ОБОБЩЕННАЯ ВЫЧИСЛИМОСТЬ И ОПРЕДЕЛИМОСТЬ
(Вычислительные системы)
1998 год Выпуск 161
УДК 519.68

ТРАНСЛЯЦИЯ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ЯЗЫКА СЕМАНТИЧЕСКИХ СПЕЦИФИКАЦИЙ В РЕЛЯЦИОННУЮ АЛГЕБРУ[1]

В.Ш. Гумиров

Введение

Объектно-ориентированные методы анализа и проектирования широко применяются в настоящее время на практике. Такие широко известные методологии, как Booch notation [8], OMT, UML [12] и др. [7], имеют коммерческие реализации в виде CASE средств поддержки анализа и проектирования. Их нотации отличаются в основном только внешне, имея при этом одинаковую семантику [13].

Не вдаваясь в подробности нотаций объектно-ориентированных методологий, рассмотрим основные понятия, используемые в них. Заметим, что анализ и проектирование базируются на разных входных данных, имея при этом похожие результирующие документы (продукты). В качестве входных данных при анализе используется информация, полученная аналитиком от заказчика. Аналитик фиксирует эту информацию в определенном виде, который понятен или может быть достаточно

[1]Работа выполнена при поддержке гранта РФФИ № 96-01-00097.

легко понят заказчиком (это нужно для верификации того, что полученная информация понята одинаково обоими сторонами). С другой стороны, этот вид понятен проектировщику, для которого аналитические продукты являются входными. Проектировщик преобразует аналитические документы в проектные решения путем модификации продуктов анализа. Основными документами, которыми оперирует и аналитик, и проектировщик являются:

- диаграмма классов,
- диаграмма взаимодействия объектов,
- диаграмма состояний объектов и др.

Отличаются эти диаграммы степенью детализации и тем, что аналитические документы должны быть понятны заказчику (для верификации), аналитику и проектировщику, а проектные документы должны быть понятны аналитику (для верификации), проектировщику и кодировщику (для реализации проектных решений в виде конкретного программного кода).

Большинство современных нотаций предполагают определение ограничений (constraints) при описании классов объектов и взаимодействии [13]. Однако при этом не определяется язык описания ограничений. Хотелось бы иметь для этой цели достаточно формальный язык, имеющий по возможности четкую семантику. При этом было бы хорошо, если бы этот язык имел конструктивную семантику [10,11].

В настоящей работе предлагается использовать для этой цели язык Σ-спецификаций [1,6]. Для этого строится его объектно-ориентированное расширение. Доказывается консервативность такого расширения. Кроме того, в качестве примера реализации строится трансляция из расширенного языка Σ-спецификаций в язык реляционной алгебры и доказывается ее корректность. Это может быть использовано для реализации так называемых быстрых Σ-предикатов [2], реализующих базовые классы (спецификации) поверх объектов базы данных.

Отношения на классах и объектах. Мы будем рассматривать классы, объекты этих классов и отношения на них. Через Cl мы будем обозначать непустое конечное или счетное множество классов, через Obj — множество объектов. Будем считать, что на множестве классов задано *отношение наследования* $(\mathrm{Cl}, \sqsubseteq_{\mathrm{Cl}})$, которое является частичным порядком.

Будем рассматривать объектную модель с метаклассами. Это означает, что классы сами являются объектами: $\mathrm{Cl} \subseteq \mathrm{Obj}$. Классы мы будем ассоциировать с их носителями (extent). То есть каждый класс можно рассматривать как множество объектов этого класса.

Символом $\in$ мы будем обозначать отношение *быть объектом*. Таким образом, запись $a \in A$ обозначает, что a является объектом класса A. Отношение *быть объектом* определяется следующими свойствами:

1) $\mathrm{Cl} \in \mathrm{Cl}$, $\mathrm{Obj} \in \mathrm{Cl}$;

2) $\mathrm{Cl} \sqsubseteq_{\mathrm{Cl}} \mathrm{Obj}$;

3) если $a, b \in \mathrm{Cl}$ и $a \sqsubseteq_{\mathrm{Cl}} b$, тогда $a \subseteq b$.

Типы. Определим множество *базовых типов* $\mathbf{BType} = \{bool, int, float, char\}$; множество типов $\mathbf{Type} = \mathrm{Cl} \cup \mathbf{BType}$. Определим введенное выше отношение наследования на типах $\sqsubseteq_{\mathbf{Type}}$ следующим образом:

1) если $t_1, t_2 \in \mathbf{BType}$, то $t_1 \sqsubseteq_{\mathbf{Type}} t_2$ тогда и только тогда, когда $t_1 = t_2$;

2) если $t_1, t_2 \in \mathrm{Cl}$, то $t_1 \sqsubseteq_{\mathbf{Type}} t_2$ тогда и только тогда, когда $t_1 \sqsubseteq_{\mathrm{Cl}} t_2$.

Отношение $\sqsubseteq_{\mathbf{Type}}$ можно считать расширением отношения наследования на классах $\sqsubseteq_{\mathrm{Cl}}$. Поэтому ниже мы оба этих отношения будем обозначать одним и тем же символом $\sqsubseteq$.

