Уфа: УГАТУ, 2003 Вестник УГАТУ Т. 4, № 1. С. 108–116

УДК 681.51.01(035.5)

В. В. МИРОНОВ, Р. Ф. АХМЕТШИН

АСИНХРОННАЯ ДЕЦЕНТРАЛИЗОВАННАЯ ИНТЕРПРЕТАЦИЯ ИЕРАРХИЧЕСКИХ СИТУАЦИОННЫХ МОДЕЛЕЙ

Рассматриваются вопросы реализации иерархических ситуационных моделей (ИСМ) на основе асинхронной децентрализованной интерпретации (АДИ). Описываются метод и математическое обеспечение АДИ, а также методы контроля текущего состояния ИСМ. Приводятся результаты практической реализации ИСМ на основе АДИ в среде Visual FoxPro. *Дискретно-событийные, иерархические, ситуационные, объектные, реляционные модели*

ВВЕДЕНИЕ

В автоматизированных системах принятия решений (АСПР) широкое применение находят дискретные динамические модели, отражающие изменение во времени процессов управления. Особый интерес представляют *встроенные* модели, позволяющие не только описать процесс на стадии проектирования, но и организовать работу СППР на стадии функционирования. Для этого применяются дискретные динамические модели различного вида (сети Петри, потоковые диаграммы, графы переходов, конечные автоматы и др.), дополненные соответствующими *интерпретаторами*, осуществляющими обработку моделей для идентификации текущей ситуации и выработки адекватных управленческих решений.

В течение ряда лет в УГАТУ ведется разработка одного из классов дискретно-событийных моделей — так называемых иерархических ситуационных моделей (ИСМ) [1–3]. Модели этого класса не только описывают ситуационное пространство принятия решений на этапе проектирования, но и непосредственно используются для формирования управленческих воздействий в процессе управления. ИСМ размещаются в системе управления в качестве встроенной базы знаний, содержащей правила обнаружения и смены ситуаций и принятия управленческих решений, ассоциированных с ситуациями. Разработан большой ассортимент как самих ИСМ, так и программных инструментальных средств их реализации, прежде всего, в технических системах, таких, как летательные аппараты (*В. В. Миронов, Н. И. Юсупова, Ю. Б. Головкин, Р. А. Ярцев, Л. Е. Гончар, О. Н. Сметанина, А. Н. Ситчихин*).

Применение встроенных динамических моделей требует реализации интерпретатора модели — специального программного обеспечения, осуществляющего контроль текущего состояния модели и на этой основе формирующего управляющие воздействия в контексте текущей ситуации. Традиционный подход к интерпретации ИСМ, который может быть назван *синхронным централизованным*, основан на применении централизованного интерпретатора, который циклически с достаточно высокой частотой выполняет обработку модели (рис. 1).

Этот подход хорошо зарекомендовал себя в технических системах, организованных по принципу «единого цикла управления», в ходе которого производится считывание текущих показаний датчиков, решение разнообразных задач управления, формирование управляющих воздействий. В более сложно организованных вычислительных средах, предусматривающих распределенные активно действующие компоненты, сложные интерфейсы с пользователями, сложное взаимодействие с базами данных, применение синхронного централизованного подхода становится неэффективным. В подобных системах наряду с синхронным применяется *асинхронный* событийный подход, в основе которого лежит контроль событий.

Событийный подход основан на обнаружении вычислительной системой моментов наступления определенных явлений — *событий*, автоматическом запуске обработчиков событий — специальных процедур, ассоциированных с событиями и выполняющих действия — реакцию на события. В этих условиях необходимо, чтобы интерпретатор ИСМ мог бы учитывать события и реагировать на них,

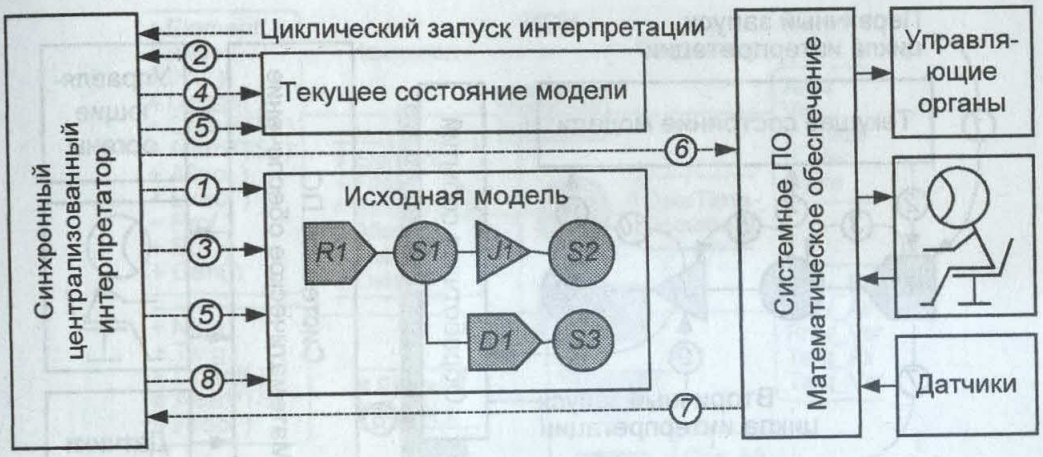


Рис. 1. Традиционный синхронный централизованный подход к интерпретации ИСМ: 1 — обращение к корневой субмодели; 2 — определение текущей (на первом цикле — начальной) ситуации субмодели; 3 — интерпретация текущей ситуации $S1$; 4 — запись текущей ситуации $S1$ в ПТС; 5 — интерпретация перехода $J1$; 6 — запрос значения предиката $\mathrm{Pr}(J1)$; 7 — возвращение значения предиката; 8 — интерпретация целевой ситуации $S2 = \mathrm{Tr}(J1)$; 9 — запись новой текущей ситуации $S2$ в ПТС

