ГИБРИДНАЯ АДАПТИВНАЯ ТЕХНОЛОГИЯ ПРОЕКТИРОВАНИЯ БИЗНЕС-ПРОЦЕССОВ

А. Ю. Атисков, С. В. Перминов

Санкт-Петербургский институт информатики и автоматизации РАН

СПИИРАН, 14-я линия ВО, д. 39, Санкт-Петербург, 199178

<atiskov@mail.ru>

УДК 004.8

Атисков А. Ю., Перминов С. В. **Гибридная адаптивная технология проектирования бизнес-процессов** // Труды СПИИРАН. Вып. 4. — СПб.: Наука, 2007.

Аннотация. *Объектом исследования является развитие автоматизированной системы трансформации компонент диаграмм языка структурного моделирования в диаграммы классов на основе гибридной адаптивной технологии проектирования бизнес-процессов, включающей IDEF0, UML, OWL, RDF, XML, XSLT. Показана возможность определения правил трансформации с использованием онтологического описания.* — Библ. 9 назв.

UDC 004.8

Atiskov A. J., Perminov V. I. **Adaptive hybrid technology of business-process modeling** // SPIIRAS Proceedings. Issue 4. — SPb.: Nauka, 2007.

Abstract. *The paper explores development of automated system for transforming diagram components of structure modeling language into class diagrams using hybrid adaptive business-processes designing technology that includes IDEF0, UML, OWL, RDF, XML, XSLT. Possibility of defining transformation rules that use ontological description is shown.* — Bibl. 9 items.

1. Постановка задачи

В работах [1, 2, 3] показана необходимость и актуальность применения функционального проектирования при описании бизнес-процессов. Там же приводится обоснованность разработки методов трансформации в объектно-ориентированную технологию. Практика применения трансформационных технологий требует введения механизмов адаптации, так как состав правил трансформации нестабилен, объемен и не всегда формализован [4]. Мы предлагаем интегрировать использование методологии IDEF0 в процесс создания информационных систем за счет создания средства построения диаграмм классов UML (и соответственно кодогенерации) по диаграммам IDEF0. Более того, чтобы следовать современным методам создания новых технологий, наше средство будет основываться на объединении таких технологий: IDEF0, UML, OWL, RDF, XML, XSLT [5, 6, 7]. Схема взаимодействия технологий показана на рис. 1.

Диаграмма IDEF0 представляется в виде XML-файла, структура которого определяется онтологическим описанием метаданных диаграмм IDEF0. Онтологическое описание позволяет нам адаптировать способ перевода IDEF0-диаграмм в UML. Адаптивный блок состоит из словаря тэгов, онтологического представления правил трансформации и преобразователя из онтологического описания в формате OWL в XSLT-преобразование (которое трансформирует XML-представление IDEF0-диаграмм в XML-представление UML-диаграмм классов). При этом из XML-представления UML-диаграмм классов получается файл, который можно открыть в средствах проектирования UML (например, Borland Together).

