**Серия «Математика»**
2012. Т. 5, № 4. С. 27—44

Онлайн-доступ к журналу:
http://isu.ru/izvestia

ИЗВЕСТИЯ
Иркутского
государственного
университета

УДК 510.62:004.82

# Объектные теории над списочными надстройками *

А.А. Малых, А.В. Манцивода
*Иркутский государственный университет*

**Аннотация.** В работе исследуются возможности использования методов семантического программирования, основанных на теории наследственно-конечных списочных надстроек (GES), для логического моделирования объектно-ориентированного подхода в программировании. На основе GES нами строится формальная система, аналогичная объектной дескриптивной логике $\mathcal{OODL}$, которая позволяет, в отличие от $\mathcal{OODL}$, естественным образом моделировать такие важные понятия программирования как упорядоченные структуры данных (например, списки и массивы). Формальная система, которая определяется и исследуется в данной работе, может служить для построения логических семантик языков программирования, в частности, объектно-ориентированного языка Libretto.

**Ключевые слова:** дескриптивная логика, объектная теория, тип данных, объектно-ориентированное программирование, семантическое программирование, язык Libretto

## 1. Введение

В работе [1] нами рассматривается логический подход к объектно-ориентированному моделированию. Определяется дескриптивная логика $\mathcal{OODL}$, которая служит логическим эквивалентом программистскому понятию объектно-ориентированных типов данных. На основе этой логики строится подход к построению логической семантики типов данных объектно-ориентированных языков программирования. Использование подобного логического формализма предоставляет большое количество возможностей. В частности, этот подход позволяет строго определять взаимодействие между объектно-ориентированным программным кодом и базами данных. Еще одной возможностью является

* Работа выполнена при финансовой поддержке Федеральной целевой программы "Научные и научно-педагогические кадры инновационной России"на 2009-2013 гг., государственные контракты 14.740.11.1001 от 23 мая 2011, 16.740.11.0137 от 01.09.2010

погружение типов данных языка программирования в структуры логических баз знаний, например, в рамках семантического веба. Это обеспечивает непосредственное использование информации из баз данных в процессе программирования. Имеются и другие интересные возможности. К сожалению, эксперименты с $\mathcal{OODL}$ показали, что дескриптивные логики не очень хорошо подходят для практического программирования. Например, в дескриптивных логиках существенно затруднено моделирование такого популярного инструмента, как упорядоченные коллекции элементов (массивы, списки и т.д.) Это резко снижает ценность дескриптивных логик как практического инструмента.

В работах [2, 3] был развит подход, который, в отличие от дескриптивных логик, использует упорядоченные структуры (списки) в качестве основных конструкторов данных. Это семантическое программирование, основанное на конструкции наследственно-конечных списочных надстроек. Использование списков вместо множеств позволяет получить более адекватный с точки зрения программирования формализм. На основе этого формализма можно строить довольно тонкие информационные модели, например, описывающие абстрактное понятие итератора.

В данной работе нами показано, что семантическое программирование может быть использовано для построения логических объектных моделей весьма естественным образом, сохраняя при этом преимущества теории GES, связанные с использованием упорядоченных структур данных. Результаты, излагаемые в работе, могут применяться как метод для описания семантики объектно-ориентированного языка Libretto [4], нацеленного на разработку приложений, работающих в распределенных информационных пространствах и включающего возможности для представления и обработки знаний в виде объектных моделей.

## 2. Объектные теории

Определим дескриптивную логику $\mathcal{OODL}$ [1]. Эта логика применяется для построения логических моделей хранилищ данных и знаний, представленных в объектно-ориентированном стиле. Язык $\mathcal{OODL}$ включает имена концептов $C_i$, имена ролей $R_i$, имена доменов $D_i$, имена объектов $O_i$, логические константы $\top$ и $\bot$ и квантор всеобщности $\forall T.F$, где $T$ – роль или атрибут, а $F$ – либо атомарный концепт, либо имя типа данных из $\mathcal{D}$. Кроме того, в $\mathcal{OODL}$ включаются

- инверсные роли при кванторах всеобщности: $\forall T^{-}.F$, где $T$ – либо роль, либо атрибут, а $F$ – либо атомарный концепт, либо имя типа данных;
- кардинальности $\geq n.T$ и $\leq n.T$, где $n \in \mathbb{N}_0$, а $T$ – либо роль, либо атрибут, либо инверсия роли/атрибута.

– имена объектов $o, o', o_1, o_2, \ldots$

**Определение 1.** *Выражениями логики $\mathcal{OODL}$ являются:*
*Концепты вида*

$$C, \quad \top, \quad \bot, \quad \forall T.F, \quad \forall T^{-}.F, \quad \geq n.T, \quad \leq n.T$$

*где $C$ – имя концепта, $F$ – произвольный концепт, $T$ – свойство (роль или атрибут).*
*Терминологические аксиомы вида $F_1 \sqsubseteq F_2$, где $F_1, F_2$ – концепты.*
*Фактологические аксиомы (факты) $C(o), R(o, o'), P(o, v)$, где $C, R, P$ – имена концепта, роли и атрибута, соответственно, $o, o'$ – имена объектов, $v$ – элемент типа данных.*

Определим интерпретацию $I$ конструкций $\mathcal{OODL}$. Выражения $\mathcal{OODL}$ интерпретируются на множестве объектов $\Delta$, причем имена объектов $o_i$ интерпретируются как элементы основного множества $o_i^I \in \Delta$, концепты – как подмножества $F^I \subseteq \Delta$, роли и атрибуты – как подмножества декартовых произведений $R^I \subseteq \Delta \times \Delta$ и $P^I \subseteq \Delta \times \mathcal{D}$, соответственно. Интерпретация $I$ выражений $\mathcal{OODL}$ на множестве объектов $\Delta$ должна удовлетворять правилам, представленным на рис. 1.

Обратим внимание на то, что с помощью кардинальности можно выразить неограниченный квантор существования $\exists$:

$$\exists R \rightleftharpoons \geq 1.R$$

Данный концепт выделяет все объекты, у которых существуют значения свойства $R$. Он эквивалентен выражению $\exists R.\top$ в более богатых логиках, чем $\mathcal{OODL}$, в частности, в логике $\mathcal{SHOIN}(D)$ [8].