т. е. учитывать *событийные предикаты* активности ИСМ. В рамках традиционного синхронно-централизованного метода интерпретации ИСМ в принципе могут быть учтены событийные предикаты (например, обработчик при наступлении событий устанавливает специальные флаги, которые проверяются и сбрасываются в ходе интерпретаций). Однако такой путь сопровождается значительной избыточностью циклов интерпретации как до, так и после наступления событий и поэтому практически не эффективен.

Таким образом, необходим подход к интерпретации динамических моделей, позволяющий эффективно интерпретировать событийные предикаты. Поэтому возникает актуальная задача *создания программных инструментальных средств разработки АСПР на основе асинхронной децентрализованной интерпретации ИСМ*.

1. ПРЕДЛАГАЕМЫЙ ПОДХОД К ИНТЕРПРЕТАЦИИ ИСМ

Предлагаемое направление решения задачи на основе асинхронной децентрализованной интерпретации (АДИ) основано на обнаружении в системе моментов наступления определенных явлений — событий, автоматическом запуске обработчиков событий — специальных процедур, ассоциированных с событиями и выполняющих действия-реакции на события (рис. 2).

Первое отличие предлагаемого подхода состоит в представлении процесса интерпретации модели как последовательной *самоинтерпретации* ее объектов-элементов. Каждый элемент ИСМ рассматривается как объект, наделенный соответствующими свойствами и методами в смысле объектно-ориентированного подхода[1], в том числе методом Exec(), обеспечивающим самоинтерпретацию элементов.

На некотором цикле интерпретации извне вызывается метод Dive.Exec() объекта-погружения Dive корневой субмодели ИСМ. Выполняя самоинтерпретацию, объект Dive определяет, какая ситуация Sit субмодели является текущей, и вызывает метод Sit.Exec() этой ситуации. В свою очередь, в ходе самоинтерпретации методом Sit.Exec() последовательно вызываются методы самоинтерпретации объектов связи, ассоциированных с объектом-ситуацией. При самоинтерпретации связи-погружения Dive выполняется интерпретация внутренней субмодели ситуации. При самоинтерпретации связи-перехода Jump обеспечивается смена текущей ситуации субмодели. Самоинтерпретация Jump.Exec() начинается с определения текущего значения предиката Pr(Jump), ассоциированного с переходом. Если Pr(Jump) = истина, т. е. связь-переход активен, то целевая ситуация перехода Tr(Jump) устанавливается в качестве текущей для данной субмодели (мето-

[1] В терминологии объектно-ориентированного подхода инкапсулированные в объект данные называются *свойствами* объекта, а программы обработки данных — его *методами*.

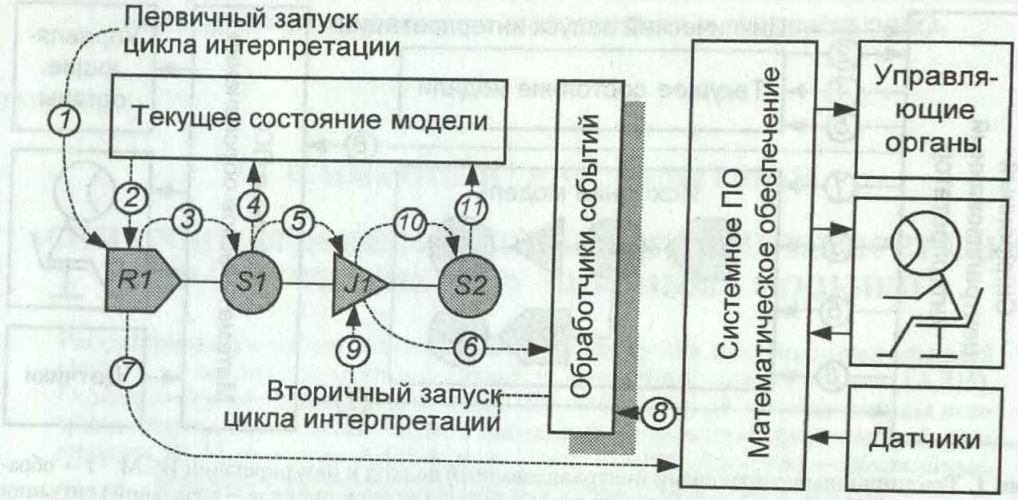


Рис. 2. Предлагаемый асинхронный децентрализованный подход к интерпретации ИСМ: *1* — вызов интерпретации корневой субмодели $R1$; *2* — определение текущей ситуации субмодели (на первом цикле — начальная ситуация $S1$); *3* — вызов интерпретации текущей ситуации $S1$; *4* — запись текущей ситуации $S1$ в ПТС; *5* — вызов интерпретации перехода $J1$ с событийным предикатом Pr ($J1$); *6* — настройка обработчика события событийного предиката $J1$; *7* — переход в режим ожидания события; *8* — вызов обработчика при наступлении события; *9* — вызов второго этапа интерпретации перехода $J1$; *10* — вызов интерпретации $S2$, целевой ситуации Tr ($J1$); *11* — запись новой текущей ситуации $S2$ в ПТС

дом Dive.SetCurSit()) и для нее вызывается метод самоинтерпретации Sit.Exec(). Указанные процедуры рекурсивно применяются как «вширь» — при переходе к другим ситуациям, так и «вглубь» — при интерпретации внутренних субмоделей.