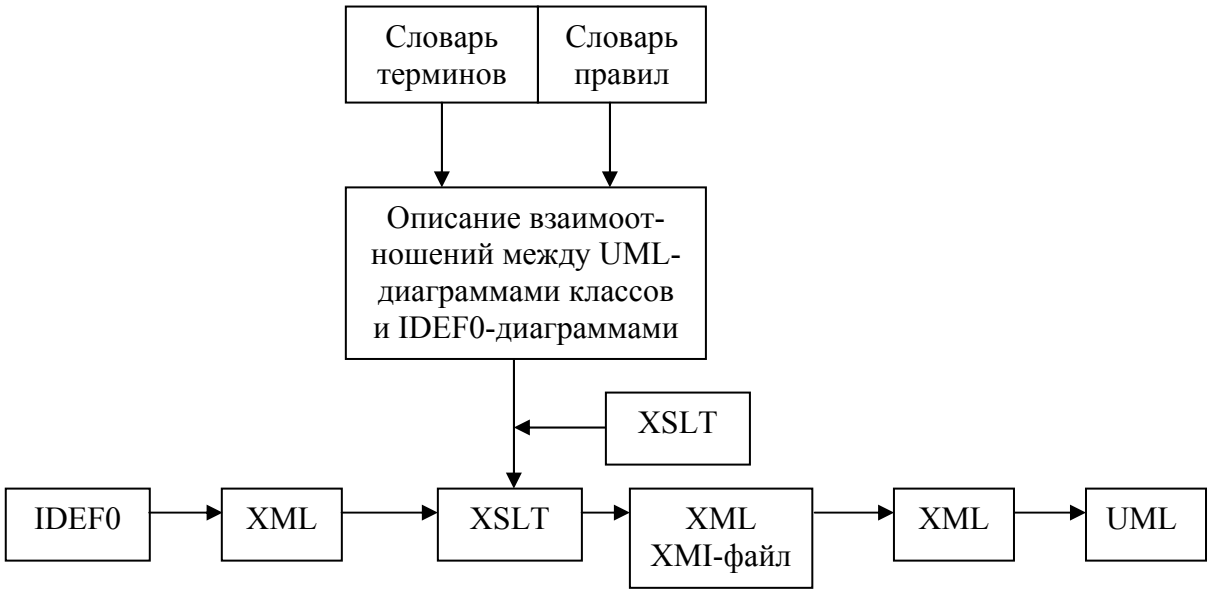

Рис. 1. Обобщенная схема разрабатываемой гибридной технологии.

Используем следующие примитивы:

1. `owl:Class` — класс, определяющий группу экземпляров, которые связаны друг с другом посредством разделения общих свойств. Классы организуются в специальную иерархию при помощи отношения `rdfs:subClassOf`;

2. `owl:Thing` — это класс, объединяющий в себе все экземпляры и являющийся суперклассом для всех других классов в OWL и соответственно описанной онтологии;

3. `rdfs:subClassOf` — иерархии классов, формирующиеся посредством определения одного или нескольких отношений между ними, которые показывают, что один из них является подклассом другого класса;

4. `rdf:Propery` — свойство. При помощи свойств определяются специальные отношения между экземплярами или от экземпляров к значениям данных. В первом случае используется `owl:ObjectProperty`, во втором — `owl:DatatypeProperty`;

5. `owl:Restriction` — определяет ограничение на использование свойства `rdf:Property` (при помощи отношения `owl:onProperty`) для экземпляра конкретного класса;

6. `owl:onProperty` — отношение, определяющее свойство `rdf:Property`, для которого установлено ограничение `owl:Restriction`;

7. `rdfs:domain` — отношение, связанное с `rdf:Property`, определяющее субъект (в логической тройке субъект–предикат–объект), а именно класс, для экземпляров которого только и может быть определено данное свойство;

8. `rdfs:range` — в отличие от `rdfs:domain` определяет объект в логической тройке;

9. `owl:cardinality` — вид ограничения. Показывает, как отношение связано с числом, описывающим точное количество элементов конкретного свойства `owl:Property`, которые должны быть определены для экземпляров конкретного класса `owl:Class`. В описанной онтологии ограничение `owl:cardinality` всегда связано с числом «1», это означает, что связанное с ним свойство должно быть определено один раз;

Труды СПИИРАН. 2007. Вып. 4. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 4. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

10. owl:minCardinality — вид ограничения. В отличие от owl:cardinality определяет минимально допустимое количество элементов конкретного свойства для конкретного класса.

Представляем метаданные для правил преобразования для XSLT-процессора в виде OWL-онтологий. Построение семантического описания структуры диаграмм IDEF0 основывается на двух типах элементов (блоки и стрелки) и отношениях между ними. Стрелки могут иметь один из четырех типов (вход, выход, управление и механизм), начинаться и заканчиваться либо на других блоках, либо на одной из сторон диаграммы (см. рис. 2).

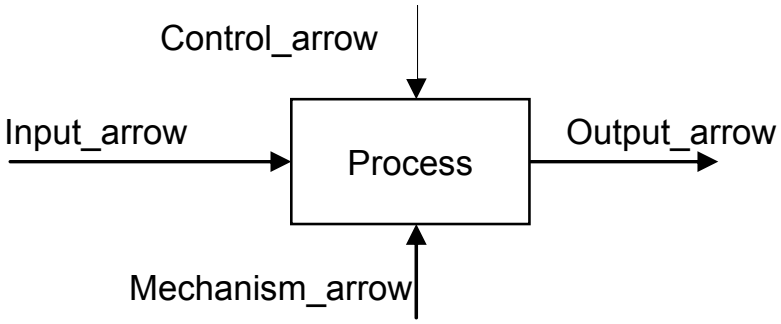

Рис. 2. Пример диаграммы IDEF0.

