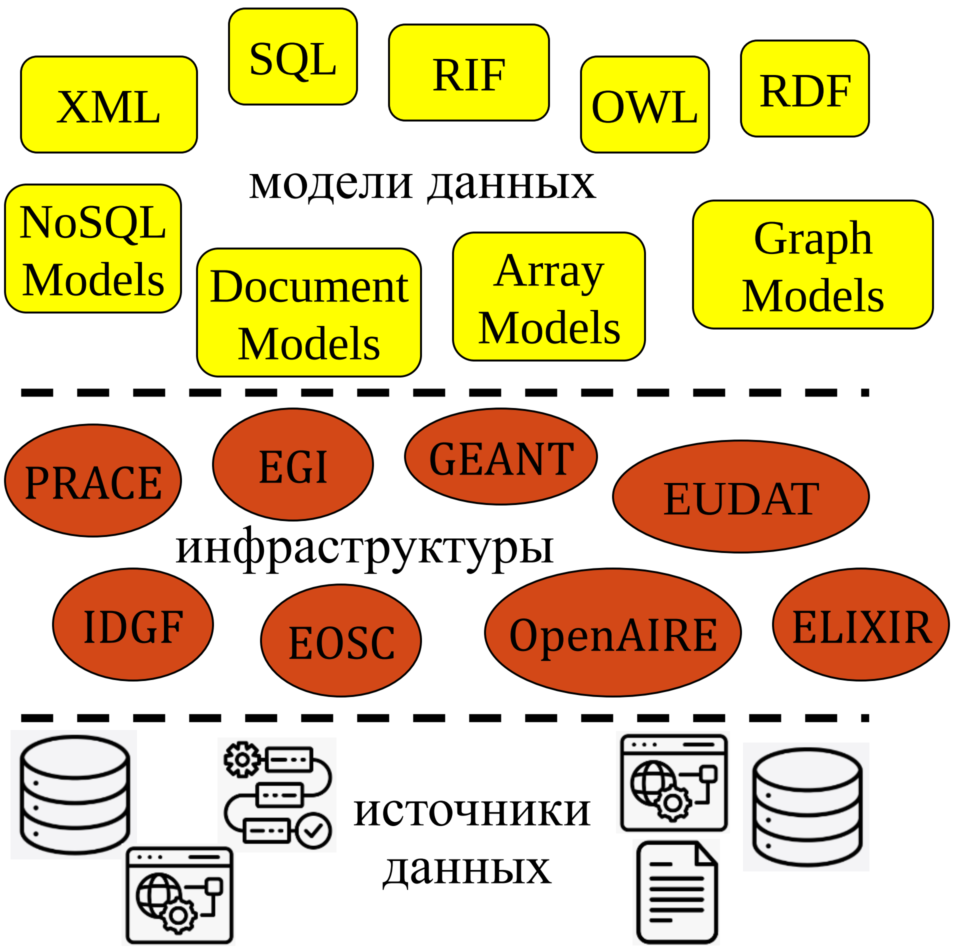
Четвертая парадигма

- Интенсивное использование данных, данные как доминирующий фактор
- Происхождение данных: наблюдательные, экспериментальные, полученные в ходе компьютерных симуляций
- Возрастающие объемы данных требуют разработки методов обработки и анализа в рамках новых технологий баз данных

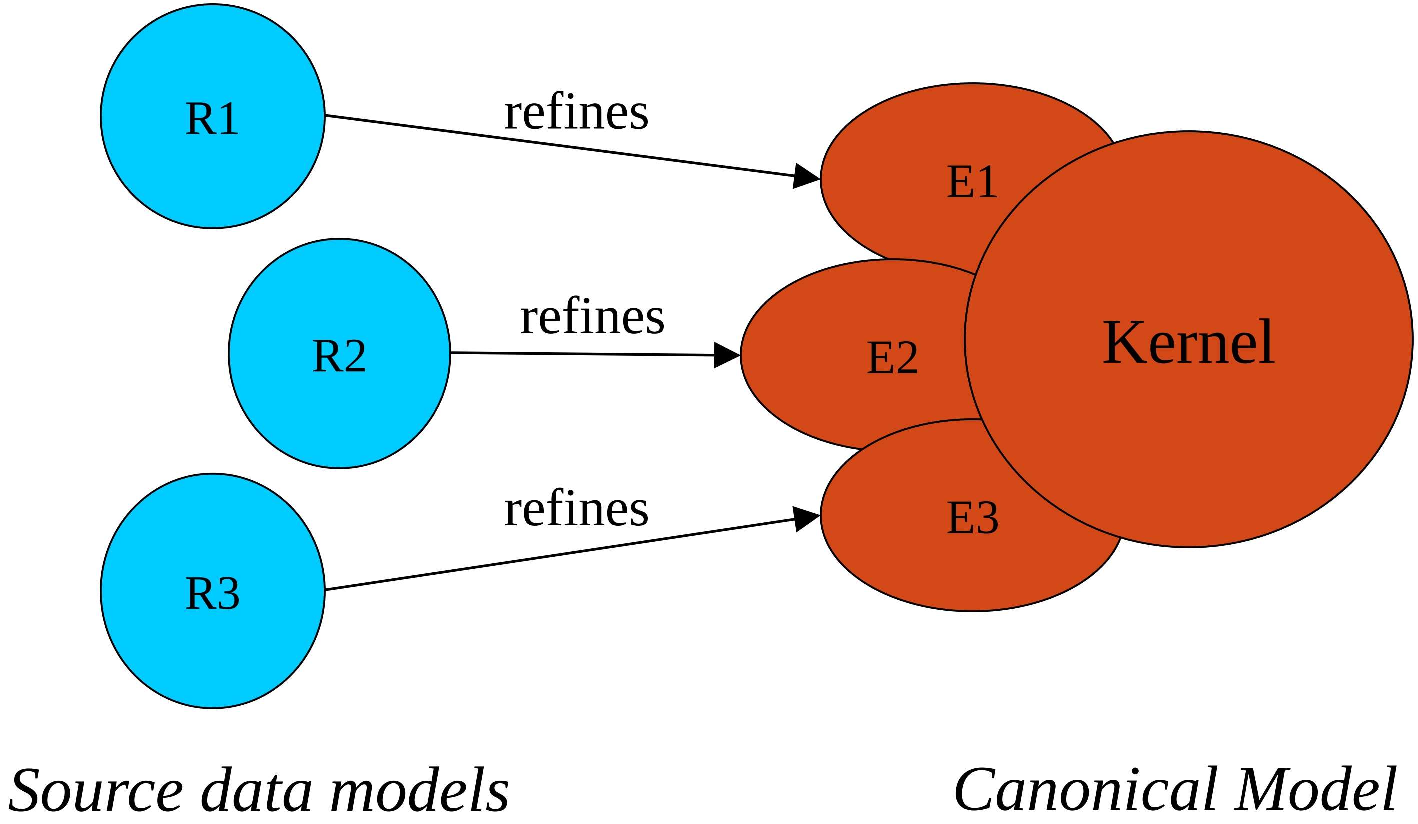
- Необходимы
  - новые подходы к концептуализации, организации и реализации информационных систем
  - методы и средства оперирования данными, объемы которых выходят за рамки возможностей современных СУБД
  - подходы, позволяющих справляться с разнообразием массово и хаотично развивающихся языков и моделей данных
    - NoSQL-модели
    - онтологические и семантические модели
    - модели, основанные на многомерных массивах
    - графовые модели
    - …

