УДК 004.435

ФОРМАЛИЗАЦИЯ ОБЪЕКТНОЙ СТРУКТУРЫ С ПОМОЩЬЮ СИСТЕМ ОБЪЕКТНЫХ УРАВНЕНИЙ

А. А. Седунов

Воронежский государственный университет

Поступила в редакцию 18.11.2010 г.

Аннотация: В данной работе представлен нижний уровень теории объектов, разработанной автором в процессе работы над инфраструктурой системы метапрограммирования. Вводятся понятия структуры, объекта, кучи, определяется структура объектов, а также математический аппарат объектных уравнений и их систем.
Ключевые слова: ООП, теория, объект.
Annotation: The paper present lower level of object theory developed by the author during his work on the infrastructure of metaprogramming system. It introduces such concepts as structure, object, heap, considers object structure and develops mathematical notation of object equation systems.
Keywords: OOP, theory, object.

СТРУКТУРА ОБЪЕКТА

Вначале мы рассмотрим общую структуру объектов и введем основные обозначения. В соответствии с методологией ООП, объект характеризуется состоянием, поведением и уникальностью. В данной работе мы опираемся на класс-ориентированный подход, поэтому считаем, что поведение не приписывается отдельным объектам, а определяется на уровне типов (в частности, классов).

Уникальность объекта характеризуется ссылкой или указателем, идентифицирующим данный объект среди всех прочих. Большинство языков программирования поддерживают объекты-значения (например, данные, относящиеся к простым типам: целые, вещественные, логические и др.), которые не обладают идентичностью и в процессе вычислений передаются непосредственно, а не в виде ссылок на разделяемую область памяти. Мы будем считать, что все объекты-значения кодируются в виде строк, заданных над фиксированным алфавитом Σ.

Состояние объекта характеризуется конечным набором атрибутов. В ходе вычислительного процесса объекты могут изменять свое состояние, сохраняя при этом уникальность, причем модификация атрибутов объекта, как правило, представляет собой многошаговую процедуру. Для отражения временного аспекта в данной модели мы будем рассматривать все состояния, через которые проходит объект как отдельные объекты (с общей уникальностью).

Рассмотрим формальное определение структуры объекта. Для этого вначале введем два фиксированных множества Σ и $\mathbf{I}$, где

1. Σ – некоторый алфавит.
2. $\mathbf{I}$ – некоторое счетное множество *ссылок*, причем $\mathbf{I} \cap \Sigma^* = \varnothing$. Мы будем использовать $\mathbf{I}$ для обозначения идентичностей объектов.

Для кодирования значений, не обладающих идентичностью, т. е. полностью определяемых своими компонентами, введем понятие структуры.

Структурой 0-го порядка назовем простые значения (строки над алфавитом Σ) или ссылки:

$$\mathbf{V}_0 \overset{def}{=} \Sigma^* \cup \mathbf{I}. \qquad (1)$$

Структурой k-го ($k > 0$) порядка назовем кортеж, состоящий из структур порядка не выше $k-1$:

$$\mathbf{V}_k \big|_{k>0} \overset{def}{=} \left(\bigcup_{j=0}^{k-1} \mathbf{V}_j \right)^* . \qquad (2)$$

Множество всех таких структур обозначим символом $\mathbf{V}$:

$$\mathbf{V} \overset{def}{=} \bigcup_{k \geq 0} \mathbf{V}_k . \qquad (3)$$

Объектом называется элемент множества $\mathbf{J} = \mathbf{I} \times \mathbf{V}$, т. е. упорядоченная пара вида $u = \langle n, d \rangle$, где n – *идентичность* объекта, а d – его текущее *состояние*:

$$\mathrm{id}\langle n, \mathrm{d} \rangle \overset{def}{=} n, \mathrm{bs}\langle n, \mathrm{d} \rangle \overset{def}{=} \mathrm{d}. \qquad (4)$$

В частности, псевдообъект null, используемый в объектно-ориентированных языках в качестве маркера для обозначения фактического отсутствия объекта, будем представлять в виде $\langle 0_{\mathbf{I}}, 0_{\mathbf{I}} \rangle$.

Будем говорить, что две структуры одного порядка k *имеют одинаковую форму*, если выполняются следующие соотношения:

$$V_1 \sim V_2 \overset{def}{\Leftrightarrow} V_1, V_2 \in \mathbf{V}_k \wedge \wedge \begin{cases} k = 0 \Rightarrow V_1, V_2 \in \mathbf{I} \vee V_1, V_2 \in \Sigma^* \\ k > 0 \Rightarrow |V_1| = |V_2| = \\ = m \wedge \forall (i \in \overline{1,m})[V_1(i) \sim V_2(i)] \end{cases}. \quad (5)$$

Поскольку структуры порядка выше 0 являются кортежами, к ним можно применить операцию модификации $[i \leftarrow V]$. Кроме того, мы можем обобщить операцию модификации на случай компонента произвольной вложенности. Пусть $V = \langle V_i \rangle_{i=1}^{m} \in \mathbf{V}_k$ и $I \in \mathbb{N}^p$ – последовательность индексов, идентифицирующих компонент структуры ($1 \le p \le k$), тогда выражение $[I \leftarrow V']V$ определяется следующим образом:

$$[I \leftarrow V']V \overset{def}{=} \begin{cases} [i \leftarrow V']V, I = \langle i \rangle \\ [I' \leftarrow V']V_i, I = \langle i \rangle I', |I'| > 0 \end{cases}. \quad (6)$$

В качестве особого случая введем также определение

$$[\langle\ \rangle \leftarrow V']V \overset{def}{=} V' \quad (7)$$

Заметим, что выражение определено только при условии корректности индексов i и финального выражения $[i \leftarrow V']V$.