Определим понятие *объектной теории* [1]. В качестве терминологических аксиом объектных теорий могут выступать только аксиомы определенных видов, а именно:

| | |
|---|---|
| $C_1 \sqsubseteq C_2$ | (аксиома наследования) |
| $\exists T \sqsubseteq C$ | (аксиома домена) |
| $\exists T^{-} \sqsubseteq C$ | (аксиома ранга) |
| $C \sqsubseteq \geq n.T$ | (аксиома min-кардинальности) |
| $C \sqsubseteq \leq n.T$ | (аксиома max-кардинальности) |

**Определение 2.** [1] *Объектной теорией $\mathbb{O}$ логики $\mathcal{OODL}$ в словаре $\mathcal{W} = \langle \mathcal{C}, \mathcal{R}, \mathcal{P}, \mathcal{O} \rangle$ назовем непротиворечивую теорию, $TBox$ которой состоит из конечного множества аксиом наследования, домена, ранга, max- и min-кардинальности, удовлетворяющую следующим условиям:*

*1) множество аксиом наследования в $\mathbb{O}$ ациклично;*

Семантика концептов:

| Концепт | Пример | Семантика |
|---|---|---|
| $C$ | $Woman$ | $C^I \subseteq \Delta$ |
| $\forall T.F$ | $\forall hasChild.Woman$ | $\{x \mid \{y \mid \langle x, y\rangle \in T^I\} \subseteq F^I\}$ |
| $\forall T^-.F$ | $\forall hasChild^-.Man$ | $\{x \mid \{y \mid \langle y, x\rangle \in T^I\} \subseteq F^I\}$ |
| $\geq n.T$ | $\geq 1.hasChild$ | $\{x \mid \ \|\{y \mid \langle x, y\rangle \in T^I\}\| \geq n\}$ |
| $\leq n.T$ | $\leq 2.hasChild$ | $\{x \mid \ \|\{y \mid \langle x, y\rangle \in T^I\}\| \leq n\}$ |
| $\top$ | $\top$ | $\top^I = \Delta$ |
| $\bot$ | $\forall hasChild.\bot$ | $\bot^I = \emptyset$ |

Семантика аксиом:

| Аксиома | Пример | Семантика |
|---|---|---|
| $F_1 \sqsubseteq F_2$ | $Girl \sqsubseteq Female$ | $F_1^I \subseteq F_2^I$ |
| $C(o)$ | $Woman(Marie)$ | $o^I \in C^I$ |
| $R(o, o')$ | $hasChild(Marie, John)$ | $\langle o^I, o'^I\rangle \in R^I, R^I \subseteq \Delta \times \Delta$ |
| $P(o, v)$ | $Age(Marie, 25)$ | $\langle o^I, v^I\rangle \in P^I, P^I \subseteq \Delta \times \mathcal{D}$ |

*Рис. 1.* Семантика конструкций логики $\mathcal{OODL}$

*2) каждое свойство $T \in (\mathcal{R} \cup \mathcal{P})$ обладает в $\mathbb{O}$ ровно одной аксиомой домена и ровно одной аксиомой ранга.*

Объектные теории развивают идею ОО-проекций [14] как логических эквивалентов понятия объектной (объектно-ориентированной) модели в информационных системах и языках программирования. В частности, именованные концепты моделируют классы, элементы множества $\mathcal{O}$ – объекты, поля которых представляются свойствами (ролями и атрибутами). Аксиомы наследования моделируют механизмы наследования в объектных моделях, аксиомы домена и ранга привязывают свойства к классам и определяют их тип, аксиомы кардинальности регулируют количество (в частности, позволяя, хотя и грубо, моделировать такие понятия, как массивы и списки). В [1] представлено подробное обсуждение этих вопросов. Там же показано, что для любой ОО-проекции можно построить эквивалентную ей объектную теорию.

## 3. Объектные теории в логике первого порядка

Как уже отмечалось выше дескриптивные логики обладают рядом особенностей, которые при решении многих задач превращаются в серьезные недостатки, наиболее ярким примером которых является отсутствие порядка на значениях свойств. Этот и следующие пункты работы посвящены переводу объектных теорий, определенных выше, в эквивалентные конструкции теории списочных надстроек [2] – логической системы, где возникающие в дескриптивных логиках проблемы решаются легко и естественно. Наш план состоит в переводе объектных теорий на язык логики первого порядка с последующим моделированием совокупностей значений свойств объектов элементами списочной алгебры. Сначала будет построен «наивный» вариант на основе плоских (линейных) списков, а затем более реальная версия – на основе итераторов, семантика которых может быть описана на языке GES.

Определим

$$\exists_{\leq n} x.F(x,\bar{y}) \rightleftharpoons \forall x_1 \ldots \forall x_n \forall x_{n+1}.(\bigwedge_{i=1}^{n+1} F(x_i,\bar{y}) \rightarrow \bigvee_{1\leq i<j\leq n+1} x_i = x_j)$$

$$\exists_{\geq n} x.F(x,\bar{y}) \rightleftharpoons \exists x_1 \ldots \exists x_n.(\bigwedge_{i=1}^{n} F(x_i,\bar{y}) \ \wedge \bigwedge_{1\leq i<j\leq n} x_i \neq x_j)$$

Основные аксиомы объектных теорий имеют следующие эквиваленты в логике первого порядка:

| Аксиома | ДЛ | ЛПП |
|---|---|---|
| наследование | $C_1 \sqsubseteq C_2$ | $\forall x.(C_1(x) \rightarrow C_2(x))$ |
| домен | $\exists T \sqsubseteq C_d$ | $\forall x \forall y.(T(x,y) \rightarrow C_d(x))$ |
| ранг | $\exists T^- \sqsubseteq C_r$ | $\forall x \forall y.(T(x,y) \rightarrow C_r(y))$ |
| min-кардинальность | $C_d \sqsubseteq \geq n.T$ | $\forall x.(C_d(x) \rightarrow \exists_{\geq n} y.T(x,y))$ |
| max-кардинальность | $C_d \sqsubseteq \leq n.T$ | $\forall x.(C_d(x) \rightarrow \exists_{\leq n} y.T(x,y))$ |

В данном подходе совокупность объектов, удовлетворяющих определенному свойству, формируется опосредованно – через совокупность фактов. Например, можно определить структуру книги

```
book(Ершов_Теория_Нумераций)
chapter(Выч_нумерации)        chapter(Категория_нум_множ)
chapter(m-сводимость)         chapter(Выч_функционалы)
has_chapter(Ершов_Теория_Нумераций, Выч_нумерации)
has_chapter(Ершов_Теория_Нумераций, Категория_нум_множ)   (NT)
```

```
has_chapter(Ершов_Теория_Нумераций, m-сводимость)
has_chapter(Ершов_Теория_Нумераций, Выч_функционалы)
```

Определение (NT) неявно формирует множество глав книги

```
{Выч_нумерации, Категория_нум_множ,
            m-сводимость, Выч_функционалы}
```