Блоки также могут быть декомпозированы (то есть один блок внутри себя может содержать множество других). А стрелки могут сливаться и разъединяться. Описание стандарта IDEF0 [8, 9] позволяет построить его семантическую интерпретацию в среде Protégé.

В процессе разработки онтологии, объединяющей функциональный подход (диаграммы IDEF0) и объектно-ориентированный (диаграммы классов UML), были сформулированы следующие понятия: классы owl:Class и свойства owl:Property, включая owl:ObjectProperty и owl:DatatypeProperty. Все понятия определены в пространстве имен (XML-namespace) xmlns:tf="http://www.spiiras.nw.ru/transform#".

tf:Functional — owl:Class, наследниками которого являются все классы, описывающие метаданные диаграмм IDEF0.

tf:ObjectOriented — в отличие от tf:Functional его наследниками являются классы, описывающие метаданные диаграмм классов UML.

К наследникам (связанным с родительским классом посредством использования предопределенного в OWL отношения owl:subClassOf) tf:Functional относятся:

1. tf:Arrow — класс, обозначающий «стрелку» в диаграммах IDEF0;

2. tf:Block — класс, обозначающий «блок» в диаграммах IDEF0;

3. tf:ArrowEnd — связь между конкретной «стрелкой» и «блоком». Связь имеет следующие разновидности: tf:Control, tf:Input, tf:Output, tf:Mechanism.

К наследникам tf:ObjectOriented относятся:

1. tf:Type — класс, представляющий условное понятие «тип». Имеет разновидности (подклассы) tf:Class и tf:PrimitiveType;

2. tf:Attribute — обозначает атрибут класса tf:Class;

3. tf:Method — обозначает метод класса tf:Class;

4. tf:Parameter — обозначает параметр метода tf:Method;

Труды СПИИРАН. 2007. Вып. 4. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 4. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

Для связи введенных нами ранее понятий были определены следующие отношения rdf:Property:

1. tf:atBlock — связывает tf:ArrowEnd и tf:Block. Определено ограничение owl:Restriction, согласно которому tf:ArrowEnd должен иметь ровно одно отношение tf:atBlock;

2. tf:hasArrowEnd — связывает tf:Arrow и tf:ArrowEnd. Определено ограничение, согласно которому tf:Arrow должен иметь не менее одного отношения (свойства) tf:hasArrowEnd;

3. tf:type — связывает tf:Attribute, а также tf:Method и tf:Parameter с tf:Type. Определено ограничение, согласно которому определенные выше субъекты логической тройки должны иметь ровно одно такое отношение;

4. tf:parameter — связывает tf:Method и tf:Parameter;

5. tf:attribute — связывает tf:Class и tf:Attribute;

6. tf:method — связывает tf:Class и tf:Method;

7. tf:links — связывает различные экземпляры tf:Class. Данное отношение имеет разновидности tf:composes, tf:inherits, tf:delegates, tf:associates.

Пример XML-файла для конкретной диаграммы IDEF0 (в соответствии с семантическим описанием):

```
<tf:Control rdf:ID="Control_3">
 <tf:atBlock>
  <tf:Block rdf:ID="Block_2">
   <tf:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >b2</tf:name>
  </tf:Block>
 </tf:atBlock>
</tf:Control>
<tf:Control rdf:ID="Control_2"><tf:atBlock rdf:resource="#Block_2"/>
</tf:Control>
<tf:Arrow rdf:ID="Arrow_2">
 <tf:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >a2</tf:name>
 <tf:hasArrowEnd rdf:resource="#Control_3"/>
</tf:Arrow>
<tf:Arrow rdf:ID="Arrow_1">
 <tf:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >a1</tf:name>
 <tf:hasArrowEnd>
  <tf:Control rdf:ID="Control_1">
   <tf:atBlock>
    <tf:Block rdf:ID="Block_1">
     <tf:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >b1</tf:name>
    </tf:Block>
   </tf:atBlock>
  </tf:Control>
 </tf:hasArrowEnd>
 <tf:hasArrowEnd rdf:resource="#Control_2"/>
 <tf:hasArrowEnd>
  <tf:Input rdf:ID="Input_1"><tf:atBlock rdf:resource="#Block_1"/>
  </tf:Input>
 </tf:hasArrowEnd>
 <tf:hasArrowEnd>
```