Структура V *ссылается* на объект v (запись $V \to v$), если один из компонентов, возможно косвенных, совпадает с $\mathrm{id}\, v$:

$$V \to v \overset{def}{\Leftrightarrow} \begin{cases} V \in \mathbf{V}_0 \Rightarrow V = \mathrm{id}\, v \\ V \in \mathbf{V}_k\big|_{k>0} \Rightarrow \exists \left(i \in \overline{1,|V|}\right)[V_i \to v] \end{cases}. \quad (8)$$

Будем также говорить, что объект u *ссылается* на объект v, если на v ссылается один из его атрибутов:

$$u \to v \overset{def}{\Leftrightarrow} \mathrm{bs}\, u \to v. \quad (9)$$

Для описания таких операций, реализующих обход структуры, мы введем функцию $\mathrm{map} : (\mathbf{V}_0 \to \mathbf{V}_0) \to \mathbf{V} \to \mathbf{V}$. Пусть $f : \mathbf{V}_0 \to \mathbf{V}_0$ – функция преобразования простых значений и V – структура, тогда под $\mathrm{map}_f V$ понимается структура, определенная следующим соотношением (во втором выражении кортеж V интерпретируется как функция):

$$\begin{cases} \mathrm{map}_f (V : V \in \mathbf{V}_0) \overset{def}{=} f(V) \\ \mathrm{map}_f (V : V \in \mathbf{V}_k, k > 0) \overset{def}{=} \mathrm{map}_f \circ V \end{cases}. \quad (10)$$

Индукцией по порядку структуры можно установить следующие свойства функции map (свойства и предполагают существование композиции gf и f^{-1} соответственно):

$$\mathrm{map}_{\mathrm{id}_{\mathbf{V}_0}} = \mathrm{id}_{\mathbf{V}}, \quad (11)$$

$$\mathrm{map}_g\, \mathrm{map}_f = \mathrm{map}_{gf}, \quad (12)$$

$$\mathrm{map}_{f^{-1}}\, \mathrm{map}_f = \mathrm{id}_{\mathbf{V}}. \quad (13)$$

Кроме того, при определенном выборе f функция map сохраняет отношение ссылочности, а именно:

Лемма 1.

Если функция $f : \mathbf{V}_0 \to \mathbf{V}_0$ расширена на множество $\mathbf{J}$ таким образом, что $\mathrm{id}\, f(u) = f(\mathrm{id}\, u)$, тогда

$$V \to v \Rightarrow \mathrm{map}_f V \to f(v). \quad (14)$$

КУЧИ

Поскольку объекты могут содержать в своих атрибутах ссылки на другие объекты, их следует рассматривать в контексте некоторого множества. Для описания таких множеств, фактически являющихся моделью динамической памяти (кучи) в объектно-ориентированных языках, мы введем понятие *кучи*.

Кучей называется конечное множество объектов H, удовлетворяющее следующим условиям:

$$\begin{cases} \mathrm{null} \in H \\ u, v \in H \wedge \mathrm{id}\, u = \mathrm{id}\, v \Rightarrow u = v \\ u \in H, u \to v \Rightarrow \mathrm{id}\, v \in \mathrm{id}[H] \\ \mathrm{bs}\, u = \mathrm{bs}\, \mathrm{null} \Rightarrow u = \mathrm{null} \end{cases}. \quad (15)$$

Множество всех куч обозначим $\mathbf{H}$. Очевидно, что $\mathbf{H} \subseteq \mathbb{F}(\mathbf{J})$ и множество $\mathbf{H}$ счетно.

Заметим, что первое условие обеспечивает предопределенность псевдообъекта null и корректность соответствующей ссылки в любой куче. Второе условие отождествляет два объекта, обладающих одинаковым значением идентичности. Это позволяет для каждой кучи H построить функцию $\mathrm{heap}_H : \mathrm{id}[H] \to H$, отображающую ссылки на соответствующие объекты: $\mathrm{heap}_H(n) = u$, где $\mathrm{id}\, u = n$. Для упрощения записи выражение $\mathrm{heap}_H(n)$ сократим до $H(n)$. Третье условие обеспечивает замкнутость кучи, не позволяя ее элементам

ссылаться на объекты, не принадлежащие ей самой.

Поскольку фактические значения атрибутов-ссылок не играют принципиальной роли (важны только связи между элементами кучи, проявляющие в виде отношений $\rightarrow$), целесообразно ввести на множестве $\mathbf{H}$ отношение эквивалентности, учитывающее внутреннюю структуру кучи вне зависимости от конкретных значений идентичности.

Предположим, что между кучами H_1 и H_2 задано некоторое взаимно однозначное соответствие $f: H_1 \rightarrow H_2$. В этом случае мы будем подразумевать, что это соответствие автоматически расширяется на идентичности объектов так, что для любого $n \in \mathrm{id}[H_1]$ выполняется равенство $f(n) = (\mathrm{id}\, fH_1)(n)$. Будем говорить, что H_1 *отождествляется с* H_2 *отображением* f (запись: $H_1 \sim H_2$), если для любого объекта $u \in H_1$ выполняется равенство

$$\mathrm{bs}\, f(u) = \mathrm{map}_{\mathrm{V}f}\, \mathrm{bs}\, u, \qquad (16),$$

где функция $\mathrm{V}f: \mathbf{V}_0 \rightarrow \mathbf{V}_0$ имеет вид:

$$\mathrm{V}f(V) \overset{def}{=} \begin{cases} f(V), V \in \mathbf{I}; \\ V, V \notin \mathbf{I}. \end{cases} \qquad (17)$$

Две кучи H_1 и H_2 называются *эквивалентными* (запись $H_1 \sim H_2$), если они отождествляются хотя бы одним отображением:

$$H_1 \sim H_2 \overset{def}{\Leftrightarrow} \exists f\left[H_1 \overset{f}{\sim} H_2\right]. \qquad (18)$$

Таким образом, эквивалентность куч сохраняет их структуру, но не конкретные значения идентичностей.