Проведенный анализ показал, что предложенная организация процесса интерпретации ИСМ дает ту же самую функциональность, что и традиционный централизованный подход, но обеспечивает большую гибкость интерпретации отдельных элементов.

Второе отличие данного подхода заключено в асинхронной обработке предикатов. Предложена двухэтапная схема обработки предикатов, позволяющая учитывать событийные предикаты. Согласно этой схеме на первом этапе метод Exec() самоинтерпретации предикативного объекта-связи ограничивается тем, что настраивает обработчик события, ассоциированного с предикатом, на запуск процедуры второго этапа. При обнаружении ассоциированного события обработчик вызывает метод DoAct(), соответствующий второму этапу интерпретации. В ходе его выполняются те действия, которые выполняются при интерпретации обычного (несобытийного) предиката после обнаружения его истинности.

На основе этого подхода разработана иерархия основных классов объектов ИСМ (рис. 3). Она специфицирует состав и наследование основных свойств и методов как объектов-элементов исходной модели, так и соответствующих элементов ее текущего состояния. Иерархия является концептуальной основой для последующей разработки и программной реализации классов объектов ИСМ, а также для создания новых элементов с модифицированной функциональностью (на рис. 3 приведен пример такого нового элемента — Dive_Inter — массива погружений).

2. МЕТОД КОНТРОЛЯ ТЕКУЩЕГО СОСТОЯНИЯ ИСМ

Рассмотренный в разд. 2 подход к интерпретации ИСМ предполагает отражение в памяти текущего состояния (ПТС) записей о текущих объектах модели, сохраняемых между циклами интерпретации. Необходим метод контроля текущего состояния применительно к новым условиям интерпретации на основе АДИ и его математическое обеспечение. Метод должен позволять однозначно идентифицировать текущие объекты и связывать их между собой в процессе АДИ. Ввиду ориентации на применение в информационных системах общего назначения необходимо также обеспечить достаточно простое взаимодействие баз данных с объектами интерфейса пользователя. Сложность однозначной идентификации текущих объектов обусловлена тем, что в условиях циклического и

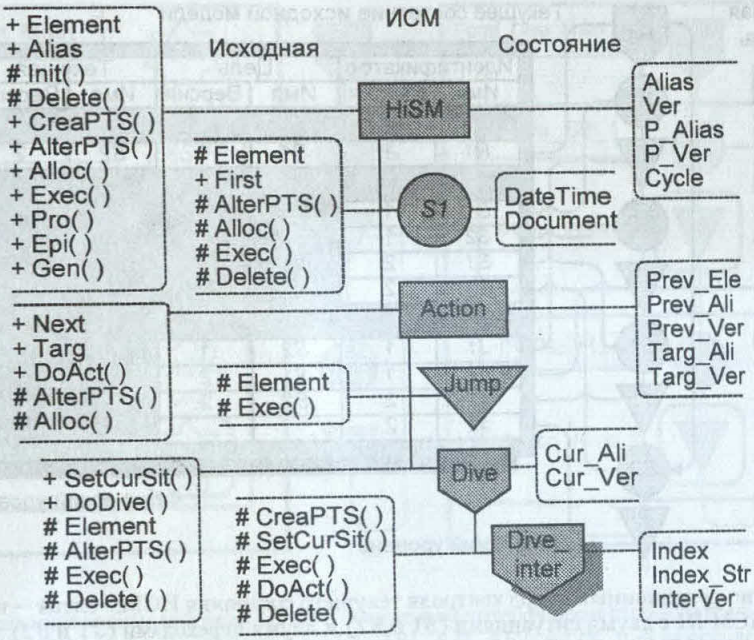


Рис. 3. Диаграмма классов объектов ИСМ на основе АДИ: слева — свойства и методы объектов исходной модели; справа — модели текущего состояния; «+» — добавляемые (новые), «#» — модифицируемые свойства/методы. Свойства элементов исходной модели: Element — тип; Alias — псевдоним; First — первая акция ситуации; Next — следующая акция; Targ — целевая ситуация. Методы элементов исходной модели: Init() — инициализация; Delete() — уничтожение; CreaPTS(), AlterPTS() — создание, модификация структуры объектов ПТС; Alloc() — размещение объектов ПТС; Exec() — интерпретация; Pro(); Gen(); Epi() — прологовая, основная, эпилоговая обработка при интерпретации объекта; DoAct() — второй этап интерпретации событийного предиката; DoDive() — то же для объекта-погружения; SetCurSit() — установка текущей ситуации субмодели. Свойства элементов текущего состояния: (Alias, Ver) — идентификатор объекта (псевдоним и версия); (P_alias, P_ver) — указатель на родительский объект; (Prev_Ele, Prev_Ali, Prev_Ver) — указатель на предшествующий объект; (Targ_Ali, Targ_Ver) — указатель целевого объекта; Cycle — идентификатор цикла интерпретации; Index, Index_Str — индексы элементов массива субмоделей

рекурсивного обхода моделей в процессе интерпретации одному объекту исходной модели может соответствовать несколько одновременно существующих объектов текущего состояния. Поэтому уникальность имен (псевдонимов) элементов модели недостаточна для обеспечения однозначности.