Труды СПИИРАН. 2007. Вып. 4. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 4. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

```
  <tf:Output rdf:ID="Output_1"><tf:atBlock rdf:resource="#Block_1"/>
  </tf:Output>
 </tf:hasArrowEnd>
</tf:Arrow>
```

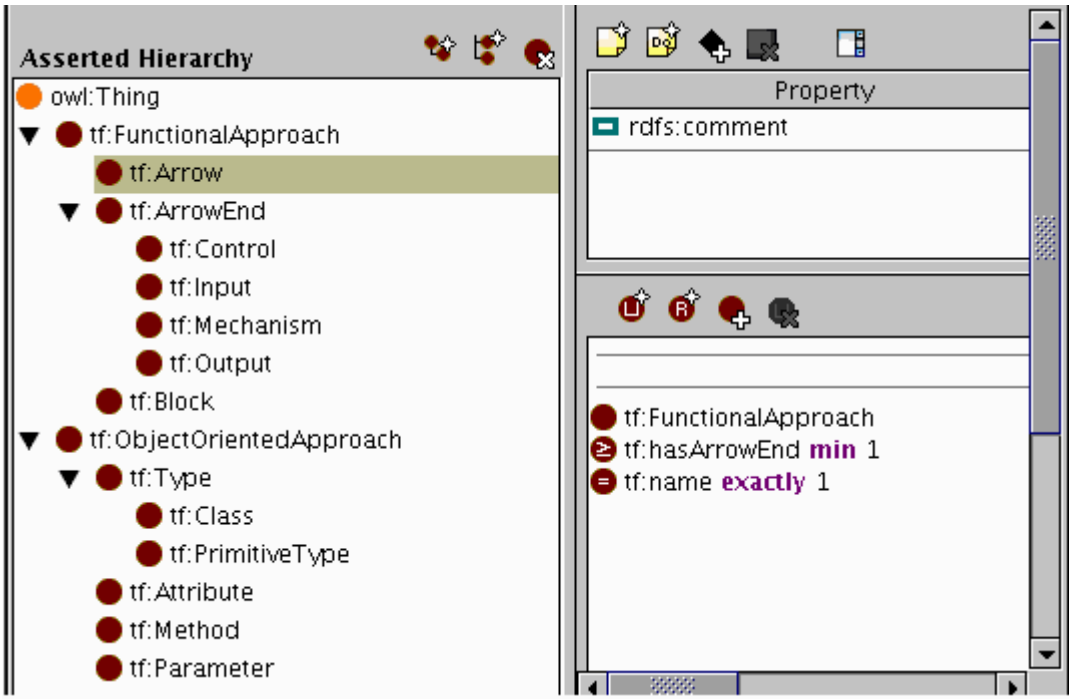

Рис. 3. Скриншот из среды Protégé построенного онтологического описания нотаций.

Окно группы «SUBCLASS EXPLORER» (рис. 3) определяет иерархию `owl:Class` (с корневым классом `owl:Thing`), описанную для онтологии, связывающей функциональный подход `tf:FunctionalApproach` (рис. 4) в диаграммах IDEF0 и объектно-ориентированный `tf:ObjectOrientedApproach` — в диаграммах классов UML (рис. 5). В нижнем окне показаны ограничения `owl:Restriction`, определенные для класса `tf:Arrow`.