Атрибуты классов. Обозначим счетное множество идентификаторов (имен) символом **Indentifier**, множество всех конечных подмножеств множества A символом $\mathcal{P}'(A)$. Будем считать, что для каждого класса $c \in \mathrm{Cl}$ задается

конечное множество атрибутов $\mathbf{Attribute}(c)$ этого класса. При этом с каждым атрибутом $a \in \mathbf{Attribute}(c)$ связывается его имя $\mathbf{Identifier}(a) \in \mathbf{Identifier}$ и тип $\mathbf{type}(a) \mathbf{Type}$:

$$\mathbf{Attribute} : \mathrm{Cl} \to \mathcal{P}'(\mathbf{Identifier} \times \mathbf{Type}).$$

Мы будем требовать, чтобы в классе не было разных атрибутов с одинаковыми именами:

$$(\forall c \in \mathrm{Cl})(\forall a, b \in \mathbf{Attribute}(c))((a \neq b) \to$$

$$\to \mathbf{Identifier}(a) \neq \mathbf{Identifier}(b)).$$

Это позволит нам использовать в большинстве контекстов символ $\mathbf{Attribute}(c)$ для обозначения множества имен атрибутов класса c. В тех же случаях, когда из контекста не ясно, о каком именно множестве идет речь, мы будем это явно оговаривать.

Методы классов. Определим множество сигнатур методов $\mathbf{MethType} = \{\langle arg, ret\rangle | ret \in \mathbf{Type} \wedge arg \in \in \bigcup_{n \in \mathcal{N}} (\mathbf{Type})^n\}$. Будем считать, что для каждого класса $c \in \mathrm{Cl}$ задается конечное множество методов $\mathbf{Meth}(c)$ этого класса. При этом с каждым методом $m \in \mathbf{Meth}(c)$ связывается его имя $\mathbf{Identifier}(m)$ и тип $\mathbf{type}(m) \in \mathbf{MethType}$:

$$\mathbf{Meth} : \mathrm{Cl} \to \mathcal{P}'(\mathbf{Identifier} \times \mathbf{MethType}).$$

При этом так же, как и для атрибутов будем требовать, чтобы в классе не было разных методов с одинаковыми именами.

Предикаты классов. Определим множество сигнатур предикатов $\mathbf{PredType} = \{arg | arg \in \bigcup_{n \in \mathcal{N}} (\mathbf{Type})^n\}$. Будем считать, что для каждого класса $c \in \mathrm{Cl}$ задается конечное множество предикатов $\mathbf{Pred}(c)$ этого класса. При этом с каждым предикатом $m \in \mathbf{Pred}(c)$ связывается его имя $\mathbf{Identifier}(m)$ и тип $\mathbf{type}(m) \in \mathbf{PredType}$:

$$\mathbf{Pred} : \mathrm{Cl} \to \mathcal{P}'(\mathbf{Identifier} \times \mathbf{PredType}).$$

При этом так же, как и для атрибутов будем требовать, чтобы в классе не было разных методов с одинаковыми именами.

Сигнатуры классов. Каждому классу $c \in \mathrm{Cl}$ поставим в соответствие сигнатуру $\mathbf{Signature}(c) \rightleftharpoons \langle \mathbf{Attribute}(c), \mathbf{Meth}(c), \mathbf{Pred}(c) \rangle$, где

$\mathbf{Attribute}(c)$ — множество имен атрибутов класса c;

$\mathbf{Pred}(c)$ — множество предикатных символов;

$\mathbf{Meth}(c)$ — множество имен методов;

$\overline{\sigma}$ — множество всех таких сигнатур.

2. Объектно-ориентированные Σ-схемы

Определим множество объектно-ориентированных термов OOT:

1) если $T \in \mathbf{Type}$ — некоторый тип, $v \in \mathbf{Var}(T)$ — переменная типа T, то $v \in \mathrm{OOT}$, при этом $\mathbf{type}(v) = T$;

2) если $T \in \mathrm{Cl}$ — некоторый класс, $a \in \mathbf{Attribute}(T)$ — атрибут класса T, $t \in \mathrm{OOT}$ — терм типа T, т.е. $\mathbf{type}(t) = T$, то $t.a \in \mathrm{OOT}$ и его тип $\mathbf{type}(t.a) = \mathbf{AttrType}(T, a)$;

3) если F — функциональный символ сигнатуры σ_0, $F : \tau_1, \ldots, \tau_n \to \tau_0$, где $\tau_{i=\overline{0,n}} \in \mathbf{Type} \backslash \mathrm{Cl}$, $t_{i=\overline{1,n}} \in \mathrm{OOT}$, $\mathbf{type}(t_i) = \tau_i$ для $i = \overline{1,n}$, то $F(t_1, \ldots, t_n) \in \mathrm{OOT}$ и тип такого терма $\mathbf{type}(F(t_1, \ldots, t_n)) = \tau_0$.