Очевидным недостатком данного определения является то, что главы книги в данной формализации явно не упорядочены, что не соответствует сути предметной области – главы в книге всегда находятся в определенной последовательности. В информационных задачах такие упорядочения возникают сплошь и рядом, поэтому отсутствие адекватных логических инструментов является очень существенным недостатком формализмов. Конечно, в тексте определения, представленного выше, порядок существует, но он является внелогическим, поскольку математически аксиомы составляют неупорядоченное множество. Отсутствие возможности упорядочения в базовых механизмах приводит к необходимости использовать для этих целей внелогические конструкции. Например, в языке пролог порядок вхождения фактов в программу объявляется существенным с точки зрения фиксированной стратегии. В реляционных базах данных порядок вводится явно, например, через использование натурального ряда. Такой подход соответствует следующему варианту определения глав:

```
has_chapter(Ершов_Теория_Нумераций, Выч_нумерации, 1)
has_chapter(Ершов_Теория_Нумераций, Категория_нум_множ, 2)
has_chapter(Ершов_Теория_Нумераций, m-сводимость, 3)
has_chapter(Ершов_Теория_Нумераций, Выч_функционалы, 4)
```

Очевидная ущербность такого решения состоит в том, что, во-первых, оно опирается не на общую логическую семантику, а на конкретную интерпретацию конкретного предиката, во-вторых, что порядок строится опосредовано – через имеющийся порядок в натуральных числах, и, в-третьих, что не имеется общего подхода к упорядочиванию – в каждой модели приходится придумывать самодеятельные инструменты.

Отметим, что для дескриптивных логик подобное решение вообще невозможно, поскольку превращает роли и атрибуты из двуместных в трехместные, что формально невозможно и ломает всю структуру дескриптивной логики. Данный краткий анализ, очевидно, приводит к следующему выводу: упорядочения важны в информационных моделях, и поэтому должны включаться в ядро логик, на базе которых информационные модели строятся.

## 4. Теория GES

Задача построения такого формализма решена в работах [2][3]. В них развита теория списочных надстроек (теория GES), которая основана на идее построения теории моделей для одного из классических типов данных – списка. Поскольку объекты в списках упорядочены, то их использование позволяет явно определять порядок на элементах. В отдельной работе, находящейся сейчас в печати, на основе списочных надстроек нами была описана теоретико-модельная семантика еще одного классического типа данных – итераторов. Именно итераторы как списки специального вида будут использоваться нами для реорганизации объектных теорий, позволяющих включить в них упорядоченные системы объектов.

Модели теории GES двусортные, состоящие из базового множества $S$ и списочной надстройки над этим множеством $I_S$. Сорта базового множества и списочной надстройки будем обозначать $\sigma$ и $\iota$, соответственно. Через $\iota\sigma$ будем обозначать сорт всех элементов. Выражение $e : \tau$ означает, что объект $e$ принадлежит сорту $\tau$. С помощью $\mathbf{r}$, возможно с индексами, будем обозначать списки (то есть, $\mathbf{r} : \iota$). Элементы сорта $\sigma$ обозачаем через $v$. Произвольные элементы (сорта $\iota\sigma$) обозначаются с помощью $e$. Сигнатура GES: $\{\mathtt{nil}, \mathtt{cons}, \mathtt{head}, \mathtt{tail}, \in, \subseteq, S\}$ (все элементы $S$ считаются выделенными), причем сорта данных функциональных и предикатных символов определяются следующим образом:

| | |
|---|---|
| $\mathtt{nil} : \iota$ | (пустой список) |
| $\mathtt{cons} : \iota\sigma \times \iota \to \iota$ | (операция добавления элемента к списку) |
| $\mathtt{head} : \iota \to \iota\sigma$ | (взятие головы списка) |
| $\mathtt{tail} : \iota \to \iota$ | (взятие хвоста списка) |
| $\in : \iota\sigma \times \iota$ | (принадлежность элемента списку) |
| $\subseteq : \iota \times \iota$ | (подсписок) |

Стандартно обозначим $x \neq y \rightleftharpoons \neg\, x = y$. Сформулируем аксиомы теории GES, которые выполняются для любых $\mathbf{r}, \mathbf{r}', \mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3 : \iota$, $e, e' : \iota\sigma$ и любой формулы $F$.

AE: Аксиомы пустого списка $\quad \neg\, e \in \mathtt{nil}, \quad \mathtt{nil} \subseteq \mathbf{r}$

AU: Аксиомы единственности $\quad \mathtt{cons}(e, \mathbf{r}) = \mathtt{cons}(e', \mathbf{r}') \Rightarrow \mathbf{r} = \mathbf{r}' \wedge e = e'$

AO: Аксиомы списочных операций:

| | |
|---|---|
| $\mathtt{head}(\mathtt{nil}) = \mathtt{nil}$ | $\mathtt{tail}(\mathtt{nil}) = \mathtt{nil}$ |
| $\mathtt{head}(\mathtt{cons}(e, \mathbf{r})) = e$ | $\mathtt{tail}(\mathtt{cons}(e, \mathbf{r})) = \mathbf{r}$ |
| $\mathbf{r} \neq \mathtt{nil} \to \mathtt{cons}(\mathtt{head}(\mathbf{r}), \mathtt{tail}(\mathbf{r})) = \mathbf{r}$ | $e \in \mathbf{r} \ \wedge \ \mathbf{r} \subseteq \mathbf{r}' \to e \in \mathbf{r}'$ |

AN: Аксиомы равнообъемности:

$$\mathbf{r}_1 = \mathbf{r}_2 \Leftrightarrow (\forall \mathbf{r}_3 \subseteq \mathbf{r}_1)(\mathbf{r}_3 \subseteq \mathbf{r}_2 \ \wedge (\mathbf{r}_3 \neq \mathbf{r}_2 \vee \mathbf{r}_3 \neq \mathbf{r}_1) \rightarrow \\ \rightarrow (\exists e \in \mathbf{r}_1)(\mathtt{cons}(e, \mathbf{r}_3) \subseteq \mathbf{r}_1 \wedge \mathtt{cons}(e, \mathbf{r}_3) \subseteq \mathbf{r}_2)$$

AN: Аксиомы для $\in$: $\quad e \in \mathtt{cons}(e', \mathbf{r}) \Leftrightarrow (e = e' \vee e \in \mathbf{r})$
AS: Аксиомы для $\subseteq$: $\quad \mathbf{r} \subseteq \mathtt{cons}(e, \mathbf{r}') \Leftrightarrow (\mathbf{r} \subseteq \mathbf{r}' \vee \mathbf{r} = \mathtt{cons}(e, \mathbf{r}'))$
AI: Аксиома $\Sigma$–индукции $\quad (F\,|\,^x_{\mathtt{nil}} \ \wedge \ \forall \mathbf{r}.\forall e.(F\,|\,^x_{\mathbf{r}} \ \Rightarrow \ F|^x_{\mathtt{cons}(e,\mathbf{r})})) \ \Rightarrow \ \forall \mathbf{r}.F\,|\,^x_{\mathbf{r}}$
AF: Аксиома $\Sigma$-фундируемости $\quad \forall \mathbf{r}.(\forall e \in \mathbf{r}.F\,|\,^x_e \Rightarrow F\,|\,^x_{\mathbf{r}}) \Rightarrow \forall e.F\,|\,^x_e$