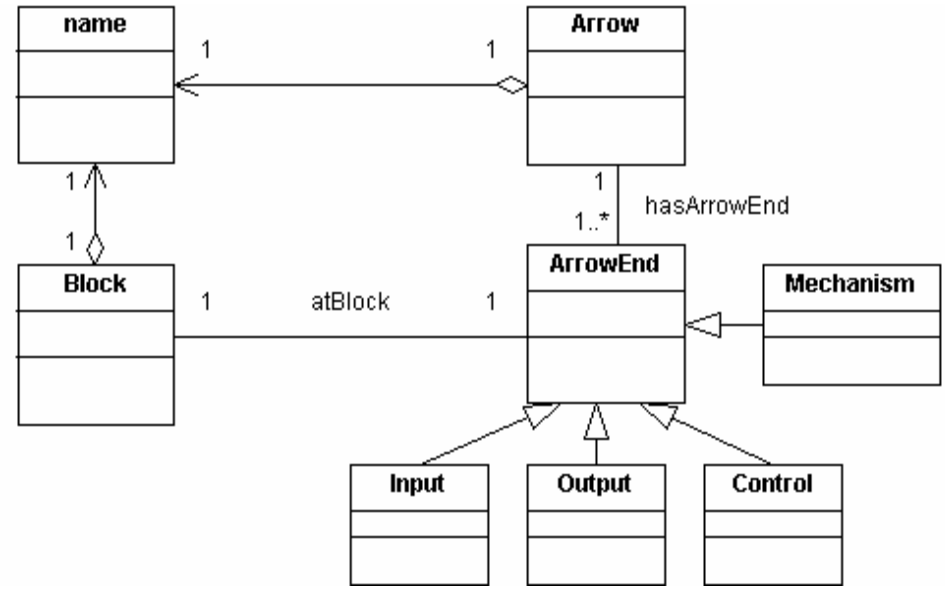

Рис. 4. Семантическое описание нотации IDEF0 в виде UML-диаграммы.

Труды СПИИРАН. 2007. Вып. 4. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 4. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

Построение онтологического описания диаграмм классов основывается на основных сущностях, которые используются при создании UML-диаграмм классов: класс, атрибут, метод, параметр метода, тип, связи (наследование, ассоциация, композиция, зависимость).