Определим множество объектно-ориентированных Σ-формул OOL:

1) если $t_1, t_2 \in \mathrm{OOT}$, $\mathbf{type}(t_1) \in \mathrm{Cl}$, $\mathbf{type}(t_2) \sqsubseteq \mathrm{Cl}$, то $\Phi \rightleftharpoons (t_1 \in t_2) \in \mathrm{OOL}$;

2) если $t_1, t_2 \in \mathrm{OOT}$, $\mathbf{type}(t_1), \mathbf{type}(t_2) \sqsubseteq \mathrm{Cl}$, то $\Phi \rightleftharpoons (t_1 \sqsubseteq t_2) \in \mathrm{OOL}$;

3) если $t_1, t_2 \in \mathrm{OOT}$, $\mathbf{type}(t_1), \mathbf{type}(t_2) \in \mathrm{Cl}$, причем $\mathbf{type}(t_1) \sqsubseteq \mathbf{type}(t_2)$ или $\mathbf{type}(t_2) \sqsubseteq \mathbf{type}(t_1)$, то $\Phi \rightleftharpoons (t_1 = t_2) \in \mathrm{OOL}$;

4) если $t_1, t_2 \in \mathrm{OOT}$, $\mathbf{type}(t_1), \mathbf{type}(t_2) \notin \mathrm{Cl}$, причем $\mathbf{type}(t_1) = \mathbf{type}(t_2)$, то $\Phi \rightleftharpoons (t_1 = t_2) \in \mathrm{OOL}$;

5) если Φ — некоторая формула OOL, $v \in \mathbf{fv}(\Phi)$ — ее свободная переменная, причем $\mathbf{type}(v) \in \mathrm{Cl}$, то $\forall v \Phi$ и $\exists v \Phi$ — формулы OOL;

6) если Φ, Ψ — формулы OOL, то $\Phi \vee \Psi$, $\Phi \wedge \Psi$ — формулы OOL;

7) если Φ — некоторая формула OOL, $v \in \mathbf{fv}(\Phi)$ — ее свободная переменная, причем $\mathbf{type}(v) \notin \mathrm{Cl}$, $l \in \mathbf{ListOf}(\mathbf{type}(v))$, то $(\forall v \in l)\Phi$, $(\exists v \in l)\Phi$ и $(\exists v)\Phi$ — формулы OOL;

8) если $v \in \mathrm{OOT}$, $\mathbf{type}(v) = T \in \mathrm{Cl}$ — класс, кроме того, класс Q — предок класса T: $T \sqsubseteq Q$, если $P(x_1, \ldots x_k) \in \sigma(Q)$ — некоторый предикатный символ из сигнатуры класса Q, $t_1, \ldots, t_k \in \mathrm{OOT}$, то формула $v.Q :: P(t_1, \ldots, t_k)$ является формулой OOL;

9) если $v \in \mathrm{OOT}$, $\mathbf{type}(v) = T \in \mathrm{Cl}$, $P(x_1, \ldots, x_k) \in \sigma(T)$ — предикатный символ из сигнатуры класса T, $t_1, \ldots, t_k \in \mathrm{OOT}$, то $v.P(t_1, \ldots, t_k)$ является формулой OOL;

10) если H — формула сигнатуры σ_0, $\Phi \in \mathrm{OOL}$, то $H \rightarrow \Phi$ будет формулой OOL.

Рассмотрим объектно-ориентированные Σ-схемы (OOΣ-схемы). Пусть $\{S_1, \ldots, S_k\}$ — возможно пустое конечное множество OOΣ-схем, тогда OOΣ-схема S класса $c \in \mathrm{Cl}$, наследующая OOΣ-схемы $\{S_1, \ldots, S_k\}$:

$$S : S_1, \ldots, S_k \; ofclass \; c$$

$$P_1(\overline{x_1}) \; \underline{def} \; \Phi_1(\overline{x_1}, this)$$

$$\ldots$$

$$P_n(\overline{x_n}) \; \underline{def} \; \Phi_1(\overline{x_n}, this).$$

Для такой OOΣ-схемы S мы будем использовать следующие обозначения:

$\mathbf{Parents}(S) = \{S_1, \ldots, S_k\}$ — множество родительских (базовых) OOΣ-схем;

$\mathbf{Defs}(S) = \{P_1, \ldots, P_n\}$ — множество определяемых предикатов OOΣ-схемы S.

OOΣ-запросом к OOΣ-схеме S будем называть такую OOL-формулу Ψ, что $\mathbf{fv}(\Psi) \subseteq \{\overline{x_1}, \ldots, \overline{x_n}, this\}$.

Базовой сигнатурой OOΣ-схемы S назовем множество:

$$\mathbf{BaseSig}(S) = \bigcup_{i=\overline{1,k}} \mathbf{Defs}(S_i).$$

Сигнатурой ООΣ-схемы S назовем множество:

$$\mathbf{Signature}(S) = \Big(\bigcup_{i=\overline{1,n}} \overline{x_i} \Big) \bigcup P_1, \ldots, P_n \bigcup \{this\} \bigcup \mathbf{BaseSig}(S) \Big).$$

3. Трансляция ООΣ-схем в Σ-спецификации

Мы хотим построить такое преобразование, которое использовало бы только статическую информацию. Фактически это алгоритм для некоторого препроцессора из OOL в обычный Σ-язык.