Исходя из этих предпосылок, предложен следующий *объектно-реляционный* метод контроля текущего состояния ИСМ (рис. 4) [4]:

1) Текущие объекты ИСМ на логическом уровне абстракции отражаются в ПТС в виде соответствующих текущих объектов логического уровня, которым на физическом уровне соответствуют записи в таблицах реляционной базы данных (БД) ПТС. Каждому классу объектов ИСМ: Dive, Sit, Jump — соответствуют свои таблицы в БД ПТС; каждому текущему объекту логического уровня соответствует запись в таблице БД ПТС. Связь объектов логической модели реализуется на основе одноименных атрибутов таблицы.

2) Для достижения однозначности идентификации объектов текущего состояния вводится двухкомпонентный идентификатор текущего объекта, представляющий собой пару $A_T = (A_и, V_A)$, где $A_и$ — идентификатор (псевдоним) соответствующего исходного объекта; V_A — версия данного текущего объекта, представляющая собой целое число, уникальное для подмножества текущих объектов данного класса с данным псевдонимом. Версия V_A вычисляется при создании записи объекта в таблице БД ПТС и хранится в течение всего времени существования объекта. Пара $(A_и, V_A)$ идентифицирует текущий объект и используется в качестве символического указателя, связывающего объекты текущего состояния. При этом первая компонента $A_и$ — псевдоним — обеспечивает доступ от объекта текущего состояния к соответствующему объекту исходной модели.

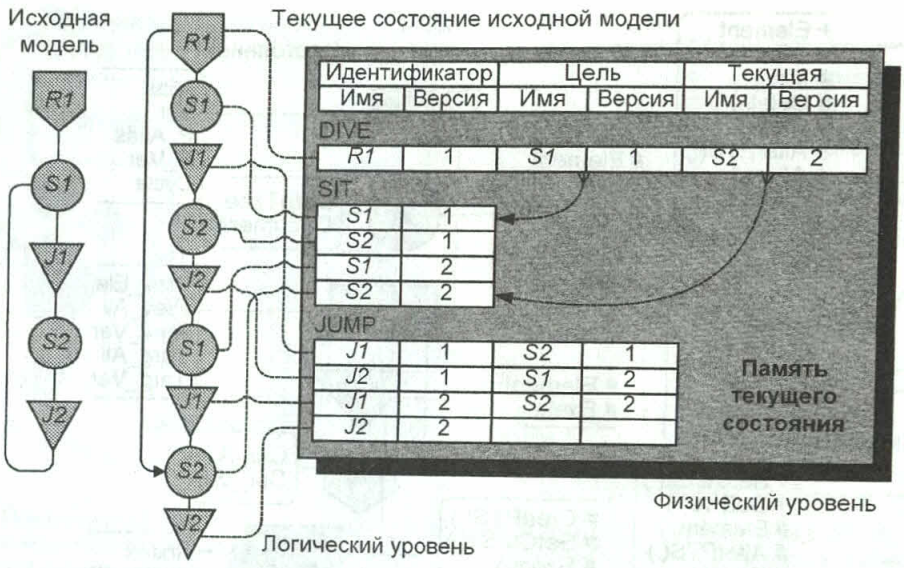


Рис. 4. Объектно-реляционный метод контроля текущего состояния ИСМ: слева — исходная одноуровневая ИСМ $R1$ с двумя ситуациями ($S1$ и $S2$) и двумя переходами ($J1$ и $J2$); в центре — текущее состояние ИСМ на логическом уровне (предыстория смены ситуаций $S1 \rightarrow S2 \rightarrow S1 \rightarrow S2$); справа — текущее состояние на физическом уровне (реляционные таблицы погружений — DIVE, ситуаций — SIT, переходов — JUMP; идентификаторы и указатели объектов в виде пар «Имя–Версия»)

В развитие указанных идей были определены состав атрибутов таблиц ПТС (рис. 3, справа), разработаны алгоритмы, реализующие методы объектов (рис. 3, слева), обеспечивающие в ходе интерпретации: создание/удаление объектов исходной модели, создание таблиц ПТС; создание записей в ПТС объектов текущего состояния; обработку событийных предикатов.