Лемма 2.

$$\begin{cases} H \overset{\mathrm{id}_H}{\sim} H; \\ H_1 \overset{f}{\sim} H_2 \Rightarrow H_2 \overset{f^{-1}}{\sim} H_1; \\ H_1 \overset{f}{\sim} H_2 \Rightarrow H_2 \overset{g}{\sim} H_3 \Rightarrow H_1 \overset{gf}{\sim} H_3. \end{cases} \qquad (19)$$

Из этой леммы следует, что отношение $\sim$, определенное в (18), действительно является отношением эквивалентности.

Теорема 1.

Если кучи H_1 и H_2 отождествляются функцией f, то образ любой кучи $H \subseteq H_1$ также является кучей и отождествляется с H сужением $f_H: H \rightarrow f[H]$ функции f:

$$\begin{cases} H_1 \overset{f}{\sim} H_2 \\ H \in \mathbf{H}, H \subseteq H_1 \end{cases} \Rightarrow \begin{cases} f[H] \in \mathbf{H} \\ H \overset{f_H}{\sim} f[H] \end{cases} \qquad (20)$$

Используя эквивалентность, введем на множестве куч упорядочивающее отношение $\leq$. Пусть H_1 и H_2 – кучи и $f: H_1 \rightarrow H_2$ – инъективная функция. Тогда функция $f': H_1 \rightarrow f[H_1]$, такая, что $f = \mathrm{inc}_{f[H_1],H_2} \circ f'$, является биекцией. Будем говорить, что *куча* H_1 *расширяется до* H_2 *отображением* f (запись $H_1 \leq H_2$), если $f[H_1]$ – куча и функция f' отождествляет H_1 с $f[H_1]$.

$$H_1 \overset{f}{\leq} H_2 \overset{def}{\Leftrightarrow} \exists\left(f': f = \mathrm{inc}_{f[H_1],H_2} \circ f'\right) \times \times\left[H_1 \overset{f'}{\sim} f[H_1]\right], \qquad (21)$$

$$H_1 \leq H_2 \overset{def}{\Leftrightarrow} \exists f\left[H_1 \overset{f}{\leq} H_2\right]. \qquad (22)$$

Непосредственно из определения следует, что эквивалентность является частным случаем отношения $\leq$.

Лемма 3.

$$\begin{cases} H \overset{\mathrm{id}_H}{\leq} H \\ H_1 \overset{f}{\leq} H_2 \wedge H_2 \overset{g}{\leq} H_3 \Rightarrow H_1 \overset{gf}{\leq} H_3 \end{cases} \qquad (23)$$

Используя доказанную лемму можно легко установить, что отношение $\leq$ образует на множестве $\mathbf{H}^2$ предпорядок.

Для идентификации отдельных объектов без привязки к идентичности дополним кучу функцией разметки. Пусть $\mathbf{L}$ - фиксированное счетное множество меток. *Размеченной кучей* называется пара вида $\widehat{H} = \langle H, l\rangle \in \widehat{\mathbf{H}}$, где H – куча, а $l: L \rightarrow H, L \subseteq \mathbf{L}$ – *функция разметки*, которая каждой метке ставит в соответствие некоторый объект кучи. Элементы размеченной кучи $\widehat{H} = \langle H, l\rangle$ обозначим $\widehat{H}.H$ и $\widehat{H}.l$ соответственно.

Расширим отношение $\sim$ на размеченные кучи так, чтобы оно обеспечивало эквивалентность исходных куч и сохраняло разметку.:

$$\widehat{H_1} \overset{f}{\sim} \widehat{H_2} \overset{def}{\Leftrightarrow} \begin{cases} \widehat{H_1}.H \overset{f}{\sim} \widehat{H_2}.H \\ \mathrm{Dom}\left(\widehat{H_1}.l\right) = \mathrm{Dom}\left(\widehat{H_2}.l\right) \\ \widehat{H_2}.l = f \circ \widehat{H_1}.l \end{cases} \qquad (24)$$

$$\widehat{H_1} \sim \widehat{H_2} \overset{def}{\Leftrightarrow} \exists f\left[\widehat{H_1} \overset{f}{\sim} \widehat{H_2}\right]. \qquad (25)$$

Можно показать, что для введенного таким образом отношения справедливы утверждения и , Следовательно, отношение – на множестве $\widehat{\mathbf{H}}^2$ также образует эквивалентность.

Аналогичным образом распространим отношения $\leq$ и $\leq$ на размеченные кучи:

$$\widehat{H}_1 \overset{f}{\leq} \widehat{H}_2 \overset{def}{\Leftrightarrow}$$

$$\overset{def}{\Leftrightarrow}\begin{cases}\widehat{H}_1.H \overset{f}{\leq} \widehat{H}_2.H \\ L_1 = \mathrm{Dom}\left(\widehat{H}_1.l\right) \subseteq \mathrm{Dom}\left(\widehat{H}_2.l\right) = L_2 \\ f \circ \widehat{H}_1.l = \widehat{H}_2.l \circ \mathrm{inc}_{L_1,L_2}\end{cases} \tag{26}$$

$$\widehat{H}_1 \leq \widehat{H}_2 \overset{def}{\Leftrightarrow} \exists f\left[\widehat{H}_1 \overset{f}{\leq} \widehat{H}_2\right]. \tag{27}$$

Как и в случае с эквивалентностью отношение $\leq$ на множестве $\widehat{\mathbf{H}}^2$ сохраняет свойство и остается предпорядком.

ОПЕРАТОРЫ НА КУЧАХ

Преобразования, переводящие одну кучу в другую, т. е. имеющие вид функций $p : \mathrm{H} \to \widehat{\mathbf{H}}$, где $\mathrm{H} \subseteq \widehat{\mathbf{H}}$, назовем операторами на кучах.

Для описания модификаций кучи мы введем 3 *элементарных* оператора:

генерация – добавление объекта с новой идентичностью;

модификация – изменение значения одного атрибута конкретного объекта;

разметка – добавление метки для объекта кучи;

Первые два оператора соответствуют элементарным действиям над объектами, поддерживаемым ОО-языками: создание нового объекта и изменение атрибута существующего объекта. Оператор разметки используется для связи существующего объекта с некоторым именем (меткой).