**Определение 3.** *Пусть $S$ и $I$ – множества, $S \cap I = \emptyset$, на которых определена константы* $\mathtt{nil} \in I$, *а также операции* $\mathtt{cons}, \mathtt{head}, \mathtt{tail}, \in$ *и* $\subseteq$ *таким образом, что аксиомы* AE–AF *выполняются. Тогда $I$ называется списочной надстройкой над $S$, а двусортная модель*

$$\mathcal{L} = \langle S, I; \mathtt{nil}, \mathtt{cons}, \mathtt{head}, \mathtt{tail}, \in, \subseteq \rangle,$$

*называется* списочной алгеброй.

**Определение 4.** *Пусть $S$ – множество сорта $\sigma$. Тогда* стандартной списочной надстройкой $I_S$ *(сорта $\iota$) над $S$ назовем минимальное множество, удовлетворяющее следующим условиям:*

- *пустой список* $\mathtt{nil} \in I_S$;
- *если* $\mathbf{r} \in I_S$ *и* $e \in S \cup I_S$, *то* $\mathtt{cons}(e, \mathbf{r}) \in I_S$.

Стандартной списочной алгеброй *назовем списочную алгебру над множеством $S$, в качестве списочной надстройки которой выступает стандартная списочная надстройка.*

В дальнейшем будем использовать для списков стандартное скобочное представление:

$$\mathtt{cons}(e_1, \mathtt{cons}(e_2, \ldots \mathtt{cons}(e_m, \mathtt{nil}) \ldots) = (e_1, \ldots, e_m)$$

например, $\mathtt{cons}(\mathtt{cons}(1, \mathtt{cons}(2, \mathtt{nil})), \mathtt{cons}(3, \mathtt{nil})) = ((1, 2), 3)$.

Еще одна особенность GES, почерпнутая из теории наследственно конечных множеств [5], состоит в том, что если в теории GES индуктивно (рекурсивно) определяется некоторая функция $f$ (с помощью $\Sigma$-формул), то можно построить такую $\Sigma$-формулу, которая определяет $f$ без индукции. Это верно как для индукции по оператору cons (теорема 1 в [9]), так и по индукции в глубину списка (переход от подсписков к списку) [11]. Данная особенность делает GES очень мощным инструментом для описания, например, абстрактных рекурсивных типов данных, поскольку семантика неподвижной точки оказывается в GES формульно выразимой, а рекурсивные определения – консервативным расширением базового формализма. В частности, это касается операции конкатенации двух списков $\mathtt{conc} : \iota \times \iota \rightarrow \iota$:

AC: Аксиома, определяющая `conc`:

$$\begin{aligned}\texttt{conc}(\mathbf{r}_1, \mathbf{r}_2) = \mathbf{r}_3 \Leftrightarrow\\ (\mathbf{r}_1 = \texttt{nil} \to \mathbf{r}_2 = \mathbf{r}_3) \wedge\\ (\mathbf{r}_1 = \texttt{cons}(e, \mathbf{r}'_1) \to \mathbf{r}_3 = \texttt{cons}(e, \texttt{conc}(\mathbf{r}'_1, \mathbf{r}_2)))\end{aligned}$$

Также легко доказывается, что `conc` является ассоциативной операцией, а алгебра $\langle I_S; \texttt{nil}, \texttt{conc}\rangle$ является моноидом с единицей `nil`.

В заключение пункта введем два полезных логических сокращения:

$$\underline{\texttt{list}}(x) \rightleftharpoons x = \texttt{nil} \ \vee \ \exists e \exists \mathbf{r}.\ x = \texttt{cons}(e, \mathbf{r})$$
$$\underline{\texttt{if}}\ P\ \underline{\texttt{then}}\ Q\ \underline{\texttt{else}}\ R \rightleftharpoons (P \to Q) \wedge (\neg P \to R)$$

Первое сокращение показывает, что понятие списка выразимо в GES. Второе позволяет представлять в читабельном виде формулы с условными выражениями.

## 5. «Наивный» вариант: линейные списки

Используя теорию GES, мы можем реализовать основную идею данной работы, связанную с явным формированием совокупности значений свойств и упорядочиванием значения внутри этих совокупностей. Это можно сделать заменой неявно определенных множеств значений свойств на явные списки с использованием ограниченной квантификации по спискам. В частности, описание (NT) может быть перестроено. Для этого определим тип предиката `has_chapter` как $\sigma \times \iota$ и заменим факты для `has_chapter` на

```
has_chapter(Ершов_Теория_Нумераций, (Выч_нумерации,
        Категория_нум_множ, m-сводимость, Выч_функционалы) )
```

Для представления совокупностей значений достаточно использовать линейные списки, т. е. те списки, элементы которых являются праэлементами из базового множества $S$. Свойство линейности списка выразимо в GES:

$$\underline{\texttt{linear}}(\mathbf{r}) \rightleftharpoons \underline{\texttt{list}}(\mathbf{r}) \ \wedge \ \forall e.(e \in \mathbf{r} \to \neg \underline{\texttt{list}}(e))$$

Легко показать, что операции `tail` и `conc`, возвращающие в качестве результата списки, замкнуты относительно множества линейных списков. Теория линейных списков в контексте списочных надстроек подробно рассматривается в [10].