Motivation for the Creation of the Canonical Data Models

- Разнообразие моделей данных
  - реляционные (диалекты SQL), объектно-реляционные, объектные
  - массив-базированные, графовые, семантические
  - NoSQL: документные, ключ-значение, …
  - …
- Необходимость решения задач над источниками данных в исследовательских инфраструктурах
- Данные должны быть совместно и повторно используемыми
- Накопление неоднородных источников данных
  - базы данных, файлы
  - веб-сервисы, потоки работ



Synthesis of the Canonical Data Model



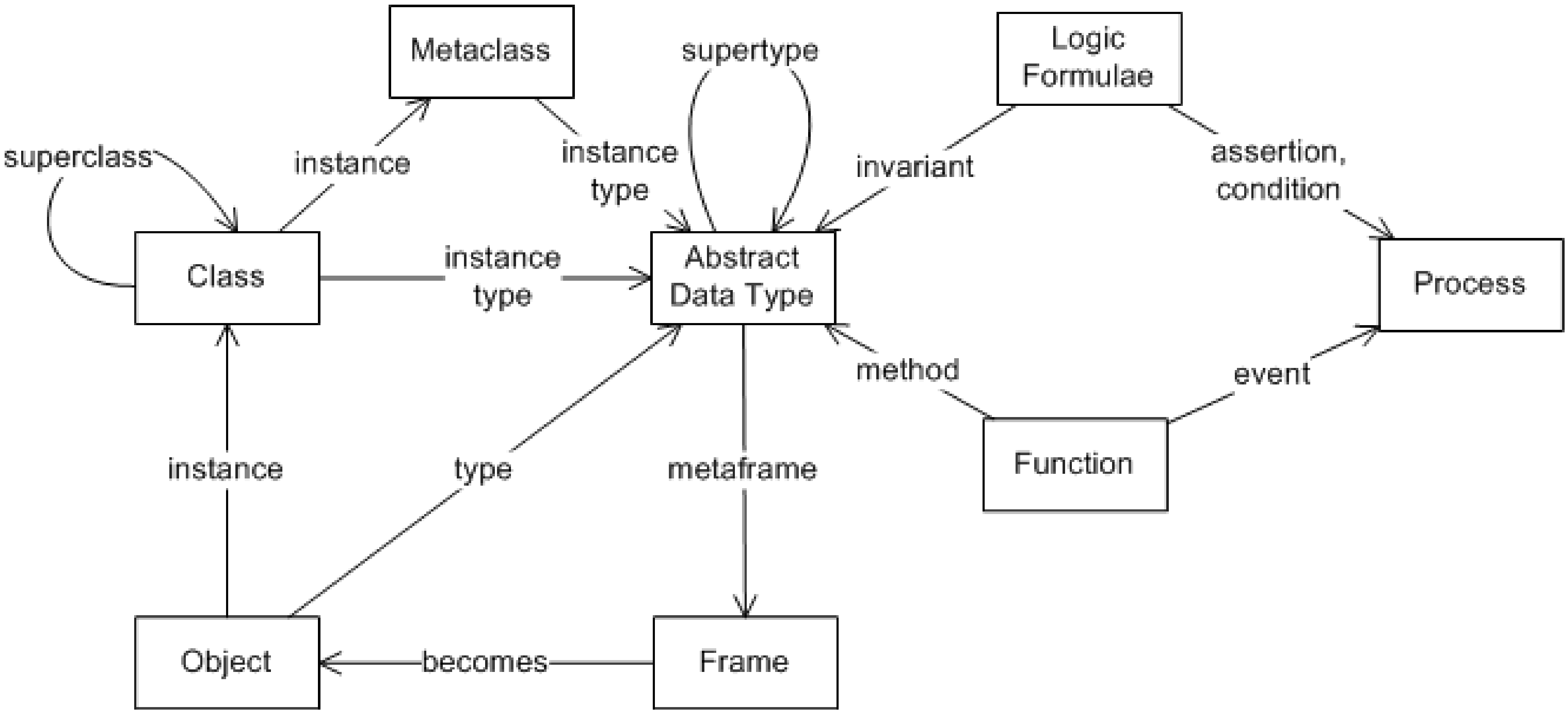
Унификация моделей данных

- *Каноническая модель* данных - общий язык, унифицирующий разнообразные модели ресурсов
- *Унификация* исходной модели данных - ее отображение в каноническую модель, сохраняющее информацию и семантику операций языка манипулирования данными (ЯМД)
  - унификация должна быть доказуемо правильной
  - унификация моделей ресурсов является необходимым предусловием для регистрации ресурсов в посреднике
- В качестве канонической модели может рассматриваться язык СИНТЕЗ - комбинированная слабоструктурированная и объектная модель данных, нацеленная на разработку предметных посредников для решения задач в средах неоднородных ресурсов