Поскольку многие современные языки программирования предполагают наличие в исполняющей системе механизма “сборки мусора”, обеспечивающего автоматическое удаление недостижимых объектов, в данной работе мы не рассматриваем операций, реализующих удаление объектов или меток.

Функцией генерации называется отображение следующего вида:

$$\begin{cases}\mathrm{new} : \mathbf{V}_+ \to \widehat{\mathbf{H}} \to \widehat{\mathbf{H}} \\ \mathrm{new}(\mathrm{V})\,\widehat{H} \overset{def}{=} \left\langle \widehat{H}.H \cup \left\{\left\langle \triangleright\, \mathrm{id}[H], \mathrm{V}\right\rangle\right\}, \widehat{H}.l\right\rangle\end{cases} \tag{28}$$

Параметр V определяет начальное состояние объекта. Следует заметить, что если структура V ссылается на некоторый объект с идентичностью n, то объект с такой же идентичностью должен присутствовать в куче-аргументе оператора, в противном случае выражение $\mathrm{new}(\mathrm{V})\,\widehat{H}$ не будет соответствовать определению кучи. Однако если все идентичности в структуре V равны $0_{\mathbf{I}}$, т. е. V ссылается только на null, оператор $\mathrm{new}(\mathrm{V})$ будет определен для любой кучи. Для этой цели мы определим операцию $\mathrm{null}(V)$, которая заменяет все идентичности в структуре V на $0_{\mathbf{I}}$:

$$\mathrm{null}(V) \overset{def}{=} \mathrm{map}_{\mathrm{null}_0} V,$$
$$\mathrm{null}_0(v) \overset{def}{=} \begin{cases}0_{\mathbf{I}}, v \in \mathbf{I} \\ v, v \notin \mathbf{I}\end{cases} \tag{29}$$

Функцией модификации (атрибута объекта) называется отображение следующего вида:

$$\begin{cases}\mathrm{mod} : \mathbf{J} \times \mathbb{N}^* \times \mathbf{V} \to \widehat{\mathbf{H}} \to \widehat{\mathbf{H}}; \\ \mathrm{mod}(u, I, V)\,\widehat{H}\big|_{u \subseteq \widehat{H}.H} \overset{def}{=} \\ \left\langle \widehat{H}.H \setminus \{u\} \cup \left\{\left\langle \mathrm{id}\, u, [I \leftarrow V]\,\mathrm{bs}\, u\right\rangle\right\}, \widehat{H}.l\right\rangle.\end{cases} \tag{30}$$

Результат применения оператора $\mathrm{mod}(u, I, V)\,\widehat{H}$ определен только при условии, что $u \in \widehat{H}.H$ и определено выражение $[I \leftarrow V]\,\mathrm{bs}\, u$.

Для упрощения записи мы будем пользоваться следующим обозначением:

$$\left[u[I] \leftarrow V\right]\widehat{H} \overset{def}{=} \mathrm{mod}(u, I, V)\,\widehat{H}. \tag{31}$$

Последовательное изменение нескольких атрибутов одного объекта реализуется с помощью композиции функций mod.

Функцией добавления метки называется отображение следующего вида:

$$\begin{cases}\mathrm{bind} : \mathbf{L} \times \mathbf{J} \to \widehat{\mathbf{H}} \to \widehat{\mathbf{H}}; \\ \mathrm{bind}(x, u)\,\widehat{H}\Big|_{\substack{u \in \widehat{H}.H \\ x \notin \mathrm{Dom}\left(\widehat{H}.l\right)}} \overset{def}{=} \\ \overset{def}{=} \left\langle \widehat{H}.H, [x \to u]\,\widehat{H}.l\right\rangle.\end{cases} \tag{32}$$

Условия, определяющие корректность операции $\mathrm{bind}(x, u)\,\widehat{H}$, означают, что метка может быть создана только для объекта, уже присутствующего в куче ($u \in \widehat{H}.H$) и только один раз, т. е. переопределение метки не допускается.

Оператор, который можно представить в виде композиции сужений элементарных операторов, назовем *выполнимым*. Если существует такой выполнимый оператор p, что $p\{\mathrm{null}\} \sim \widehat{H}$, то куча $\widehat{H}$ также называется *выполнимой*.

Теорема 2.

Любая куча выполнима.

Таким образом, с помощью элементарных операторов всегда можно построить кучу, эквивалентную любой заданной.

Перейдем теперь к описанию формализма, представляющего структуру кучи в виде системы уравнений специального вида.

ОБЪЕКТНЫЕ ТЕРМЫ, УРАВНЕНИЯ И СИСТЕМЫ УРАВНЕНИЙ

Объектный терм представляет собой обобщение структуры и объекта, в котором вместо идентичностей используются метки или переменные. Пусть $\mathbf{W}$ – фиксированное счетное множество *переменных*. Множеством *объектных термов* называется минимальное множество TO, удовлетворяющее следующим условиям:

$$\begin{cases} \mathrm{TO}_0 = \mathbf{L} \cup \mathbf{W} \cup \mathbf{\Sigma}^* \overset{def}{\subseteq} \mathrm{TO}; \\ t_i \big|_{i=1}^{k} \in \mathrm{TO} \overset{def}{\Rightarrow} \langle t_i \rangle_{i=1}^{k} \in \mathrm{TO}; \\ t \in \mathrm{TO} \overset{def}{\Rightarrow} t* \in \mathrm{TO}. \end{cases} \qquad (33)$$

Первая часть определения указывает элементарные компоненты – элементы множества TO_0, которые мы будем называть *простыми термами*. Вторая часть описывает *структурные термы*, которые по аналогии со структурами высокого порядка представляют собой кортежи более простых термов. Третья часть описывает *ссылочные термы* вида $t*$, соответствующие объектам с идентичностью. Если $t = t'*$, то будем также писать $t' = *t$ для извлечения терма, завернутого в конструктор $*$.