Определим объектные теории в списочной алгебре. Введем следующие определения:

| Аксиома | $\mathcal{OODL}$ | Списочные надстройки |
|---|---|---|
| наследование | $C_1 \sqsubseteq C_2$ | $\forall x.(C_1(x) \to C_2(x))$ |
| домен | $\exists T \sqsubseteq C_D$ | $\forall x \forall \mathbf{r}.(T(x,\mathbf{r}) \to C_D(x))$ |
| ранг | $\exists T^- \sqsubseteq C_R$ | $\forall x \forall \mathbf{r}.(T(x,\mathbf{r}) \to C_R^\star(\mathbf{r}))$ |
| min-кардинальность | $C \sqsubseteq \geq n.T$ | $\forall x \forall \mathbf{r}.(C(x) \wedge T(x,\mathbf{r}) \to \mathtt{min}_n(\mathbf{r}))$ |
| max-кардинальность | $C \sqsubseteq \leq n.T$ | $\forall x \forall \mathbf{r}.(C(x) \wedge T(x,\mathbf{r}) \to \mathtt{max}_n(\mathbf{r}))$ |

*Рис. 2.* Выразимость аксиом объектных теорий формулами ЛПП

$$C^\star(\mathbf{r}) \rightleftharpoons \forall x.(x \in \mathbf{r} \to C(x))$$
$$\mathtt{max}_n(\mathbf{r}) \rightleftharpoons \underbrace{\mathtt{tail}(\ldots\mathtt{tail}}_{n}(\mathbf{r})\ldots) = \mathtt{nil}$$
$$\mathtt{min}_n(\mathbf{r}) \rightleftharpoons \underbrace{\mathtt{tail}(\ldots\mathtt{tail}}_{n-1}(\mathbf{r})\ldots) \neq \mathtt{nil}$$

Отметим, что в последних двух определениях термы слева от равенства и неравенства могут содержать несколько вхождений одного и того же элемента. Это соответствует семантике не множеств, а мультимножеств, и значительно ближе к практике, поскольку и списки, и массивы могут содержать множественные вхождения одного и того же значения.

Основные аксиомы объектных теорий на языке списочной надстройки выглядят так, как это представлено на рис. 2. Пусть $\mathbb{O}$ – объектная теория в OODL. Обозначим через $\hat{\mathbb{O}}$ – теорию на языке GES, полученную следующим образом:

1) Включим в сигнатуру $\hat{\mathbb{O}}$ сигнатуру теории GES.

2) Каждый символ объекта из $\mathbb{O}$ включим как константу в сигнатуру $\hat{\mathbb{O}}$.

3) Для каждого символа концепта $C$ из $\mathbb{O}$ включим в сигнатуру $\hat{\mathbb{O}}$ символ одноместного предиката $C$, имеющего тип $\sigma$.

4) Для каждой роли и атрибута $T$ из $\mathbb{O}$ включим в сигнатуру $\hat{\mathbb{O}}$ символ двуместного предиката $T$, имеющего тип $\sigma \times \iota$.

5) Для каждой объектной аксиомы $A \in \mathbb{O}$ включим в $\hat{\mathbb{O}}$ аксиому $\hat{A}$, полученную из $A$ по соответствующему правилу из рис. 2.

**Теорема 1.** *$\mathcal{OODL} \vdash F$ тогда и только тогда, когда $GES + \hat{\mathbb{O}} \vdash \hat{F}$.*

*Доказательство.* Утверждение доказывается рутинной проверкой эквивалентности теоретико-множественной семантики аксиомы $\mathcal{OODL}$ и семантики соответствующего варианта из логики первого порядка. $\mathcal{OODL}$, очевидно, является разрешимой логикой (с табличным алгоритмом в качестве разрешающей процедуры [12]). Упорядочение значений свойств в виде линейных списков задает стратегию доказательства (последовательность перебора элементов) при проверке выводимости формулы из объектной теории.

## 6. «Реальный» вариант: итераторы

Семантика, основанная на линейных списках, прозрачна и понятна для понимания. К сожалению, эта семантика не соответствует механизмам, действующим в реальной работе, например, в программах на языке Libretto. Реальная модель работы списков в рамках вычислений значительно сложнее элементарной схемы, основанной на линейных списках. Дело в том, что в реальной обстановке в дело вмешиваются следующие процедурные особенности исполнения программ: ленивые вычисления, возможность параллельной обработки и возможность совместного доступа к объектам. В процессе последовательной обработки элементов списка, сам этот список может меняться. Это резко усложняет процедурную семантику, делая линейные списки неадекватным методом построения соответствующих моделей. Значительно более точным инструментом для построения этих моделей являются объекты абстрактного типа данных «итератор», семантика которых как списков, обладающих специальным поведением, хорошо описывается средствами теории GES (этому посвящена отдельная работа авторов, находящаяся сейчас в печати).

Вместо линейных списков используются элементы итераторной алгебры

$$\hat{\mathcal{L}}_M = \langle M_\Diamond, It_M; \mathtt{nil}, \Diamond, \mathtt{get}, \mathtt{next}, \mathtt{hasNext}, \mathtt{abv}, \mathtt{abn}\rangle \qquad (6.1)$$

которая строится на базе списочной алгебры введением выделенного элемента $\Diamond$, обозначающего «останов» итератора и определение функций $\mathtt{get}, \mathtt{next}, \mathtt{abv}, \mathtt{abn}$ и отношения $\mathtt{hasNext}$, которые мы называем итераторными функциями и отношением:

$\mathtt{get}(\mathbf{r})$ – функция, получающая текущий элемент итератора $\mathbf{r}$;

$\mathtt{next}(\mathbf{r})$ – функция перехода к следующему элементу $\mathbf{r}$;

$\mathtt{hasNext}(\mathbf{r})$ – истинно, если можно у $\mathbf{r}$ получить значение;

$\mathtt{abn}(\mathbf{r}, \mathbf{r}')$ – (`add-by-name`) добавление к $\mathbf{r}$ итератора $\mathbf{r}'$ по имени;

$\mathtt{abv}(\mathbf{r}, \mathbf{r}')$ – (`add-by-value`) добавление $\mathbf{r}'$ к $\mathbf{r}$ по значению.

В отличие от стандартных функций `head` и `tail` функции `get` и `next` работают с элементами базового множества праэлементов, например,

$$\texttt{get}((((1,2),3),4)) = 1 \qquad \texttt{head}((((1,2),3),4)) = ((1,2),3)$$
$$\texttt{next}((((1,2),3),4)) = (((2),3),4) \qquad \texttt{tail}((((1,2),3),4)) = (4)$$

и кроме того, специальным образом обрабатывают символ останова $\Diamond$. Функции `abn` и `abv` добавляют новые элементы к итератору. Все эти функции и отношение индуктивно определимы в GES, а значит в силу $\Sigma$-определимости представляют собой консервативное расширение базового формализма. Например, функция `get` определяется следующим образом:

**IG** Аксиома для `get`:

$$\begin{array}{l}\texttt{get}(\texttt{cons}(e,\mathbf{r})) = e' \Leftrightarrow \\ \quad \underline{\texttt{if}}\ \ \underline{\texttt{list}}(e)\ \underline{\texttt{then}}\ \ ( \\ \qquad \underline{\texttt{if}}\ \ (\Diamond = \texttt{get}(e))\ \ \underline{\texttt{then}}\ e' = \texttt{get}(\mathbf{r}) \\ \qquad \underline{\texttt{else}}\ e' = \texttt{get}(e)\ ) \\ \quad \underline{\texttt{else}}\ e' = e\end{array}$$

Здесь $\underline{\texttt{void}}$ определяется так: $\underline{\texttt{void}}(\mathbf{r}) \rightleftharpoons \forall\, v.\neg\ v\ \hat{\in}\ \mathbf{r}$.