Вначале мы определим преобразование множества определений ООΣ-схемы S в множество Σ-определений $\Sigma(S)$. Параллельно мы будем строить базовую сигнатуру этого множества $\mathbf{BaseSig}(\Sigma(S))$[2]. В заключение мы определим денотационную семантику ООΣ-схем.

Предполагаем, что $S_1, \ldots, S_n$ — базовые ООΣ-схемы нашей схемы S.

УТВЕРЖДЕНИЕ 1. *Существует алгоритм трансляции ООΣ-схем в Σ-спецификации.*

ДОКАЗАТЕЛЬСТВО. Пусть S — ООΣ-схема. Мы будем строить Σ-спецификацию S^Σ. При этом вначале в первых трех шагах мы определим трансляцию ООΣ-схемы S в Σ-схему $\Sigma(S)$, затем в последнем шаге мы получим уже Σ-спецификацию S^Σ.

ШАГ 1. Формулы OOL и термы OOT преобразуем в обычные Σ-формулы и Σ-термы. Каждому терму $t \in$ OOT поставим в соответствие Σ-терм $\Sigma(t)$. И соответственно каждой формуле $\Phi \in$ OOL поставим в соответствие Σ-формулу $\Sigma(\Phi)$.

1) Пусть $t \in$ OOT, $\mathbf{type}(t) = T \in$ Cl, $a \in \mathbf{Attribute}(T)$. Тогда T и a поставим в соответствие новый функциональный символ $T_a : T \longrightarrow \mathbf{type}(\mathbf{Attribute}(T, a))$ и добавим его в $\mathbf{BaseSig}(\Sigma(S))$. OOT-терму $t.a$ поставим в соответствие Σ-терм $\Sigma(t.a)$, определяемый следующим образом: $\Sigma(t.a) := T_a(\Sigma(t))$.

---

[2]Используются обозначения из [6].

2) Пусть $\Psi \rightleftharpoons (\mathbf{Q}v)\Phi$ — OOL-формула, где $\mathbf{Q} \in \{\wedge, \vee\}$, $v \in \mathbf{fv}(\Phi)$, $\mathbf{type}(v) = T \in \mathrm{Cl}$. Тогда OOL-формуле Ψ поставим в соответствие Σ-формулу $\Sigma(\Psi)$, определяемую следующим образом: $\Sigma(\Psi) := (\forall v \in T)(\Sigma(\Phi))$.

3) Пусть $\Psi \rightleftharpoons v.Q :: P(t_1, \ldots, t_k)$ — OOL-формула, где $Q \in \mathbf{Ancesstors}(S)$, $P \in \mathbf{All}(Q)$, $v \in \mathrm{OOT}$, $\mathbf{type}(v) = T \in \mathrm{Cl}$, $t_1, \ldots, t_k \in \mathrm{OOT}$. Тогда паре Q и P поставим в соответствие новый предикатный символ P^Q арности $k+1$, который добавим в $\mathbf{BaseSig}(\Sigma(S))$, и OOL-формуле Ψ поставим в соответствие следующую Σ-формулу: $\Sigma(\Psi) := P^Q(\Sigma(v), \Sigma(t_1), \ldots, \Sigma(t_k))$.

4) Пусть $\Psi \rightleftharpoons v.P(t_1, \ldots, t_k)$ — OOL-формула, где $P \in \mathbf{Defs}(S)$, $v \in \mathrm{OOT}$, $\mathbf{type}(v) = T \in \mathrm{Cl}$, $t_1, \ldots, t_k \in \mathrm{OOT}$. Тогда OOL-формуле Ψ поставим в соответствие следующую Σ-формулу: $\Sigma(\Psi) := P(\Sigma(v), \Sigma(t_1), \ldots, \Sigma(t_k))$.

ШАГ 2. Теперь займемся преобразованием определений ООΣ-схемы S. Пусть в схеме S имеется следующее определение:

$$\mathrm{D}: \qquad P(\overline{x}) \ \underline{def}\ \Phi(\overline{x}, this).$$

Тогда в Σ-схеме $\Sigma(S)$ мы поставим в соответствие этому определению следующее определение:

$$\mathrm{D}^{\Sigma}: \qquad P^S(\overline{x}, this) \ \underline{def}\ \Sigma(\Phi(\overline{x}, this)).$$

При этом добавим новый предикатный символ P^S арности $k+1$, где k — арность предикатного символа P, в сигнатуру $\mathbf{BaseSig}(\Sigma(S))$.

ШАГ 3. На этом шаге мы решаем проблему резолюции переопределяемых предикатов (полиморфизм).