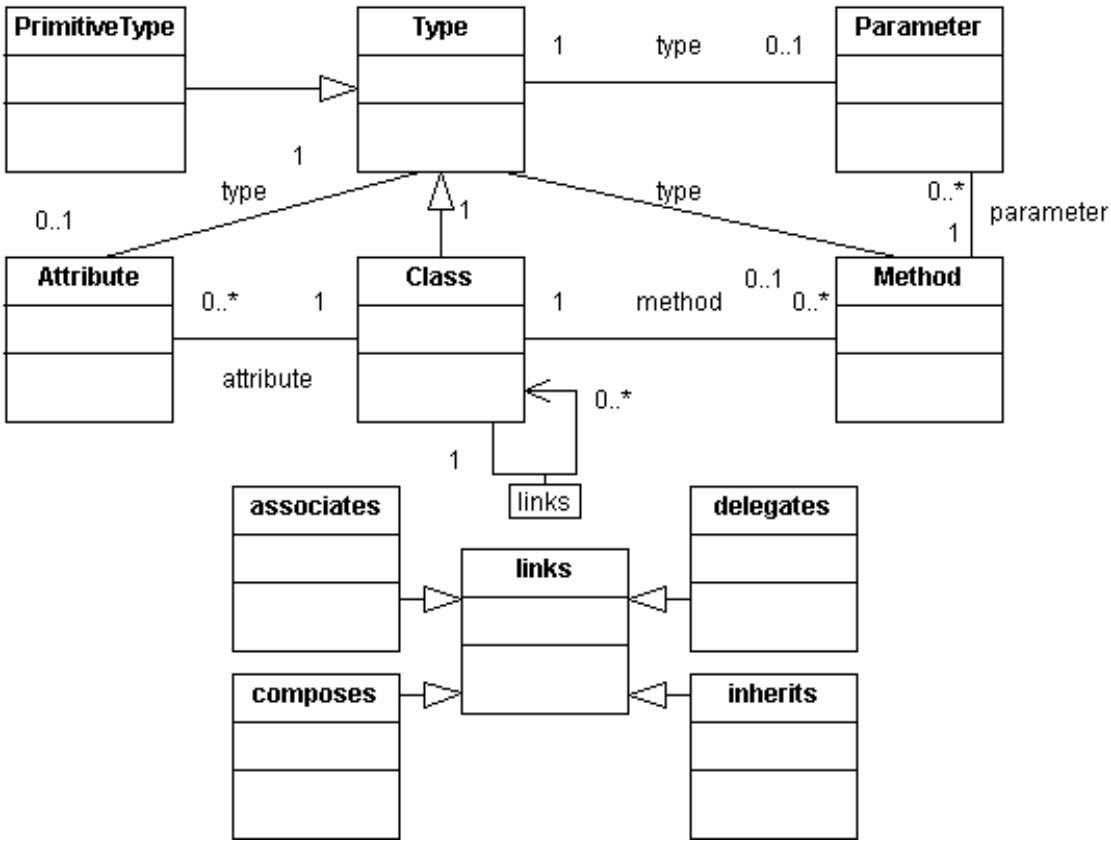

Рис. 5. Семантическое описание нотации UML в виде UML-диаграммы.

2. Описание правил трансформации

Для формального (в виде множества логических троек) описания правил в онтологию были введены дополнительные понятия и отношения, а также использован ряд предопределенных понятий и отношений из языка RDF (табл. 1).

Таблица 1

Использованные понятия и отношения из языка RDF

Выражение из RDF	**Описание выражения**
rdf:Statement	Обозначает условие (представленное в виде логической тройки) правила
rdf:parseType=«Collection»	Обозначает конечное объединение множества понятий
rdf:subject	Связывает условие rdf:Statement с субъектом, входящим в логическую тройку, которая описывает данное условие
rdf:predicate	Связывает rdf:Statement с предикатом
rdf:object	Связывает rdf:Statement с объектом

Труды СПИИРАН. 2007. Вып. 4. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 4. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

Введенные понятия:

1. `tf:Transform` — понятие, обозначающее правило;
2. `tf:Object` — обозначает переменную, примененную в правиле.

Введенные отношения (свойства):

1. `tf:functionalTemplate` — свойство правила `tf:Transform`, связывающее его с объединением множества условий `rdf:Statement`, которое описывает часть правила, относящуюся к диаграмме IDEF0. На менее формальном языке можно сказать, что данное свойство связывает правило с шаблоном (как множеством условий) из диаграммы IDEF0;
2. `tf:objectOrientedTemplate` — в отличие от `tf:functionalTemplate` связывает правило с частью, относящейся к диаграмме классов UML.

Определим несколько правил трансформации метаданных IDEF0 в метаданные UML. При этом необходимо заметить, что правила нельзя трактовать как устойчивые и неизменяемые. Другими словами, мы лишь определяем возможную совокупность правил, позволяющих нам реализовать преобразование конкретной диаграммы.

Правила представляются в виде взаимосвязи двух множеств — множества трансформируемых элементов из диаграммы бизнес-процессов и множества элементов из диаграммы классов. Но для непротиворечивого определения используем список логических троек (субъект–предикат–объект), причем этот список должен быть достаточно полным для однозначного определения нужного элемента при конкретной трансформации.

Пример правила преобразования: «все стрелки механизмов на диаграмме IDEF0 трансформируются в имена классов диаграммы классов UML». Вследствие ограниченных возможностей в ядре Protégé по описанию взаимосвязей между классами `owl:Class` используем более абстрактную форму описания, а именно опишем данное правило в виде множества логических троек. Такое представление можно интерпретировать как совокупность объектов, связи между которыми показаны на рис. 6.

