УДК 681.324

*С. А. Зинкин*

# СЕТИ АБСТРАКТНЫХ МАШИН ВЫСШИХ ПОРЯДКОВ В ПРОЕКТИРОВАНИИ СИСТЕМ И СЕТЕЙ ХРАНЕНИЯ И ОБРАБОТКИ ДАННЫХ (БАЗОВЫЙ ФОРМАЛИЗМ И ЕГО РАСШИРЕНИЯ)

Рассматриваются проблемы интеграции формальных представлений распределенных процессов и объектов, взаимодействующих через общее структурированное пространство памяти. Предложен новый формализм для описания согласованных взаимодействий процессов и объектов в системах и сетях хранения и обработки данных, базирующийся на декларативном и процедурном подходах к представлению знаний о функционировании распределенных систем.

## Введение

К настоящему времени разработан ряд технологий проектирования распределенных вычислительных систем, в том числе распределенных систем управления базами данных, систем управления промышленными объектами. Принципы построения программного обеспечения нередко скрыты от пользователя. Однако в последнее время наметилась тенденция создания такого сетевого программного обеспечения, в котором стираются грани между сетевой операционной системой, распределенной системой управления базой данных и распределенным приложением, а проект должен быть видим разработчику на всех этапах. Основным направлением решения поставленной задачи, расширяющим функциональность и улучшающим эксплуатационные характеристики проектируемой системы, является использование современных объектно- и агентно-ориентированных технологий и связанных с ними технологий согласования и координации объектов и процессов. На основе использования формальных методов в проектировании систем согласования и координации объектов и процессов в настоящей работе делается попытка создания методологии, которая позволила бы упростить переход к реализациям механизмов непосредственно исполняемых спецификаций систем хранения и обработки данных. Поставленная задача может быть решена путем создания сетевой операционной среды, адекватной решаемой задаче, т.е. сетевому приложению. Подобная среда должна обеспечить как обработку неструктурированной информации, так и выполнение операций над базами данных.

## Определение базового формализма

При обсуждении принципов построения распределенных систем хранения и обработки данных рассматривается ряд парадигм, определяющих оригинальные методологии их логического проектирования. Среди известных принципов наиболее распространены нисходящее проектирование, функциональная декомпозиция, объектно-ориентированное программирование. Для концептуального представления принципов организации распределенных систем с целью выработки единых в концептуальном плане подходов в настоящей работе предлагается использовать формализм

сетей абстрактных машин, берущих свое начало от машин Колмогорова–Успенского–Шёнхаге [1–3] и Гуревича [4, 5]. При определении декларативной составляющей формализма используется понятие алгебраических систем (в смысле работ [6–9]) и логик высших порядков [10], а для представления процедурной составляющей используются некоторые элементы алгебры алгоритмов Глушкова [11, 12], в частности операции α-дизъюнкции и α-итерации для всюду определенных условий.

Сеть абстрактных машин (CeAM), или эволюционирующая многоосновная алгебраическая система (ЭМАС), определяется следующим набором:

$$N = \left(A_1, ..., A_n, \Sigma, F_{\mathrm{C}}, F_{\mathrm{Ob}}, P_{\mathrm{C}}, P_{\mathrm{Ob}}, I(t), I(t_0), \Sigma', F', P', J, \mu_a, U_{\mathrm{F}}, U_{\mathrm{P}}, M,\right.$$

$$\left.\Omega_{\mathrm{Top}}, \Omega_{\mathrm{Cop}}, L, \Omega_{\mathrm{Lop}}, \Phi, \mathrm{B}, \Omega_{\mathrm{B}}, \mu_{\mathrm{update}}, \mu_{\mathrm{test}}, \mu_{\mathrm{block}}, \mu_{\mathrm{B}}, \mu_{\mathrm{MB}}, \mu_{\mathrm{L}}, C, Q, W\right), \quad (1)$$