Для произвольного терма t определяются множества входящих в него переменных $\mathrm{var}\, t$ и меток $\mathrm{lab}\, t$:

$$\begin{cases} u \in \mathbf{\Sigma}^* \overset{def}{\Rightarrow} \mathrm{lab}\, u = \mathrm{var}\, u = \varnothing; \\ u \in \mathbf{L} \overset{def}{\Rightarrow} \mathrm{lab}\, u = \{u\}, \mathrm{var}\, u = \varnothing; \\ w \in \mathbf{W} \overset{def}{\Rightarrow} \mathrm{lab}\, w = \varnothing, \mathrm{var}\, w = \{w\}; \\ t = \langle t_i \rangle \big|_{i=1}^{k} \overset{def}{\Rightarrow} \mathrm{lab}\, t = \bigcup_{i=1}^{k} \mathrm{lab}\, t_i, \mathrm{var}\, t = \bigcup_{i=1}^{k} \mathrm{var}\, t_i; \\ t = t'* \overset{def}{\Rightarrow} \mathrm{lab}\, t = \mathrm{lab}\, t', \mathrm{var}\, t = \mathrm{var}\, t'. \end{cases} \qquad (34)$$

Терм t *зависит от переменной* w, если $w \in \mathrm{var}\, t$. Терм называется *базовым*, если он не зависит от переменных.

Функцией замены переменных называется произвольная конечная биекция $\sigma : W' \to W''$, где $W', W'' \subseteq \mathbf{W}$.

Переименованием терма t с функцией замены σ, где $\mathrm{var}\, t \in \mathrm{Dom}\, \sigma$, называется терм $t[\sigma]$, определенный следующим образом:

$$\begin{cases} t \in \mathbf{L} \cup \mathbf{\Sigma}^* \overset{def}{\Rightarrow} t[\sigma] = t \\ t \in \mathbf{W} \overset{def}{\Rightarrow} t[\sigma] = \sigma(t) \\ t = \langle t_i \rangle_{i=1}^{k} \overset{def}{\Rightarrow} t[\sigma] = \langle t_i[\sigma] \rangle_{i=1}^{k} \\ t = t'* \overset{def}{\Rightarrow} t[\sigma] = (t'[\sigma])* \end{cases} \qquad (35)$$

Переименованием кортежа переменных $\mathrm{w} = \langle w_i \rangle_{i=1}^{k}$ называется кортеж $\mathrm{w}[\sigma] \overset{def}{=} \langle w_i[\sigma] \rangle_{i=1}^{k} = \langle \sigma(w_i) \rangle_{i=1}^{k}$.

Заметим, что, в отличие от объектов, ссылочные термы могут быть вложены друг в друга. В данной работе мы ограничимся случаем термов, в которых конструктор $*$ встречается не более одного раза. Термы, не содержащие $*$, назовем *нормальными*.

Нормальные термы изоморфны структурам, отличаясь от них только тем, что вместо идентичностей $\mathbf{I}$ в них используются значения из множества $\mathbf{L} \cup \mathbf{W}$. Это дает нам основания расширить рассмотренные выше свойства структур на нормальные термы.

В частности, индукцией по порядку t можно показать, что переименование нормального терма $t[\sigma]$ представимо в виде $t[\sigma] = \mathrm{map}_{\mathrm{V}\sigma}\, t$, где функция $\mathrm{V}\sigma$ имеет вид

$$\begin{cases} \mathrm{V}\sigma(t : t \in \mathbf{L} \cup \mathbf{\Sigma}^*) = t \\ \mathrm{V}\sigma(t : t \in \mathbf{W}) = \sigma(t) \end{cases} \qquad (36)$$

Кроме того, множество нормальных термов замкнуто относительно операции переименования.

Объектной интерпретацией называется пара функций $\rho = \langle s, l \rangle$, где $s : W \in X$ (*функция подстановки*), $l : L \subseteq X'$ (*функция разметки*), $W \subseteq \mathbf{W}$, $L \subseteq \mathbf{L}$ и $X, X' \subseteq \mathbf{J}$.

Значением нормального терма t *при интерпретации* ρ, где $\mathrm{lab}\, t \subseteq \mathrm{Dom}(\rho.l)$ и $\mathrm{var}\, t \subseteq \mathrm{Dom}(\rho.s)$ называется структура $\mathrm{val}_\rho\, t = \mathrm{map}_{\mathrm{V}\rho}\, t$, где функция $\mathrm{V}\rho : \mathrm{TO}_0 \to \mathbf{V}_0$ определяется следующим образом:

$$\begin{cases} \mathrm{V}\rho(v : v \in \mathbf{\Sigma}^*) \overset{def}{=} v \\ \mathrm{V}\rho(x : x \in \mathbf{L}) \overset{def}{=} \mathrm{id}\, \rho.l(x) \\ \mathrm{V}\rho(w : w \in \mathbf{W}) \overset{def}{=} \mathrm{id}\, \rho.s(w) \end{cases} \qquad (37)$$

Иначе говоря, операция val заменяет метки и переменные, входящие в терм, конкретными значениями идентичностей, тем самым преобразуя его в структуру.

Объектным уравнением называется пара $e = \langle v, t \rangle$, где $v \in \mathbf{W}$ и t – ссылочный терм. Переменная v называется *левой частью* (запись: $\mathrm{lhs}\, e$), а терм t – *правой частью* ($\mathrm{rhs}\, e$) уравнения. Множество всех объектных уравнений обозначается OE. Уравнение называется *нормальным*, если его правая часть имеет вид t^*, где t – нормальный терм.