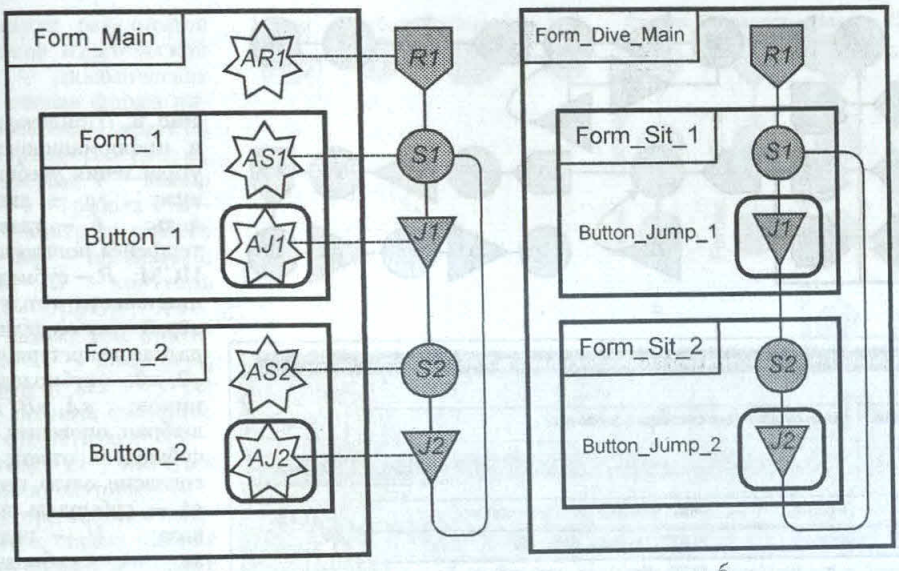
Разработанное алгоритмическое обеспечение является основой для программной реализации ИСМ в конкретной инструментальной среде. Для этого в работе использована среда Microsoft Visual FoxPro, дающая полномасштабные объектно-ориентированные возможности создания приложений, ориентированных на базы данных и интерфейс пользователя. Разработанные алгоритмы реализованы и отлажены в виде библиотеки классов объектов, содержащих все основные классы объектов ИСМ: 12 классов; 25 методов; 8 свойств; 53,2 Кбайт программного кода.

3. ОРГАНИЗАЦИЯ ВЗАИМОДЕЙСТВИЯ ИСМ С ОБЪЕКТНЫМ ИНТЕРФЕЙСОМ ПОЛЬЗОВАТЕЛЯ

Обсудим подходы к организации взаимодействия объектов ИСМ с визуальными объектами интерфейса пользователя: подход на основе объектов-агентов и альтернативный подход на основе встраивания объектов ИСМ в объекты интерфейса.

Взаимодействие с современным интерфейсом пользователя, организованным на основе визуальных объектов наряду с взаимодействием с БД, является важной частью автоматизированного принятия решений в информационных системах общего назначения. Через этот интерфейс пользователю предъявляются автоматизированные варианты решений в контексте текущей ситуации и фиксируется его выбор. Авторами предложено и исследовано два подхода к решению этой задачи (рис. 5): на основе независимой ИСМ и объектов-агентов; на основе встраивания объектов ИСМ в объекты интерфейса.

Первый подход (рис. 5, *а*) предусматривает разработку объектов-агентов, обеспечивающих связь объектов интерфейса с объектами ИСМ. Объекты-агенты, являющиеся «собственностью» соответствующих объектов ИСМ, встраиваются в объекты-контейнеры интерфейса (формы, кнопки, переключатели и т. п.). Агенты получают управление в ходе интерпретации соответствующих объектов ИСМ и выполняют действия по управлению объектами интерфейса: видимостью форм, доступностью кнопок и др. Кроме того, агенты настраивают обработчики событий элементов управления интерфейса (кнопок, переключателей и т. п.) на запуск методов об-



а *б*

Рис. 5. Организация взаимодействия ИСМ с объектным интерфейсом пользователя: *а* — на основе независимой ИСМ и объектов-агентов ($AR1$, $AS1$, $AJ1$, $AS2$, $AJ2$); *б* — на основе встраивания объектов ИСМ в объекты интерфейса. Модель $R1$ содержит ситуации $S1$ и $S2$ и переходы $J1$ и $J2$. Интерфейс с основной формой Form_Main и двумя подчиненными формами Form 1 и 2, содержащими кнопки Button_1 и Button_2. Текущая ситуация модели управляет видимостью подчиненных форм и доступностью кнопок; воздействия пользователя на кнопки управляют активностью переходов

работки событийных предикатов объектами ИСМ.

Второй подход (рис. 5, *б*) предусматривает встраивание в объекты интерфейса свойств и методов объектов ИСМ. В этом случае объекты интерфейса становятся «по совместительству» объектами ИСМ и управление ими осуществляется в рамках одного объекта контейнера. Проведенное сравнение подходов показало, что если первый из них обладает потенциально большей гибкостью в обеспечении взаимодействия, то второй — большей простотой и удобством в применении, экономичностью программного кода.

Для реализации предложенных подходов разработано программное обеспечение в среде объектного программирования Visual FoxPro: в первом случае — набор классов для порождения объектов-агентов, во втором — набор классов визуальных объектов интерфейса со встроенными свойствами и методами объектов ИСМ.