где $A_1, ..., A_n$ – непустые непересекающиеся множества (основы), на которых определены множества функций и предикатов, имена которых составляют «текущую» (с изменяющейся, или эволюционирующей, в процессе функционирования сети интерпретацией $I(t)$ функциональных и предикатных символов) и «статическую» (с неизменной интерпретацией) сигнатуры;

$A = \{A_1, \ ..., \ A_n\}$ – множество основ;

$\Sigma = F_{\mathrm{C}} \cup F_{\mathrm{Ob}} \cup P_{\mathrm{C}} \cup P_{\mathrm{Ob}} = F \cup P$ – каузально-объектная сигнатура $\left(F = F_{\mathrm{C}} \cup F_{\mathrm{Ob}}, P = P_{\mathrm{C}} \cup P_{\mathrm{Ob}}\right)$;

$F_{\mathrm{C}}$ – множество «каузальных» функций;

$F_{\mathrm{Ob}}$ – множество «объектных» функций;

$P_{\mathrm{C}}$ – множество «каузальных» предикатов;

$P_{\mathrm{Ob}}$ – множество «объектных» предикатов;

$I(t)$, $t \geq t_0$ – текущая интерпретация функциональных и предикатных символов, составляющих сигнатуру $\Sigma$;

$I(t_0)$, $t_0 \geq 0$ – начальная интерпретация функциональных и предикатных символов, составляющих сигнатуру $\Sigma$;

$\Sigma' = F' \cup P'$ – статическая сигнатура (с не изменяемой в процессе функционирования CeAM-интерпретацией функциональных символов из множества $F'$ и предикатных символов из множества $P'$);

$J$ – интерпретация функциональных символов из множества $F'$ и предикатных символов из множества $P'$;

$\mu_a : (\Sigma \cup \Sigma') \to \{1, 2, ..., n_a\}$ – отображение арности, где $n_a = \mathrm{const}$; в процессе функционирования CeAM арности функций и предикатов остаются неизменными;

$U_{\mathrm{F}}$ – множество элементарных обновлений функций, сопоставленных функциональным символам из $\Sigma$;

$U_{\mathrm{P}}$ – множество элементарных обновлений предикатов, сопоставленных предикатным символам из $\Sigma$;

$U = U_{\mathrm{F}} \cup U_{\mathrm{P}}$ – система образующих алгебры модулей;

$M$ – множество модулей CeAM, реализующих локальные преобразования (обновления) текущей интерпретации $I(t)$ сигнатуры $\Sigma$; под модулем

подразумевается СеАМ-выражение, не содержащее свободных вхождений предметных переменных;

$\Omega_{\mathrm{Top}}$ – система темпоральных операций вида $\omega_{\mathrm{Top}}: M \times M \to M$, принимающих значения в множестве модулей $M$;

$\Omega_{\mathrm{Cop}}$ – система дополнительных операций вида $\omega_{\mathrm{Cop}}: L \times M \times M \to M$ и $\omega'_{\mathrm{Cop}}: L \times M \to M$ («управляющих конструкций» СеАМ), принимающих значения в множестве модулей $M$;

$L$ – множество логических условий;

$\Omega_{\mathrm{Lop}}$ – система логических операций вида $\omega_{\mathrm{Lop}}: L \times L \to L$ и $\omega'_{\mathrm{Lop}}: L \to L$, принимающих значения в множестве логических условий $L$;

$\Phi$ – множество атомарных формул (атомов) – система образующих алгебры логических условий;

$B$ – множество блоков СеАМ;

$\Omega_{\mathrm{B}}$ – множество операций, выполняемых в блоках модулей СеАМ;

$\mu_{\mathrm{update}}: M \to \mathbf{P}(\Sigma) \setminus \varnothing$ – отображение, сопоставляющее каждому модулю подмножество функций и предикатов, которые им модифицируются, $\mathbf{P}$ – символ булеана;

$\mu_{\mathrm{test}}: M \to \mathbf{P}(\Sigma) \setminus \varnothing$ – отображение, сопоставляющее каждому модулю подмножество функций и предикатов, которые им проверяются;