  - Разработан прототип программных средств для поддержки среды предметных посредников
  - Проведена унификация структурированных, онтологических, сервисных, процессных моделей

Canonical Data Model

- SYNTHESIS – a language intended to provide a uniform (canonical) representation of heterogeneous data, programs and processes for their use as interoperable entities – contains:
  - universal constructor of arbitrary abstract data types
  - comprehensive collection of the built-in types
  - type composition operations and type lattice
    - **T1 JOIN T2** contains all the information of T1 and T2
    - **T1 MEET T2** contains common information of T1 and T2
  - functions are given by predicative specifications expressed by mixed pre- and post-conditions formulae of typed first order predicate logic

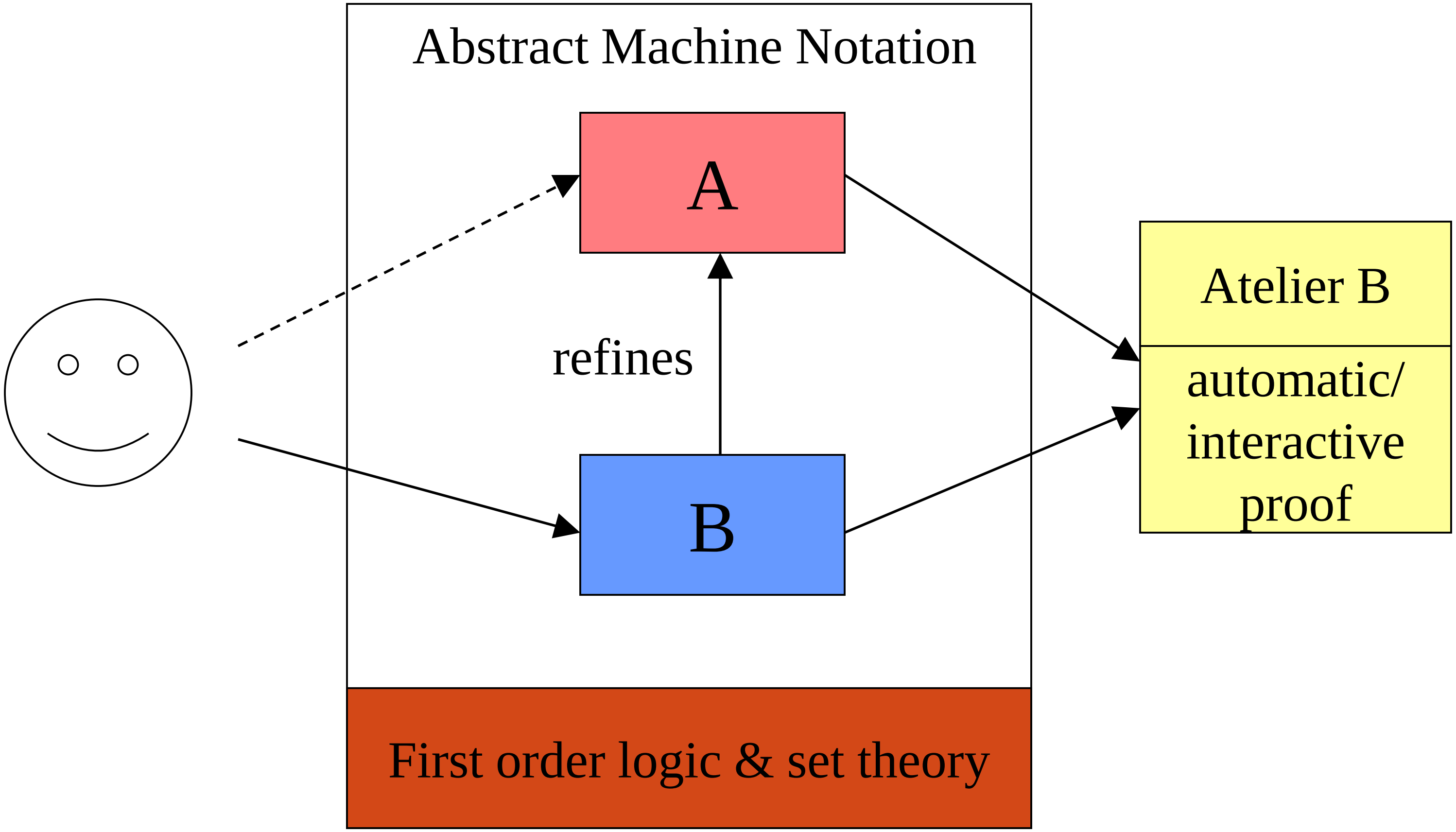
Canonical Model Kernel - SYNTHESIS language



Canonical Model Kernel Extension Facilities

- Metaclasses
- Metaframes
- Parameterized constructions (including assertions)
- Generic data types