5. ПРАКТИЧЕСКАЯ АПРОБАЦИЯ РЕЗУЛЬТАТОВ

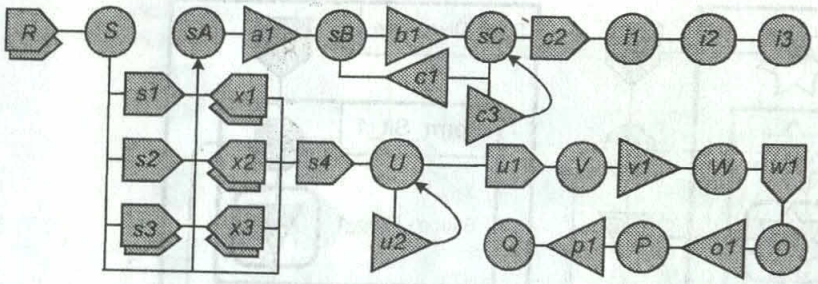
Достоверность и эффективность результатов была проверена путем их апробации в составе: 1) информационной системы для управления учебным процессом в вузе; 2) информационной издательской системы.

В первом случае (рис. 6) результаты использовались при реализации динамической модели и интерфейса пользователя для задачи «Формирование состава государственной аттестационной комиссии» в составе автоматизированной системы управления учебным процессом на уровне кафедры вуза[1], причем взаимодействие с интерфейсом пользователя было реализовано методом объектов-агентов.

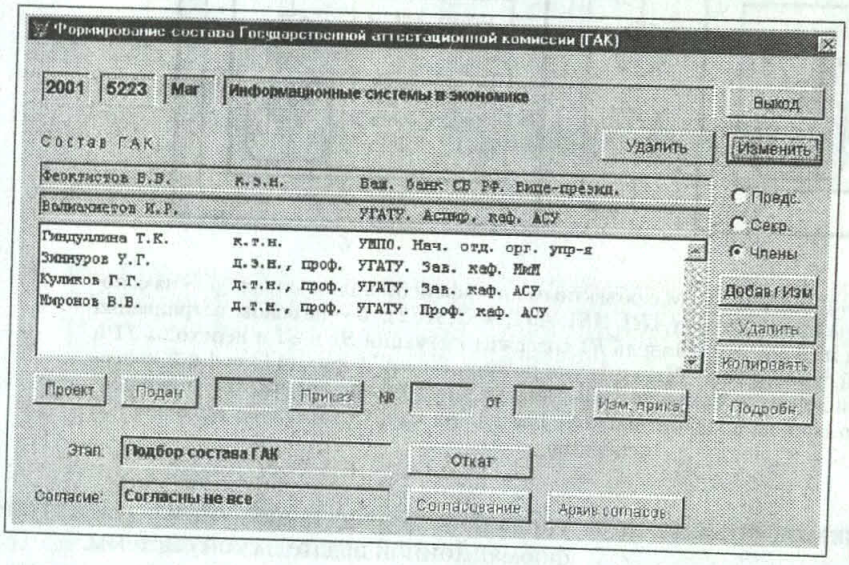
Разработанная динамическая модель в виде ИСМ включает массив субмоделей верхнего уровня, соответствующих отдельным специальностям или направлениям подготовки и включающим, в свою очередь, субмодели для подбора состава ГАК и согласования участия, подготовки проекта приказа и выпуска приказа, а также при необходимости — изменений к нему. Возможности ИСМ позволили достаточно просто учесть необходимость многократного изменения состава ГАК и соответствующего изменения приказа.

Разработанный интерфейс пользователя, обслуживающий задачу формирования состава ГАК, включает 10 экранных форм (на рис. 6 приведена только главная форма интерфейса), на которых размещены 28 информационных полей и списков для отображения необ-

[1] В разработке принимал участие А. Н. *Ситчихин*.



а



б

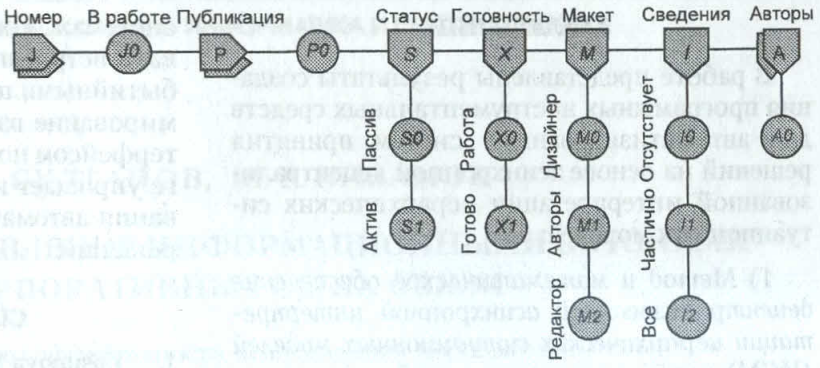
Рис. 6. Приложение результатов в информационной системе для управления учебным процессом в вузе: *а* — динамическая модель; *б* — главная форма интерфейса пользователя. Элементы ИСМ: R — субмодель выбора специальности и года обучения; $s1$, $s2$, $s3$ — субмодели выбора председателя, секретаря, членов ГАК; $x1$, $x2$, $x3$ — субмодели архива участников; sA, sB, sC — участник выбран, оповещен, ответил; $c2$ — субмодель ответа; $i1$, $i2$, $i3$ — согласие, отказ, неопределенность; $s4$ — субмодель подготовки приказа; U — подготовка приказа; $u1$ — субмодель этапов подготовки приказа; $u2$ — изменение приказа; V — подбор состава; $v1$ — укомплектован; W — оформление; $w1$ — субмодель состояния приказа; O — проект; $o1$ — передача в учебное управление; P — в учебном управлении; $p1$ — утверждение; Q — вышел; $a1$ — оповещение; $b1$ — ожидание ответа; $c1$ — повторное оповещение; $c3$ — повторный ответ

ходимой информации, а также 25 элементов управления, предназначенных для ввода решений пользователей и управления системой в процессе формирования состава ГАК.