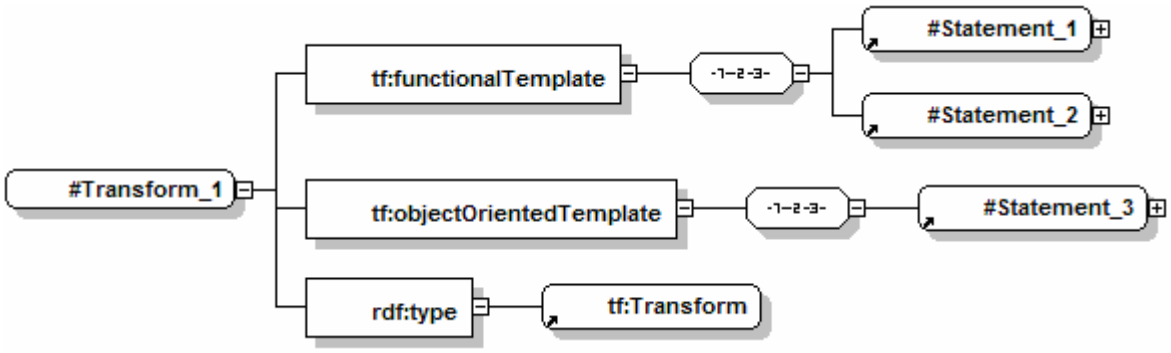

Рис. 6. Семантическая сеть примера правила трансформации (где Statement_1 раскрыт на рис. 7, Statement_2 — на рис. 8, а Statement_3 — на рис. 9).

Труды СПИИРАН. 2007. Вып. 4. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 4. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

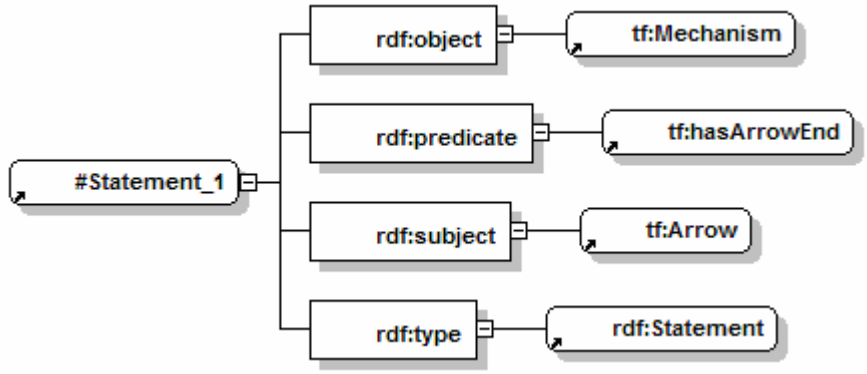

Рис. 7. Выражение для поиска стрелки механизма.

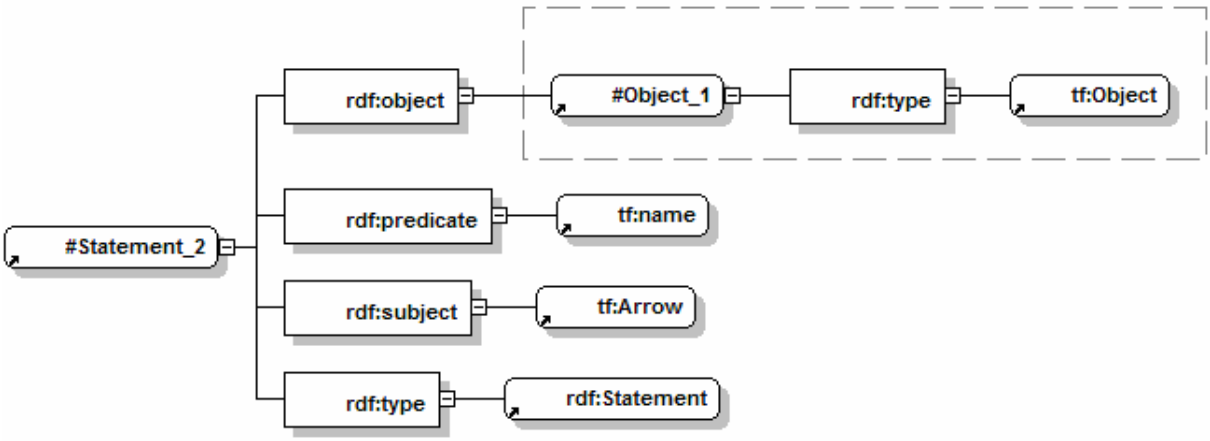

Рис. 8. Выражение для поиска стрелки с определенным именем.