Рассмотрим случай, когда некоторый определяемый в ООΣ-схеме S предикатный символ P переопределяется (overriden), т.е. $P \in \mathbf{Overrides}(S)$. Тогда в Σ-схему $\Sigma(S)$ мы добавляем следующие два определения:

$$P_o^S(\overline{x}, this) \ \underline{def}\ \mathbf{type}(this) = \mathrm{Cl}(S) \wedge P^S(\overline{x}, this),$$

$$P(\overline{x}, this) \ \underline{def}\ P_o^S(\overline{x}, this).$$

Добавляем в множество определений Σ-схемы $\Sigma(S)$ определения Σ-схем $\Sigma(S_1), \ldots, \Sigma(S_n)$. После этого, для каждого $P \in \mathbf{Overrides}(S)$ соответствующие правила вида $P(\overline{x}, this) \underline{def} P_o^S(\overline{x}, this)$, где $S \in \mathbf{S}$ и $\mathbf{S} = \{S | S \in \{S, S_1, \ldots, S_n\} \wedge P \in \mathbf{Defs}(S)\}$ сливаем в одно правило вида:

$$P(\overline{x}, this) \underline{def} \bigvee_{S \in \mathbf{S}} P_o^S(\overline{x}, this).$$

ШАГ 4. Определим трансляцию ООΣ-схемы S : $S_1, \ldots, S_k$ в Σ-спецификацию S^Σ.

1) Базовая сигнатура Σ-схемы S^Σ задается следующим образом:

$$\mathbf{BaseSig}(S^\Sigma) := \Big(\bigcup_{i=\overline{1,k}} \mathbf{BaseSig}(S_i^\Sigma) \Big) \cup \mathbf{BaseSig}(\Sigma(S)).$$

2) Σ-схема: $\mathbf{Schema}(S^\Sigma) := \Sigma(S) \cup \Big(\bigcup_{i=\overline{1,k}} \mathbf{Schema}(S_i^\Sigma) \Big)$.

3) $\mathbf{SigDef}(S^\Sigma) := \mathbf{SigDef}(\Sigma(S))$.

4) $\mathbf{SigOut}(S^\Sigma) := \mathbf{SigOut}(\Sigma(S))$.

Конец доказательства.

Денотационной семантикой ООΣ-схемы S будем называть денотационную семантику Σ-схемы $\Sigma(S)$.

4. Трансляция ООΣ в реляционную алгебру

Реляционная алгебра RA. *Схемой R* некоторого отношения мы будем называть конечное множество имен атрибутов $\{A_1, \ldots, A_n\}$:

$$R = \{A_1, \ldots, A_n\} \rightleftharpoons R[A_1, \ldots, A_n] \rightleftharpoons A_1 \ldots A_n.$$

Доменом схемы отношения $R[A_1, \ldots, A_n]$ будем называть множество $D = \bigcup_{i=1}^n D_i$, где $D_i = dom(A_i)$ — домен атрибута A_i.

Отношением r схемы R будем называть конечное множество отображений $\{t : R \to D | \ t(A_i) \in D_i$ для любого $i = \overline{1,n}\}$.

Будем использовать следующие записи для обозначения того, что отношение r имеет схему $R[A_1, \ldots, A_n]$:

$$r(R), r(A_1, \ldots, A_n), R = sch(r).$$

Ключем отношения $r(R)$ будем называть такое $K \subseteq R$, что для любых разных $t_1, t_2 \in r$ найдется атрибут ключа $B \in K$, на котором значения t_1 и t_2 будут отличаться: $t_1(B) \neq t_2(B)$.

Активным доменом атрибута $A \in R$ отношения $r(R)$ будем называть множество значений, принимаемых отношением r на этом атрибуте:

$$adom(A, r) \rightleftharpoons dom(A) \cap r(A) = \{d \in dom(A) |\ (\exists t \in r)(t(A) = d)\}.$$

Определим операторы реляционной алгебры.

1. Оператор выборки. Пусть $r(R)$ — отношение. Тогда для атрибута $A \in R$ и $a \in dom(A)$ определен оператор выборки, ставящий в соответствие отношению $r(R)$ следующее отношение: $\sigma_{A=a}(r) \rightleftharpoons \{t \in r |\ t(A) = a\}$. Схемой этого отношения будет $sch(\sigma_{A=a}(r)) \rightleftharpoons R$.

2. Оператор проекции. Пусть у нас имеется отношение $r(R)$. Тогда для $X \subseteq R$ будет определен оператор проекции отношений схемы R на это подмножество: $\pi_X(r) \rightleftharpoons \{t(X) |\ t \in r\}$, где $t(X)$ — сужение отображения t на X, определяемое обычным образом. Схемой результирующего отношения будет $sch(\pi_X(r)) = X$.