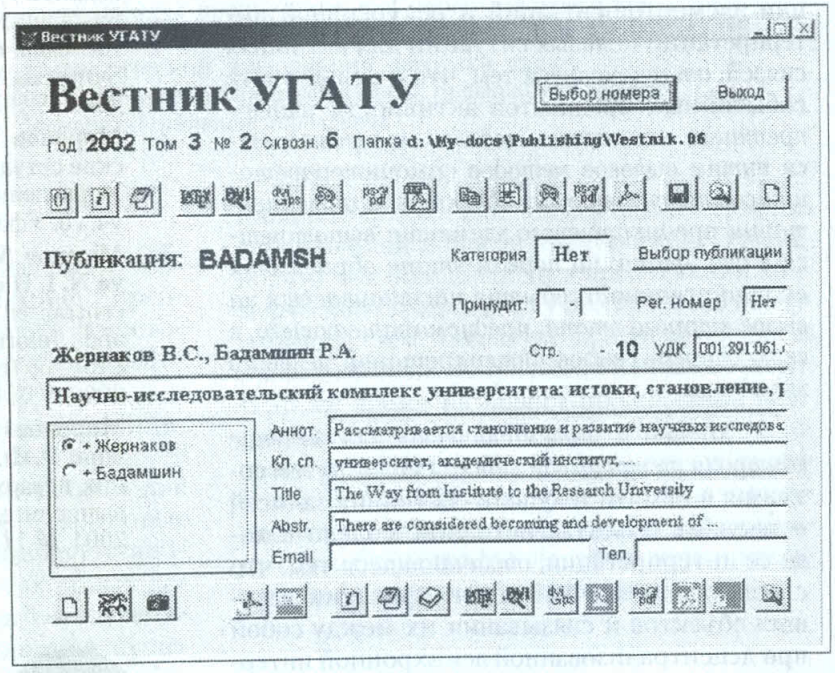
Динамическая модель через 19 объектов-агентов управляет состоянием интерфейса пользователя: видимостью экранных форм, доступностью и активностью элементов управления, характером отображаемой пользователями информации. В свою очередь, решения пользователей управляют состоянием динамической модели через объекты-агенты, воздействующие на модель при срабатывании элементов управления экранных форм.

Во втором случае (рис. 7) результаты использовались при реализации динамической модели и интерфейса пользователя для редактирования, верстки и изготовления оригинал-макета научного журнала «Вестник УГАТУ», причем взаимодействие с интерфейсом пользователя было реализовано методом встраивания объектов ИСМ в объекты интерфейса.

Динамическая модель представляет собой 4-уровневую ИСМ, обеспечивающую на первом уровне доступ к отдельным номерам журнала, на втором — к отдельным публикациям выбранного номера, на третьем — к отдельным элементам выбранной публикации (отдельным соавторам, названию, аннотации, ключевым словам и т. д.). Для каждой публикации динамическая модель контролирует состояние ее готовности и в этом контексте предоставляет пользователю доступ к инструментальным средствам для подготовки и исправления исходных текстов элементов, разметки их форматирования, компиляции, просмотра, печати, подготовки и отправки электронной почты соавторам с запросом недостающих сведений, получение ответов, выполнение других действий в рамках информационной технологии подготовки издания к выходу в свет. При этом ИСМ вместе с интерфейсом пользователя выполняет функции контекстно-ситуационного интегратора разнородных инструментальных средств (ориентированных на выполнение специализированных функций): текстовых редакторов, графических редакторов иллюстраций, про-



а



б

Рис. 7. Приложение результатов в информационной издательской системе: *а* — динамическая модель; *б* — главная форма интерфейса пользователя. Элементы ИСМ: J – выбор номера журнала; $J0$ – работа с номером; P — выбор публикации; $P0$ — работа с публикацией; S – контроль статуса публикации; $S0$ — пассивный; $S1$ – активный; X — контроль готовности; $X0$ – в работе; $X1$ – готова; M — нахождение оригинал-макета; $M0$ — у дизайнера; $M1$ – у авторов; $M2$ — у редактора; I — полнота сведений; $I0$ – сведения отсутствуют; $I1$ – имеются частично; $I2$ — имеются все; A – селекция соавтора; $A0$ – сведения о соавторе. Кнопки панелей инструментов, справа налево: верхней панели (номер журнала): «Новая публикация»; «Папка номера»; «Диск А»; «Просмотр Portable Document File (PDF)»; «Изготовление PDF Booklet»; «Просмотр Post Script (PS) Booklet»; «Изготовление PS Booklet»; «Раскладка PS»; «Просмотр PDF»; «Изготовление PDF»; «Просмотр PS»; «Изготовление PS»; «Просмотр Device Independent File (DVI)»; «Изготовление DVI»; «Файл Main»; «Файл Info»; «Файл Preamble»; нижней правой (публикация): «Папка публикации»; «Письмо с приложением PDF»; «Просмотр PDF»; «Просмотр PS»; «Изготовление PS»; «Просмотр DVI»; «Изготовление DVI»; «Файл Bibl»; «Файл Main»; «Файл Info»; «Письмо авторам»; «Письмо с приложением образцов файлов»; нижней левой (соавтор): «Фото автора»; «Сведения об авторе»; «Новый соавтор»

грамм набора и верстки, конвертации, электронной почты и др.