$\mu_{\mathrm{block}}: M \to \mathbf{P}(\Sigma) \setminus \varnothing$ – отображение, сопоставляющее каждому модулю подмножество функций и предикатов, с которыми оперирует данный модуль, причем

$$(\forall m \in M)\left(\mu_{\mathrm{block}}(m) = \mu_{\mathrm{update}}(m) \cup \mu_{\mathrm{test}}(m)\right)$$

(в общем случае для корректной работы сети функции и предикаты, с которыми оперирует конкретный модуль, должны быть временно заблокированы в целях недопущения нежелательных взаимоблокировок конкурирующих модулей);

$\mu_B: B \to \mathbf{P}\left(U_{\mathrm{F}} \cup U_{\mathrm{P}}\right) \setminus \varnothing$ – отображение, ставящее в соответствие каждому блоку подмножество совместимых элементарных обновлений сигнатуры $\Sigma$;

$\mu_{\mathrm{MB}}: M \to \mathbf{P}(B) \setminus \varnothing$ – отображение, сопоставляющее каждому модулю СеАМ подмножество реализуемых им блоков;

$\mu_{\mathrm{L}}: M \to \mathbf{P}(L) \setminus \varnothing$ – отображение, сопоставляющее каждому модулю подмножество проверяемых им логических условий;

$C$ – отношение каузации, представленное областью истинности одноименного предиката. Этот предикат определяется следующим образом:

$$\mathrm{C}\left(m_i, m_j\right) = p_{\mathrm{inter}}\left(\mu_{\mathrm{update}}(m_i), \mu_{\mathrm{test}}\left(m_j\right)\right),$$

где

$$p_{\mathrm{inter}}\left(\mu_{\mathrm{update}}(m_i), \mu_{\mathrm{test}}\left(m_j\right)\right) = \begin{cases} \text{true}, & \text{если} \quad \mu_{\mathrm{update}}(m_i) \cap \mu_{\mathrm{test}}\left(m_j\right) \neq \varnothing, \\ \text{false}, & \text{если} \quad \mu_{\mathrm{update}}(m_i) \cap \mu_{\mathrm{test}}\left(m_j\right) = \varnothing. \end{cases}$$

Для сети $N$ может быть дополнительно задана следующая пара множеств:

$Q$ – множество предикатов вида $q$: $M \to \{\text{true, false}\}$, характеризующих рабочее или нерабочее состояние модулей из множества $M$; данные предикаты не могут модифицироваться никакими модулями и используются только в формулах логических условий из множества $L$ для проверки фактов исполнения некоторых модулей в текущий момент времени;

$W$ – множество предикатов вида $w$: $M \to \{\text{true, false}\}$, используемых для «внешнего» или «внутреннего» управления текущей конфигурацией сети $N$: в процессе функционирования сети из нее могут исключаться некоторые модули либо они снова могут подключаться к сети; такие манипуляции с сетью могут осуществляться модулями с помощью включаемых в блоки правил обновления предикатов из множества $W$.

**Примечание**. Для удобства при описании операционной семантики сетей в дальнейшем мы будем использовать понятие $S$-пространства, объединяющего в одно множество $S = F \cup P$ множества функций $F$ и предикатов $P$.

Для используемых в логико-алгебраических моделях функций вида

$$f : A_{i1} \times A_{i2} \times ... \times A_{in} \to A_j$$

и предикатов вида

$$p : A_{i1} \times A_{i2} \times ... \times A_{in} \to \{\text{true, false}\}$$

введем дополнительно следующие (статические) функции:

$f_a(q)$ – функция арности предикатного или функционального символа $q$;

$\sigma_f(f)$, $\sigma_p(p)$ – функции типов функционального $f$ или предикатного $p$ символов соответственно (в многосортных или многоосновных системах тип $n$-арного предикатного символа – это кортеж $(i_1, i_2, ..., i_n, j)$, а тип $n$-арного предикатного символа – это кортеж $(i_1, i_2, ..., i_n)$, где $i_1, i_2, ..., i_n, j$ – названия (сорта) для основ или носителей);

$\pi_i(p)$ – функция проекции по $i$-й предметной переменной в области истинности $n$-арного предиката $p$;

$\sigma_{fp}(f)$ – функция, ставящая в соответствие $n$-арной функции $f$ $(n+1)$-арный предикат $p$.