## 7. Итераторные объектные модели

Выше были представлены два формализма:

1) *теории в* $\mathcal{OODL}$ – задают логические структуры, ведущие себя как объектные модели;

2) *итераторные алгебры* – описывают в терминах семантического программирования понятие и поведение итераторов.

Поведение Libretto описывается как сочетание этих двух формализмов – это итераторный язык, действующий на объектных моделях с множественным наследованием. В данном пункте будет показано, как эти формализмы взаимодействуют друг с другом. Для этого мы построим понятие *итераторных объектных моделей*, воспользовавшись тем, что описание предметных областей оо-проекциями изоморфно системе типов данных языка программирования с множественным наследованием [14].

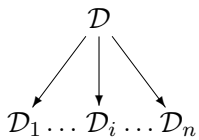
Строительство итераторных объектных моделей будем проводить последовательным расширением итераторных алгебр конструкциями, взятыми из оо-проекций.

Как отмечалось в пункте 2, базовым материалом для построения объектов и модели в целом является некоторая фиксированная область данных:

$$\mathcal{D} = \langle \mathcal{D}_1, \ldots, \mathcal{D}_k; \Omega \rangle$$

С помощью этого материала строятся *объекты* – основные обитатели итераторных объектных моделей. Каждый объект может принадлежать одному или нескольким *классам*. Классы могут определяться с помощью одноместных предикатов, выделяющих их элементы. Кроме того, объекты обладают свойствами. Свойства бывают двух типов - *о-свойства* (роли), образующие связи объектов с другими объектами, и *т-свойства* (атрибуты), привязывающие к объектам некоторые значения типов данных, рассматриваемые как характеристики этих объектов. В соответствии с определением оо-проекций в общем случае свойство объекта может принимать произвольное конечное число значений. Особенность языка Libretto состоит в том, что он интерпретирует эти последовательности значений как *итераторы*. Таким образом, в качестве значений объектов служат итераторы, и поскольку свойства бывают двух типов, то и итераторы также должны быть двух типов – генерирующие последовательности объектов (для о-свойств), и генерирующие последовательности данных (для т-свойств). В первом случае мы говорим об *о-итераторах*, а во втором – о *т-итераторах*. Смешение этих двух типов в Libretto запрещено. Это означает, в частности, что невозможно применить к итераторам разных типов операцию `cons`.

Напомним также, что области определения и области значений свойств в оо-проекциях типизированы (с помощью оператора типизации $\theta$). Типизация подчиняется иерархиям по наследованию. В частности, для т-свойств используется простая двухуровневая иерархия:




где

$$\mathcal{D} = \mathcal{D}_1 \cup \ldots \cup \mathcal{D}_k$$

Одноместные предикатные символы $D_1, \ldots, D_k$ интерпретируются как предикаты, выделяющие соответствующий тип данных. Если же т-свойство типизируется с помощью $D$, то это означает, что в качестве его значений могут выступать любые данные из $\mathcal{D}$ (но не объекты!).

В то время, как иерархия типов данных фиксирована, иерархия классов полностью зависит от описываемой предметной области. В оо-проекциях иерархия классов строится с помощью оо-определений

вида

$$C \sqsubseteq C_1 \sqcap \ldots \sqcap C_f \sqcap \forall R_1.B_1 \sqcap \ldots \sqcap \forall B_g.E_g \sqcap \forall P_1.D'_1 \sqcap \ldots \sqcap \forall P_h.D'_h$$

где $C$ – определяемый класс (концепт), $C_1, \ldots, C_f$ – надклассы $C$, а $D'_1, \ldots, D'_h \in \{D_1, \ldots, D_k\}$. Исходя из правил перехода, представленных в таблице на рис. 1, данное определение будет эквивалентно следующей формуле логики первого порядка:

АМ: Классовые аксиомы

$$C(x) \to C_1(x) \wedge \ldots \wedge C_f(x) \wedge \bigwedge_{i=1}^{g} \forall y.(R_i(x,y) \to B_i(y)) \wedge \bigwedge_{i=1}^{h} \forall y.(P_i(x,y) \to D'_i(y))$$

Каждому концепту $C_i$ в дескриптивной логике соответствует одноместный предикат $C_i(\cdot)$, а каждому о-свойству $R_i$ и т-свойству $P_i$ соответствуют двуместные предикаты $R_i(\cdot,\cdot)$ и $P_i(\cdot,\cdot)$, соответственно.

Будем говорить, что $C$ определен классовой аксиомой АМ. Так же, как и в определении оо-проекции, будем говорить, что в данной аксиоме $C$ *непосредственно наследует* от $C_1, \ldots, C_f$. Пусть $\mathcal{A} = \{A_1, \ldots, A_s\}$ – множество классовых аксиом вида АМ. Понятие *наследования* в $\mathcal{A}$ определяется как транзитивное замыкание непосредственного наследования. Система аксиом $\mathcal{A}$ называется *ацикличной*, если ни один определяемый этой системой класс $C$ не наследует сам от себя. $\mathcal{A}$ называется *замкнутой*, если каждый класс, имя которого встречается в $\mathcal{A}$, определен ровно одной классовой аксиомой.

**Определение 5.** *Система классовых аксиом $\mathcal{A} = \{A_1, \ldots, A_s\}$ называется* корректной, *если выполняются следующие условия:*

- *$\mathcal{A}$ – ацикличная;*
- *$\mathcal{A}$ – замкнутая;*
- *имя каждого о- и т-свойства входит в систему $\mathcal{A}$ ровно один раз.*

Словарем системы $\mathcal{A}$ назовем четверку $\langle \mathcal{C}, \mathcal{R}, \mathcal{P}, \mathcal{D} \rangle$, где $\mathcal{C} = \{C_1, \ldots, C_l\}$, $\mathcal{R} = \{R_1, \ldots, R_m\}$, $\mathcal{P} = \{P_1, \ldots, P_n\}$ – множества имен классов, о- и т-свойств, встречающихся в системе $\mathcal{A}$, соответственно, а предикатные символы $\mathcal{D} = \{D_1, \ldots, D_k\}$ предназначены для предикатов, выделяющих соответствующие типы данных.