Refinement Formalization



- Atelier B (ClearSy System Engineering, France)
  - tool support for AMN (Abstract Machine Notation) language based on the first order predicate logic and set theory;
  - tools support for automatic/interactive proving of refinement
  - is used for model's semantics formalization and refinement verification

Abstract Machine Notation

- Based on first order predicate logic and Zermelo-Frenkel set theory with axiom of choice
- Allows to consider specifications of state space and behaviour in an integrated way
- State is introduced by *state variables* together with *invariants*
- Behaviour is introduced by *operations* defined as generalized substitutions – predicate transformers
- Refinement is formalized by formulating *proof obligations*

Refinement Formalization in AMN

REFINEMENT M
REFINES K
CONSTANTS c_M
PROPERTIES P_M
VARIABLES v
INVARIANT I_M
INITIALISATION $Init_M$
OPERATIONS
  y ← op(x) =
  PRE $Pre_{op,M}$
  THEN
    $Def_{op,M}$
  END

REFINEMENT N
REFINES M
CONSTANTS c_N
PROPERTIES P_N
VARIABLES w
INVARIANT I_N
INITIALISATION $Init_N$
OPERATIONS
  y ← op(x) =
  PRE $Pre_{op,N}$
  THEN
    $Def_{op,N}$
  END

Theorem of joint state non-emptiness

$$P_M \wedge P_N \Rightarrow \exists\, \mathbf{(v,w)}\, (I_M \wedge I_N)$$

Theorem of initialization refinement

$$P_M \wedge P_N \Rightarrow [Init_N] \neg\, [Init_M] \neg\, I_N$$

Theorem of operation refinement

$$P_M \wedge P_N \wedge I_M \wedge I_N \wedge Pre_{op,M} \Rightarrow$$
$$Pre_{op,N} \wedge [Def_{op,N}\{y \to y'\}] \neg\, [Def_{op,M}] \neg$$
$$(I_N \wedge y'=y)$$

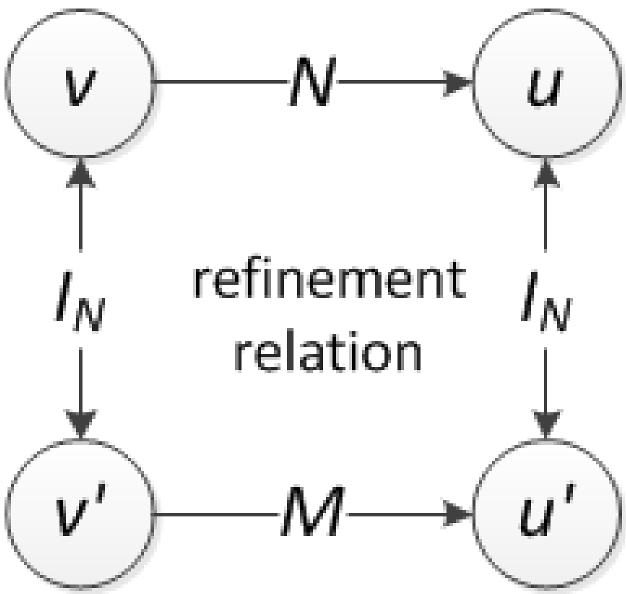
- "Operation refinement"
  - Under the refinement relation and the precondition of the more abstract operation, the precondition of the more concrete operation holds;
  - For every execution of concrete operation there is a corresponding execution of abstract operation from the same initial state which establishes the same external result values and reestablishes the refinement relation between the post-states.

Refinement Formalization Meaning

- Non-emptiness of joint state $P_M \wedge P_N \Rightarrow \exists\ \mathbf{(v,w)}\ \mathbf{(}I_M \wedge I_N\mathbf{)}$
  - There is a combined abstract (v) and concrete (w) state which satisfies the refinement relation and abstract machine invariant
    - If this failed there would not be an executable implementation of the abstract specification via the refinement
- Initialization refinement $P_M \wedge P_N \Rightarrow \mathbf{[}Init_N\mathbf{]} \neg \mathbf{[}Init_M\mathbf{]} \neg I_N$
  - The concrete initialization is a refinement of the abstract one under thy assumption of the constraints and properties of both machines

Refinement Formalization Meaning

- Operation refinement $P_M \wedge P_N \wedge I_M \wedge I_N \wedge Pre_{op,M} \Rightarrow$
$$Pre_{op,N} \wedge [Def_{op,N}\{y \rightarrow y'\}] \neg [Def_{op,M}] \neg (I_N \wedge y'=y)$$
  - Under the refinement relation (I_N) and abstract precondition ($Pre_{op,M}$)
    - concrete precondition ($Pre_{op,N}$) holds (preconditions may be weakened in refinements, or concrete operation has a wider range of behavior) and
    - for every execution of concrete operation ($Def_{op,N}$) there is a corresponding execution of abstract operation ($Def_{op,M}$) from the same initial state (under the relation I_N) which establishes the same external result values ($y'=y$) and re-establishes the refinement relation between post-states.