Рассматриваемые нами абстрактные машины отличаются от других тем, что их память состоит не из элементарных ячеек, а из некоторых структур.

Известно, что при изучении алгебраических систем становится возможным сочетать как алгебраические, так и логические понятия и методы. При этом рассматривается язык $L(F, P)$, алфавит которого содержит счетное число переменных $x_1, x_2, ..., x_n, ...$ функциональные символы из $F$, предикатные символы из $P$, символ равенства, логические связки, кванторы всеобщности и существования. Формулы и термы составляются по известным в многосортном исчислении предикатов первого и высших порядков [6–10] правилам. Отличительной особенностью предлагаемого формализма является использование термов вида

$$(\tilde{\exists}!x \in X)p(x),\ (\tilde{\exists}!!x \in X)p(x),\ (\tilde{\forall}x \in X)p(x),\ (\tilde{\forall}!!x \in X)p(x) \qquad (2)$$

в блоках и в $\alpha$-выражениях модулей СеАМ (т.е. в записи формул для логических условий в операциях $\alpha$-дизъюнкции и $\alpha$-итерации). Назовем данные операторы квантифицированными операторами выбора кортежей из областей истинности $n$-арных $(n \geq 1)$ предикатов. Например, в случае применения введенных операторов к бинарным предикатам соответствующие термы будем записывать в следующем виде:

$$\begin{gathered}\left(\tilde{\exists}!x \in X, y \in Y\right) q(x, y),\ \left(\tilde{\exists}!!x \in X, y \in Y\right) q(x, y), \\ \left(\tilde{\forall} x \in X, y \in Y\right) q(x, y),\ \left(\tilde{\forall}!!x \in X, y \in Y\right) q(x, y).\end{gathered} \tag{3}$$

Рассмотрим группу термов (2). Результатом выполнения оператора $\tilde{\exists}!$ является единственное значение предметной переменной $x$, выбранное произвольным образом из области истинности унарного предиката $p$. Оператор $\tilde{\exists}!!$ выбирает значение $x$ при условии, что оно является единственным в области истинности унарного предиката $p$. Оператор $\tilde{\forall}$ позволяет выбрать все значения переменной $x$ из области истинности предиката $p$. Оператор $\tilde{\forall}!!$ выбирает все значения переменной $x$ из области истинности предиката $p$ при условии, что эта область совпадает с областью определения данного предиката. Выбранные значения для обоих операторов составляют результирующие унарные отношения.

Одноименные операторы в термах, заданных выражениями (3), выполняют аналогичные действия над бинарными предикатами. В результате вычисления термов $\left(\tilde{\exists}!x \in X, y \in Y\right) q(x, y)$ и $\left(\tilde{\exists}!!x \in X, y \in Y\right) q(x, y)$ находятся по одному кортежу, а результатами вычисления термов $\left(\tilde{\forall} x \in X,\ y \in Y\right)\ q(x, y)$ и $\left(\tilde{\forall}!!x \in X, y \in Y\right) q(x, y)$ являются отношения. Результаты вычисления термов обеих групп используются в блоках согласованных обновлений интерпретации сигнатуры $\sum$. Аналогично определяются и $n$-арные квантифицированные операторы выбора.

В термах вида (2), если это специально оговорено, переменная $x$ может пробегать по кортежам некоторого $n$-арного отношения, представленного областью истинности соответствующего предиката. Для выбора $i$-го элемента кортежа $x$ в этом случае необходимо использовать операцию проекции $pr_i(x)$, $i = 1, 2, ..., n$.