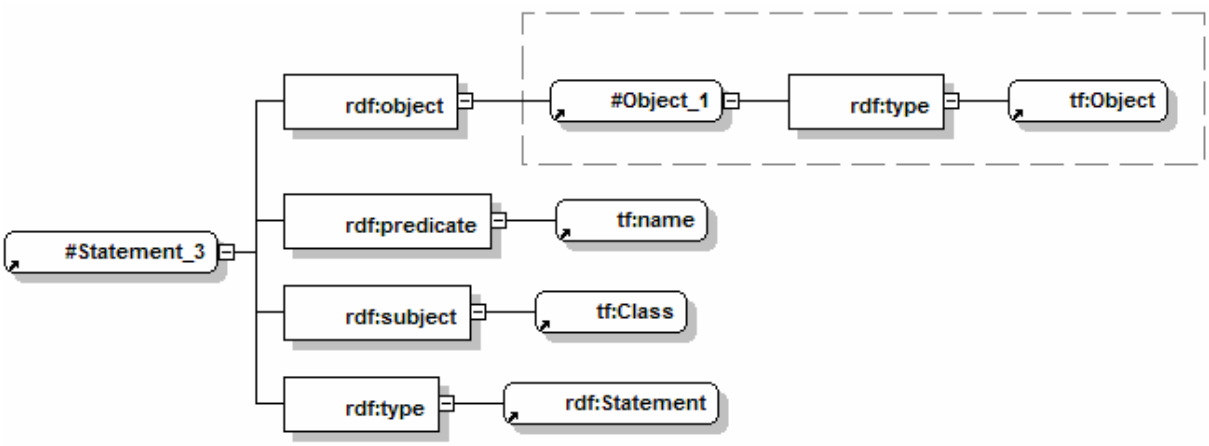

Рис. 9. Выражение для поиска класса с определенным именем.

3. Заключение

Предлагаемая технология, заключающаяся в комплексном использовании технологий IDEF0, UML, OWL, RDF, XML, XSLT позволяет решить следующие задачи:

1. Создать гибридную технологию проектирования программных систем от IDEF0-диаграмм до имплементации кода;

2. Обеспечить возможность адаптации технологии к меняющимся условиям применения (в частности, корректировать входящие и исходящие нотации,

Труды СПИИРАН. 2007. Вып. 4. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 4. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

введение новых и изменение существующих правил трансформации);

3. Реализовать программный комплекс для автоматизированной поддержки данной технологий, включающей инструментальную поддержку компонент гибридной технологии;

4. Создать условия для обмена RDF-описаниями правил между заинтересованными лицами и организациями в рамках инфраструктуры Semantic Web [5].

Основное направление дальнейшего развития технологии лежит в определении адаптируемых способов задания как правил трансформации, так и описаний других видов нотаций, между которыми происходит трансформация.

Литература

1. *Noran O.* UML vs IDEF: An ontology-oriented comparative study in view of business modelling [Электронный ресурс]: Paper accepted at the 6th International Conference on Enterprise Information Systems. 2003. 53 p. // <http: // www.cit.gu.edu.au/~noran> (по состоянию на 29.03.2006).
2. *Фаулер Мартин.* Архитектура корпоративных программных приложений. М. : Издательский дом «Вильямс», 2006. 544 с.
3. *Маклаков С. В.* Моделирование бизнес-процессов с AllFusion. М.: Диалог-Мифи, 2003. 224 с.
4. *Атисков А. Ю., Воробьев В. И.* Автоматизированная система трансформации диаграмм бизнес-процессов в диаграммы классов // Труды СПИИРАН. Вып. 3, т. 2., СПб.: Наука, 2006. 406 с.
5. *Berners-Lee T., Hendler J., Lassila O.* The Semantic Web. A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities // Scientific American. 2001. № 5. P. 34–43.
6. *Manola F., Miller E.* RDF Primer. W3C Recommendation 10 February 2004 [Электронный ресурс] // <http://www.w3.org/TR/rdf-primer/> (по состоянию на 21.02.2007).
7. *McGuinness D., Harmelen F.* OWL Web Ontology Overview. W3C Recommendation 10 February 2004 [Электронный ресурс] // <http://www.w3.org/TR/owl-features/> (по состоянию на 21.02.2007).
8. *Вендров А. А.* CASE-технологии. Современные методы и средства проектирования информационных систем. М.: Финансы и статистика, 1998. 176 с.
9. Методология функционального моделирования IDEF0. Госстандарт России. М.: ИПК Издательство стандартов, 2000.