Разработанный интерфейс пользователя, обслуживающий эту задачу, включает 5 экранных форм, на которых размещены 28 информационных полей и списков для отображения необходимой информации, а также 49 элементов управления, предназначенных для ввода решений пользователей и доступа к специализированным инструментальным средствам.

По результатам апробации проведен анализ эффективности предложенного подхода при построении интерактивных информационных систем. Отмечено, что предложенный подход обеспечивает:

1) возможность интерактивного управления активностью предикатов ИСМ, что расширяет сферу использования ИСМ в человеко-машинных информационных системах;

2) большую «понятность» внешнего представления ИСМ, что облегчает процессы проектирования (и перепроектирования) информационной системы;

3) упрощение программирования динамического управления состояниями и свойствами экранных форм пользовательского интерфейса, что сокращает сроки создания информационной системы. Экспертные оценки показывают, что использование ИСМ с предысторией в 2–3 раза сократило сроки программирования рассмотренных систем.

ЗАКЛЮЧЕНИЕ

В работе представлены результаты создания программных инструментальных средств для автоматизированных систем принятия решений на основе асинхронной децентрализованной интерпретации иерархических ситуационных моделей:

1) *Метод и математическое обеспечение децентрализованной асинхронной интерпретации иерархических ситуационных моделей* (ИСМ) путем последовательной интерпретации предикативных связей при интерпретации элементов-ситуаций и рекурсивной интерпретации целевых ситуаций для активных связей, *отличающиеся* тем, что с целью учета событийных предикатов активности *интерпретация элементов модели осуществляется путем вызовов методов самоинтерпретации соответствующих объектов, а интерпретация предикативного элемента выполняется в два этапа: на первом этапе обработчик ассоциированного события настраивается на вызов второго этапа, предусматривающего, в свою очередь, вызов интерпретации целевого элемента.*

2) *Метод и математическое обеспечение контроля текущего состояния ИСМ* путем создания в памяти текущего состояния записей о текущих объектах исходной модели в ходе ее интерпретации, *отличающиеся* тем, что с целью однозначной идентификации текущих объектов и связывания их между собой при децентрализованной асинхронной интерпретации *в состав записи о текущем объекте включаются имена класса и элемента исходной модели, а также номер версии, уникальный для подмножества объектов данного класса с данным именем.*

3) *Метод и математическое обеспечение организации взаимодействия ИСМ с пользователем* на основе создания визуальных объектов интерфейса пользователя и их связи с объектами ИСМ, *отличающиеся* тем, что с целью повышения экономичности и снижения трудоемкости программной реализации *в качестве родительских классов исходной ИСМ используются классы визуальных объектов интерфейса пользователя.*

Разработанные методы и их математическое обеспечение реализованы в виде компьютерного программного обеспечения и апробированы в реальных информационных системах. Их применение позволяет создавать встроенные динамические модели с событийными предикатами, облегчает программирование взаимодействия с объектным интерфейсом пользователя, что в конечном итоге упрощает и сокращает процесс проектирования автоматизированных систем принятия решений.

СПИСОК ЛИТЕРАТУРЫ

1. **Yusupova N. I., Mironov V. V.** Hierarchical situational models and linguistic means of their realization // Proc. of World Multiconference on Systemics, Cybernetics and Informatics and 5th Int. Conf. on Information Systems Analysis and Synthesis. Orlando, Florida, 1999. V. 7. P. 233–236.
2. **Миронов В. В., Ситчихин А. Н.** Иерархические ситуационные модели с предысторией // Управление в сложных системах: Межвуз. науч. сб. Уфа: УГАТУ, 1999. С. 55–68.
3. **Mironov V. V., Akhmetshin R. F., Yussupova N. I.** Hierarchical situational models with decentralized interpreting // Computer Science and Information Technologies (CSIT'2000): Proc. of the 2nd Int. Workshop. Ufa, Russia, 2000. V. 3. P. 29–31.
4. **Миронов В. В., Ситчихин А. Н., Ахметшин Р. Ф.** Объектно-реляционная реализация иерархических ситуационных моделей в вычислительной среде // Вестник УГАТУ. 2001. № 1 (3). С. 185–189.

ОБ АВТОРАХ

Миронов Валерий Викторович, профессор кафедры автоматизированных систем управления УГАТУ. Дипл. радиофизик (Воронежск. гос. ун-т, 1975), д-р техн. наук по управлению техническими системами (УГАТУ, 1995). Исследования критических ситуаций и ситуационного управления.

Ахметшин Радик Фагимович, аспирант той же кафедры. Дипл. инж. по информационным системам (УГАТУ, 1999). Подготовил диссертацию об объектно-ориентированных иерархических ситуационных моделях.