Набор всех переменных (меток), входящих в уравнение e, обозначается $\mathrm{var}\, e$ (соответственно $\mathrm{lab}\, e$):

$$\begin{cases} \mathrm{var}\, e \overset{def}{=} \mathrm{var}\, \mathrm{rhs}\, e \cup \{\mathrm{lhs}\, e\} \\ \mathrm{lab}\, e \overset{def}{=} \mathrm{lab}\, \mathrm{rhs}\, e \end{cases} \tag{38}$$

Переименованием уравнения e с функцией замены σ, где $\mathrm{var}\, e \subseteq \mathrm{Dom}\, \sigma$, называется уравнение

$$e[\sigma] \overset{def}{=} \langle (\mathrm{lhs}\, e)[\sigma], (\mathrm{rhs}\, e)[\sigma] \rangle. \tag{39}$$

Системой объектных уравнений порядка k, *заданной над множеством меток* L, называется кортеж $E = \langle e_i \rangle_{i=1}^{k} \in \mathrm{EO}^k$, в котором все левые части уравнений e_i различны. Множество всех таких систем обозначим $\mathrm{EO}^{(k)}$.

Для обозначения множества всех переменных (меток), входящих в систему E, используем обозначение $\mathrm{var}\, E$:

$$\begin{aligned} \mathrm{var} \langle e_i \rangle_{i=1}^{k} &\overset{def}{=} \bigcup_{i=1}^{k} \mathrm{var}\, e_i, \\ \mathrm{lab} \langle e_i \rangle_{i=1}^{k} &\overset{def}{=} \bigcup_{i=1}^{k} \mathrm{lab}\, e_i. \end{aligned} \tag{40}$$

Система E называется *замкнутой*, если каждая входящая в нее переменная является левой частью одного из уравнений, т.е. $\mathrm{var}\, E \subseteq \mathrm{lhs}\, E$.

Переименованием системы $E = \langle e_i \rangle_{i=1}^{k}$ с функцией замены σ, где $\mathrm{var}\, E \subseteq \mathrm{Dom}\, \sigma$, называется система $E[\sigma] \overset{def}{=} \langle e_i [\sigma] \rangle_{i=1}^{k}$

Система называется *нормальной*, если все ее уравнения являются нормальными.

РЕШЕНИЕ НОРМАЛЬНЫХ СИСТЕМ

Основным назначением объектных систем является представление структуры кучи и ее объектов без использования конкретных значений идентичностей, реализуемое через понятие *решения* системы.

Пусть $E = \langle e_i \rangle_{i=1}^{k}$ – замкнутая нормальная система объектных уравнений. Интерпретация ρ с инъективной подстановкой $\rho.s$ называется *решением системы* E (запись: $\rho \sim E$), если при всех $i = \overline{1,k}$ определены значения $\mathrm{val}_\rho\, \mathrm{lhs}\, e_i$ и $\mathrm{val}_\rho\, \mathrm{rhs}\, e_i$, причем $\mathrm{bs}\, \rho.s\, (\mathrm{lhs}\, e_i) = \mathrm{val}_\rho{}^*\, \mathrm{rhs}\, e_i$.

Лемма 4.

Пусть σ – функция замены переменных, заданная для нормальной системы E. Если $\rho = \langle s, l \rangle$ – решение системы E, то $\rho[\sigma] = \langle s \circ \sigma^{-1}, l \rangle$ – решение системы $E[\sigma]$.

Замкнутая система $E = \langle e_i \rangle_{i=1}^{k}$ *расширяет кучу* $\widehat{H}$ *до кучи* $\widehat{H}' \overset{f}{\geq} \widehat{H}$ (запись: $\widehat{H} \overset{E}{\geq} \widehat{H}'$), если $\mathrm{lab}\, E \in \mathrm{Dom}\, (\widehat{H}.l)$, $\mathrm{Dom}\, (\widehat{H}'.l) = \mathrm{Dom}\, (\widehat{H}.l)$ и существует такая биективная функция $s : \mathrm{var}\, E \to \widehat{H}'.H \setminus f[\widehat{H}.H]$, что интерпретация $\rho = \langle s, \widehat{H}'.l \rangle$ является решением системы E.

Теорема 3.

Все кучи, полученные расширением данной кучи $\widehat{H}$ системой уравнений E, эквивалентны друг другу:

$$\widehat{H} \overset{E}{\to} \widehat{H}' \wedge \widehat{H} \overset{E}{\to} \widehat{H}'' \Rightarrow \widehat{H}' \sim \widehat{H}''. \tag{41}$$

Утверждение означает, что для поиска кучи $\widehat{H}'$ в расширении $\widehat{H} \overset{E}{\to} \widehat{H}'$ достаточно найти одно решение, поскольку все остальные решения будут приводить к эквивалентным кучам. Рассмотрим процедуру построения подходящего решения.

Пусть E – замкнутая непустая система, $\mathrm{lab}\, E \subseteq \mathrm{Dom}\, (\widehat{H}.l)$ и $|\mathrm{var}\, E| = k$, построим кортеж $I = \langle I_j \rangle_{j=1}^{k}$ из k идентичностей таким образом, что

$$\begin{cases} I_1 =\triangleright \mathrm{id}[\widehat{H}.H] \\ I_{j+1} =\triangleright I_j, j = \overline{1,k-1} \end{cases} \tag{42}$$

Здесь символом $\triangleright n$, где n – идентичность обозначен следующий за n элемент множества $\mathbf{I}$ в предположении, что на $\mathbf{I}$ задано полное отношение порядка (мы считаем его произвольным, но фиксированным).