Ю. Гуревичем [4, 5] предложено использовать в машинах абстрактных состояний специальные операции – элементарные обновления функций и предикатов. Элементарное правило обновления функции или предиката запишем в виде правила вывода

$$\frac{t_1, t_2, ..., t_k, t_{k+1}}{s(t_1, t_2, ..., t_k) \leftarrow t_{k+1}}, \tag{4}$$

где $t_1, t_2, ..., t_k$ – термы различных сортов, в том числе термы, определенные выражениями (2); $s$ – функциональный или предикатный символ.

В случае если $s$ – функциональный символ, $t_{k+1}$ – суть терм любого сорта, а если $s$ – предикатный символ, то $t_{k+1}$ – булево выражение.

**Расширение базового формализма**

В отличие от модели Ю. Гуревича, мы допускаем использование в блоках согласованных обновлений операций реляционной алгебры, рассматривая каждое алгебраическое выражение как функцию, которая отображает множество отношений в одно отношение (отношения определены областями истинности одноименных предикатов с именами из $\sum$). Символы операторов реляционной алгебры в нашем случае являются сокращающими символами для обозначения множества согласованных обновлений интерпретации сигнатуры $\sum$. Другим отличием является возможность использования элементов исчисления предикатов высших порядков.

В отличие от обычных термов, используемых в многоосновном исчислении предикатов, значения термов с квантифицированными операторами выбора $\tilde{\exists}!$, $\tilde{\exists}!!$, $\tilde{\forall}$, $\tilde{\forall}!!$ не всегда могут быть вычислены, т.е. не удается осуществить выбор ни одного кортежа из области истинности некоторого предиката. В этом случае будем считать, что данный оператор выполняет «тождественное» обновление $R^E$, т.е. он не вносит ни одного изменения в интерпретацию сигнатуры $\sum$. Дадим попутно еще одно определение модуля как замкнутого выражения в алгебре модулей, все вхождения переменных в котором связаны обычными кванторами или квантифицированными операторами выбора.

Дополним сигнатуру $\sum$ выражениями следующего вида:

$$\varphi_i : D_{i_1} \times D_{i_2} \times ... \times D_{i_k} \to D_{i_{k+1}},$$
$$\psi_j : D_{j_1} \times D_{j_2} \times ... \times D_{j_r} \to \{\text{true}, \text{false}\},$$

где символами $D_q$ с соответствующими индексами обозначены множества $A_q \in A$ или булеаны $\mathbf{P}(A_q)$, $q = 1, 2, ..., n$. Подобное дополнение позволяет более компактным образом представлять групповые связи между объектами в логико-алгебраических моделях дискретных систем.

Для обеспечения возможности изменения групповых связей множества правил $U_{\mathrm{F}}$ и $\mathrm{U_P}$ обновления интерпретации текущей сигнатуры $\sum$ необходимо дополнить правилами вида (4), где значениями некоторых термов могут быть элементы булеанов множеств $A_q \in A$, $q = 1, 2, ..., n$.

Данное расширение СеАМ особенно удобно использовать для спецификации групповых операций, выполняемых в сетях хранения и обработки данных. Для таких сетей характерны операции, затрагивающие множество узлов, сообщений, каналов передачи данных, баз данных.

**Основные виды сетей абстрактных машин**

Выражения для модулей абстрактных машин предлагается использовать в качестве основы для создания системы исполняемых спецификаций, реализуемых аппаратно или программно в сетевой среде. В зависимости от характеристик среды реализации сетевого приложения может быть выбрана различная сложность заданных с помощью формализма СеАМ спецификаций. В этой связи определим следующие разновидности СеАМ.