Refinement Formalization Meaning

- S[R] means "For every execution of S starting from this state, the execution terminates in a state satisfying R"
- ¬ S ¬ [R] means "There exists an execution of S starting from this state, which either fails to terminate or terminates in a state satisfying R"
- T¬ S ¬ [R] means "For every execution of T, there exists an execution of S starting from this state, which either fails to terminate or terminates in a state satisfying R"

AMN Specification Example

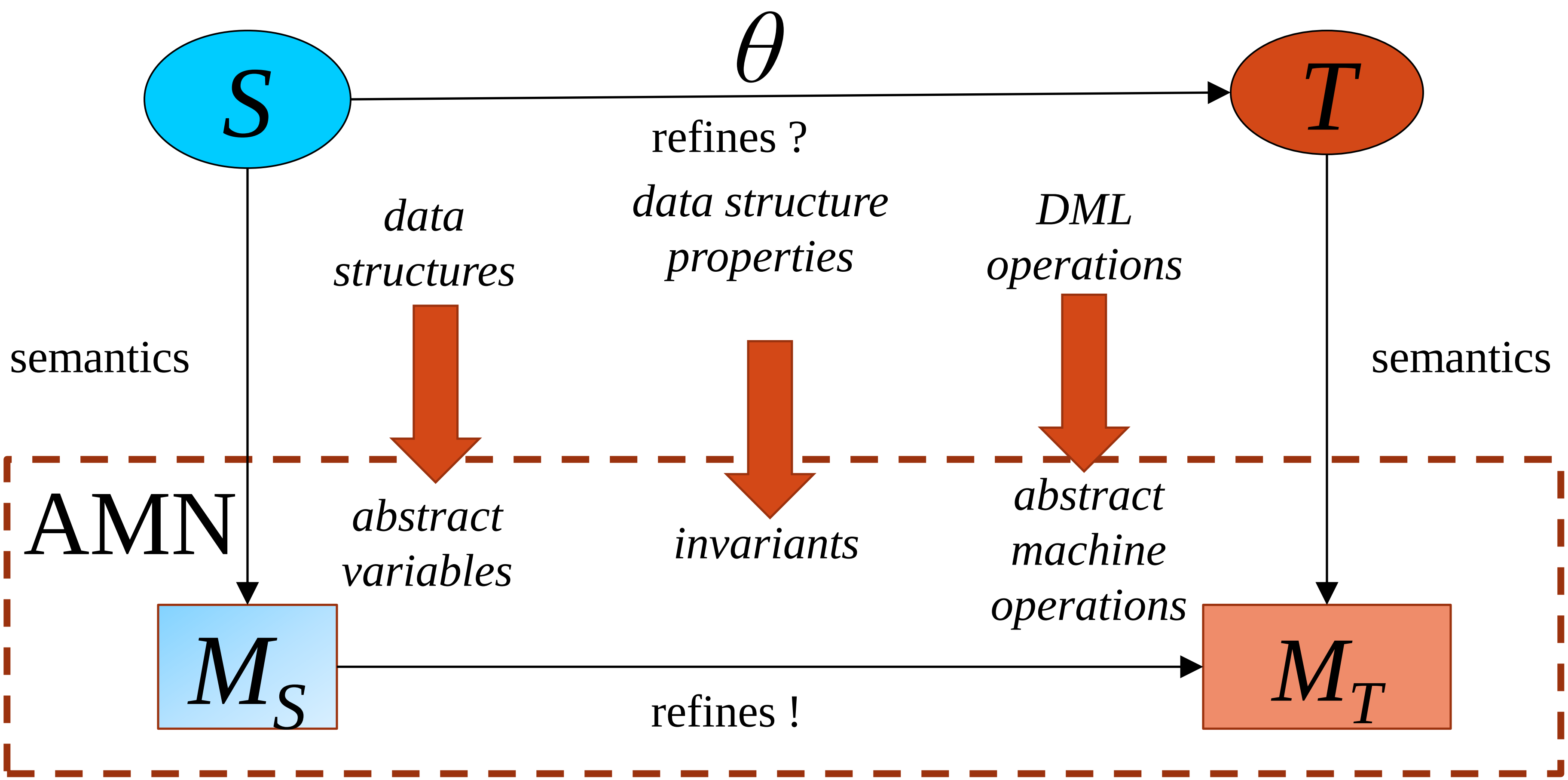
```
MACHINE FileTransfer
SETS Byte, CopyState = { Idle, Remember}
DEFINITIONS File == seq(Byte)
VARIABLES copy, buf
INVARIANT copy: CopyState & buf: File

OPERATIONS
send(f) =
PRE f: File THEN
 SELECT copy = Idle THEN
  copy:= Remember ||  buf:= f
 END
END;
g <-- receive() =
SELECT copy = Remember THEN
 copy:= Idle ||  g:= buf
END
END
```

AMN Refinement Example

```
REFINEMENT FileTransferRefinement
REFINES FileTransfer
VARIABLES afile, bfile, byteWise

INVARIANT afile: File & bfile: File &
 byteWise: {BW, Transfer, Transfer1} &

 (byteWise = Transfer => buf = bfile^afile) &
 (byteWise = Transfer1 => buf = bfile) &

 (copy = Idle <=> byteWise = BW ) &
 (copy = Remember <=>
    byteWise: {Transfer, Transfer1} )
```

```
OPERATIONS
send (f) =
PRE f: File THEN SELECT byteWise = BW THEN
  byteWise:= Transfer ||  afile:= f
END
transBlock =
SELECT byteWise = Transfer THEN
 IF afile = []  THEN
  byteWise:= Transfer1
 ELSE
  bfile:= bfile^first(afile) ||   afile:= tail(afile)
 END
END;
g <-- receive =
SELECT byteWise = Transfer1 THEN
 g:= bfile ||  byteWise:= BW
END
```

Unifying of a Data Model



- construct a mapping θ of a source model S into a target model T
- express semantics of S and T as the AMN abstract machines
- construct the gluing invariant binding states of M_S and M_T
- θ preserves information and operation semantics iff M_S refines M_T

Сохранение информации и семантики операций ЯМД при отображении

- AMN - теоретико-модельная нотация, основанная на теории множеств и типизированном языке первого порядка
  - Спецификации AMN - абстрактные машины
  - Интегрированно рассматриваются спецификация пространства состояний и поведения машины
  - Формализуется отношение *уточнения* (спецификация B уточняет спецификацию A, если пользователь может использовать B вместо A, не замечая факта замены A на B)