3. Оператор соединения. Для двух схем отношений R и S определим оператор соединения, сопоставляющий отношениям $r(R)$ и $s(S)$ следующее отношение схемы $T = R \cup S$:

$$q_T(r, s) = r \bowtie s \rightleftharpoons \{t |\ sch(t) = T \wedge t(R) \in r \wedge t(S) \in s\}.$$

4. Оператор деления. Для схем R и S, таких что $S \subseteq R$, определим оператор деления, ставящий в соответствие двум отношениям $r(R)$ и $s(S)$ следующее отношение схемы $R - S$:

$$r \div s \rightleftharpoons \{t |\ (\forall t_s \in s)(\exists t_r \in r)(t(R - S) = t \wedge t_r(S) = t_s)\}.$$

Трансляция ООΣ–схем в программы реляционной алгебры RA. Пусть $D \subseteq Dom(P)$. Каждому ООΣ–определению $Q: \quad P(\overline{x})$ **def** $\Psi(\overline{x}, this)$ и множеству D мы поставим в соответствие определение в реляционной алгебре $\mathrm{RA}_D(Q)$: $\mathrm{RA}_D(P)$ **def** $\mathrm{RA}_D(\Psi(\overline{x}, this))$ по нижеприведенным правилам.

1) Пусть $\Phi \rightleftharpoons t_1 \in t_2$ или $\Phi \rightleftharpoons t_1 \sqsubseteq t_2$ или $\Phi \rightleftharpoons t_1 = t_2$ — подформула формулы Ψ. Тогда $\mathrm{RA}_D(\Phi) \rightleftharpoons \sigma_\Phi(D)$.

2) Пусть $\Phi \rightleftharpoons (\exists v)(\Phi_1(v))$ — подформула формулы Ψ. Тогда $\mathrm{RA}_D(\Phi) = \pi_D(\mathrm{RA}_{D \bowtie \mathrm{MT}(\mathbf{type}(v))}(\Phi_1))$.

3) Пусть $\Phi \rightleftharpoons (\forall v)(\Phi_1(v))$ — подформула формулы Ψ. Тогда $\mathrm{RA}_D(\Phi) \rightleftharpoons \mathrm{RA}_{D \bowtie \mathrm{MT}(\mathbf{type}(v))}(\Phi_1(v)) \div \mathrm{MT}(v)$.

4) Пусть $\Phi \rightleftharpoons \Phi_1 \vee \Phi_2$ — подформула формулы Ψ. Тогда $\mathrm{RA}_D(\Phi) \rightleftharpoons \mathrm{RA}_D(\Phi_1) \cup \mathrm{RA}_D(\Phi_2)$.

5) Пусть $\Phi \rightleftharpoons \Phi_1 \wedge \Phi_2$ — подформула формулы Ψ. Тогда $\mathrm{RA}_D(\Phi) \cap \mathrm{RA}_D(\Phi_1) \cap \mathrm{RA}_D(\Phi_2)$.

6) Пусть $\Phi \rightleftharpoons H \to \Phi_1$ — подформула формулы Ψ. Тогда $\mathrm{RA}_D(\Phi) \rightleftharpoons \mathrm{RA}_D(\neg H) \cup \mathrm{RA}_D(\Phi_1) = \sigma_H(D) \cup \mathrm{RA}_D(\Phi_1)$.

7) Пусть $\Phi \rightleftharpoons (\forall v \in l)\Phi_1(v)$ или $\Phi \rightleftharpoons (\exists v \in l)\Phi_1(v)$, или $\Phi \rightleftharpoons (\exists v)\Phi_1(v)$ — подформула формулы Ψ, причем $l \in \mathbf{listOf}(\mathbf{type}(v))$ и $\mathbf{type}(v) \in \mathbf{Type} \backslash \mathrm{Cl}$. Тогда

$$\mathrm{RA}_D((\forall v \in l)\Phi_1(v)) \rightleftharpoons$$

$$\rightleftharpoons \pi_D(\mathrm{RA}_{\sigma_{v \in l}(Dom(f_v(\Phi_1)))}(\Phi_1) \div \sigma_{v \in l}(Dom(\{v\}))),$$

$$\mathrm{RA}_D((\exists v \in l)\Phi_1(v)) \rightleftharpoons \pi_D(\mathrm{RA}_{\sigma_{v \in l}(dom(f_v(\Phi)))}(\Phi_1)),$$

$$\mathrm{RA}_D((\exists v)\Phi_1(v)) \rightleftharpoons \pi_D(\mathrm{RA}_{dom(f_v(\Phi_1))}(\Phi_1)).$$

8) Пусть $v \in \mathrm{OOT}, \quad \mathbf{type}(v) = T \in \mathrm{Cl}, \quad T \sqsubseteq Q \in \mathrm{Cl}, \quad P \in \sigma(Q), \quad \Phi \rightleftharpoons v.Q :: P(t_1, \ldots, t_k)$. Тогда

$$\mathrm{RA}_D(\Phi) \rightleftharpoons \pi_D(\pi_{sch((f_v(\overline{x}=\overline{t})\backslash\{\overline{x}\})\cup\{v\})}(\mathrm{RA}(P)) \bowtie$$

$$\bowtie \mathrm{RA}_D(\overline{x} = \overline{t} \wedge \mathbf{type}(v) = Q)).$$