Очевидно, что идентичности I_j не встречаются у объектов кучи $\widehat{H}$. Сопоставим каждой переменной системы компонент I_j, согласно номеру этой переменной в системе (здесь $\ell : \mathrm{var}\, E \to \mathrm{Rng}\, I$):

$$\ell\, (\mathrm{lhs}\, E)_j \overset{def}{=} I_j. \tag{43}$$

В качестве подстановки определим следующую функцию $s : \mathrm{var}\, E \to \mathbf{J}$:

$$s\left(\mathrm{lhs}\, E\right)_j \overset{def}{=} \left\langle I_j, \mathrm{map}_I{}^*\left(\mathrm{rhs}\, E\right)_j \right\rangle, \qquad (44)$$

где функция I определяется на основе одноименного кортежа и функции разметки кучи $\widehat{H}$:

$$\begin{cases} I\left(v : v \in \mathbf{\Sigma}^*\right) \overset{def}{=} v \\ I\left(x : x \in \mathbf{L}\right) \overset{def}{=} \mathrm{id}\, \widehat{H}.l\left(x\right) \\ I\left(w : w \in \mathbf{W}\right) \overset{def}{=} \ell\left(w\right) \end{cases} \qquad (45)$$

Последним шагом построим кучу $\widehat{H}'$ так, чтобы

$$\begin{cases} \widehat{H}'.H = \widehat{H}.H \cup s\left[\mathrm{var}\, E\right] \\ \widehat{H}'.l = \widehat{H}.l \end{cases} \qquad (46)$$

Пару $\rho = \left\langle s, \widehat{H}'.l \right\rangle$, полученную в ходе описанной процедуры, назовем *основным решением*.

Теорема 4.

"Основное решение" является решением в смысле отношения $\widehat{H} \overset{E}{\to} \widehat{H}'$.

Так как все кучи, полученные расширением фиксированной кучи $\widehat{H}$ системой E, эквивалентны, то мы можем заменить отношение $\widehat{H} \overset{E}{\to} \widehat{H}'$ оператором $\overset{E}{\to}$, считая, что $\overset{E}{\to} \widehat{H} = \widehat{H}'$, где $\widehat{H}'$ определено выражением .

Теорема 5.

Оператор $\overset{E}{\to}$ выполним.

В силу выполнимости оператора $\overset{E}{\to}$ расширение, соответствующее системе E всегда можно выразить через исходную кучу с помощью элементарных операторов.

Обобщим рассмотренные свойства на случай размеченных куч.

РАЗМЕЧЕННЫЕ СИСТЕМЫ УРАВНЕНИЙ

Размеченной системой объектных уравнений называется пара $\widehat{E} = \left\langle E, l \right\rangle$, где E – объектных система уравнений, а $l : L \to \mathrm{var}\, E, L \subseteq \mathbf{L}$ – *функция разметки*, причем $\mathrm{lab}\, E \cap \mathrm{Dom}\left(\widehat{E}.l\right) = \varnothing$.

Будем говорить, что размеченная система $\widehat{E}$ *расширяет кучу* $\widehat{H}$ *до кучи* $\widehat{H}'$ (запись: $\widehat{H} \overset{\widehat{E}}{\to} \widehat{H}'$), если $\mathrm{Dom}\left(\widehat{E}.l\right) \cap \mathrm{Dom}\left(\widehat{H}.l\right) = \varnothing$ и существует такая промежуточная куча $\widehat{G}'$, что $\widehat{H} \overset{\widehat{E}.E}{\to} \widehat{G}'$ и $\widehat{H}'.H = \widehat{G}'.H$, причем функция разметки $\widehat{H}'.l$ удовлетворяет следующим условиям:

$$\begin{cases} \mathrm{Dom}\left(\widehat{H}'.l\right) = \mathrm{Dom}\left(\widehat{H}.l\right) \cup \mathrm{Dom}\left(\widehat{E}.l\right) \\ u \in \mathrm{Dom}\left(\widehat{H}.l\right) \Rightarrow \widehat{H}'.l\left(u\right) = \widehat{G}'.l\left(u\right) \\ u \in \mathrm{Dom}\left(\widehat{E}.l\right) \Rightarrow \widehat{H}'.l\left(u\right) = \rho.s\left(\widehat{E}.l\left(u\right)\right) \end{cases} \qquad (47)$$

где ρ – решение, соответствующее $\widehat{H} \overset{\widehat{E}.E}{\to} \widehat{G}'$.

Теорема 6.

Все кучи, полученные расширением данной кучи $\widehat{H}$ размеченной системой уравнений $\widehat{E}$, эквивалентны друг другу:

$$\widehat{H} \overset{\widehat{E}}{\to} \widehat{H}' \wedge \widehat{H} \overset{\widehat{E}}{\to} \widehat{H}'' \Rightarrow \widehat{H}' \sim \widehat{H}''. \qquad (48)$$

Расширим процедуру построения основного решения для исходной кучи $\widehat{H}$ на случай размеченной системы $\widehat{E}$. Пусть ρ – основное решение системы $\widehat{E}.E$ и $\widehat{G}$ – соответствующая куча, определенная в . Определим $\widehat{H}'$ следующим образом:

$$\begin{cases} \widehat{H}' = \left\langle \widehat{G}.H, l \right\rangle \\ l = \begin{cases} \widehat{G}.l\left(x\right), x \in \mathrm{Dom}\left(\widehat{G}.l\right) \\ \left(\rho.s \circ \widehat{E}.l\right)\left(x\right), x \in \mathrm{Dom}\left(\widehat{E}.l\right) \end{cases} \end{cases} \qquad (49)$$

Очевидно, что $\widehat{H}'$ является кучей в силу определения решения ρ. Под *основным решением размеченной системы* будем понимать пару $\rho' = \left\langle \rho.s, \widehat{H}'.l \right\rangle$.

Как легко заметить, из определений $\widehat{H} \overset{\widehat{E}}{\to} \widehat{H}'$ и функции l в непосредственно вытекает корректность построения ρ'.

Теорема 7.

"Размеченное основное решение" является решением в смысле отношения $\widehat{H} \overset{\widehat{E}}{\to} \widehat{H}'$.

Используя построенное основное решение можно расширить оператор $\overset{E}{\to}$ на размеченные кучи и понимать под $\overset{\widehat{E}}{\to} \widehat{H}$ расширение кучи $\widehat{H}$, соответствующее основному решению ρ'. Выполнимость при этом сохраняется.

Теорема 8.

Оператор $\overset{\widehat{E},\mathrm{w}}{\to} \left(\overset{\widehat{E}}{\to}\right)$ выполним.