- Метод доказательства сохранения информации и семантики операций
  - θ - отображение модели исходной модели S в целевую модель T
  - семантика моделей представляется в виде абстрактных машин AMN M_S и M_T
    - структуры данных моделей – классы, массивы представляются переменными машин
    - структур данных представляются инвариантами машин
    - операции моделей данных представляются операциями машин
  - отображение θ сохраняет информацию и семантику операций, если машина M_S уточняет машину M_T

Унификация графовых моделей

- Унификация графовой модели данных для *виртуальной* или *материализованной* интеграции ресурсов при создании федеративных баз данных или хранилищ данных

- СУБД, основанные на графовых моделях - новый вид ресурсов для интеграции вместе с привычными ресурсами – реляционными и объектными СУБД, веб-сервисами, …
- Графовые модели
  - Информация о взаимосвязях между данными или их топологии является более важной (или настолько же важной), как сами данные
  - Данные и/или схема – граф
  - Манипулирование данными
    - трансформация графов
    - пути, подграфы, связность …
  - *Применения* - системы управления и анализа сложных сетей – социальных, биологических, информационных, транспортных, телекоммуникационных …

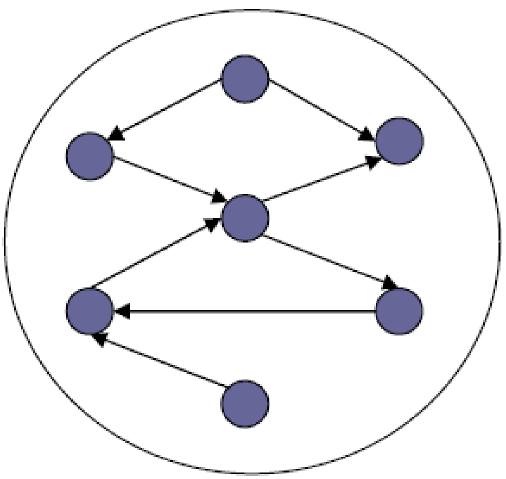
Графы, гипервершины, гиперграфы

$\mathbf{N}$ = set of simple nodes

$\mathbf{H}$ = set of hypernodes

Graph

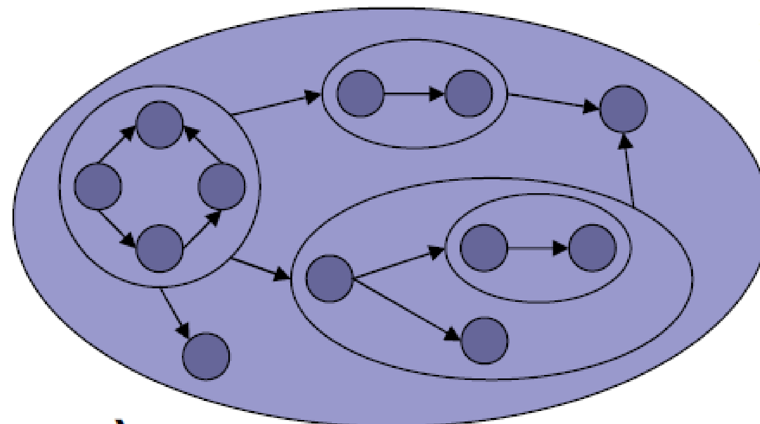
$V \subseteq N$ $\quad$ $E \subseteq V \times V$