**Определение 6.** *Пусть $\mathcal{A}$ – корректная система классовых аксиом со словарем $\mathcal{W} = \langle \mathcal{C}, \mathcal{R}, \mathcal{P} \rangle$, $O$ – некоторое конечное множество, называемое множеством объектов. Тогда* объектной моделью над $\mathcal{A}$ *назовем четырехсортную модель вида:*

$$\mathfrak{M} = \langle \mathcal{D}, I_{\mathcal{D}}, O, I_O; \mathtt{nil}, \Diamond, \mathtt{conc}, \mathtt{get}, \mathtt{next}, \mathcal{W} \rangle$$

*такую, что*

1) $\mathcal{D} = \mathcal{D}_1 \cup \ldots \cup \mathcal{D}_k \cup \{\Diamond\}$;

2) $I_\mathcal{D}$ *и* $I_O$ *– множества итераторов над* $\mathcal{D}$ *и* $O$*, соответственно;*

3) $O = \{o_1, \ldots, o_n, \Diamond\}$ *– конечное множество такое, что* $O \cap \mathcal{D} = \{\Diamond\}$;

4) *Символы* $C \in \mathcal{C}$, $R \in \mathcal{R}$ *и* $P \in \mathcal{P}$ *имеют типы* $\omega$, $\omega \times \iota\omega$ *и* $\omega \times \iota\delta$, *соответственно, где* $\omega$ *– сорт основного множества* $O$, $\delta$ *– сорт объединенного основного множества* $\mathcal{D}$, *а* $\iota\delta$ *и* $\iota\omega$ *– сорта итератов для сортов* $\delta$ *и* $\omega$, *соответственно.*

5) $\mathfrak{M} \models \mathcal{A}$.

6) $\langle \mathcal{D}, I_\mathcal{D}; \mathtt{conc}, \mathtt{get}, \mathtt{next} \rangle$ *и* $\langle O, I_O; \mathtt{conc}, \mathtt{get}, \mathtt{next} \rangle$ *являются итераторными алгебрами.*

Пункт 4 определения говорит о том, что для любого символа $C \in \mathcal{C}$, его интерпретация $C^\mathfrak{M}$ в модели $\mathfrak{M}$ должна удовлетворять $C^\mathfrak{M} \subseteq O$. Аналогично для символов $R \in \mathcal{R}$ и $P \in \mathcal{P}$: $P^\mathfrak{M} \subseteq O \times I_\mathcal{D}$ и $R^\mathfrak{M} \subseteq O \times I_O$. Таким образом, $C_i$ выделяют в множестве объектов классы, а $P_i$ и $R_i$ привязывают к объектам итераторы, содержащие значения свойств.

**Замечание 1.** Обратите внимание на то, что, как это и требовалось нами выше, в объектной модели $\mathfrak{M}$ отсутствуют итераторы, в которых смешаны объекты и элементы типов данных. В частности, отсутствуют конкатенации вида $\mathtt{conc}(\mathbf{r}_1, \mathbf{r}_2)$, если $\mathbf{r}_1 \in I_D$ и $\mathbf{r}_2 \in I_O$.

**Пример 1.** Построим пример объектной итераторной модели, описывающий некоторые отношения между людьми. Словарь модели:

$$\begin{aligned}
\mathcal{C} &= \{\texttt{Mammal, Female, Male, Person, Man, Woman}\},\\
\mathcal{R} &= \{\texttt{has-child}\}\\
\mathcal{P} &= \{\texttt{age, name}\}\\
O &= \{\texttt{Ann, Marie, Tom}\}\\
\mathcal{D} &= \{D_{\texttt{int}}, D_{\texttt{string}}\}
\end{aligned}$$

Система объектных аксиом $\mathcal{A}$:

$$\begin{aligned}
\texttt{Mammal}(x) &\to \forall y.(\texttt{age}(x,y) \to D_{\texttt{int}}(y)) \wedge \forall y.(\texttt{has-child}(x,y) \to \texttt{Mammal}(y))\\
\texttt{Female}(x) &\to \texttt{Mammal}(x)\\
\texttt{Male}(x) &\to \texttt{Mammal}(x)\\
\texttt{Person}(x) &\to \forall y.(\texttt{name}(x,y) \to D_{\texttt{string}}(y))\\
\texttt{Woman}(x) &\to \texttt{Person}(x) \wedge \texttt{Female}(x)\\
\texttt{Man}(x) &\to \texttt{Person}(x) \wedge \texttt{Male}(x)
\end{aligned}$$

Конкретная информация об объектах – следующие атомарные формулы истинны в модели:

$$\begin{gathered}\texttt{Woman}(Ann)\\ \texttt{Woman}(Marie)\\ \texttt{Man}(Tom)\\ \texttt{age}(Ann,(23))\\ \texttt{age}(Tom,(3))\\ \texttt{has-child}(Ann,(Marie,Tom))\end{gathered}$$

Легко проверяется, что данная модель удовлетворяет условиям определения 6. Итераторы, определяющие значения свойств, даны в скобочном синтаксисе. Из модели следует, что у Анны двое детей, поскольку в качестве значения свойства `has-child` указан двухэлементный итератор. Заметим, что классы `Man` и `Woman` определяются через множественное наследование от двух классов каждый. □

**Замечание 2.** Строгая типизация предикатов позволяет нам опускать угловые скобки итераторов, когда в них находится один элемент. Например, вместо $\texttt{age}(Ann,(23))$ будем писать $\texttt{age}(Ann,23)$ понимая второй аргумент как одноэлементный итератор. В Libretto также работает это соглашение. □

Пусть $\mathcal{K} = TBox \cup ABox$ – некоторая оо-проекция. Определим на $\mathcal{K}$ отображение $\mathcal{T}(\cdot)$, переводящее множества формул дескриптивной логики в множества формул первого порядка того же словаря в соответствии с правилами из таблицы на рис. 1, плюс следующее правило, объединяющее в один итератор все значения, относящиеся к свойству объекта:

Правило Q. Если для некоторого объекта $o$ множество $\{e_1,\ldots,e_m\}$ содержит все значения такие, что $\texttt{prop}(o,e_i) \in ABox$ для некорого свойства `prop`, то

$$\mathcal{T}(\{\texttt{prop}(o,e_1),\ldots,\texttt{prop}(o,e_m)\}) = \texttt{prop}(o,(e_1,\ldots,e_m)).$$

**Теорема 2.** *Если $\mathcal{K} = TBox \cup ABox$ – оо-проекция, то существует модель $\mathfrak{M}$, $\mathfrak{M} \models \mathcal{T}(\mathcal{K})$, являющаяся итераторной объектной моделью.*

*Доказательство.* Доказательство леммы проводится построением $\mathfrak{M}$ из материала, предоставляемого $\mathcal{K}$. То, что $\mathfrak{M}$ является объектной моделью в соответствии с определением 6, следует из следующих наблюдений:

1) $TBox$ оо-проекции переводится в набор аксиом вида AM, которые являются корректной системой аксиом $\mathcal{A}$ для модели $\mathfrak{M}$.