9) Пусть $v \in \mathrm{OOT}, \quad \mathbf{type}(v) = T \in \mathrm{Cl}, \quad P \in \sigma(T), \quad \Phi \rightleftharpoons v.P(t_1, \ldots, t_k)$. Тогда

$$\mathrm{RA}_D(\Phi) \rightleftharpoons \pi_D(\pi_{sch(fv(\overline{x}=\overline{t})\cup\{v\}\setminus\{\overline{x}\})}(\mathrm{RA}(P) \bowtie$$

$$\bowtie \mathrm{RA}(\overline{x} = \overline{t} \wedge v \in \mathbf{type}(v)))).$$

Пусть S — OOΣ-схема состоящая из определений $\{Q_1, \ldots, Q_k\}$. Поставим ей в соответствие реляционную программу $\mathrm{RA}(S)$, состоящую из определений отношений $\{\mathrm{RA}_{Dom(Q_1)}(Q_1), \ldots, \mathrm{RA}_{Dom(Q_k)}(Q_k)\}$.

Реляционной семантикой OOΣ**-схемы** $S = \{Q_1, \ldots, Q_k\}$ будем называть денотационную семантику реляционной программы $\mathrm{RA}(S)$: $\mathrm{LFP}\rho_{\mathrm{RA}(S)}$, где

$$\rho_{\mathrm{RA}(S)}(Q_1, \ldots, Q_k) = (\mathrm{RA}_{Dom(Q_1)}(Q_1), \ldots, \mathrm{RA}_{Dom(Q_k)}(Q_k)).$$

УТВЕРЖДЕНИЕ 2. *Для любой формулы* Φ *исчисления* OOL

$$\mathrm{RA}_{\overline{\mathbf{D}}}(\Phi) = \left\{\overline{x}|\overline{x} \in \overline{\mathbf{D}} \wedge \mathcal{N} \models (\Phi)|^{\overline{x}|_{fv(\Phi)}}_{fv(\Phi)}\right\},$$

где $\mathbf{D} = D_1 \times \ldots \times D_k$ *и существуют* $i_1, \ldots, i_l \leq k$ *такие, что* $D_{i_1} \times \ldots \times D_{i_l} = \mathbf{fv}(\Phi)$.

ДОКАЗАТЕЛЬСТВО проведем по индукции построения формулы OOL.

Для краткости приведем доказательства лишь для некоторых правил трансляции OOS-схем в RA-программы.

Для правила 1:

$$\mathrm{RA}_D(t_1 = t_2) = \sigma_{t_1=t_2}(D) = \left\{x \in D|\mathcal{N} \models (t_1 = t_2)|^{x|_{\mathbf{fv}(t_1 = t_2)}}_{\mathbf{fv}(t_1=t_2)}\right\}.$$

Для правила 2:

$$\mathrm{RA}_D(\exists v\Phi_1(v)) = \pi_D(\mathrm{RA}_{D \bowtie \mathrm{MT}(\mathbf{type}(v))}(\Phi_1)),$$

$$\mathrm{RA}_D(\exists v\Phi_1(v)) =$$

$$= \left\{x|x \in D \wedge \mathcal{N} \models (\exists v\Phi_1(v))|^{x|_{\mathbf{fv}(\Phi_1)\setminus\{v\}}}_{\mathbf{fv}(\Phi_1)\setminus\{v\}}\right\},$$

$$\pi_D(\mathrm{RA}_{D \bowtie \mathrm{MT}(\mathbf{type}(v))}(\Phi_1)) =$$

$$= \pi_D\left(\left\{y | y \in D \bowtie \mathbf{MT}(\mathbf{type}(v)) \wedge \mathcal{N} \models (\Phi_1)|^{y|_{\mathbf{fv}(\Phi_1)}}_{\mathbf{fv}(\Phi_1)}\right\}\right) =$$

$$= \left\{y(D) | y \in D \bowtie \mathbf{MT}(\mathbf{type}(v)) \wedge \mathcal{N} \models (\Phi_1)|^{y|_{\mathbf{fv}(\Phi_1)}}_{\mathbf{fv}(\Phi_1)}\right\} =$$

$$= \left\{x | x = y(D) \wedge y \in\in D \bowtie \mathbf{MT}(\mathbf{type}(v)) \wedge \mathcal{N} \models\right.$$

$$\left. \models (\Phi_1)|^{y|_{\mathbf{fv}(\Phi_1)},\, y|_v}_{\mathbf{fv}(\Phi),v}\right\} =$$

$$= \left\{x \in D | \exists z \in \mathbf{MT}(\mathbf{type}(v)) z(sch(D) \cup sch(\mathbf{MT}(\mathbf{type}(v)))) =\right.$$

$$= x(sch(D) \cup sch(\mathbf{MT}(\mathbf{type}(v))))\wedge$$

$$\left. \wedge \mathcal{N} \models (\Phi_1)|^{y|_{\mathbf{fv}(\Phi_1)},\, y|_v}_{\mathbf{fv}(\Phi),v}\right\} = \mathrm{RA}_D(\exists v \Phi_1(v)).$$