$G = (V, E)$

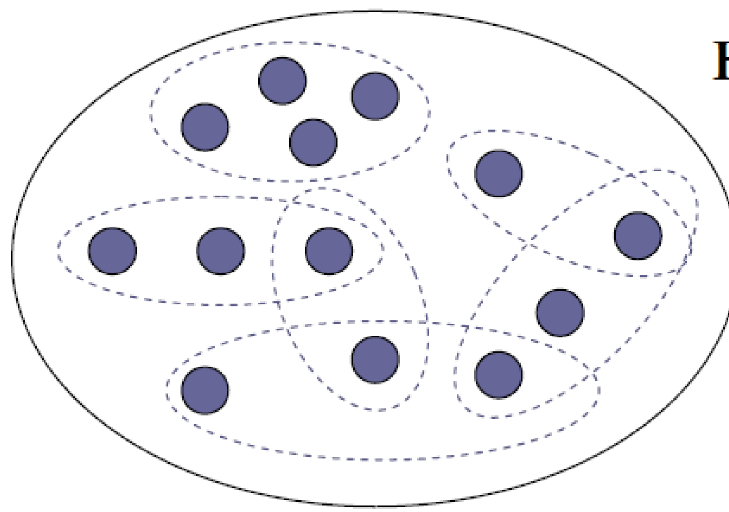
Hypernode

$V = N \cup H$



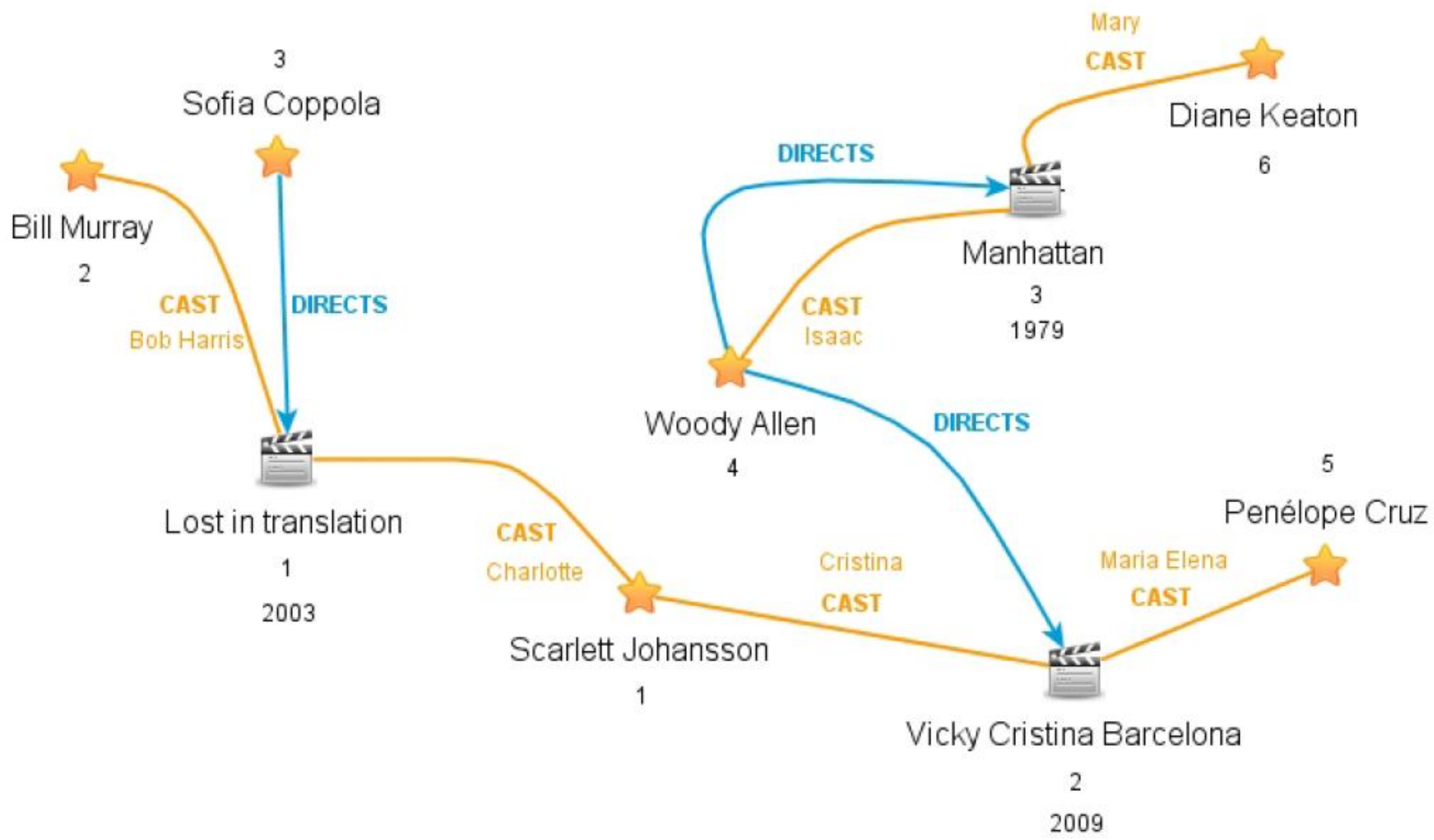
Hypergraph

$E \subseteq P(V)$



Модель данных атрибутированных графов

- *Атрибутированный граф* - атрибуты (свойства) приписываются ребрам и/или вершинам графа
  - Neo4j, Dex, InfiniteGraph, OrientDB, VertexDB, Filament, OQGraph, Horton, InfoGrid



ЯМД графовой модели

- API, скриптовые языки
  - доступ к структуре графа
  - методы обхода графа
  - алгоритмы на графах

- Декларативные языки (Cypher)
  - Поддержан Neo4j
  - Родство с SQL и SPARQL
  - Возможности
    - смежность вершин и ребер
    - достижимость по путям фиксированной длины
    - достижимость по простым регулярным путям
    - поиск кратчайших путей
    - поиск подграфов по образцу
    - …

Отображение ЯОД (I)

Графовая модель

VT(Cinema) = { people, movie }

ET(Cinema) = { cast, directs }

A(movie) = {

<id, long>,

<title, string>,

<year, integer>}

A(people) = {

<id, long>,

<name, string>}

cast = <{<character, string>}, false, false, undefined, undefined>

directs = < ∅, true, true, people, movie>

Каноническая объектная модель

{ Cinema; in: module;

{ vertices; in: class; … },

{ edges: in: class; … };

{ movie; in: class; superclass: vertices;

instance_type: {

id: long;

title: string;

year: integer; }; }

{ directs; in: class; superclass: edges;

instance_type: {

edgeConstr: {in: invariant;

{{ all e/directs.inst (directs(e) ->

people(e.startVertex) &

movie(e.endVertex)) }}

}; };

} }

Отображение ЯОД (II)

- *Схема* отображается в одноименный модуль, включающий классы, содержащие вершины и ребра графа
- *Тип вершины* представляется одноименным классом - подклассом класса всех вершин vertices
- *Тип ребра* представляется одноименным классом - подклассом класса всех вершин edges
- *Атрибуты* типов вершин и ребер представляются атрибутами типа экземпляров соответствующих классов
- Между встроенными типами графовой модели (int8, int64, double и т.д.) и встроенными типами объектной модели (short, long, double) устанавливается взаимно-однозначное соответствие
- …

Виртуальная и материализованная интеграция

- При *виртуальной* интеграции отображение ЯМД обеспечивает возможность трансляции программ на языке посредника в запросы на языке ресурсов
- При *материализованной* интеграции данные извлекаются из ресурса и представляются в хранилище в канонической модели. При этом программы на языке канонической модели исполняются непосредственно на данных.
  - Отображение ЯМД нужно лишь для того, чтобы убедиться, что отображение моделей сохраняет информацию и семантику операций.
  - Семантически правильное отображение служит базой для процесса Извлечения-Преобразования-Загрузки (ETL), формирующего из данных ресурса данные хранилища: ETL-процесс может быть выражен только в терминах канонической модели.