1. СеАМ I рода. В выражениях для модулей не используются квантифицированные операторы выбора; в блоках допускается лишь одна темпоральная операция «|» (возможно одновременное выполнение согласованных элементарных правил обновления интерпретации сигнатуры ЭМАС); $\Omega_{\mathrm{Top}} = \varnothing$, т.е. в явной форме темпоральные операции не вводятся (вместо них определены каузальные (причинно-следственные) связи модулей через общую часть интерпретации сигнатуры $\sum$); множество операций $\Omega_{\mathrm{Cop}}$ содержит единственную тернарную операцию из алгебры алгоритмов Глушкова – «$\alpha$-дизъюнкцию» вида $Op_1 : L \times M \times M \to M$.

2. СеАМ II рода. В отличие от сетей СеАМ I рода, здесь в выражениях для модулей могут использоваться квантифицированные операторы выбора.

3. СеАМ III рода отличаются от СеАМ II рода тем, что в блоках модулей разрешены 3 операции: упомянутая ранее операция «|», операция «$\uparrow$» – непосредственное следование правил обновления интерпретации сигнатуры, и операция «$\updownarrow$» неодновременного выполнения правил. Для удобства чаще будем использовать упрощенные обозначения для указанных в предыдущем предложении темпоральных операций: «,», «;» и «:» соответственно.

Во всех трех предыдущих разновидностях СеАМ темпоральные операции используются лишь в блоках, причем в блоках разрешено применение только элементарных правил обновления интерпретации сигнатуры $\sum$ и реляционных операторов. Использование выражений общего вида для модулей в блоках не допускается.

4. СеАМ IV рода допускают использование квантифицированных операторов выбора, множество $\Omega_{\mathrm{Top}}$ содержит набор темпоральных операций. Данный набор при необходимости может быть расширен. Далее, $\Omega_{\mathrm{B}} = \Omega_{\mathrm{Top}} \cup \Omega_{\mathrm{Cop}}$, т.е. в блоках допустимы темпоральные и дополнительные операции над модулями в полном объеме. Множество $\Omega_{\mathrm{Cop}}$ содержит, помимо $\alpha$-дизъюнкции, бинарную операцию «$\alpha$-итерация» из алгебры алгоритмов Глушкова: $Op_2 : L \times M \to M$. СеАМ IV рода назовем «расширенными» или «развитыми» (РСеАМ).

Использование основных идей из алгебры алгоритмов Глушкова, например суперпозиций темпоральных и дополнительных операторов, позволяет естественным образом проектировать сложные иерархические расширенные сети абстрактных машин (ИРСеАМ). В этой связи имеет смысл выделить два класса сетей абстрактных машин. В относительно простых сетях первого класса подсети, или модули РСеАМ, начиная с тривиальных подсетей – элементарных обновлений интерпретации текущей сигнатуры, составляющих систему образующих, участвуют в операциях однократно. В сложных сетях второго класса допускается многократное участие указанных подсетей – модулей РСеАМ в произвольных суперпозициях подсетей. В обоих случаях формируются иерархические сети.

В дальнейшем будет часто использоваться общая аббревиатура СеАМ для обозначения технологии или реализации сетей, если это не противоречит контексту; там же, где следует различать конкретные виды выражений, описывающих сети абстрактных машин, будем использовать аббревиатуры СеАМ, РСеАМ и ИРСеАМ.

На основании изложенного дадим следующие определения для сетей СеАМ и РСеАМ.

**Определение 1**. Модуль сети называется $n$-арным, если выражение, которым он описан, содержит по крайней мере один $n$-арный предикатный или функциональный символ, входящий в локальную сигнатуру этого модуля и других предикатных и функциональных символов более высокой арности нет.

**Определение 2**. Сеть называется $n$-арной, если она содержит по крайней мере один $n$-арный модуль и других модулей более высокой арности нет.

**Определение 3**. Сеть называется семафорной (унарной), если она содержит только унарные модули.

**Определение 4**. Сеть имеет $n$-й порядок, если при описании по крайней мере одного модуля используется логика $n$-го порядка и логика более высоких порядков не используется.