Для правила 4:

$$\mathrm{RA}_D(\Phi \vee \Psi) = \left\{x \in D | \mathcal{N} \models (\Phi \vee \Psi)|^{x|_{fv_1},\, x|_{fv_2},\, x|_{fv_3}}_{fv_1,\, fv_2,\, fv_3}\right\},$$

где $fv_1 = \mathbf{fv}(\Phi) \backslash \mathbf{fv}(\Psi)$, $fv_2 = \mathbf{fv}(\Phi) \cap \mathbf{fv}(\Psi)$, $fv_3 = \mathbf{fv}(\Psi) \backslash \mathbf{fv}(\Phi)$. Отсюда получаем

$\mathrm{RA}_D(\Phi \vee \Psi) =$

$$= \left\{x \in D | \mathcal{N} \models \Phi|^{x|_{fv_1},\, x|_{fv_2}}_{fv_1,\, fv_2} \text{ или } \mathcal{N} \models \Phi|^{x|_{fv_2},\, x|_{fv_3}}_{fv_2,\, fv_3}\right\} =$$

$$= \mathrm{RA}_D(\Phi) \cap \mathrm{RA}_D(\Psi).$$

Конец доказательства.

ТЕОРЕМА. *Трансляция* **RA** *корректна в том смысле, что денотационная семантика любой* **ООΣ**-*схемы совпадает с ее реляционной семантикой.*

ДОКАЗАТЕЛЬСТВО.

1. Оператор $\Gamma_{\Sigma(S)}(\overline{Q}) = \langle\{\overline{a} | \mathcal{N}_o \models \overline{B}(\overline{a})\}\rangle$.
2. Оператор $\rho_{\mathrm{RA}(S)}(\overline{Q}) = \langle\{\overline{a} | \overline{a} \in \mathrm{RA}(\overline{B}(\overline{a})\}\rangle$.

Надо доказать, что $\mathrm{LFP}(\Gamma_{\Sigma(S)}) = \mathrm{LFP}(\rho_{\mathrm{RA}(S)})$.

Имеем

$$\rho_{\mathrm{RA}(S)}(\overline{Q}) = \{\{\overline{a}|\overline{a} \in Dom(\mathbf{fv}(\overline{B})) \wedge \mathcal{N} \models$$

$$\models (\overline{B})|_{\mathbf{fv}(\overline{B})\overline{a}}\}) = \Gamma_{\Sigma(S)}(\overline{Q}),$$

что и требовалось доказать. Конец доказательства.

В заключение автор хотел бы поблагодарить Д.И. Свириденко, С.С. Гончарова за помощь в подготовке статьи, О.Г. Юрченко, В.В. Ващенко, Д.Е. Пальчунова, С.А. Лугового, в беседах с которыми и зародились идеи, заложенные в статье.

Литература

1. ГОНЧАРОВ С.С., СВИРИДЕНКО Д.И. Математические основы семантического программирования. Докл. АН СССР. — 1986. — Т.289, № 6.

2. ГОНЧАРОВ С.С., СВИРИДЕНКО Д.И. Σ^+-программы и их семантики //Логические методы в программировании. — Новосибирск, 1987. — Вып. 120: Вычислительные системы. — С. 24–51.

3. МАНЦИВОДА А.В. Проект ФЛЕНГ. — Иркутск, Иркутский Университет, 1993. — (Препринт.)

4. СВИРИДЕНКО Д.И. Проект СИГМА. Цели и задачи //Логические методы в программировании. — Новосибирск, 1990. — Вып. 133: Вычислительные системы. — С.69–94.

5. ГУМИРОВ В.Ш. Логические спецификации и отношение реализации на них //Логические методы в информатике. — Новосибирск, 1993. — Вып. 148: Вычислительные системы. — С. 18–31.

6. ГУМИРОВ В.Ш. Объектно-ориентированный вариант языка Σ–спецификаций //Теория вычислений и языки спецификаций. — Новосибирск, 1995. — Вып. 152: Вычислительные системы. — С. 3–19.

7. SHLAER S., MELLOR S.J. Object Lifecycles: Modelling the World in States. — Englwood Cliffs, NJ: Yourdon Press, 1992.

8. BOOCH G. Object-Oriented Analysis and Design with Applications. Benjamin/Cummings Publishing Company, Inc — 1994.

9. The Object Database Standard: ODMG-93 Release 1.2, editted by R.G.G. Cattell–Morgan Kaufmann Publishers, Inc., CA, 1995

10. DAVISON A. Polka: A Parlog Object–Oriented Language. PhD Thesis — Imperial Colledge: 1989

11. CLARK K.L., WANG T.I. Distributed Object–Oriented Logic Programming. Dept. of Computing — Imperial Colledge: 1995.

12. Rational Software Corp. Unified Modeling Language, Notation Guide — http: //www.rational.com, Santa Clara, CA: 1997.

13. IBM Corp., ObjecTime Limited OMG OA&D RFP Response, Document Version 1.0 - http://www.omg.org, 97-01-18.pdf OMG: 1997.

Поступила в редакцию
10 июня 1997 года