ПРИМЕРЫ СИСТЕМ ОБЪЕКТНЫХ УРАВНЕНИЙ

В качестве примера использования построенного формализма приведем систему уравне-

ний, описывающую следующий фрагмент библиотеки коллекций Java (описания интерфейсов несколько упрощены). Вначале рассмотрим вариант без использования полиморфных типов.

```
public interface Iterable {
    Iterator iterator();
}

public interface Iterator {
boolean hasNext();
    Object next();
}

public interface Collection extends Iterable
{
    boolean isEmpty();
    int size();
    boolean add(Object item);
    boolean remove(Object item);
}
```

Состояние объектов, описывающих Java-интерфейсы будем представлять в виде <"Intf", Parents, Methods>, где Parents – кортеж ссылок на базовые типы , а Methods – кортеж описаний методов, каждое из которых имеет вид <Name, Params, Result> (имя метода, кортеж пар, содержащих имя и ссылку на тип параметра и ссылка на тип результата). Мы также подразумеваем, что объекты, описывающие типы boolean, int и Object уже определены и ассоциированы с соответствующими метками.

```
Iterable: Iterable =
*<"Intf", <>, <<"iterator", <>, Iterator>>>
Iterator: Iterator =
*<
"Intf",
<>,
<<"hasNext", <>, boolean>, <"next", <>,
Object>>
>
Collection: Collection =
*<
"Intf",
<Iterable>,
<
<"isEmpty", <>, boolean>,
<"size", <>, int>,
<"add", <<"item", Object>>, void>,
<"remove", <<"item", Object>>, void>
>
```

Для удобства записи мы выбрали имена переменных совпадающими с именами соответствующих меток, присвоенных уравнениям системы.

В данном случае набор методов хранится в структуре. Для того, чтобы смоделировать поведение массивов, аналогичное Java, т. е. реализовать их как объекты, нам потребуется представить их в виде ссылочных термов ("обернуть" в конструктор *). В этом случае, например, определение Iterator принимает вид

```
Iterator: Iterator =
*<
"Intf",
<>,
*<<"hasNext", <>, boolean>, <"next", <>,
Object>>
>
```

Это уравнение уже не является нормальным, поэтому к нему нельзя напрямую применить рассмотренное выше понятие решения. В рамках данного формализма мы разработали процедуру нормализации, переводящую систему общего вида в нормальную с сохранением структуры объектов, описываемых исходным набором уравнений. В силу ограниченного объема статьи мы не приводим здесь строгое определение нормализации с формулировкой необходимых свойств, но поясним ее на примере. Основная идея состоит в последовательном извлечении ссылочных термов из правых частей системы с одновременным добавлением новых уравнений, для которых эти термы являются правыми частями. В случае уравнения для Iterator мы получаем два уравнения:

```
Iterator: Iterator = *<"Intf", <>, X>
X = *<<"hasNext", <>, boolean>, <"next",
<>, Object>>
```

Здесь первое уравнение представляет собой модификацию исходного с заменой вложенного ссылочного терма новой переменной X, а второе содержит связь этой переменной с извлеченным ссылочным термом.

Аналогичная ситуация возникает при описании варианта данного примера с учетом параметрического полиморфизма Java.

```
public interface Iterable<T> {
    Iterator<T> iterator();
}
```

```
public interface Iterator<T> {
boolean hasNext();
    T next();
}

public interface Collection<T> extends
Iterable<T> {
    boolean isEmpty();
    int size();
    boolean add(T item);
    boolean remove(T item);
}
```

Мы расширим структур объекта, описывающего интерфейс до <"Intf", Parents, Methods, Args>, где Args – кортеж переменных типа, представленных в форме <"TVar", Name> (Name – имя переменной). Для описания экземпляров полиморфных типов, например, Iterator<T> мы используем объект с состоянием вида <"TInst", Type, <"TVar", Name>> (например, <"TInst", Iterator, <"TVar", "T">>).

```
Iterable: Iterable =
*<
"Intf",
<"T">,
<>,
*<<"iterator", <>, *<"TInst", Iterator,
*<"TVar", "T">>>>
>
Iterator: Iterator =
*<
"Intf",
<"T">,
<>,
*<<"hasNext", <>, boolean>, <"next", <>,
*<"TVar", "T">>>
>
```

В данном случае при нормализации первое уравнение распадается на 4, а второе – на 3 нормальных уравнения.

Заметим, что для учета границ, которые могут быть наложены на переменные типа в языке Java, потребовалось бы дополнительно расширить структуру объекта с меткой "TVar".

ЗАКЛЮЧЕНИЕ

Таким образом, представленный формализм позволяет описывать объектные структуры в терминах строгого математического аппарата систем объектных уравнений. Данная работы представляет альтернативный вариант построения математической модели, который в большей степени соответствует императивной методологии объектно-ориентированных языков, подобных Java, в отличие от формализмов типа ζ-исчисления, основанных на λ-нотации и ориентированных на языки, использующие, главным образом, функциональный стиль. В настоящий момент исследуются вопросы развития построенной теории для описания различных аспектов поведения, связанного с объектными системами.

СПИСОК ЛИТЕРАТУРЫ

1. *Bruce K. B.* Foundations of Object-Oriented Languages. Types and Semantics. – MIT Press, 2002.

2. The Java Language Specification (Third Edition) http://java.sun.com/docs/books/jls/download/langspec-3.0.pdf

3. *Abadi M., Cardelli L.* A Theory of Objects. Monograps in Computer Science – Springer, 1996.

Седунов Алексей Александрович – ассистент каф. "Программирование и информационные технологии" Воронежского государственного университета, тел. 8-961-182-31-28, e-mail: alexey.sedunov@gmail.com.

Sedunov A. A. – Assistant Department of Programming and Information Technologies, Voronezh State University, tel. 8-961-182-31-28, e-mail: alexey.sedunov @ gmail.com.