**Определение 5**. Сеть является $n$-сортной, или $n$-основной, если $|A| = n$, где $A$ – множество основ (основных множеств).

**Определение 6**. Модуль – это абстрактная СеАМ(РСеАМ)-машина, интерпретирующая замкнутое СеАМ(РСеАМ)-выражение (выражение, не содержащее свободных предметных переменных) и модифицирующая локальные функции и предикаты, составляющие текущую интерпретацию некоторого подмножества символов из сигнатуры $\Sigma$ .

**Определение 7**. Сеть называется сетью первого класса, если при ее формировании допускается лишь однократное участие ее компонент в операциях из $\Omega_{\text{Top}} \cup \Omega_{\text{Cop}} \cup \Omega_{\text{B}}$ (темпоральных операциях из множества $\Omega_{\text{Top}}$, дополнительных операциях из множества $\Omega_{\text{Cop}}$ и блочных операциях из множества $\Omega_{\text{B}}$).

**Определение 8**. Сеть называется сетью второго класса, если при ее формировании допускается многократное участие подсетей в заданных операциями $\Omega_{\text{Top}}$, $\Omega_{\text{Cop}}$ и $\Omega_{\text{B}}$ в произвольных суперпозициях подсетей.

**Примечание**. Определения 1–6 относятся к сетям СеАМ, а определения 1–8 – к сетям РСеАМ.

### Сети абстрактных машин и логика высших порядков

Выше рассмотрены сети абстрактных машин, выражения для модулей которых основаны на исчислении первого порядка. Кванторы в таком исчислении и введенные нами квантифицированные операторы выбора могут связывать только предметные переменные, но не могут связывать функторы или предикаты. В логике предикатов второго порядка дополнительно вводятся переменные, пробегающие по признакам индивидов – их свойствам и отношениям между ними. Данные переменные можно связывать кванторами, получая выражения вида $(\forall p \in P)A$ – «для всякого свойства $p$ верно, что $A$», $(\exists r \in P)A$ – «существует отношение $r$ такое, что $A$». В логике предикатов третьего порядка используется квантификация по признакам признаков индивидов и т.д. [10].

Расширим выразительные возможности формализма СеАМ путем допущения квантификации по признакам и признакам признаков индивидов, что позволяет осуществить более гибкое управление взаимодействием моду-

лей через общую сигнатуру $\sum$. Например, при реализации модуля СеАМ ВП (высшего порядка)

$$m_{\text{ВП}} = \left[(\tilde{\exists}!x \in X)\, p(x)\right]\left(\left[(\tilde{\exists}!y \in Y)\, q(y)\right]\left(\left[(\tilde{\exists}!s \in \Sigma)\, s(x, y)\right]\right.\right.$$
$$\left.\left.\left(\{\ldots s(x, y) \leftarrow \text{false} \ldots\} \vee R^E\right) \vee R^E\right) \vee R^E\right) \tag{5}$$

сначала выбирается некоторое значение $x_i$ предметной переменной $x$ из области истинности унарного предиката $p$, затем аналогичным образом выбирается значение $y_j$ предметной переменной $y$. На последнем этапе интерпретации выражения (5) в сигнатуре $\sum$ отыскивается такое отношение $s$, что имеет место $s\left(x_i, y_j\right)$, а затем в блоке модуля (запись в фигурных скобках) эта связь между объектами $x_i$ и $y_j$ аннулируется. В выражении (5) три оператора «подчеркивания» подразумеваются по умолчанию, т.к. логические $\alpha$-выражения ($\alpha$-условия) содержат только подчеркнутые термы. Поскольку в выражении (5) первые два квантифицированные оператора выбора не зависят друг от друга, выражение (5) может быть записано в следующем эквивалентном виде:

$$m_{\text{ВП}} = \left[\underline{(\tilde{\exists}!x \in X)\, p(x)}\ \&\ \underline{(\tilde{\exists}!y \in Y)\, q(y)}\right]\left(\left[(\tilde{\exists}!s \in \Sigma) s(x, y)\right]\right.$$
$$\left.\left(\{\ldots s(x, y) \leftarrow \text{false} \ldots\} \vee R^E\right) \vee R^E\right). \tag{6}$$

В последнем выражении (6) операторы «_» позволяют «сформировать» пару атомарных формул, объединенных в первое $\alpha$-условие символом конъюнкции. На выполнение операций поиска значений предметных переменных $x$ и $y$ операторы «_» не влияют.