2) Элементы $ABox$, переводимые в атомарные формулы с помощью правила Q, не могут содержать итераторов, в которых смешиваются объекты и типы данных. Это обеспечивается строгим разделением свойств на о-свойства и т-свойства. Поэтому любой сгенерированный итератор попадает либо в множество $I_O$, либо в множество $I_{\mathcal{D}}$ модели $\mathfrak{M}$.

Следующее утверждение обосновывает существенное для нас качество итераторных объектных моделей, а именно, эквивалентность их поведения поведению оо-проекций.

**Теорема 3.** *Пусть* $\mathcal{K} = TBox \cup ABox$ *– некоторая оо-проекция, тогда существует итераторная объектная модель* $\mathfrak{M}$ *теории* $\mathcal{T}(\mathcal{K})$ *такая, что для любой* $\mathcal{SHOIN}(D)$*-формулы* $F$

$$\mathcal{K} \models F \text{ тогда и только тогда, когда } \mathfrak{M} \models \mathcal{T}(F)$$

*Доказательство.* На основе теоремы 2 строится свободная объектная модель, соответствующая оо-проекции $\mathcal{K}$. И необходимость, и достаточность утверждения доказывается индукцией по определению семантики дескриптивных формул. Сначала рассматривается случай, когда длина всех итераторов в определениях объектов не больше 1. В этом случае результат получается без использования теории GES – только на основании базового описания семантики формул дескриптивной логики через формулы логики первого порядка. Это описание представлено в таблице на рис. 1 (см также пп. 2.2.1.3 и 4.2 книги [12])). Случай произвольных итераторов рассматривается индукцией по их длине с применением аксиомы индукции AI.

## Список литературы


1. Малых А. А. Объектно-ориентированная дескриптивная логика / А. А. Малых, А. В. Манцивода // Изв. Иркут. гос. ун-та. Сер. Математика. – № 1. – 2011. – С. 57–72.
2. Гончаров С. С. Σ-программирование / С. С. Гончаров, Д. И. Свириденко // Логико-математические проблемы МОЗ. – Новосибирск, 1985. – С. 3–29. – (Вычислительные системы ; вып. 107).
3. Ershov Yu. L. Semantic Programming / Yu. L. Ershov, S. S. Goncharov, D. I. Sviridenko // Information processing, Proc. IFIP 10th World Comput. Congress, Dublin. – 1986. – Vol. 10. – P. 1093–1100.
4. Object-oriented Programming Language Libretto [Electronic resource]. – URL: http://librettolang.org.
5. Барвайс Дж. Справочная книга по математической логике. Ч. 1 / Дж. Барвайс. – М. : Наука, 1982. – 392 с.
6. Кокорин А. И. Линейно упорядоченные группы / А. И. Кокорин, В. М. Копытов. – М. : Наука, 1972. – 200 с.

7. Malykh A. A Query Language for Logic Architectures / A. Malykh, A. Mantsivoda // Proceedings of 7th International Conference "Perspectives of System Informatics". – Springer-Verlag Berlin Heidelberg, Lecture Notes in Computer Science. – 2010. – Vol. 5947. – P. 294–305.
8. Horrocks I. Reducing OWL entailment to description logic satisfiability / I. Horrocks, P. F. Patel-Schneider // Fensel D., Sycara K. and Mylopoulos J. (eds.). Proc. of the 2003 International Semantic Web Conference (ISWC 2003), number 2870 in Lecture Notes in Computer Science, 17-29. Springer.
9. Гончаров С. С. Замечание об аксиомах списочной надстройки GES / С. С. Гончаров // Логические вопросы теории типов данных. – Новосибирск, 1986. – С. 11–15. – (Вычислительные системы ; вып. 114).
10. Гончаров С. С. Теория списков и ее модели / С. С. Гончаров // Логические вопросы теории типов данных. – Новосибирск, 1986. – С. 84–95. – (Вычислительные системы ; вып. 114).
11. Гаврюшкина А. А. Теория списков и Σ–определимость / А. А. Гаврюшкина // Изв. Иркут. гос. ун-та. Сер. Математика. – 2011. – № 4. С. 27–38.
12. The Description Logic Handbook: Theory, Implementation, Applications / F. Baader, D. Calvanese , D. L. McGuinness, D. Nardi, P. F. Patel-Schneider. – Cambridge, 2003. – 574 p.
13. Schmidt-Schauss M. Attributive concept descriptions with complements / M. Schmidt-Schauss, G. Smolka // Artificial Intelligence. – 1991. – Vol. 48. – P. 1–26.
14. Малых А. А. Логические архитектуры и объектно-ориентированный подход / А. А. Малых, А. В. Манцивода, В. С. Ульянов // Вестн. НГУ. Сер. Математика, механика, информатика. – 2009. – Т. 9, № 3. – С. 64–85.
15. The Semantic Web [Electronic resource]. – URL: http://www.w3.org/2001/sw.
16. The NCBI Entrez Taxonomy [Electronic resource]. – URL: http://www.ncbi.nlm.nih.gov/sites/entrez?db=taxonomy.
17. Web Ontology Language (OWL) [Electronic resource]. – URL: www.w3.org/2004/OWL.
18. Berners-Lee T. The Semantic Web / T. Berners-Lee, J. Hendler , O. Lassila // Scientific American. – 2001. – Vol. 5. – P. 34–43.

---

## A. Malykh, A. Mantsivoda
## Object Theories over List Superstructures

**Abstract**. In this paper the potential of semantic programming methods based on the theory of hereditarily finite list superstructures (GES) for the logical simulation of the object-oriented approach is considered and estimated. Based on GES, we develop a formal system, which is analogous to the description logic $\mathcal{OODL}$, but in contrast with $\mathcal{OODL}$, it allows the natural simulation of ordered data structures (e.g. lists and arrays). The formal system, which is introduced and investigated in this paper, can help for the development of the logical semantics of programming languages, in particular, the object-oriented programming language Libretto.

**Keywords:** description logic, object theory, datatype, object-oriented programming, semantic programming, Libretto

Малых Антон Александрович, Институт математики, экономики и информатики, Иркутский государственный университет, 664003, Иркутск, ул. К. Маркса, 1 тел.: (3952)242210 (malykh@baikal.ru)

Манцивода Андрей Валерьевич, Институт математики, экономики и информатики, Иркутский государственный университет, 664003, Иркутск, ул. К. Маркса, 1 тел.: (3952)242210 (andrei@baikal.ru)

Anton Malykh, Irkutsk State University, 1, K. Marks St., Irkutsk, 664003 Phone: (3952)242210 (malykh@baikal.ru)

Andrei Mantsivoda, Irkutsk State University, 1, K. Marks St., Irkutsk, 664003 Phone: (3952)242210 (andrei@baikal.ru)