Отображение ЯМД

Каноническая объектная модель

- Datalog-подобный язык в объектной среде

q([colleague_name]) :-
people(scarlett/[name]),
movies(m),
people(colleague/
    [collegue_name: name]),
cast(c1),
cast(c2),
c1.isValidEdge(m, scarlett),
c2.isValidEdge(m, colleague),
scarlett.name =
  “Scarlett Johansson”,
colleague.name.like(“*Cruz*”).

Графовая модель

- Cypher

START scarlett =
  node:node_auto_index(
      name = 'Scarlett Johansson')
MATCH
  m-[c1:cast]-scarlett,
  m-[c2:cast]-collegue
WHERE
  collegue.name =~ /*Cruz*/
RETURN collegue.name

Вернуть имена актеров по фамилии Круз, игравших в фильмах вместе со Скарлетт Йохансон

Семантика объектной модели в AMN

```
REFINEMENT  ObjectDM
CONSTANTS
  c_edges, c_vertices,  a_startVertex, a_endVertex, c_edges_instance_type
ABSTRACT_VARIABLES
  m_directed, m_restricted, m_startVertexType, m_endVertexType, isValidEdge, ...
INVARIANT
  c_edges: classNames & c_vertices: classNames &
  isValidEdge: objectsOfClass(c_edges) * objectsOfClass(c_vertices) * objectsOfClass(c_vertices) --> BOOL &
   …
OPERATIONS
     deleteVertex(attr, cond) =
     PRE
        attr : dom(attributeNames) &  cond : INT --> BOOL & attributeType(attr) = Integer
     THEN
          objectsOfClass(c_vertices) :=
          objectsOfClass(c_vertices) -
          { vert | vert: objectsOfClass(c_vertices) & vert: dom(adtAttributeValue(attr)) &
                    cond(integerAttributeValue(attr)(vert)) = TRUE }
     END
END
```

Семантика графовой модели в AMN

```
REFINEMENT GraphDM
REFINES ObjectDM
ABSTRACT_VARIABLES
  vertexTypeIDs, edgeTypeIDs, attributeIDs, attributes, vertices, vertixType, edges, edgeType, ...
INVARIANT
  vertexTypeIDs: POW(NAT) & edgeTypeIDs: POW(NAT) & attributeIDs: POW(NAT) &
  attributes: vertexTypeIDs & vertices: POW(NAT) & vertixType: vertices --> vertexTypeIDs &
  edges: POW(NAT) & edgeType: edges --> edgeTypeIDs &
   ...
OPERATIONS
deleteVertex(attr, cond) =
   PRE
      attr: attributeIDs & cond: INT --> BOOL & attributeTyping(attr) = Integer
   THEN
      vertices := vertices -
      { vert | vert: vertices & attr: attributes(vertixType(vert)) &
               cond(g_integerAttributeValue(vert, attr)) = TRUE }
   END
END
```

Инвариант уточнения

- Связывает переменные уточняемой и уточняющей машин, добавляется в инвариант уточняющей машины

- Множество имен типов графовой модели совпадает с множеством имен классов объектной модели (за исключением предопределенных классов c_edges, c_vertices)

  ran(typeName) = classNames - {c_edges, c_vertices}

- Вершины и ребра графовой базы данных соответствуют объектам классов c_vertices и c_edges:

  vertices = objectsOfClass(c_vertices) & edges = objectsOfClass(c_edges)

- Значения атрибутов вершин и ребер графовой модели совпадают со значениями соответствующих атрибутов соответствующих объектов

```
!(vert, attr).(vert: vertices & attr: attributeIDs =>
 ((vert |-> attr) : dom(g_integerAttributeValue) =>
    g_integerAttributeValue(vert, attr) = integerAttributeValue(attr)(vert)) )
!(edg, attr).(edg: edges & attr: attributeIDs =>
 ((edg |-> attr) : dom(g_integerAttributeValue) =>
    g_integerAttributeValue(edg, attr) = integerAttributeValue(attr)(edg))   )
```

Доказательство уточнения

- Машины ObjectDM и GraphDM загружены в инструментальное средство доказательства уточнения Atelier B
- Автоматически сгенерированы теоремы уточнения
  - для операции deleteVertex – 15 теорем
- Теоремы доказываются автоматически или интерактивно
  - для операции deleteVertex – все теоремы доказаны автоматически
- Доказательство проводится для всех операций ЯМД

Литература

- Kalinichenko L.A., Stupnikov S.A., Martynov D.O. SYNTHESIS: a Language for Canonical Information Modeling and Mediator Definition for Problem Solving in Heterogeneous Information Resource Environments. Moscow: IPI RAN, 2007. - 171 p. - http://goo.gl/SknBs4
- Dominique Cansell, Dominique Mery. Foundations of the B Method. Computing and Informatics, Vol. 22, 2003, 1–31. - http://www.imm.dtu.dk/~dibj/cai/cai-b.pdf
- Atelier B, the industrial tool to efficiently deploy the B Method. - http://www.atelierb.eu/index-en.php
- R. Angles. A Comparison of Current Graph Database Models. Proc. IEEE 28th International Conference on Data Engineering Workshops (ICDEW), 2012. – P. 171-177.
- С. А. Ступников. Отображение графовой модели данных в каноническую объектно-фреймовую информационную модель при создании систем интеграции неоднородных информационных ресурсов // CEUR Workshop Proceedings, Vol. 934, P. 42-52. http://ceur-ws.org/Vol-934/