**Выводы**

1. На основе интеграции формальных представлений распределенных процессов и объектов, взаимодействующих через общее структурированное пространство памяти, предложен новый формализм для описания согласованных взаимодействий процессов и объектов, базирующийся на декларативном и процедурном подходах к представлению знаний о функционировании распределенных систем.

2. Данный формализм может быть положен в основу технологии проектирования распределенных систем хранения и обработки данных, а сами формальные представления могут непосредственно интерпретироваться в узлах вычислительной сети.

3. Предложен алгебраический подход к определению операционной семантики распределенных систем хранения и обработки данных, основанный на описании данных систем сетями абстрактных машин, при определении которых используется логика предикатов высших порядков. Данный подход дает возможность построения сложных иерархических сетей абстрактных машин и при необходимости позволяет использовать мощные аппараты алгебры алгоритмов и эволюционирующих алгебраических систем.

### *Список литературы*


1. **Gurevich, Y.** On Kolmogorov machines and related issues. The logic in computers science column / Y. Gurevich // Bulletin of Europian Assoc. for Theor. Comp. Science. – 1998. – № 35. – P. 71–82.
2. **Dexter, S.** Gurevich abstract state machines and Shönhage storage modification machines / S. Dexter, P. Doyle, Y. Gurevich // Journal of Universal Comp. Science. – 1997. – V. 3. – № 4. – P. 279–303.
3. **Григорьев, Д. Ю.** Алгоритмы Колмогорова сильнее машин Тьюринга / Д. Ю. Григорьев // Зап. науч. семинаров Ленинград. отд. матем. ин-та АН. – 1979. – № 88. – С. 15–18.
4. **Gurevich, Y.** Evolving Algebras 1993: Lipari Guide, Specification and Validation Methods, ed. E. Börger / Y. Gurevich // Oxford University Press. – 1995. – P. 9–36.
5. **Gurevich, Y.** Draft of the ASM Guide / Y. Gurevich // University of Michigan EECS Department Technical Report CSE-TR-336-97. – 1997. – May. – P. 1–27.
6. **Плоткин, Б. И.** Универсальная алгебра, алгебраическая логика и базы данных / Б. И. Плоткин. – М. : Наука, 1991. – 448 с.
7. **Плесневич, Г. С.** Логические модели / Г. С. Плесневич // Искусственный интеллект : в 3-х кн. Кн. 2. Модели и методы : справочник / под ред. Д. А. Поспелова. – М. : Радио и связь, 1990. – С. 14–28.
8. **Ершов, Ю. Л.** Математическая логика / Ю. Л. Ершов, Е. А. Палютин. – М. : Наука, 1979. – 320 с.
9. **Артамонов, В. А.** Общая алгебра / В. А. Артамонов, В. Н. Салий, Л. А. Скорняков [и др.]. – М. : Наука, 1991. – 480 с.
10. **Маркин, В. И.** Логика предикатов / В. И. Маркин // Новая философская энциклопедия : в 4-х т. – М. : Мысль, 2000. – 2 т.
11. **Глушков, В. М.** Алгебра. Языки. Программирование / В. М. Глушков, Г. Е. Цейтлин, Е. Л. Ющенко. – Киев : Наукова думка, 1978. – 320 с.
12. **Глушков, В. М.** Методы символьной мультиобработки / В. М. Глушков, Г. Е. Цейтлин, Е. Л. Ющенко. – Киев : Наукова думка, 1980. – 252 с.